Block-encoding dense and full-rank kernels using hierarchical matrices: applications in quantum numerical linear algebra

Quynh T. Nguyen[1,2], Bobak T. Kiani[1,3], and Seth Lloyd[3,4,5]

[1]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA
[2]Department of Physics, Massachusetts Institute of Technology, USA
[3]Research Laboratory of Electronics, Massachusetts Institute of Technology, USA
[4]Department of Mechanical Engineering, Massachusetts Institute of Technology, USA
[5]Turing Inc., Cambridge, MA, USA

Many quantum algorithms for numerical linear algebra assume black-box access to a block-encoding of the matrix of interest, which is a strong assumption when the matrix is not sparse. Kernel matrices, which arise from discretizing a kernel function $k(x, x')$, have a variety of applications in mathematics and engineering. They are generally dense and full-rank. Classically, the celebrated fast multipole method performs matrix multiplication on kernel matrices of dimension N in time almost linear in N by using the linear algebraic framework of hierarchical matrices. In light of this success, we propose a block-encoding scheme of the hierarchical matrix structure on a quantum computer. When applied to many physical kernel matrices, our method can improve the runtime of solving quantum linear systems of dimension N to $O(\kappa \operatorname{polylog}(\frac{N}{\varepsilon}))$, where κ and ε are the condition number and error bound of the matrix operation. This runtime is near-optimal and, in terms of N, exponentially improves over prior quantum linear systems algorithms in the case of dense and full-rank kernel matrices. We discuss possible applications of our methodology in solving integral equations and accelerating computations in N-body problems.

1 Introduction

One of the most promising applications of quantum information processing is in solving linear algebraic problem in high dimensions efficiently. Since the Harrow-Hassidim-Lloyd (HHL) algorithm was first developed for solving linear systems of equations [1], a number of improvements have been proposed. Significantly, the breakthrough algorithms in [2] and [3] solved sparse linear systems of equations in time polylogarithmic in the system size N and the target error bound ε. When these algorithms are applied to dense matrices, however, their complexity scales linearly in N (see Table 1). Using a quantum random access memory (QRAM) data structure [4], the work of [5] improved the runtime to $\widetilde{O}(\sqrt{N})$. Classically, quantum-inspired algorithms [6, 7] use an analog of QRAM called sample-query access and apply sketching techniques [8] to solve linear systems with runtimes independent of N but scaling poorly with the rank k of the matrix as $\widetilde{O}(k^6)$ (see Appendix A).

In fact, achieving a runtime logarithmic in N for dense and full-rank matrices is generally challenging unless there are specific symmetries or structures inherent in the matrix.

For example, quantum algorithms have been developed to achieve polylogarithmic complexity in N for Toepliz systems [9], Hankel matrices [10], and linear group convolutions [11] which all feature the group quantum Fourier transforms (QFT) [12] to implement operations as sparse matrices in the Fourier basis. This approach of using group QFTs, however, only applies for certain groups and requires the matrix entries to be *exactly* uniformly sampled from a generating function, thus inapplicable to more general settings (e.g., matrix entries corresponding to atoms on a lattice vibrating around their equilibrium positions).

Kernel matrices are a class of matrices whose entries are calculated as a function of a kernel $k(x, x')$. They have applications in mathematical physics [13], engineering [14] and machine learning [15], where they are often obtained by sampling or discretizing a smooth kernel $k(x, x')$ that reflects the physical model at hand. In general, kernel matrices are dense and full-rank. Hierarchical matrices ($\mathcal{H}$-matrices) [16, 17] are a classical framework for efficiently performing matrix operations on commonly found kernel matrices by hierarchically splitting the "space" dimension of the matrix entries. In terms of the matrix dimension N, classically performing matrix-vector multiplication with $\mathcal{H}$-matrices takes time $O(N \operatorname{polylog} N)$ which is a significant improvement over the generic $O(N^2)$ time. This $\mathcal{H}$-matrix structure lies at the heart of the celebrated fast multipole method [13]. This improvement intuitively arises because kernel matrices often decay or change slowly along entries that are farther from the diagonals of the matrix. $\mathcal{H}$-matrices progressively approximate the matrix in larger low-rank blocks for farther off-diagonal entries to efficiently implement kernel matrices.

We present an efficient quantum algorithm to implement $\mathcal{H}$-matrices in the block-encoding framework, which embeds matrices inside a unitary operator that can be queried multiple times. This construction, which is at the core of many quantum algorithms, allows one to conveniently perform matrix computations on a quantum computer [18–21]. As many recently proposed algorithms use block-encodings as a black box, one of the major challenges for quantum numerical linear algebra in this framework is to efficiently block-encode matrices so that the asymptotic runtimes are efficient even when the block-encoding complexity is included. Our $\mathcal{H}$-matrix block-encoding procedure, combined with existing quantum linear systems algorithms, *e.g.,* adiabatic solver in [22], provides near-optimal runtime in solving quantum linear systems for many typical kernels that are neither sparse nor low-rank, *e.g.,* $k(x, x') = \|x - x'\|^{-1}$ (see Table 1 for comparisons to prior work). Apart from the $\mathcal{H}$-matrix block-encoding, we also provide in the Appendix a list of common kernel matrices that can be efficiently block encoded using existing techniques.

We proceed as follows. First, we overview classical hierarchical matrices in Section 2 and the quantum block-encoding framework in Section 3. We then present an algorithm for efficiently block-encoding hierarchical matrices and apply this to kernel matrices in Section 4. In Section 4.4, we generalize our methodology to construct block encodings for a more general class of matrices whose structure is derived from hierarchical matrices. Finally, we discuss applications of our algorithm in calculating N-body gravitational potentials and solving integral equations in Section 5.

2 Background in $\mathcal{H}$-matrices and hierarchical splitting

To observe why hierarchical matrices are useful, consider a physical model which requires summation or integration over spacetime. The N-body gravitational force problem [23],

Result	Complexity		Input model	Algorithm
	Forward $\widetilde{O}(\cdot)$	Inversion $\widetilde{O}(\cdot)$		
Classical [16]	N	N	Classical	Classical $\mathcal{H}$-matrix
GST [6]/SM [7]	-	$\frac{N^6\kappa^8}{\varepsilon^2}$	Sample query	Classical sketching
CKS [2]/GSLW [3]	$N\kappa$	$N\kappa$	Oracle access[a]	Lemma 5.2[b]
KP [4]/WZP [5]	$\sqrt{N}\kappa$	$\sqrt{N}\kappa$	QRAM[c]	Lemma 5.2
This paper	$\operatorname{polylog}(N)\kappa$	$\operatorname{polylog}(N)\kappa$	Oracle access	Lemma 4.3, Lemma 5.2

[a] Naive block-encoding using oracle access via Lemma 48 or 49 in [3] (also see Lemma 3.2)
[b] Lemma 5.2 uses the adiabatic solver in [22] which has optimal runtime dependence on κ
[c] This input model is strictly more powerful than the oracle access model

Table 1: Runtimes of previous methods versus our method for matrix multiplication (forward) and solving linear systems (inversion) of dense and full-rank kernel matrices, *e.g.*, $k(x,x') = \|x-x'\|^{-1}$. For purposes of comparison, we assume the operator norm of the matrix is bounded. The $\widetilde{O}$ notation hides terms that grow as $\operatorname{polylog}(\frac{N}{\varepsilon})$. Here, κ is the condition number of the matrix operation and ε is the error bound on the output. For quantum algorithms, we assume the input vector is given as a quantum state and the runtime is equal to the circuit depth times the number of queries needed to obtain the output as a quantum state. We note that classical $\mathcal{H}$-matrix literature and this work focus on kernel matrices, whereas the quantum algorithms we compare to are "general-purpose". For these general-purpose algorithms, runtimes are calculated assuming they are applied directly on kernel matrices. More details about prior works can be found in Appendix A.

for example, requires evaluating N potentials of the form

$$\Phi\left(\mathbf{x}_j\right) = \sum_{\substack{i=1 \\ i\neq j}}^{N} \frac{m_i}{\|\mathbf{x}_j - \mathbf{x}_i\|^2}, \qquad 1 \le j \le N \tag{1}$$

where m_i and $\mathbf{x}_i$ denote the mass and location of particle i. Summing or integrating over all particles is equivalent to performing matrix-vector multiplication, generally requiring $O(N^2)$ operations unless the underlying matrix is sparse or has special properties. As we will show, when particles are placed in a favorable arrangement (*e.g.,* approximately uniformly), this matrix-vector multiplication can be approximately performed in time $O(N \operatorname{polylog} N)$.

Hierarchical matrices ($\mathcal{H}$-matrices) [16, 17, 24] are a powerful algebraic framework for accelerating matrix computations involving displacement kernels $k(\mathbf{x},\mathbf{x}') = k(\|\mathbf{x}-\mathbf{x}'\|)$ which satisfy certain properties outlined below. The key idea is to (i) split the kernel matrix into hierarchically off-diagonal blocks and (ii) approximate each block by a low-rank matrix. This low rank approximation is justified because off-diagonal blocks represent distant interactions that are small or slowly changing with distance. We now detail the two ingredients of $\mathcal{H}$-matrices, namely the hierarchical splitting and the off-diagonal low-rank approximation.

2.1 Hierarchical splitting into admissible blocks

Hierarchical splitting partitions a kernel matrix into so-called "admissible" blocks where entries representing more distant interactions are grouped together into larger blocks. Consider a system of N particles, whose pairwise interaction is described by the kernel $k(x,x')$. The kernel matrix of this system is defined as

$$\mathbf{K} = \left(k(x_i,x_j)\right)_{i,j=0}^{N-1}, \tag{2}$$

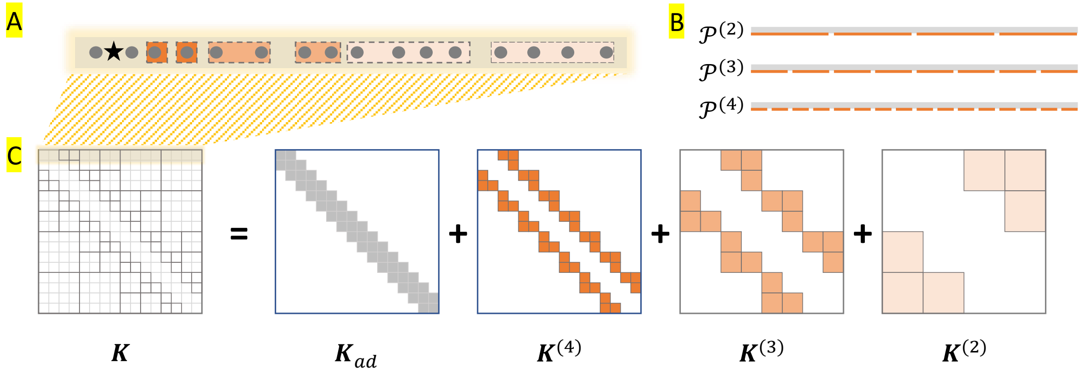

Figure 1: (A) Structure of a hierarchical splitting for a 1D array of 16 particles (gray circles), which are close to uniformly distributed. The accumulative potential at the destination location (star, which can also be one of the sources) is decomposed into contributions from clusters of source particles whose sizes grow with the distance to the destination. (B) Cluster size decreases exponentially with the level of the hierarchical splitting. (C) The associated kernel matrix of this 16-particle system takes the form of an $\mathcal{H}$-matrix, assuming that the destination locations are the same as the source locations. This matrix is decomposed into hierarchically structured blocks using the hierarchical splitting in (A), where each block is numerically low-rank.

where x_i is the location of particle i. For ease of analysis, we assume the particles are uniformly distributed and choose $x_i = i/N$, for any $i \in \{0, 1, \dots, N-1\}$ and $N = 2^L$. For example, this is the case when the kernel matrix is obtained by discretizing an integral operator over the domain $[0, 1]$ (see [16]). In general, a kernel matrix need not be square and the source locations x_i need not be 1-dimensional nor equally spaced and we will show how to generalize the current setup to these cases in later sections. We refer the interested readers to [17, 25–28] for further details.

A set of m particles located at $\{x_{j_1}, \dots, x_{j_m}\}$ is denoted as a *cluster* σ, storing the list of individual indices $\sigma = \{j_1, \dots, j_m\}$.

Definition 2.1 (Admissible blocks). *Let* $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$ *and* $\mathcal{N} = \{0, 1, \dots, N-1\}$. *Let* $\mathcal{P}(\mathcal{N})$ *be a partition of* $\mathcal{N}$, *i.e.,* $\mathcal{N} = \bigcup_{\sigma \in \mathcal{P}(\mathcal{N})} \sigma$. *For a cluster* $\sigma \in \mathcal{P}(\mathcal{N})$, *define the center of* σ *as* $c_\sigma = \mathrm{mean}\{x_i | i \in \sigma\}$ *and the radius of* σ *as* $r_\sigma = \max_{i \in \sigma} |x_i - c_\sigma|$. *Furthermore, for* $\sigma, \rho \in \mathcal{P}(\mathcal{N})$, *define the distance between the two clusters* σ *and* ρ *as* $\mathrm{dist}(\sigma, \rho) = \min_{i \in \sigma, j \in \rho} |x_i - x_j|$. *Then, the matrix block* $\mathbf{K}^{\sigma,\rho} = (k(x_i, x_j))_{i \in \sigma, j \in \rho}$ *is called admissible if* $\eta \cdot \max\{r_\sigma, r_\rho\} \le \mathrm{dist}(\sigma, \rho)$, *where* $\eta = 2$.

Informally, a block is admissible if the distance between its two corresponding clusters is larger than their radii (other values of η can give rise to different forms of the hierarchical splitting, see [16, 25]). We now split the matrix $\mathbf{K}$ into a hierarchy of admissible blocks consisting of $L = \log_2 N$ levels of decreasing radii. At level ℓ, define the following partition of $\mathcal{N} = \{0, 1, \dots, N-1\}$:

$$\mathcal{P}^{(\ell)} = \{\sigma_I^{(\ell)} | I \in \{0, \dots, 2^\ell - 1\}\}, \tag{3}$$

where

$$\sigma_I^{(\ell)} = \{i \in \mathcal{N} | x_i \in [I/2^\ell, (I+1)/2^\ell)\}. \tag{4}$$

For example, $\mathcal{P}^{(0)} = \{\mathcal{N}\}$, $\mathcal{P}^{(L)} = \{\{i\} | i \in \mathcal{N}\}$, and $\mathcal{P}^{(\ell)}$ for $\ell \in \{2, 3, 4\}$ are shown in Figure 1B. Note that admissible blocks only exist for $\ell \ge 2$. We now construct a hierarchical splitting of our kernel matrix $\mathbf{K}$ into admissible blocks starting from the coarsest level $\ell = 2$.

First, we split $\mathbf{K}$ into admissible and inadmissible blocks in $\mathcal{P}^{(2)}$

$$\mathbf{K} = \mathbf{K}^{(2)} + \mathbf{K}^{(2)}_{inadmissible}.$$

Next, we split $\mathbf{K}^{(2)}_{inadmissible}$ into admissible and inadmissible blocks in $\mathcal{P}^{(3)}$ (excluding the admissible blocks of $\mathcal{P}^{(3)}$ that are already covered in $\mathbf{K}^{(2)}$)

$$\mathbf{K}^{(2)}_{inadmissible} = \mathbf{K}^{(3)} + \mathbf{K}^{(3)}_{inadmissible}.$$

Recursively applying the decomposition $\mathbf{K}^{(\ell-1)}_{inadmissible} = \mathbf{K}^{(\ell)} + \mathbf{K}^{(\ell)}_{inadmissible}$ up to level L, we obtain a hierarchically structured representation of $\mathbf{K}$

$$\mathbf{K} = \sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}, \tag{5}$$

where $\mathbf{K}_{ad} = \mathbf{K}^{(L)}_{inadmissible}$ is a tridiagonal matrix which represents the "adjacent" interaction of neighboring particles (see Figure 1C). Observe that the matrices $\mathbf{K}^{(\ell)}$ are block-sparse; that is, each block-row or block-column of $\mathbf{K}^{(\ell)}$ contains at most three non-zero admissible blocks. Furthermore, these blocks represent "far-field" interactions which can be approximated by low-rank matrices if the kernel $k(x, x')$ satisfies some additional properties discussed in the next section.

In Appendix D.3, we describe a generalization of the hierarchical splitting to higher dimensions, where the two key properties still hold, i.e., there are only $\log N$ levels and each level $\mathbf{K}^{(\ell)}$ is block-sparse.

2.2 Low-rank approximation of admissible blocks

For far-field interactions in admissible blocks, the kernel $k(x, x')$ can be well approximated by a truncated Taylor series if it satisfies the following *asymptotic smoothness* condition:

$$|\partial^q_{x'} k(x, x')| \leq C q! |x - x'|^{-q} \quad (\forall q \in \mathbb{N}), \tag{6}$$

where C is a constant. For instance, the log kernel $k(x, x') = \log(|x - x'|)$ directly satisfies the above. This condition enables approximating any admissible block $\mathbf{K}^{\sigma,\rho}$ by a rank-p matrix

$$\mathbf{K}^{\sigma,\rho} \approx \widetilde{\mathbf{K}}^{\sigma,\rho} = \mathbf{\Psi}^{\sigma,\rho} \mathbf{D} \left(\mathbf{\Phi}^{\rho}\right)^{\dagger}, \tag{7}$$

where $\mathbf{D} = \mathrm{Diag}(1/q!)_{q \in P} \in \mathbb{R}^{p \times p}$, $P = \{0, 1, \ldots, p-1\}$ and

$$\mathbf{\Psi}^{\sigma,\rho} = (\partial^q_{x'} k(x_i, c_\rho))_{i \in \sigma, q \in P} \in \mathbb{R}^{|\sigma| \times p}, \tag{8}$$

$$\mathbf{\Phi}^{\rho} = ((x'_j - c_\rho)^q)_{j \in \rho, q \in P} \in \mathbb{R}^{|\rho| \times p}. \tag{9}$$

Here, p regulates the precision of the approximation (typically $p = \mathrm{polylog}(N/\varepsilon)$ for an error bound ε). A detailed example and error analysis of this far-field approximation can be found in Appendix D. In addition, we note that conditions other than that of Equation 6 can also be used to justify the approximation (see [29–31]) of admissible blocks by low-rank kernels $k(x, x') = \sum_{q=0}^{p-1} \psi_q(x) \varphi_q(x')$.

Combining this low-rank approximation and the hierarchical splitting forms a complete $\mathcal{H}$-matrix. In Appendix D, we show that matrix-vector multiplication with an $\mathcal{H}$-matrix can be performed in time $O(Np \log N)$. In addition, other $\mathcal{H}$-matrix operations such as matrix-matrix multiplication and matrix inversion run in time $O(Np^2 \log^2 N)$, we refer the reader to [16, 24, 25] for further details. The rank p can be chosen at each level to further optimize computational complexity versus approximation error [26].

3 Block-encoding arithmetic

Throughout this paper, we employ the block-encoding framework to perform matrix computations on a quantum computer [18]. Here, an s-qubit (non-unitary) operator $A \in \mathbb{C}^{2^s \times 2^s}$ (let $N = 2^s$) is embedded in a unitary operator $U \in \mathbb{C}^{2^{(s+a)} \times 2^{(s+a)}}$ using a ancilla qubits such that the top left block of U is proportional to A:

$$U = \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{10}$$

where α is the *normalization factor* of the block-encoding. Formally, the $(s+a)$-qubit unitary U is an (α, a, ε)-block-encoding of A if

$$\|A - \alpha(\langle 0^a| \otimes I_s)U(|0^a\rangle \otimes I_s)\| \leq \varepsilon, \tag{11}$$

where $\|\cdot\|$ denotes the operator norm of a matrix and I_s is the identity operator on s qubits. In other words, applying the unitary U to a quantum state $|\psi\rangle$ and post-selecting on the measurement outcome $|0^a\rangle$ on the ancilla qubits is equivalent to applying the operation A on $|\psi\rangle$:

$$U|0^a\rangle|\psi\rangle = \frac{1}{\alpha}|0^a\rangle A|\psi\rangle + |G\rangle, \tag{12}$$

where $|G\rangle$ is a garbage state that is orthogonal to the subspace $|0^a\rangle$, *i.e.*, $(\langle 0^a| \otimes I_s)|G\rangle = 0$. The probability of successfully post-selecting $|0^a\rangle$ is thus equal to $\|A|\psi\rangle\|^2\alpha^{-2} \leq (\|A\|/\alpha)^2$. Observe that $\alpha \geq \|A\|$ is required to embed A inside a unitary matrix. Further, since the probability of success of the post-selection step depends on the *ratio* $\|A\|/\alpha$, a block-encoding U is *optimal* if $\alpha = \Theta(\|A\|)$. More generally, if the ratio $\alpha/\|A\|$ grows logarithmically with the size of the matrix N, the probability of success also only changes logarithmically with N. More generally, as we will see some examples in Section 5, the complexity of quantum algorithms [3] in the block-encoding framework depends on the ratio $\alpha/\|A\|$. This important observation motivates the following definition of a "good" block-encoding.

Definition 3.1 (Good block-encoding)**.** *Given a matrix $A \in \mathbb{C}^{N\times N}$, an (α, a, ε)-block-encoding U of A is good if and only if the ratio $\alpha/\|A\| = O(\text{polylog}(N))$, and additionally, $a = O(\text{polylog}(\frac{N}{\varepsilon}))$ and the circuit depth of U is $O(\text{polylog}(\frac{N}{\varepsilon}))$. The block-encoding is called* optimal *if $\alpha = \Theta(\|A\|)$.*

Throughout this paper, we assume that matrices are normalized to have the largest entry at most 1, and we analyze the optimality of block-encodings by comparing the normalization factor α to the operator norm $\|A\|$.

Before proceeding to block-encode a hierarchical matrix, we provide various methods for block-encoding the individual matrix blocks in the hierarchical splitting. Depending on the locations of the matrix blocks and how the values of the kernel entries can be accessed by a quantum computer, there are various different ways one can block-encode a given matrix. Previous work has proposed efficient means to block-encode sparse matrices, purified density matrices, etc., as well as transformations of these matrices in the block-encoding framework [3, 18, 32]. We list some previously established results in this framework in Appendix B, among which we particularly use Lemma B.5 to block-encode sparse matrices with oracle access to entries, Lemma B.3 to linearly combine block-encoded matrices, and Lemma B.4 to multiply block-encoded matrices. Later, we will apply and combine these individual block-encoding Lemmas to form a block-encoding of the complete hierarchical matrix.

First, we provide a version of Lemma B.5 applied to dense matrices, which we refer to as the "naive" block-encoding approach since it is unaware of the inherent structure of the block-encoded matrix.

Lemma 3.2 (Naive block-encoding of dense matrices with oracle access)**.** *Let $A \in \mathbb{C}^{N\times N}$ (where $N = 2^s$) and let $\hat{a} \geq \max_{i,j} |a_{ij}|$. Suppose the following oracle is provided*

$$\mathcal{O}_A : |i\rangle\, |j\rangle\, |0^b\rangle \to |i\rangle\, |j\rangle\, |\tilde{a}_{ij}\rangle\,,$$

where $0 \leq i, j < N$ and $\tilde{a}_{ij}$ is the (exact) b-qubit description of $a_{ij}/\hat{a}$. Then one can implement a $(N\hat{a}, s+1, \varepsilon)$-block-encoding of A with two uses of $\mathcal{O}_A$, $O(\mathrm{polylog}(\frac{\hat{a}N}{\varepsilon}))$ one- and two-qubit gates and $O(b, \mathrm{polylog}(\frac{\hat{a}N}{\varepsilon}))$ extra qubits (which are discarded before the post-selection step).

Assuming $\hat{a} = 1$, the normalization factor of this block-encoding is N, which is suboptimal if $\|A\| \ll N$, *e.g.,* this situation arises when the matrix A has many small entries. This suboptimal block-encoding could lead to an exponentially small probability of success in the post-selection step. Therefore, a structure-aware block-encoding procedure is needed to achieve an optimal runtime in terms of N.

To directly implement the $\mathcal{H}$-matrices outlined in Section 2, we need procedures to block-encode block-sparse and low-rank matrices which we provide in the Lemmas below (all proofs are deferred to Appendix B). First we define state preparation pairs, which are frequently used in this study to prepare the coefficients in a linear combination of matrices.

Definition 3.3 (State preparation pair [3])**.** *Let $y \in \mathbb{C}^m$ and $\|y\|_1 \leq \beta$, the pair of unitaries (P_L, P_R) is called a (β, n, ε)-state-preparation-pair of y if $P_L\,|0^n\rangle = \sum_{j=0}^{2^n-1} c_j\,|j\rangle$ and $P_R\,|0^n\rangle = \sum_{j=1}^{2^n-1} d_j\,|j\rangle$ such that $\sum_{j=0}^{m-1} \left|\beta c_j^* d_j - y_j\right| \leq \varepsilon_1$ and $c_j^* d_j = 0$ for any $j \in \{m, \ldots, 2^n - 1\}$.*

Next, we show how to implement a block-encoding of low-rank matrices.

Lemma 3.4 (Block-encoding of low-rank operators with state preparation unitaries, inspired by Lemma 1 of [33])**.** *Let $A = \sum_{i=0}^{p-1} \sigma_i \mathbf{u}_i \mathbf{v}_i^\dagger \in \mathbb{C}^{2^s\times 2^s}$ where $\|\mathbf{u}_i\| = \|\mathbf{v}_i\| = 1$. Let $r = \lceil \log p \rceil$ and $\sum_{i=0}^{p-1} |\sigma_i| \leq \beta$. Suppose the following $(r+s)$-qubit unitaries are provided:*

$$G_L : |i\rangle\, |0^s\rangle \to |i\rangle\, |\mathbf{u}_i\rangle\,,$$
$$G_R : |i\rangle\, |0^s\rangle \to |i\rangle\, |\mathbf{v}_i\rangle\,,$$

where $0 \leq i < p$ and $|\mathbf{u}_i\rangle$ ($|\mathbf{v}_i\rangle$) is the quantum state whose amplitudes are the entries of $\mathbf{u}_i$ ($\mathbf{v}_i$). Let (P_L, P_R) be a (β, r, ε)-state-preparation-pair for the vector $[\sigma_0, \ldots, \sigma_{p-1}]$. Then, one can construct a $(\beta, r+s, \varepsilon)$-block-encoding of A with one use of each of $G_L, G_R^\dagger, P_L^\dagger, P_R$, and a SWAP *gate on two s-qubit registers.*

The above lemma provides one method to block-encode the approximately low-rank matrices described in Section 2, but as we will see, is not the only option one has at their disposal. When the rank of a matrix is small and bounded, the time needed to prepare unitaries P_L, P_R is typically negligible; *e.g.,* they can be efficiently implemented with an oracle providing access to σ_i (and β is close to the sum of the singular values). In addition, the unitaries G_R, G_L can be constructed using existing state preparation methods based on the structure of the entries in the singular vectors (*e.g.,* [34] utilized efficiently integrable distributions, [4] used a QRAM data structure). In Appendix B.3, we also present an approach to efficiently construct G_R, G_L from oracles providing access to the entries of $\mathbf{u}_i, \mathbf{v}_i$ for cases where the entries change slowly (Lemma B.8).

Next, the following lemma block-encodes a block-sparse matrix, whose blocks are also block-encoded.

Lemma 3.5 (Block-encoding of block-sparse matrices)**.** *Let $A = \sum_{i,j=0}^{2^t-1} |i\rangle\langle j| \otimes A^{ij}$ be a d_r-row-block-sparse and d_c-column-block-sparse matrix, where each A^{ij} is an s-qubit operator. Let U^{ij} be an $(\alpha_{ij}, a, \varepsilon)$-block-encoding of A^{ij}. Suppose that we have the access to the following $2(t+1)$-qubit oracles*

$$\mathcal{O}_r : |i\rangle |k\rangle \to |i\rangle |r_{ik}\rangle ,$$
$$\mathcal{O}_c : |l\rangle |j\rangle \to |c_{lj}\rangle |j\rangle ,$$

where $0 \le i < 2^t, 0 \le k < d_r$, $0 \le j < 2^t, 0 \le l < d_c$, r_{ik} is the index for the k-th non-zero block of the i-th block-row of A, or if there are less than k non-zero blocks, then $r_{ik} = k+2^t$, and similarly c_{lj} is the index for the l-th non-zero block of the j-th block-column of A, or if there are less than l non-zero blocks, then $c_{lj} = l + 2^t$. Additionally, suppose the following oracle is provided:

$$\mathcal{O}_\alpha : |i\rangle |j\rangle |z\rangle \to |i\rangle |j\rangle |z \oplus \tilde{\alpha}_{ij}\rangle$$

where $0 \le i, j < 2^t$ and $\tilde{\alpha}_{ij}$ is the b-qubit description of α_{ij} (if i or j is out of range then $\tilde{\alpha}_{ij} = 0$), and let the $(2t+2+s+a)$-qubit unitary $W = \sum_{i,j:A^{ij}\neq 0}(|i\rangle |j\rangle \langle i| \langle j|) \otimes U^{ij} + \Big(I - \sum_{i,j:A^{ij}\neq 0}(|i\rangle |j\rangle \langle i| \langle j|)\Big) \otimes I_{s+a}$. Let $\hat{\alpha} = \max_{i,j} \alpha_{ij}$. Then one can implement an $(\hat{\alpha}\sqrt{d_r d_c}, t+a+3, 2\sqrt{d_r d_c}\varepsilon)$-block-encoding of A, with one use of each of $\mathcal{O}_r, \mathcal{O}_c$, and W, two uses of $\mathcal{O}_\alpha$, $O(t+\mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ additional one- and two-qubit gates, and $O(b, \mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ added ancilla qubits.

Remark 3.6. *Lemma 3.5 is efficient when the matrix is sparse over blocks (where each block may be dense), which is not in general the same as the matrix being sparse. It generalizes Lemma 48 of [3] (also see Lemma B.5) for block-encoding matrices with oracle access to entries. In addition, if α_{ij} are the same for all blocks, we do not need the oracle $\mathcal{O}_\alpha$, hence some ancilla qubits and gates can be saved (see proof in Appendix B.4).*

4 Block-encoding of kernel matrices via hierarchical splitting

In this section, we apply the block-encoding techniques delineated in Section 3 to block-encode $\mathcal{H}$-matrices. Our goal is to obtain good block-encodings (Definition 3.1) of these matrices where the ratio between the normalization factor and the operator norm is at most $O(\mathrm{polylog}(N))$. We provide an optimal block-encoding procedure for kernel matrices arising from a variety of polynomially decaying and exponentially decaying kernels. We show that using only the hierarchical splitting *without subsequent low-rank approximation* gives an optimal block-encoding for these kernel matrices, enabling matrix transformations (*e.g.,* multiplication, inversion, polynomial transformations) with optimal query and gate complexities. Then, we provide a block-encoding procedure for general $\mathcal{H}$-matrices whose admissible blocks are approximated by low-rank matrices and discuss the applicability of our procedure on other variants of hierarchical splitting.

4.1 Polynomially decaying kernels

We use the hierarchical splitting described in Section 2 to implement an optimal block-encoding procedure for polynomially decaying kernels, which take the following form

$$k(x, x') = \begin{cases} C & \text{if } x = x' \\ \|x - x'\|^{-p} & \text{otherwise} \end{cases}, \tag{13}$$

where $p > 0$ and $|C| \leq 1$ represents self-interaction terms (*e.g.,* $C = 0$ in Coulomb point source interactions). For concreteness, we consider the problem of computing pairwise interactions in a system of N particles. Particularly, consider a 1D system of N particles $\{(x_i, m_i)\}_{i=0}^{N-1}$, where x_i and m_i are the location and mass of the particle i, respectively. The potential at x_i is the sum over the pairwise interactions

$$\Phi(x_i) = \sum_{j=0}^{N-1} m_j k(x_i, x_j). \tag{14}$$

For simplicity and ease of analysis, we consider a system of equally distanced particles in which $x_j = j$ and $m_j = 1$ for any $j \in [N]$, where $N = 2^L$. Note that we choose the domain to be $[1, N]$ instead of $[0, 1]$ as considered in Section 2, so that the maximum entry in the kernel matrix is 1. Had the particles been placed in the region $[0, 1]$, we could simply downscale the matrix by dividing it by N^p so that the entries are bounded by 1. Recall from Section 3 that doing so does not change the block-encoding and algorithmic runtime as these depend on the ratio between the operator norm and the normalization factor. This ratio is unchanged under global rescaling of the matrix because the normalization factor will rescale accordingly in the block-encoding procedure.

To access the kernel function on a quantum computer, we assume the following oracle is given

$$\mathcal{O}_k : |i\rangle\, |j\rangle\, |0\rangle^{\otimes b} \to |i\rangle\, |j\rangle\, |\tilde{k}(x_i, x_j)\rangle\,, \tag{15}$$

where $\tilde{k}(x_i, x_j)$ is the b-qubit description of $k(x_i, x_j)$. In fact, if the function $k(x_i, x_j)$ can be computed via an efficient classical circuit, one can construct a comparably efficient quantum circuit for $\mathcal{O}_k$ [35]. In particular, this is the case when the distribution of particles is known and efficiently computable (e.g. x_j are equally spaced points in the discretization of integral operators).

We optimally block-encode the kernel matrix $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$ up to error ε with a normalization factor of $\Theta(\|\mathbf{K}\|)$ by using only two queries to $\mathcal{O}_k$ and $O(\text{polylog}(N, \frac{1}{\varepsilon}))$ additional resources (i.e., one- and two-qubit gates and ancilla qubits). At a high level, our method proceeds as follows. First, we decompose $\mathbf{K}$ into a hierarchical structure of admissible blocks as described in Section 2

$$\mathbf{K} = \sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}. \tag{16}$$

Note that this hierarchical decomposition can also be performed in higher dimensional systems (see Appendix D.3). Next, for each admissible block (which is a dense matrix) in the level $\mathbf{K}^{(\ell)}$, we block-encode it by using the naive procedure for dense matrices (Lemma 3.2). Then, we use the procedure for block-sparse matrices (Lemma 3.5) to form a block-encoding of $\mathbf{K}^{(\ell)}$. Finally, we sum over hierarchical levels to obtain a block-encoding of $\mathbf{K}$.

Specifically, at level ℓ of the hierarchy, each admissible block has dimension $2^{L-\ell}$. In addition, the minimum pairwise distance between particles in an admissible block in $\mathbf{K}^{(\ell)}$ is $d_{\text{min}}^{(\ell)} = 2^{L-\ell}$, so the maximum entry of an admissible block $\mathbf{K}^{\sigma,\rho}$ in $\mathbf{K}^{(\ell)}$ is bounded by $k_{\text{max}}^{(\ell)} = d_{\text{min}}^{-p} = 2^{-(L-\ell)p}$. It follows that an admissible block at level ℓ can be $(\alpha_\ell, L - \ell + 1, \frac{\varepsilon}{3})$-block-encoded via the naive procedure for dense matrices (Lemma 3.2), where $\alpha_\ell = 2^{(L-\ell)(1-p)}$, using two queries to $\mathcal{O}_k$ and $O(\text{polylog}(\frac{N}{\varepsilon}))$ additional one- and two-qubit gates. Note that in using Lemma 3.2 to block-encode a block $\mathbf{K}^{\sigma,\rho}$ at level ℓ, one actually needs to query $k(x_i, x_j)/k_{\text{max}}^{(\ell)}$ rather than $k(x_i, x_j)$ itself. This is readily achieved from the oracle $\mathcal{O}_k$ by conditioning on a register that indexes the level ℓ and the block indices σ, ρ (see Appendix C.1 for details). From here, each $\mathbf{K}^{(\ell)}$, which is a block-sparse matrix of block-sparsity 3, can be $(3\alpha_\ell, L+3, \varepsilon)$-block-encoded via Lemma 3.5 (particularly Remark 3.6). Observe that while $\mathbf{K}^{(\ell)}$ is a $2^L \times 2^L$ matrix, the normalization factor of this block-encoding is only $3\alpha_\ell$, which is better than the normalization factor of N that one would get when naively applying Lemma 3.2 to block-encode $\mathbf{K}^{(\ell)}$. In addition, we still only need two queries to $\mathcal{O}_k$ and $O(\text{polylog}(\frac{N}{\varepsilon}))$ additional gates because all blocks $\mathbf{K}^{\sigma,\rho}$ are constructed in parallel in Lemma 3.5 (see detailed analysis in Appendix C.1).

The adjacent interaction part $\mathbf{K}_{ad}$, which is a tridiagonal matrix, can be $(3, L+3, \varepsilon)$-block-encoded with the procedure for sparse matrices (Lemma B.5), using two queries to $\mathcal{O}_k$ and $O(\text{polylog}(\frac{N}{\varepsilon}))$ one- and two-qubit gates. Finally, we use the procedure for linearly combining block-encoded matrices (Lemma B.3) to obtain a block-encoding of $\mathbf{K} = \sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}$ as $U = \sum_{\ell=2}^{L} 3\alpha_\ell U^{(\ell)} + 3U_{ad}$. This step requires a $(\log L)$-qubit state preparation pair (Definition 3.3) which prepares the coefficients in the linear combination. We neglect the error in implementing this state preparation pair since it can be performed with $\log L = \log\log N$ qubit operations (see Appendix C.1). This entire procedure results in a $(\alpha, L+\log L+3, \alpha\varepsilon)$-block-encoding of the *exact* kernel matrix $\mathbf{K}$, where for $L = \log N$,

$$\begin{aligned}\alpha &= 3 + 3\sum_{\ell=2}^{L} 2^{(L-\ell)(1-p)} \\ &= \begin{cases} 3(1 + \frac{2^{1-p}-N^{1-p}}{2^{1-p}-4^{1-p}}) \leq \Theta(1) & \text{if } p > 1 \\ 3\log N & \text{if } p = 1 \\ 3(1 + \frac{N^{1-p}-2^{1-p}}{4^{1-p}-2^{1-p}}) & \text{if } p < 1 \end{cases}.\end{aligned} \tag{17}$$

As before, the linear combination step does not significantly increase the circuit complexity since all levels can be constructed in parallel with only two queries to the oracle $\mathcal{O}_k$. See Appendix C.1 for more detailed analysis of the entire block-encoding procedure.

Remark 4.1 (Optimality)**.** *The block-encoding of the polynomially decaying kernels in Equation 13 is optimal in the normalization factor α. Indeed, a lowerbound on the operator norm of $\mathbf{K}$ can be obtained by evaluating the norm of the vector $\mathbf{K}\,|\mathbf{1}\rangle$, where $|\mathbf{1}\rangle$ is the normalized all-ones state. Observe that, $\|\mathbf{K}\| \geq \|\mathbf{K}\,|\mathbf{1}\rangle\| \geq \int_1^N x^{-p}dx$. Thus, for $p \neq 1$, $\|\mathbf{K}\| \geq \frac{N^{1-p}-1}{1-p}$ and for $p = 1$, $\|\mathbf{K}\| \geq \ln N$. Clearly, the existence of our block-encoding procedure also implies that these (up to a constant factor) are an upper bound for $\|\mathbf{K}\|$.*

Remark 4.2 (Generalized polynomially decaying kernel)**.** *Since the above analysis only relies on the maximum entry in admissible blocks, it still holds for the case of non-uniform (bounded) masses m_i or more general kernels of the form $k(x, x') = |x - x'|^{-p}G(x, x')$, where $G(x, x')$ is a bounded function such as $G(x, x') = \frac{1}{\sqrt{1+\epsilon|x-x'|^2}}$, $G(x, x') = e^{-i|x-x'|}$, etc. In these cases, we simply scale the kernel matrix such that the maximum entry is 1 (if needed) and apply the above analysis. This generalized class includes a variety of kernels, ranging from power-law interactions to common Green's functions [36].*

The above block-encoding procedure similarly applies for higher-dimensional settings where coordinates x_j are in a d-dimensional space with d constant. In Appendix D.3, we show that there are still only $\log N$ hierarchical levels in this higher dimensional setting and that each level $\mathbf{K}^{(\ell)}$ is still block-sparse. The two key features of the hierarchical splitting that allow for optimal block-encodings are maintained, and the normalization factor is still $\alpha = \Theta(\|\mathbf{K}\|)$. The above optimal block-encoding construction and remarks are summarized in the following lemma.

Lemma 4.3 (Optimal block-encoding of polynomially decaying kernel matrix using $\mathcal{H}$-matrix)**.** *Let $k(x,x') = \|x - x'\|^{-p} G(x,x')$, where $p > 0$ and $G(x,x')$ is a bounded function, i.e. $|G(x,x')| < C$. Assume there exists an efficient quantum circuit $\mathcal{O}_k$ that performs $\mathcal{O}_k : |i\rangle |j\rangle |z\rangle \to |i\rangle |j\rangle |z \oplus \tilde{k}(x_i,x_j)\rangle$, where $\tilde{k}(x_i,x_j)$ is a qubit string description of $k(x_i,x_j)$. Then the matrix $\mathbf{K} = (k(x_i,x_j))_{i,j=0}^{N-1}$ can be $(\alpha, \operatorname{polylog} N, \varepsilon)$-block-encoded with $\alpha = \Theta(\|\mathbf{K}\|)$ using the hierarchical matrix splitting. That is, the block-encoding is optimal according to Definition 3.1. This block-encoding can be constructed with two queries to $\mathcal{O}_k$ and $O(\operatorname{polylog} \frac{N}{\varepsilon})$ additional one- and two-qubit gates.*

Our block-encoding procedure is optimal for common kernel matrices, which are neither sparse nor low-rank, thus expanding the scope of application of quantum computers in the block-encoding framework. In Section 5, for example, we discuss applications of this block-encoding in N-body force computations and solving quantum linear systems.

Remark 4.4. *When sources are not close to uniformly distributed, one must first apply classical adaptive methods [27, 28] which discretize the domain to form a finer mesh, such that the number of nonzero-mass sources in each mesh site is $O(1)$. The dimension of the kernel matrix in this method depends on the inverse of the smallest pairwise distances between the actual sources. Therefore, the runtime of the block-encoding remains polylogarithmic in the number of particles N so long as the minimum pairwise distance is not exponentially small (see Appendix C.3).*

4.2 Exponentially decaying kernels

The procedure used in the previous section readily generalizes to exponentially decaying kernels. Consider, for example, the following kernel

$$k(x,x') = e^{-\|x-x'\|^q} P(\|x - x'\|), \tag{18}$$

where $q > 0$ and $P(\|x - x'\|)$ is a function growing polynomially or slower. Specifically, $|P(y)| \leq O(y^k)$. Typical kernels of this form include the Laplace and Gaussian kernels, which are commonly used in machine learning [15].

As before, we first hierarchically decompose the matrix into admissible blocks. For each admissible block $\mathbf{K}^{\sigma,\rho}$ at level ℓ, the minimum pairwise distance between particles is $d_{\min} = 2^{L-\ell}$ and therefore the maximum entry in the block can be bounded as

$$\begin{aligned} \max_{(i,j)\in(\sigma,\rho)} |k(x_i,x_j)| &\leq \max_{(i,j)\in(\sigma,\rho)} \frac{|x_i - x_j|^k}{\sum_{t=0}^{\infty} |x_i - x_j|^{qt}/t!} \\ &\leq \max_{(i,j)\in(\sigma,\rho)} \frac{t!}{|x_i - x_j|^{qt-k}} \\ &\leq t! 2^{-(L-\ell)(qt-k)}, \end{aligned} \tag{19}$$

where we choose an integer t that satisfies $qt - k > 1$.

Thus, an admissible block $\mathbf{K}^{\sigma,\rho}$ at level ℓ can be $(\alpha_\ell, L - \ell + 3, \varepsilon/3)$-block-encoded via the naive procedure for dense matrices (Lemma 3.2), where $\alpha_\ell = t! 2^{(L-\ell)(1-(qt-k))}$. From here, an efficient block-encoding of $\mathbf{K}$ can be obtained via the same procedure described in the previous section, where p is now replaced by $qt - k > 1$. It can be verified that the normalization factor of this block-encoding is $\Theta(1)$, which is optimal since the maximum entry of $\mathbf{K}$ is of magnitude $\Theta(1)$.

We note, however, that one can make use of a sparsification approach, in which the $\mathbf{K}$ is approximated as a sparse band matrix, and apply the block-encoding procedure for a sparse matrix (Lemma B.5). This approach yields a block-encoding with normalization factor logarithmic in the sparsification error (see Section 5.3).

4.3 Block-encoding of general $\mathcal{H}$-matrices

In the setting most consistent with that of classical $\mathcal{H}$-matrix literature, one may be provided oracle access to the low-rank vectors in $\mathbf{\Psi}^{\sigma,\rho}$ and $\mathbf{\Phi}^{\rho}$ which approximate an admissible block $\widetilde{\mathbf{K}}^{\sigma,\rho} = \mathbf{\Psi}^{\sigma,\rho}\mathbf{D}\left(\mathbf{\Phi}^{\rho}\right)^{\dagger}$ (as seen in Equation 7). Such a situation may also arise when one wants to apply an arbitrary $\mathcal{H}$-matrix which has no connection to any kernel function, *e.g.,* see [37]. In this case, we can use the block-encoding procedure for low-rank matrices (Lemma 3.4) to block-encode $\widetilde{\mathbf{K}}^{\sigma,\rho}$.

After block-encoding the admissible blocks in each level of the hierarchical splitting, we follow the same procedure in previous sections to obtain the entire kernel matrix. Namely we apply the procedure for block-sparse matrices (Lemma 3.5) to construct a block-encoding of $\mathbf{K}^{(\ell)}$ for each level ℓ of the hierarchy; then we sum over all levels using the procedure for linear combination of block-encoded matrices (Lemma B.3) to obtain a block-encoding of $\mathbf{K}$. The normalization factor of the block-encoding of $\mathbf{K}$ is bounded as

$$\alpha = O\left(\sum_{\ell=2}^{\log N} \alpha_\ell\right), \tag{20}$$

where α_ℓ is normalization factor of the admissible blocks $\mathbf{K}^{\sigma,\rho}$ at level ℓ (for ease of analysis we assume admissible blocks at the same level have the same block-encoding normalization factor). As before, assuming the maximum entry of $\mathbf{K}$ is 1, the naive block-encoding (Lemma 3.2) yields a block-encoding with a normalization factor of N, the dimension of the matrix. Whereas, $\mathcal{H}$-matrix block-encoding can yield an exponentially better normalization factor over the naive approach when α_ℓ are $O(\mathrm{polylog}(N))$. This is *not* the case for some kernels studied in classical $\mathcal{H}$-matrix literature [13, 25] such as the log kernel $k(x,x') = \log(|x-x'|)$ and the multiquadric kernel $k(x,x') = \sqrt{c+|x-x'|^2}$. In fact, we show in Appendix C.6 that these particular kernels and most polynomially *growing* kernels can already be optimally block-encoded using the naive approach of Lemma 3.2. The naive approach works intuitively because these kernels are numerically low-rank and their top singular vectors are nearly uniform.

4.4 Variants of hierarchical splitting

The form of the hierarchical splitting can be slightly modified for other types of decaying kernels. For example, consider kernels of the form $k(x,x') = \frac{1}{|(x-x')+c|^p}$ on the domain $\Omega = [0, N-1)$ and $-N < c < N$ is an integer. In this case, we can use a "shifted" hierarchical splitting to obtain an optimal block-encoding of the kernel matrix $\mathbf{K} = (|i-j+c|^{-p})_{i,j=0}^{N-1}$, in which the hierarchy is shifted in the horizontal direction (along the rows). Similarly, for kernels of the form $k(x,x') = \frac{1}{(|x-x'|-c)^p}$ $(0 \le c < N)$, we can apply a shifted hierarchical splitting along the skew-diagonal direction. We illustrate this approach in Appendix C.2.

In Appendix C.4, we also describe a more general block-encoding that exploits the structure of entry magnitudes in a given matrix even if the entries of the admissible blocks are not neighboring (in fact, Definition 2.1 does not require the entries of an admissible block to be contiguous). At a high level, we use a pointer oracle which, conditioned on the level ℓ, points to the entries whose magnitudes are in the range $[2^{-(\ell+1)}, 2^{-\ell})$. When this oracle can be efficiently implemented, this block-encoding procedure might find applications in other classes of dense matrices.

Proposition 4.5 (Block-encoding using generalized hierarchical splitting)**.** *Let $A \in \mathbb{R}^{N\times N}$ be a Hermitian matrix with non-negative entries which are provided via an oracle $\mathcal{O}_A$: $|i\rangle|j\rangle|z\rangle \to |i\rangle|j\rangle|z \oplus \tilde{a}_{ij}\rangle$, where $\tilde{a}_{ij}$ is an exact bit string description of a_{ij}. Suppose*

$N = 2^L$ and $2^{-L} \le a_{ij} \le 1$. For any $0 \le \ell < L = \log N$ and a column j, define its level-ℓ row-index subset to be $I_\ell(j) = \{i | 2^{-(\ell+1)} < a_{ij} \le 2^{-\ell}\}$ and let $n_\ell(j) = |I_\ell(j)|$. Suppose there exist n_ℓ and γ such that $\gamma n_\ell \le n_\ell(j) \le n_\ell$ for any j. Furthermore, suppose there exists an invertible *function $f_j(\ell, k)$ which returns the row index of the k-th element (according to some fixed ordering, e.g. from top to bottom in the column) in $I_\ell(j)$. Then, one can construct a block-encoding of A with a normalization factor of at most $\frac{2\log N}{\gamma}\|A\|$ using two queries to $\mathcal{O}_A$, $O(\mathrm{polylog}(\frac{N}{\varepsilon}))$ elementary gates, and $O(\mathrm{polylog}(\frac{N}{\varepsilon}))$ ancilla qubits.*

Intuitively, γ measures how balanced the magnitudes of the matrix entries are across rows and columns. If $\frac{1}{\gamma} = \mathrm{polylog}(N)$, this yields a good block-encoding according to Definition 3.1. For instance, $\frac{1}{\gamma}$ is a constant for kernel matrices. In the proof of Proposition 4.5, we also present an efficient state preparation procedure using methods of $\mathcal{H}$-matrix splitting in Appendix C.4, which could be of independent interest.

5 Applications

In this section, we discuss applications of our quantum hierarchical block-encoding method in two commonly studied settings: (i) gravitational potential calculation [23] and (ii) solving integral equations [13]. The first problem requires performing matrix-vector multiplication on a quantum computer, while the second problem is equivalent to solving a quantum linear system. We also compare our procedure to the sparsification approach and address potential issues with large condition numbers.

We first state the following lemmas, which allow one to apply a block-encoded matrix or its inverse to a quantum state. Other polynomial transformations of block-encoded matrices can be performed by the quantum singular value transform (Theorem 17 of [3]).

Lemma 5.1 (Applying block-encoded matrices)**.** *If U is an (α, a, ε)-block-encoding of the matrix $A \in \mathbb{C}^{2^s \times 2^s}$ and $U_{\mathbf{x}}$ is a unitary that prepares the s-qubit quantum state $|\mathbf{x}\rangle$, then it takes $O\left(\frac{\alpha}{\|A\|}\kappa\right)$ queries to U and $U_{\mathbf{x}}$ to obtain a quantum state $|\mathbf{y}\rangle$ such that $\| |\mathbf{y}\rangle - |A\mathbf{x}\rangle \| \le \varepsilon$, where κ is the condition number of A.*

Proof. Applying the operator $U(I_a \otimes U_{\mathbf{x}})$ on the state $|0^a\rangle \otimes |0^s\rangle$ we obtain

$$|0^a\rangle \frac{A}{\alpha} |\mathbf{x}\rangle + |\mathrm{garbage}\rangle, \tag{21}$$

where $|\mathrm{garbage}\rangle$ is orthogonal to the subspace $|0^a\rangle$. By post-selecting the subspace $|0^a\rangle$, we obtain the desired quantum state $|A\mathbf{x}\rangle$. This step succeeds with a probability of $\left\|\frac{A|\mathbf{x}\rangle}{\alpha}\right\|^2 = \Omega((\frac{\|A\|}{\alpha\kappa})^2)$. Using amplitude amplification [38], this can be improved to $\Omega(\frac{\|A\|}{\alpha\kappa})$, giving us the specified query complexity. □

Lemma 5.2 (Applying inverse block-encoded matrices, adapted from [22])**.** *If U is an (α, a, ε)-block-encoding of an invertible matrix $A \in \mathbb{C}^{2^s \times 2^s}$ and $U_{\mathbf{b}}$ is a unitary that prepares the s-qubit quantum state $|\mathbf{b}\rangle$, then it takes $O\left(\frac{\alpha\kappa}{\|A\|}\log(1/\varepsilon)\right)$ queries to U and $U_{\mathbf{b}}$ to obtain a quantum state $|\mathbf{y}\rangle$ such that $\| |\mathbf{y}\rangle - |A^{-1}\mathbf{b}\rangle \| < \varepsilon$.*

Proof. We apply the main theorem of [22], which assumed $\alpha = \|A\| = 1$ to achieve a query complexity of $O(\kappa \log(1/\varepsilon))$. In general, κ needs to be replaced by the inverse of the minimum singular value of the block-encoded part. In our case, this singular value is $\frac{\|A\|}{\alpha\kappa}$. □

As noted in Section 3, the query complexities of the above procedures, and more generally quantum algorithms in the block-encoding framework [3], only depend on the ratio $\alpha/\|A\|$. Hence, a block-encoding is optimal (Definition 3.1) when $\alpha = \Theta(\|A\|)$. In the following, we apply such optimal block-encodings in Section 4 to two specific applications of kernel matrices.

5.1 Gravitational potential calculation: Quantum fast multipole method

We apply the block-encoding obtained in Section 4.1 to solve the gravitational potential problem [23] on a quantum computer. Since we use a quantum version of the hierarchical splitting, this is a quantum analogue to the classical fast multipole method [39]. Consider a 1D system of N particles which are approximately uniformly distributed in the domain $[0, O(N)]$, where the mass and location of particle j are m_j and x_j, respectively. Furthermore, assume that m_j are $\Theta(1)$ and the smallest pairwise distance between the particles is $\Omega(1)$. We would like to compute the accumulated potential at every particle i, stored in a vector Φ with entries $\Phi_i = \sum_{j\neq i} m_j |x_i - x_j|^{-1}$.

Proposition 5.3 (Quantum fast multipole method)**.** *Given an oracle $\mathcal{O}_k$ that performs $\mathcal{O}_k : |i\rangle\,|j\rangle\,|z\rangle \to |i\rangle\,|j\rangle\,|z \oplus \tilde{k}_{ij}\rangle$, where $\tilde{k}_{ij}$ is a b-qubit string description of $k(x_i, x_j)$, and a procedure P_m that prepares the quantum state $|\mathbf{m}\rangle$ which stores the normalized masses of the particles, we can obtain a quantum state $|\phi\rangle$ that ε-approximates the normalized vector $\Phi/\|\Phi\|$, whose j-th entry is the accumulated potential at particle j, using $O(1)$ queries to P_m, $O\left(\operatorname{polylog}\frac{N}{\varepsilon}\right)$ one- and two-qubit gates, and $O\left(b, \operatorname{polylog}\frac{N}{\varepsilon}\right)$ additional qubits.*

Proof. We first use P_m to create the state $|\mathbf{m}\rangle = \sum_{j=0}^{N-1} \frac{m_j}{\|\mathbf{m}\|}\,|j\rangle$ (assume $\log N$ is an integer). Let $\mathbf{K} = (k(x_i, x_j))_{i,j}$, observe that $|\Phi\rangle = \mathbf{K}\,|\mathbf{m}\rangle = \frac{\Phi}{\|\mathbf{m}\|}$. Using Lemma 4.3, we can construct a unitary U, which is an (α, a, ε)-block-encoding of $\mathbf{K}$, with the specified gate complexity and wherein $\alpha = \Theta(\log N)$ and $a = O(\log N)$. Then, we apply Lemma 5.1, with a more careful analysis on the post-selection step. Here, the probability of succesfully post-selecting the subspace $|0^a\rangle$ is $\|\frac{\Phi}{\alpha\|\mathbf{m}\|}\|^2 = \Theta(1)$ since $\|\mathbf{m}\| = \Theta(\sqrt{N})$ and $\|\Phi\| = \Omega(\sqrt{N}\log N)$ (by noting that the particle masses are $\Theta(1)$ and using similar calculations to those in Remark 4.1). This gives us the specified query complexity. $\square$

We remark that the (non-unitary) state preparation procedure P_m can be efficiently implemented when $m_j = \Theta(1)$ (see Theorem 1 of [40]). In addition, if particles are not (nearly) uniformly distributed, we can discretize the domain to form a finer mesh as described in Remark 4.4. Here we have assumed these distances are $\Omega(1)$, hence the size of the mesh (and the size of the kernel matrix) is $O(N)$. This method also enables computing the potential at locations other than the sources themselves. This proposition also applies to N-body calculations in higher-dimensional systems as the hierarchical splitting and Lemma 4.3 can be generalized to these cases (see Appendix D.3).

5.2 Integral equation: Quantum algorithm for linear systems of kernel matrix

We provide an application of our hierarchical block-encoding procedure in solving a quantum linear systems of kernel matrices. These problems arise in many research areas, such as Gaussian processes [15] and integral equations [14]. Consider the following integral equation over the circular strip of unit radius and height and width $\lambda \ll 1$ as shown in Figure 2.

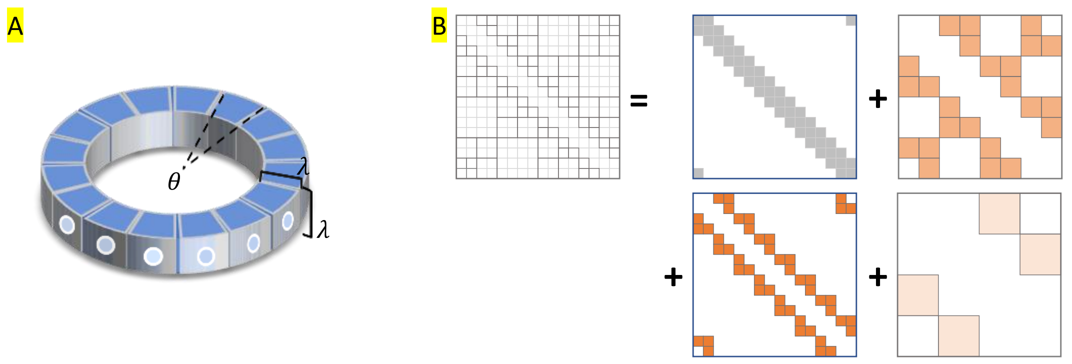

Figure 2: (A) A thin circular strip of unit radius and height $\lambda \ll 1$. When using the collocation method to solve an integral equation on its surface, one divides the strip into N "panels" such that $\lambda = O(1/N)$. (B) The "cyclic" hierarchical splitting of the kernel matrix that arises when applying the collocation method to solve for an integral equation defined over the strip (Equation 22).

$$g(\mathbf{x}) = f(\mathbf{x}) + \int_{\Omega} k(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') d\mathbf{x}', \tag{22}$$

where $k(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^{-p}$ with $0 < p \leq 2$, $g(\mathbf{x})$ is a known smooth function, and $f(\mathbf{x})$ is the unknown function we would like to solve. Equations of this form are commonly found in, *e.g.*,, Dirichlet problems [14], where $k(\mathbf{x}, \mathbf{x}')$ is the Coulomb potential, $g(\mathbf{x})$ is the given potential on the strip, and $f(\mathbf{x})$ is the unknown charge density. Here, we use the collocation method to numerically solve this integral equation which represents the unknown function in a basis $f(\mathbf{x}) = \sum_{i=1}^{N} f_i \varphi_i(x)$ and solves for the coefficients f_i such that the integral equation is satisfied at specified locations. Particularly, we use the piecewise-constant function as the basis $\varphi_i(\theta) = 1$ if $\theta \in [2\pi i/N, 2\pi(i+1)/N)$ and 0 otherwise. We refer to the segment $[2\pi i/N, 2\pi(i+1)/N)$ as *panel* i. Then, the integral on the RHS of Equation 22 can be rewritten as

$$\sum_{i=1}^{N} f_i \int_{\text{panel } i} \frac{1}{\|\mathbf{x} - \mathbf{x}'\|^p} d\mathbf{x}', \tag{23}$$

In the collocation method, we solve for the integral equation at the centroids of the panels located at $c_i = 2\pi(i + 0.5)/N$. In other words, we solve the following linear system

$$\mathbf{g} = (I + \mathbf{K})\mathbf{f}, \tag{24}$$

where bold text indicates discretized values, *e.g.*,, $\mathbf{g}_i = g(c_i)$ and $\mathbf{K}_{ij} = \int_{\text{panel } j} \|c_i - \mathbf{x}'\|^{-p} d\mathbf{x}'$.

Observe that this collocation method can handle the singularity when $i = j$ as $\mathbf{K}_{ii} = O(\lambda^{3-p})$. Furthermore, when $|i-j|$ is large, $\mathbf{K}_{ij} \approx \text{distance}^{-p} \cdot \text{panel volume} \approx \frac{\lambda^2}{N(2\sin(\pi|i-j|/N))^p}$. For panels i, j close to each other ($e.g.$, $|i-j| < O(1)$), simple adaptive methods can be used to approximate the integral. For instance, one can subdivide the panel j into eight smaller panels and approximate the integral as the sum of these eight sub-panels; we neglect the details and refer the interested reader to [41]. In summary, the entries of $\mathbf{K}$ are simple to calculate and they approximately are

$$\mathbf{K}_{ij} \approx \begin{cases} \frac{\lambda^2}{N(2\sin(\pi|i-j|/N))^p} & \text{for } i \neq j \\ O(\lambda^{3-p}) & \text{for } i = j \end{cases}. \tag{25}$$

The error bound and the uniqueness of the solution in the collocation method are well-studied in classical literature, we refer to [14, 41, 42] for more details.

We now proceed to block-encode this kernel matrix. Observe that $\frac{1}{N(2\sin(\pi|i-j|/N))^p} \le \frac{1}{(\pi|i-j|)^p}$ for any panels i, j such that $|i-j| \le N/2$, hence we can apply a "cyclic" version of the hierarchical splitting in Section 2 to *optimally* block-encode $\mathbf{K}$ (see Figure 2B). As analyzed in Section 4.1, this block-encoding only has a circuit depth of $O(\operatorname{polylog}\frac{N}{\varepsilon})$ and a normalization factor of $\alpha = \Theta(\|\mathbf{K}\|)$, as the operator norm can be lowerbounded using the same technique as in Remark 4.1 by noticing that $\frac{1}{N(2\sin(\pi|i-j|/N))^p} \ge \frac{1}{(2\pi|i-j|)^p}$ for any $|i-j| \le N/2$.

Finally, it is straightforward to use Lemma B.3 (linear combination of block-encoded matrices) to obtain an optimal block-encoding of $I+\mathbf{K}$. Then, we directly apply Lemma 5.2 to solve the linear system in Equation 24. Since the block-encoding is optimal, the query complexity is $O(\kappa\log(1/\varepsilon))$, for a total runtime of $O(\kappa\operatorname{polylog}(N/\varepsilon))$ when including the circuit complexity of the block-encoding (Lemma 4.3).

Proposition 5.4 (Solving discretized integral equations)**.** *Consider the integral equation on a ring as described in Equation 22. We first cast the equation into the $N \times N$ linear system in Equation 24 using the collocation method with N discretized points. Then, we can obtain its numerical solution as a quantum state in time $O(\kappa\operatorname{polylog}(N/\varepsilon))$, where κ is the condition number of the matrix $I+\mathbf{K}$ in Equation 24 and ε is the error bound of the quantum operation.*

We address two practical aspects when applying Proposition 5.4. First, we must prepare $\mathbf{g}$ as a quantum state. In Appendix E, we present an efficient procedure for preparing $|\mathbf{g}\rangle$ in the case where $\mathbf{g}$ is sampled from a smooth function. At a high level, our procedure prepares the quantum state in the Fourier regime, which is approximately sparse due to the smoothness of g; then we apply the quantum Fourier transform to obtain the desired real-regime state. To prove Proposition 4.5, we also present another state preparation procedure using ideas from $\mathcal{H}$-matrix splitting in Appendix C.4 which could be of independent interest. Second, the query complexity of the block-encoding depends on the condition number κ of the matrix $I+\mathbf{K}$, which can be bounded as follows

$$\kappa \le \frac{1+\|\mathbf{K}\|}{1-\|\mathbf{K}\|}. \tag{26}$$

Note that here, $\|\mathbf{K}\|$ scales with λ. Since we require $\lambda \ll 1$ for the collocation method to work, the operator norm $\|\mathbf{K}\|$, and thus κ, is bounded. Therefore, the runtime of Proposition 5.4 is $O(\operatorname{polylog}\frac{N}{\varepsilon})$.

5.3 Notes on sparsification approach

Since the entries of kernel matrices often decay along a row or column, one may assume that they can be approximated by only considering a sparse set of matrix entries and applying the naive block-encoding method for sparse matrices (Lemma B.5). Here, we analyze this approach showing that it is only favorable when kernel entries decay exponentially and not polynomially. In fact, the runtime of this method is exponentially worse in the target error bound ε compared to that of our approach. Consider the polynomially decaying kernel in Equation 13. One could attempt to approximate this kernel matrix by a band matrix $\mathbf{K}_b$ of sparsity d. Defining the error to be $\|\mathbf{K}-\mathbf{K}_b\|$, it can be seen that the error is of magnitude $\int_d^N x^{-p}dx$. Since this integral diverges when $p \le 1$, one can only sparsify the matrix when $p > 1$. In this case, to achieve an error bound of ε_s, the sparsity d needs to be $\Omega(\varepsilon_s^{\frac{1}{1-p}})$. Then, one can apply Lemma B.5 (block-encoding of sparse matrices) to directly block-encode the sparsified matrix with a normalization factor of $\Theta(\varepsilon_s^{\frac{1}{1-p}})$ using $O(\log N + \text{polylog}(\frac{1}{\varepsilon}))$ extra gates and $O(1)$ oracle queries. This results in an exponentially worse runtime in terms of the error bound compared to our method, which has a normalization factor of $O(1)$.

For the exponential kernels, however, the sparsification method requires keeping $\Theta(\log(1/\varepsilon_s))$ or more entries per row or column. With Lemma B.5, the sparsified block-encoding has a normalization factor of $\Theta(\log(1/\varepsilon_s))$ using the same gate and query complexities as above. Thus, exponential kernels may admit a sparsification approach with an extra factor of $\Theta(\log(1/\varepsilon))$ in the runtime.

5.4 Notes on condition number

The condition number κ and its relation to the dimension N of the matrix play a central role in the runtime of our algorithms. Numerical experiments shown in Appendix C.5 indicate that for purely polynomially decaying kernels without any regularization and ignoring self-interacting terms (diagonal entries set to zero), the condition number grows polynomially with N, *i.e.,* the minimum singular value decays with N. In practice, however, the input vector often lives in the well-conditioned subspace where these smaller singular values can be ignored as observed in the gravitation example earlier. For force calculation problems, this is the case when particles have comparable masses/charges. For numerical integration or integral equations, this is equivalently the case when the input functions change slowly.

We note that Lemma 5.2 for matrix inversion uses the discrete adiabatic theorem, whose runtime depends directly on the condition number κ of the matrix operation. Hence, in the case where the matrix operation A is ill-conditioned but the input vector lives in the well-conditioned subspace, we can use techniques that allow for filtering out small eigenvalues before the inversion step such as the QSVT [3]. The query complexity of the block-encoding in matrix inversion in this case is $O(\frac{\log(1/\zeta, 1/\varepsilon)}{\zeta \| A^{-1}|b\rangle \|})$, where ζ is a chosen threshold bounding the singular values in the well-conditioned subspace.

Fortunately, for applications in numerical analysis or machine learning, the kernel matrix is often regularized by an added matrix, *e.g.,* the addition of an identity matrix as in Fredholm integral equations of the second kind or the noise matrices $\sigma_n^2 I$ in Gaussian processes and kernel ridge regression [14, 15]. In the integral equation setting, this regularization is equivalent to scaling the magnitude of self-interacting terms (diagonal entries). Furthermore, if the input vector is potentially in the ill-conditioned subspace, preconditioning methods in [43] may be used to improve the condition number.

6 Conclusion

Our results show that factoring certain kernel matrices into hierarchical or $\mathcal{H}$-matrices leads to efficient algorithms for implementing such kernel matrices on a quantum computer, thus expanding the scope of application of quantum computers to a class of matrices which are neither low-rank nor sparse. In fact, our approach to implementing such hierarchical matrices, in a sense, blends prior quantum methods for implementing unitary block encodings by hierarchically splitting a matrix into a sparse matrix component close to the diagonal and progressively larger blocks farther away from the diagonals. Looking forward, it is an interesting question whether any extensions to the $\mathcal{H}$-matrix framework can lead to further improvements in the quantum setting. Classically, the most significant extension of the $\mathcal{H}$-matrix framework is perhaps the development of $\mathcal{H}^2$-matrices, which incorporates a further level of hierarchical nested bases to provide a logarithmic speedup in the matrix dimension for performing matrix operations [24, 44]. Furthermore, recent classical work has parameterized $\mathcal{H}$-matrices to integrate them into neural networks and design learning algorithms with inductive biases that match the form of the hierarchical splitting [37, 45]. Future work can study whether such parameterizations can be performed as well in the quantum setting via variational quantum algorithms or other quantum analogues to classical neural networks [46].

Acknowledgements

We thank the anonymous reviewers for helpful feedback and corrections on this paper. This project was funded by AFOSR, ARO, and DARPA. BTK acknowledges funding from the MIT Energy Initiative fellowship.

References

[1] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Physical Review Letters* 103.15 (2009). ISSN: 1079-7114. DOI: https://doi.org/10.1103/PhysRevLett.103.150502.

[2] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. “Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision”. In: *SIAM Journal on Computing* 46.6 (2017), 1920–1950. ISSN: 1095-7111. DOI: https://doi.org/10.1137/16M1087072.

[3] András Gilyén et al. “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 193–204. DOI: https://doi.org/10.48550/arXiv.1806.01838.

[4] Iordanis Kerenidis and Anupam Prakash. *Quantum Recommendation Systems*. 2016. DOI: https://doi.org/10.48550/arXiv.1603.08675.

[5] Leonard Wossnig, Zhikuan Zhao, and Anupam Prakash. “Quantum Linear System Algorithm for Dense Matrices”. In: *Physical Review Letters* 120.5 (2018). ISSN: 1079-7114. DOI: http://dx.doi.org/10.1103/PhysRevLett.120.050502.

[6] András Gilyén, Zhao Song, and Ewin Tang. “An improved quantum-inspired algorithm for linear regression”. In: *Quantum* 6 (2022), p. 754. DOI: https://doi.org/10.22331/q-2022-06-30-754.

[7] Changpeng Shao and Ashley Montanaro. *Faster quantum-inspired algorithms for solving linear systems*. 2021. DOI: https://doi.org/10.48550/arXiv.2103.10309.

[8] David P. Woodruff. “Sketching as a tool for numerical linear algebra”. In: *Foundations and Trends in Theoretical Computer Science* 10.1–2 (2014), 1–157. ISSN: 1551-3068. DOI: 10.1561/0400000060. URL: `http://dx.doi.org/10.1561/0400000060`.

[9] Lin-Chun Wan et al. “Asymptotic quantum algorithm for the Toeplitz systems”. In: *Physical Review A* 97.6 (2018). ISSN: 2469-9934. DOI: 10.1103/physreva.97.062322. URL: `http://dx.doi.org/10.1103/PhysRevA.97.062322`.

[10] A Mahasinghe and J B Wang. “Efficient quantum circuits for Toeplitz and Hankel matrices”. In: *Journal of Physics A: Mathematical and Theoretical* 49.27 (2016), p. 275301. ISSN: 1751-8121. DOI: 10.1088/1751-8113/49/27/275301. URL: `http://dx.doi.org/10.1088/1751-8113/49/27/275301`.

[11] Grecia Castelazo et al. “Quantum algorithms for group convolution, cross-correlation, and equivariant transformations”. In: *Physical Review A* 106.3 (2022), p. 032402. DOI: 10.1103/PhysRevA.106.032402.

[12] Andrew M. Childs and Wim van Dam. “Quantum algorithms for algebraic problems”. In: *Reviews of Modern Physics* 82.1 (2010), 1–52. ISSN: 1539-0756. DOI: 10.1103/revmodphys.82.1. URL: `http://dx.doi.org/10.1103/RevModPhys.82.1`.

[13] R Beatson and Leslie Greengard. “A short course on fast multipole methods”. English (US). In: *Wavelets, multilevel methods, and elliptic PDEs*. Ed. by M Ainsworth and al. Numerical Mathematics and Scientific Computation. Oxford University Press, 1997, pp. 1–37. URL: `https://nyuscholars.nyu.edu/en/publications/a-short-course-on-fast-multipole-methods`.

[14] Kendall Atkinson and Weimin Han. “Numerical solution of fredholm integral equations of the second kind”. In: *Theoretical Numerical Analysis*. Springer, 2009, pp. 473–549. DOI: https://doi.org/10.1007/978-1-4419-0458-4_12.

[15] Carl Edward Rasmussen. “Gaussian processes in machine learning”. In: *Summer school on machine learning*. Springer. 2003, pp. 63–71. DOI: https://doi.org/10.1007/978-3-540-28650-9_4.

[16] W. Hackbusch. “A Sparse Matrix Arithmetic Based on $\mathcal{H}$-Matrices. Part I: Introduction to $\mathcal{H}$-Matrices”. In: *Computing* 62.2 (1999), pp. 89–108. ISSN: 1436-5057. DOI: 10.1007/s006070050015. URL: `https://doi.org/10.1007/s006070050015`.

[17] W. Hackbusch and B. N. Khoromskij. “A Sparse $\mathcal{H}$-Matrix Arithmetic. Part II: Application to Multi-Dimensional Problems”. In: *Computing* 64.1 (2000), pp. 21–47. ISSN: 1436-5057. DOI: 10.1007/PL00021408. URL: `https://doi.org/10.1007/PL00021408`.

[18] Guang Hao Low and Isaac L. Chuang. “Hamiltonian Simulation by Qubitization”. In: *Quantum* 3 (2019), p. 163. ISSN: 2521-327X. DOI: 10.22331/q-2019-07-12-163. URL: `http://dx.doi.org/10.22331/q-2019-07-12-163`.

[19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. “The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation”. In: *ICALP*. 2019. DOI: https://doi.org/10.4230/LIPIcs.ICALP.2019.33.

[20] Guang Hao Low and Isaac L. Chuang. “Optimal Hamiltonian Simulation by Quantum Signal Processing”. In: *Physical Review Letters* 118.1 (2017). ISSN: 1079-7114. DOI: 10.1103/physrevlett.118.010501. URL: `http://dx.doi.org/10.1103/PhysRevLett.118.010501`.

[21] Joran van Apeldoorn and András Gilyén. “Improvements in Quantum SDP-Solving with Applications”. In: *ICALP*. 2019. DOI: https://doi.org/10.4230/LIPIcs.ICALP.2019.99.

[22] Pedro CS Costa et al. “Optimal Scaling Quantum Linear-Systems Solver via Discrete Adiabatic Theorem”. In: *PRX Quantum* 3.4 (2022), p. 040303. DOI: 10.1103/PRXQuantum.3.040303.

[23] Josh Barnes and Piet Hut. “A hierarchical O(N log N) force-calculation algorithm”. In: *Nature* 324.6096 (1986), pp. 446–449. ISSN: 1476-4687. DOI: 10.1038/324446a0. URL: `https://doi.org/10.1038/324446a0`.

[24] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch. “Introduction to hierarchical matrices with applications”. In: *Engineering Analysis with Boundary Elements* 27.5 (2003), pp. 405–422. ISSN: 0955-7997. DOI: https://doi.org/10.1016/S0955-7997(02)00152-2.

[25] Markus Fenn and Gabriele Steidl. *FMM and H-matrices: A Short Introduction to the Basic Idea*. Tech. rep. TR-2002-008. University of Mannheim, Department for Mathematics and Computer Science, 2002. URL: https://madoc.bib.uni-mannheim.de/744/1/TR-02-008.pdf.

[26] W. Hackbusch, B. Khoromskij, and S. A. Sauter. "On $\mathcal{H}^2$-matrices". In: *Lectures on Applied Mathematics*. Ed. by Hans-Joachim Bungartz, Ronald H. W. Hoppe, and Christoph Zenger. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 9–29. ISBN: 978-3-642-59709-1. DOI: https://doi.org/10.1007/978-3-642-59709-1_2.

[27] J. Carrier, L. Greengard, and V. Rokhlin. "A Fast Adaptive Multipole Algorithm for Particle Simulations". In: *SIAM J. Sci. Stat. Comput.* 9.4 (1988), 669–686. ISSN: 0196-5204. DOI: 10.1137/0909044. URL: https://doi.org/10.1137/0909044.

[28] Jaswinder Pal Singh et al. "A parallel adaptive fast multipole method". In: *Proceedings of the 1993 ACM/IEEE Conference on Supercomputing*. 1993, pp. 54–65. DOI: 10.1145/169627.169651.

[29] E. Tyrtyshnikov. "Mosaic-Skeleton approximations". In: *Calcolo* 33 (June 1996), pp. 47–57. DOI: 10.1007/BF02575706.

[30] Achi Brandt. "Multilevel computations of integral transforms and particle interactions with oscillatory kernels". In: *Computer Physics Communications* 65.1 (1991), pp. 24–38. ISSN: 0010-4655. DOI: https://doi.org/10.1016/0010-4655(91)90151-A. URL: https://www.sciencedirect.com/science/article/pii/001046559190151A.

[31] Gregory Beylkin, Ronald R. Coifman, and Vladimir Rokhlin. "Fast wavelet transforms and numerical algorithms I". In: *Communications on Pure and Applied Mathematics* 44 (1991), pp. 141–183. DOI: https://doi.org/10.1002/cpa.3160440202.

[32] Robin Kothari. "Efficient algorithms in quantum query complexity". PhD thesis. University of Waterloo, 2014. URL: https://uwspace.uwaterloo.ca/handle/10012/8625.

[33] Yihui Quek and Patrick Rebentrost. *Fast algorithm for quantum polar decomposition, pretty-good measurements, and the Procrustes problem*. 2021. DOI: 10.48550/arXiv.2106.07634.

[34] Lov Grover and Terry Rudolph. *Creating superpositions that correspond to efficiently integrable probability distributions*. 2002. DOI: 10.48550/arXiv.quant-ph/0208112.

[35] Michael A. Nielsen and Isaac Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2011. Chap. 3. DOI: 10.1017/CBO9780511976667.

[36] George B. Arfken, Hans J. Weber, and Frank E. Harris. "Chapter 10 - Green's Functions". In: *Mathematical Methods for Physicists (Seventh Edition)*. Ed. by George B. Arfken, Hans J. Weber, and Frank E. Harris. Seventh Edition. Boston: Academic Press, 2013, pp. 447–467. ISBN: 978-0-12-384654-9. DOI: https://doi.org/10.1016/B978-0-12-384654-9.00010-4. URL: https://www.sciencedirect.com/science/article/pii/B9780123846549000104.

[37] Yuwei Fan et al. "A Multiscale Neural Network Based on Hierarchical Matrices". In: *Multiscale Modeling & Simulation* 17.4 (2019), pp. 1189–1213. DOI: 10.1137/18M1203602. eprint: https://doi.org/10.1137/18M1203602. URL: https://doi.org/10.1137/18M1203602.

[38] Gilles Brassard et al. "Quantum amplitude amplification and estimation". In: *Quantum Computation and Information* (2002), 53–74. ISSN: 0271-4132. DOI: 10.1090/conm/305/05215. URL: http://dx.doi.org/10.1090/conm/305/05215.

[39] L. Greengard and V. Rokhlin. "A fast algorithm for particle simulations". In: *Journal of Computational Physics* 73.2 (1987), pp. 325–348. ISSN: 0021-9991. DOI: 10.1016/0021-9991(87)90140-9. URL: `https://www.sciencedirect.com/science/article/pii/0021999187901409`.

[40] Kosuke Mitarai, Masahiro Kitagawa, and Keisuke Fujii. "Quantum analog-digital conversion". In: *Physical Review A* 99.1 (2019), p. 012301. DOI: 10.1103/PhysRevA.99.012301.

[41] P. K. Kythe. *An introduction to boundary element methods (1st ed.)* CRC Press, 1995. ISBN: 9780367449148. URL: `https://www.routledge.com/An-Introduction-to-Boundary-Element-Methods/Kythe/p/book/9780367449148`.

[42] W. Hackbusch. "The Panel Clustering Method for BEM". In: *Discretization Methods in Structural Mechanics*. Ed. by Günther Kuhn and Herbert Mang. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 299–306. ISBN: 978-3-642-49373-7. DOI: 10.1007/978-3-642-49373-7_28.

[43] Yu Tong et al. "Fast inversion, preconditioned quantum linear system solvers, fast Green's-function computation, and fast evaluation of matrix functions". In: *Physical Review A* 104.3 (2021). ISSN: 2469-9934. DOI: 10.1103/physreva.104.032422. URL: `http://dx.doi.org/10.1103/PhysRevA.104.032422`.

[44] Lin Lin, Jianfeng Lu, and Lexing Ying. "Fast construction of hierarchical matrix representation from matrix–vector multiplication". In: *Journal of Computational Physics* 230.10 (2011), pp. 4071–4087. DOI: 10.1016/j.jcp.2011.02.033.

[45] Yuwei Fan et al. "A multiscale neural network based on hierarchical nested bases". In: *Research in the Mathematical Sciences* 6.2 (2019), pp. 1–28. DOI: 10.1007/s40687-019-0183-3.

[46] Marco Cerezo et al. "Variational quantum algorithms". In: *Nature Reviews Physics* 3.9 (2021), pp. 625–644. DOI: 10.1038/s42254-021-00348-9.

[47] Dominic W. Berry et al. "Efficient Quantum Algorithms for Simulating Sparse Hamiltonians". In: *Communications in Mathematical Physics* 270.2 (2007), pp. 359–371. ISSN: 1432-0916. DOI: 10.1007/s00220-006-0150-x. URL: `https://doi.org/10.1007/s00220-006-0150-x`.

[48] Andris Ambainis. *Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations*. 2010. DOI: 10.48550/arXiv.1010.4458.

[49] Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. "Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing". In: *Phys. Rev. Lett.* 122 (6 2019), p. 060504. DOI: 10.1103/PhysRevLett.122.060504. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.122.060504`.

[50] Dong An and Lin Lin. "Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm". In: *ACM Transactions on Quantum Computing* 3.2 (2022), pp. 1–28. DOI: 10.1145/3498331. URL: `https://doi.org/10.1145%2F3498331`.

[51] Lin Lin and Yu Tong. "Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems". In: *Quantum* 4 (2020), p. 361. DOI: 10.22331/q-2020-11-11-361. URL: `https://doi.org/10.22331%2Fq-2020-11-11-361`.

[52] Jonathan M. Borwein and Peter B. Borwein. *Pi and the AGM: A Study in the Analytic Number Theory and Computational Complexity*. USA: Wiley-Interscience, 1987. ISBN: 0471831387. URL: https://www.wiley.com/en-us/Pi+and+the+AGM%3A+A+Study+in+Analytic+Number+Theory+and+Computational+Complexity-p-9780471315155.

[53] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. 1st. USA: Cambridge University Press, 2009. ISBN: 0521424267. URL: https://doi.org/10.1017/CBO9780511804090.

[54] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch. *Lecture notes on Hierarchical Matrices*. Jan. 2003. URL: https://www.mis.mpg.de/publications/other-series/ln/lecturenote-2103.html.

[55] Wajih Halim Boukaram, George Turkiyyah, and David E. Keyes. *Hierarchical Matrix Operations on GPUs: Matrix-Vector Multiplication and Compression*. 2019. DOI: https://doi.org/10.48550/arXiv.1902.01829.

A Prior works on block-encoding and quantum linear systems algorithms

In this section, we provide a brief overview and discuss the runtimes of previous works in Table 1. As a reminder, the quantum linear system problem (QLSP) seeks an approximate solution $|x'\rangle$ to

$$A\vec{x} = \vec{b} \tag{27}$$

such that

$$\left\| |x'\rangle - \frac{\vec{x}}{\|\vec{x}\|} \right\| \leq \varepsilon. \tag{28}$$

Let N be the dimension; d be the sparsity; κ be the condition number; and k be the rank of the matrix A, which is assumed to be square, normalized ($\|A\| = 1$), and Hermitian [1]. The original HHL algorithm [1] runs in time $O(d^2\kappa^2 \operatorname{polylog}(N)/\varepsilon)$ by performing a Hamiltonian simulation of A (i.e. e^{-iAt}) and quantum phase estimation (QPE) subroutines. Here, the Hamiltonian simulation [47] and QPE steps together contribute a factor of $O(d^2\kappa \operatorname{polylog}(N)/\varepsilon)$, while the post-selection step contributes the remaining factor of $O(\kappa)$.

Later, [2] noticed that the QPE step can be circumvented by utilizing the fact that the inverse function $\frac{1}{x}$ can be written, to the desired precision ε/κ, as a linear combination of $O(\kappa \operatorname{polylog}(\kappa/\varepsilon))$ Chebyshev polynomials. Combining this expansion with the variable-time amplitude amplification technique in [48], they achieved a runtime of $O(d\kappa \operatorname{polylog}(N\kappa/\varepsilon))$. This polynomial expansion approach is generalized in the block-encoding framework by [3, 19] to perform almost any polynomial matrix transformation with the same complexity. All these works use a naive block-encoding in the oracle-access model (Lemma 48 in [3], same as Lemma B.5 or Lemma 3.2).

In another approach, [5] used a QRAM data structure to implement a unitary W whose eigenvalues are related to the singular values of A (this algorithm does not require A to be square or Hermitian) via its Frobenius norm $\|A\|_F$. Thus, performing QPE on W yields a quantum singular value estimation algorithm for A. Matrix multiplication and inversion can then be implemented immediately with a rotation operator controlled on the register storing the singular values, just like done in HHL. The runtime of both operations,

[1]If not, one can consider the equivalent linear system $\begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix} \vec{y} = \begin{pmatrix} \vec{b} \\ 0 \end{pmatrix}$, whose solution is $\vec{y} = \begin{pmatrix} 0 \\ \vec{x} \end{pmatrix}$.

including the post-selection step, is $O(\kappa^2\|A\|_F\,\mathrm{polylog}(N)/\varepsilon)$. For a full-rank matrix whose singular values are $O(1)$, we have $\|A\|_F = \sqrt{N}$. This QRAM model, however, is strictly stronger than the oracle-access model as it requires the entries be loaded and pre-processed in the data structure.

Recent adiabatic algorithms for QLSP [49–51] also enjoy success and culminate in [22], which achieved an optimal query complexity of $O(\kappa \log(1/\varepsilon))$ by constructing a sophisticated schedule function for the adiabatic evolution. This query complexity is better than the algorithms in [2, 3, 5] and has been shown to be optimal [22]. However, all of these assume that the matrix A is perfectly block-encoded inside a unitary $U = \begin{pmatrix} \frac{A}{\|A\|} & \cdot \\ \cdot & \cdot \end{pmatrix}$ which is generally hard to construct without prior knowledge of the structure of A. The exact query complexity and total runtime in the general case is stated in Lemma 5.2 and governed by the ratio $\frac{A}{\alpha}$, where α is the normalization factor in the block-encoding.

The runtimes shown in Table 1 are obtained by combining the optimal adiabatic solver in [22] (see Lemma 5.2) with the corresponding block-encoding used in [2–5, 19]. In particular, the naive oracle-access block-encoding in [2, 3, 19] yields a normalization factor of $\alpha = d = N$ for dense matrices. The QRAM-based block-encoding in [4, 5] yields $\alpha = \|A\|_F = \sqrt{N}$ for full-rank matrices. The runtime for matrix multiplication is simply obtained by computing the probability of success in the post-selection step after applying the block-encoding, as described in Section 3 and Lemma 5.1.

Quantum-inspired algorithms are a class of classical algorithms equipped with sample-query access, a classical analogue to QRAM. The currently fastest quantum-inspired algorithms for QLSP [6, 7] are based on stochastic gradient descent (specifically randomized Kaczmarz method) for a runtime of $\widetilde{O}(\|A\|_F^6\kappa^8/\varepsilon^2)$ (where the notation $\widetilde{O}$ suppresses the log factors). Assuming $\kappa = O(1)$ and the matrix has rank k, this runtime is $\widetilde{O}(k^6/\varepsilon^2)$, which is probitive when the matrix is full-rank.

We note that all the algorithms above are "general-purpose" since they do not assume any inherent structures in the matrix A. Whereas, our work builds on these past works and focuses on optimizing the performance of QLSP algorithms for dense and full-rank kernel matrices and potentially other classes of matrices that suit the $\mathcal{H}$-matrix framework.

B Details of block-encodings

We list some previously established results in the block-encoding framework and prove the helper lemmas presented in the main text. Most of the following results are drawn from Section 4 of [3]. As a reminder, a block-encoding is defined as follows.

Definition B.1 (Block-encoding [3])**.** *Suppose that A is an s-qubit operator, $\alpha, \varepsilon > 0$, then we say that the $(s+a)$-qubit unitary U is an (α, a, ε)-block-encoding of A, if*

$$\left\|A - \alpha(\langle 0|^{\otimes a} \otimes I_s)U(|0\rangle^{\otimes a} \otimes I_s)\right\| \leq \varepsilon,$$

where $\|\cdot\|$ denotes the operator norm of a matrix.

B.1 Linear combinations and multiplications of block-encodings

Combinations of block-encoded matrices can be efficiently combined linearly or multiplied with each other. First we define state-preparation-pairs used to prepare the coefficients in a linear combination.

Definition B.2 (State preparation pair [3])**.** *Let $y \in \mathbb{C}^m$ and $\|y\|_1 \leq \beta$, the pair of unitaries (P_L, P_R) is called a (β, n, ε)-state-preparation-pair of y if $P_L |0\rangle^{\otimes n} = \sum_{j=0}^{2^n-1} c_j |j\rangle$ and $P_R |0\rangle^{\otimes n} = \sum_{j=1}^{2^n-1} d_j |j\rangle$ such that $\sum_{j=0}^{m-1} \left|\beta c_j^* d_j - y_j\right| \leq \varepsilon_1$ and $c_j^* d_j = 0$ for any $j \in \{m, \ldots, 2^n - 1\}$.*

We can then implement a linear combination of block-encoded matrices using an above state preparation pair.

Lemma B.3 (Linear combination of block-encoded matrices, adapted from [3])**.** *Let $A = \sum_{j=0}^{m-1} y_j A_j$ be an s-qubit operator where $\|y\|_1 \leq \beta$. Suppose (P_L, P_R) is a $(\beta, n, \varepsilon_1)$-state-preparation-pair for y, $W = \sum_{j=0}^{m-1} |j\rangle\langle j| \otimes U_j + \left(\left(I - \sum_{j=0}^{m-1} |j\rangle\langle j|\right) \otimes I_a \otimes I_s\right)$ is an $(n+a+s)$-qubit unitary such that for all $j \in [m]$ we have that U_j is an $(\alpha, a, \varepsilon_2)$-block-encoding of A_j. Then the unitary $\widetilde{W} = (P_L^\dagger \otimes I_a \otimes I_s) W (P_R \otimes I_a \otimes I_s)$ is an $(\alpha\beta, a+n, \alpha\varepsilon_1 + \beta\varepsilon_2)$-block-encoding of A.*

Proof. We have:

$$\begin{aligned}
&\left\|A - \alpha\beta \left(\langle 0|^n \langle 0|^a \otimes I_s\right) \widetilde{W} \left(|0\rangle^n |0\rangle^a \otimes I_s\right)\right\| \\
=&\left\|A - \alpha \sum_{j=0}^{m-1} \beta \left(c_j^* d_j\right) \left(\langle 0|^a \otimes I_s\right) U_j \left(|0\rangle^a \otimes I_s\right)\right\| \\
\leq&\alpha\varepsilon_1 + \left\|A - \alpha \sum_{j=0}^{m-1} y_j \left(\langle 0|^a \otimes I_s\right) U_j \left(|0\rangle^a \otimes I_s\right)\right\| \\
\leq&\alpha\varepsilon_1 + \sum_{j=0}^{m-1} |y_j| \left\|A_j - \alpha \left(\langle 0|^a \otimes I_s\right) U_j \left(|0\rangle^a \otimes I_s\right)\right\| \\
\leq&\alpha\varepsilon_1 + \beta\varepsilon_2. \quad \square
\end{aligned}$$

We note that in the original work [3], Lemma 52 states that the error of the block-encoding is $\alpha\varepsilon_1 + \alpha\beta\varepsilon_2$, which is a typo. Lemma B.3 also applies to cases where P_L and P_R are non-unitary. As long as block-encodings of P_L, P_R are provided, we can still implement the unitary $\widetilde{W}$ in Lemma B.3 by using the following lemma, which allows one to multiply block-encoded matrices.

Lemma B.4 (Product of block-encoded matrices [3])**.** *If U is an (α, a, δ)-block-encoding of an s-qubit operator A, and V is an (β, b, ε)-block-encoding of an s-qubit operator B then $(I_b \otimes U)(I_a \otimes V)$ is an $(\alpha\beta, a+b, \alpha\varepsilon + \beta\delta)$-block-encoding of AB. Note that here I_a (I_b) acts on the ancilla qubits of U (V).*

Proof.

$$\begin{aligned}
&\|AB - \alpha\beta(\langle 0^{a+b}| \otimes I_s)(I_b \otimes U)(I_a \otimes V)(|0^{a+b}\rangle \otimes I_s)\| \\
=&\|AB - \underbrace{\alpha(\langle 0^a| \otimes I_s)U(|0^a\rangle \otimes I_s)}_{\widetilde{A}} \underbrace{\beta(\langle 0^b| \otimes I_s)V(|0^b\rangle \otimes I_s)}_{\widetilde{B}}\| \\
=&\|AB - \widetilde{A}B + \widetilde{A}B - \widetilde{A}\widetilde{B}\| \\
=&\|(A - \widetilde{A})B + \widetilde{A}(B - \widetilde{B})\| \\
\leq&\|A - \widetilde{A}\|\beta + \alpha\|B - \widetilde{B}\| \\
\leq&\alpha\varepsilon + \beta\delta. \quad \square
\end{aligned}$$

B.2 Block-encoding of oracle-access matrices

The following lemma can be used to block-encode any matrix whose entries are provided by an oracle.

Lemma B.5 (Block-encoding of oracle-access matrices, adapted from Lemma 48 of [3])**.** *Let $A \in \mathbb{C}^{2^s \times 2^s}$ be a d_r-row-sparse and d_c-column-sparse matrix. Suppose that we have the access to the following $2(s+1)$-qubit oracles*

$$\begin{aligned} \mathcal{O}_r &: |i\rangle |k\rangle \to |i\rangle |r_{ik}\rangle \quad 0 \le i < 2^s, 0 \le k < d_r, \\ \mathcal{O}_c &: |l\rangle |j\rangle \to |c_{lj}\rangle |j\rangle \quad 0 \le j < 2^s, 0 \le l < d_c, \end{aligned}$$

where r_{ik} is the index for the k-th non-zero entry of the i-th row of A, or if there are less than k non-zero entries, then $r_{ik} = k + 2^s$, and similarly c_{lj} is the index for the l-th non-zero entry of the j-th column of A, or if there are less than l non-zero entries, then $c_{lj} = l + 2^s$. Additionally, let $\hat{a} \ge \max_{i,j} |a_{ij}|$ and suppose the following oracle is provided

$$\mathcal{O}_A : |i\rangle |j\rangle |0\rangle^{\otimes b} \to |i\rangle |j\rangle |\tilde{a}_{ij}\rangle, \quad 0 \le i, j < 2^s,$$

where $\tilde{a}_{ij}$ is the (exact) b-qubit description of $a_{ij}/\hat{a}$, and if i or j is out of range then $\tilde{a}_{ij} = 0$. Then one can implement an $(\hat{a}\sqrt{d_r d_c}, s+3, \varepsilon)$-block-encoding of A with a single use of $\mathcal{O}_r, \mathcal{O}_c$, two uses of $\mathcal{O}_A$ and additionally $O(s + \operatorname{polylog}(\frac{\hat{a}\sqrt{d_r d_c}}{\varepsilon}))$ one- and two-qubit gates while using $O(b, \operatorname{polylog}(\frac{\hat{a}\sqrt{d_r d_c}}{\varepsilon}))$ ancilla qubits (which are discarded before the post-selection step).

Proof. Let G_d be the $(s+1)$-qubit unitary that performs: $|0\rangle \to \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} |i\rangle$, which can be implemented with $O(s)$ Hadamard gates (assume d is a power of 2 and note that $d \le 2^s$). Additionally, define the following $2(s+1)$-qubit operators: $V_R = \mathcal{O}_c(G_{d_c} \otimes I_{s+1})$ and $V_L = \mathcal{O}_r(I_{s+1} \otimes G_{d_r})\,\mathrm{SWAP}_{s+1}$, where SWAP_{s+1} swaps two $(s+1)$-qubit registers (which uses s two-qubit swap gates). For any $0 \le i, j < 2^s$, we have that

$$\begin{aligned} &\langle 0^{s+1}| \langle i| V_L^\dagger V_R |0^{s+1}\rangle |j\rangle \\ =&\left(\frac{1}{\sqrt{d_r}} \sum_{k=0}^{d_r-1} \langle i| \langle r_{ik}| \right) \left(\frac{1}{\sqrt{d_c}} \sum_{l=0}^{d_c-1} |c_{lj}\rangle |j\rangle \right) \\ =&\frac{1}{\sqrt{d_r d_c}} \text{ if } a_{ij} \ne 0 \text{ and } 0 \text{ otherwise.} \end{aligned}$$

Observe that after applying V_R, instead of applying $V_L^\dagger$ right away, we can first append a b-qubit register and call $\mathcal{O}_A$ to query the matrix entry, then append the register $|0\rangle$ and perform the controlled rotation: $|0\rangle |\tilde{a}_{ij}\rangle \to (a_{ij}/\hat{a}\,|0\rangle + \sqrt{1 - |a_{ij}/\hat{a}|^2}\,|1\rangle) |\tilde{a}_{ij}\rangle$, and finally call $\mathcal{O}_A$ again to uncompute the b-qubit ancilla register. Conditioning on the ancilla qubit being $|0\rangle$, we obtain the desired block-encoding. We only count $s+3$ qubits as ancilla qubits in the block-encoding notation because the others can be discarded immediately after the uncomputation step.

We analyze the circuit complexity of the controlled rotation step above. Observe that, assuming the b-qubit description $|\tilde{a}_{ij}\rangle$ is exact, if the controlled rotation can be done with an accuracy of $O(\frac{\varepsilon}{\hat{a}\sqrt{d_r d_c}})$ then we achieve the overall accuracy of the block-encoding. This step implements circuits to compute elementary functions such as the square root and trigonometric functions with $O(\log(\frac{\hat{a}\sqrt{d_r d_c}}{\varepsilon}))$ additional qubits. Implementing these elementary functions on these additional qubits requires $O(b, \operatorname{polylog}(\frac{\hat{a}\sqrt{d_r d_c}}{\varepsilon}))$ elementary logic gates [52, 53]. After computing the rotation angles, it takes $O(\log(\frac{\hat{a}\sqrt{d_r d_c}}{\varepsilon}))$ controlled quantum gates to perform the controlled rotation. $\square$

We can see that the normalization factor of the above block-encoding depends on the sparsity of the matrix. Without loss of generality, assume $\hat{a} = 1$. Then, the above lemma when applied to dense matrices produces a block-encoding with a normalization factor of $O(N)$. This is not optimal (see definition in Section 3) if the operator norm of the matrix is significantly smaller than $O(N)$. Hence, we refer to it as the "naive" block-encoding. We provide some examples in which this lemma works optimally in appendix C.6.

Remark B.6 ("Naive" block-encoding of dense oracle-access matrices (same as Lemma 3.2)). *In the dense case, the oracles $\mathcal{O}_r, \mathcal{O}_c$ are not needed; we can simply use s Hadamard gates to query all matrix entries. It can be easily verified that in this case we can obtain an $(\hat{a}2^s, s+1, \varepsilon)$-block-encoding with two uses of $\mathcal{O}_A$ and additionally $O(s + \operatorname{polylog}(\frac{\hat{a}2^s}{\varepsilon}))$ one- and two-qubit gates while using $O(b, \operatorname{polylog}(\frac{\hat{a}2^s}{\varepsilon}))$ ancilla qubits.*

If the matrix A is rectangular, that is $A \in \mathbb{C}^{2^s \times 2^t}$, where we assume $s > t$ without loss of generality, we can block-encode A by applying the above lemmas on the 2^s by 2^s matrix $\widetilde{A}$ defined as

$$\widetilde{A}_{ij} = \begin{cases} A_{ij} & \text{if } i < 2^s \text{ and } j < 2^t \\ 0 & \text{otherwise} \end{cases}. \tag{29}$$

Then for an input state $|\psi\rangle$, we can obtain $A|\psi\rangle$ by applying $\widetilde{A}$ on the state $|0^{s-t}\rangle \otimes |\psi\rangle$.

B.3 Block-encoding of low-rank matrices

Recall Lemma 3.4, copied below.

Lemma B.7 (Block-encoding of low-rank operators with state preparation unitaries, inspired by Lemma 1 of [33]). *Let $A = \sum_{i=0}^{p-1} \sigma_i \mathbf{u}_i \mathbf{v}_i^\dagger \in \mathbb{C}^{2^s \times 2^s}$. Let $r = \lceil \log p \rceil$ and $\sum_{i=0}^{p-1} |\sigma_i| \leq \beta$. Suppose the following $(r+s)$-qubit unitaries are provided:*

$$G_L : |i\rangle |0^s\rangle \to |i\rangle |\mathbf{u}_i\rangle, G_R : |i\rangle |0^s\rangle \to |i\rangle |\mathbf{v}_i\rangle,$$

where $0 \leq i < p$ and $|\mathbf{u}_i\rangle$ ($|\mathbf{v}_i\rangle$) is the quantum state whose amplitudes are the entries of $\mathbf{u}_i$ ($\mathbf{v}_i$). Let (P_L, P_R) be a (β, r, ε)-state-preparation-pair for the vector $[\sigma_0, \ldots, \sigma_{p-1}]$. Then, one can construct a $(\beta, r+s, \varepsilon)$-block-encoding of A with one use of each of $G_L, G_R^\dagger, P_L^\dagger, P_R$ and a SWAP gate on two s-qubit registers.

Proof. Observe that the $(2s+r)$-qubit unitary $U = (P_L^\dagger \otimes I_{2s})(G_R^\dagger \otimes I_s)(I_r \otimes \mathrm{SWAP}_s)(G_L \otimes I_s)(P_R \otimes I_{2s})$ is a $(\beta, r+s, \varepsilon)$-block-encoding of A. Indeed, for any $0 \leq m, n < 2^s$ we have that

$$\begin{aligned}
&(\langle 0^{r+s}| \otimes \langle m|) U (|0^{r+s}\rangle \otimes |n\rangle) \\
&= \left(\sum_{j=0}^{p-1} c_j^* \langle j| \langle \mathbf{v}_j| \langle m| \right) (I_r \, \mathrm{SWAP}_s) \left(\sum_{i=0}^{p-1} d_i |i\rangle |\mathbf{u}_i\rangle |n\rangle \right) \\
&= \sum_{i=0}^{p-1} c_i^* d_i \langle \mathbf{v_i}| n\rangle \langle m |\mathbf{u}_i\rangle = \langle m| \left(\sum_{i=0}^{p-1} c_i^* d_i |\mathbf{u}_i\rangle \langle \mathbf{v_i}| \right) |n\rangle,
\end{aligned}$$

and

$$
\begin{aligned}
&\left\|\beta\left(\sum_{i=0}^{p-1} c_i^* d_i \left|\mathbf{u}_i\right\rangle\left\langle\mathbf{v_i}\right|\right) - A\right\| \\
&= \left\|\left(\sum_{i=0}^{p-1}(\beta c_i^* d_i - \sigma_i)\left|\mathbf{u}_i\right\rangle\left\langle\mathbf{v_i}\right|\right)\right\| \\
&\le \sum_{i=0}^{p-1} \left|(\beta c_i^* d_i - \sigma_i)\right| \le \varepsilon. \quad \square
\end{aligned}
$$

The unitaries G_R, G_L can be constructed using existing state preparation methods based on the structure of the entries in the singular vectors (*e.g.,* [34] utilized efficiently integrable distributions, [4] used a QRAM data structure). As an example, when given oracle-access to the entries of the vectors $\mathbf{u}, \mathbf{v}$, one can use the following lemma to block-encode rank-1 operators.

Lemma B.8 (Block-encoding of rank-1 operators with oracle access entries)**.** *Let* $A = \mathbf{u}\mathbf{v}^\dagger \in \mathbb{C}^{2^s \times 2^s}$ *be a rank-1 operator. Suppose the following oracles are provided:*

$$
\mathcal{O}_{\mathbf{u}} : \left|i\right\rangle \left|z\right\rangle^b \to \left|i\right\rangle \left|z \oplus \tilde{u}_i\right\rangle, \mathcal{O}_{\mathbf{v}} : \left|j\right\rangle \left|z\right\rangle^b \to \left|j\right\rangle \left|z \oplus \tilde{v}_j\right\rangle,
$$

where $\tilde{u}_i(\tilde{v}_j)$ *is the b-qubit exact description of* $u_i(v_j)$*. Let* $\max_i |u_i| \le \hat{u}$ *and* $\max_j |v_j| \le \hat{v}$*. Then one can obtain a* $(2^s\hat{u}\hat{v}, s + 2b + 2, 0)$*-block-encoding of* A *with two uses of* $\mathcal{O}_{\mathbf{u}}$*,* $\mathcal{O}_{\mathbf{v}}$ *each.*

Proof. Let $\mathrm{CR}_{\hat{u}}$ denote the conditioned rotation operator: $\mathrm{CR}_{\hat{u}} \left|\tilde{u}_i\right\rangle\left|0\right\rangle = \left|\tilde{u}_i\right\rangle\left(\frac{u_i}{\hat{u}}\left|0\right\rangle + \sqrt{1 - \left|\frac{u_i}{\hat{u}}\right|^2}\left|1\right\rangle\right)$. Let $G_{\mathbf{u}} = (\mathcal{O}_{\mathbf{u}} \otimes I)(I_s \otimes CR_{\hat{u}})(\mathcal{O}_{\mathbf{u}} \otimes I)(H^{\otimes s} \otimes I_{b+1})$. Observe that

$$
(I_s \otimes \left\langle 0\right|^{b+1}) G_{\mathbf{u}} \left|0\right\rangle^{s+b+1} = 2^{-s/2} \sum_{i=0}^{2^s-1} \frac{u_i}{\hat{u}} \left|i\right\rangle.
$$

Define $G_{\mathbf{v}}$ similarly as above. Then we have[2]

$$
\begin{aligned}
&\left\langle 0\right|^s \left\langle m\right| \left\langle 0\right|^{b+1} \left\langle 0\right|^{b+1} (G_{\mathbf{v}}^\dagger \otimes I)(I \otimes \mathrm{SWAP}_s)(G_{\mathbf{u}} \otimes I) \left|0\right\rangle^s \left|n\right\rangle \left|0\right\rangle^{b+1} \left|0\right\rangle^{b+1} \\
&= \frac{1}{2^s} \sum_{i,j=0}^{2^s-1} \left\langle j|n\right\rangle \left\langle m|i\right\rangle \frac{u_i}{\hat{u}} \frac{v_j}{\hat{v}} = \frac{1}{2^s\hat{u}\hat{v}} u_m v_n. \quad \square
\end{aligned}
$$

This low-rank block-encoding can also be applied for rectangular matrices. Suppose $\mathbf{u} \in \mathbb{C}^{2^m}$ and $\mathbf{v} \in \mathbb{C}^{2^n}$, where we assume without loss of generality $m > n$, then we can apply the lemmas on the matrix $\widetilde{A} = \mathbf{u}(\left\langle 0\right|^{m-n} \otimes \mathbf{v}^\dagger)$.

Remark B.9. *The normalization factor* $2^s\hat{u}\hat{v}$ *is to ensure that the block-encoding is unitary. Thus, Lemma B.8 works best when the entries* u_i *and* v_j *change slowly. If* $\mathbf{u}$ *or* $\mathbf{v}$ *is sparse, we can apply the techniques in Lemma B.5 to improve the normalization factor of this low-rank block-encoding. In general, however, the above lemma is rather redundant since we can directly apply Lemma B.5 naively by constructing the matrix entry-access oracle from* $\mathcal{O}_{\mathbf{u}}$ *and* $\mathcal{O}_{\mathbf{v}}$*. We simply present this lemma here for completeness.*

[2]Here, $G_{\mathbf{u}}$ and $G_{\mathbf{v}}$ act on each other register $\left|0\right\rangle^{b+1}$.

B.4 Block-encoding of block-sparse matrices (Lemma 3.5)

In this section, we provide a block-encoding procedure for block-sparse matrices. First we consider the block-diagonal case and later use these techniques for constructing general block-sparse matrices.

Lemma B.10 (Block-encoding of block-diagonal matrices)**.** *Let $\{A^j : 0 \leq j \leq 2^t - 1\}$ be a set of s-qubit operators and each U^j be an $(\alpha_j, a, \varepsilon)$-block-encoding of A^j. Let $W = \sum_{j=0}^{2^t-1} |j\rangle\langle j| \otimes U^j$ and $\hat{\alpha} = \max_j \alpha_j$. Furthermore, suppose that the following oracle is provided: $\mathcal{O}_\alpha : |j\rangle|z\rangle \to |j\rangle|z \oplus \tilde{\alpha}_j\rangle$, where $\tilde{\alpha}_j$ is the (exact) b-qubit description of α_j and z is a b-qubit string. Then one can obtain an $(\hat{\alpha}, a+1, 2\varepsilon)$-block-encoding of $A = \sum_{j=0}^{2^t-1} |j\rangle\langle j| \otimes A^j$ with two uses of $\mathcal{O}_\alpha$, one use of W, and additionally $O(\mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ one- and two-qubit gates while using $O(b, \mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ ancilla qubits (which are discarded before the post-selection step).*

Proof. Let CR denote the controlled rotation operator: $\mathrm{CR}\,|\tilde{\alpha}_j\rangle|0\rangle = |\tilde{\alpha}_j\rangle\left(\bar{\alpha}_j|0\rangle + \sqrt{1-|\bar{\alpha}_j|^2}\,|1\rangle\right)$, with $|\bar{\alpha}_j - \frac{\alpha_j}{\hat{\alpha}}| \leq \frac{\varepsilon}{\hat{\alpha}}$. Implementing this operator requires $O(\mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ one- and two-qubit gates while using $O(b, \mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ ancilla qubits (see explanation in the proof of Lemma B.5). Observe that $\widetilde{W} = (\mathcal{O}_\alpha \otimes I_{s+a+1})(\mathrm{CR} \otimes I_{s+t+a})(\mathcal{O}_\alpha \otimes I_{s+a+1})(I_{b+1} \otimes W)$ is the desired block-encoding[3]. Indeed, we have that

$$\widetilde{W}(|0\rangle \otimes |0\rangle^b \otimes I_{s+t+a}) = \sum_{j=0}^{2^t-1}\left(\bar{\alpha}_j|0\rangle + \sqrt{1-|\bar{\alpha}_j|^2}\,|1\rangle\right) \otimes |0\rangle^b \otimes |j\rangle\langle j| \otimes U^j,$$

Therefore,

$$\begin{aligned}
&\left\|\hat{\alpha}\left(\langle 0|^{1+b+a} \otimes I_{s+t}\right)\widetilde{W}\left(|0\rangle^{1+b+a} \otimes I_{s+t}\right) - A\right\| \\
=&\left\|\bigoplus_{0\leq j<2^t}\left(\hat{\alpha}\bar{\alpha}_j \langle 0|^a U^j |0\rangle^a - A^j\right)\right\| \\
\leq&\left\|\bigoplus_{0\leq j<2^t}\left(\alpha_j \langle 0|^a U^j |0\rangle^a - A^j\right)\right\| + \left\|\bigoplus_{0\leq j<2^t}(\hat{\alpha}\bar{\alpha}_j - \alpha_j)\langle 0|^a U^j |0\rangle^a\right\| \\
\leq&\max_{0\leq j<2^t}\left\|\alpha_j \langle 0|^a U^j |0\rangle^a - A^j\right\| + \max_{0\leq j<2^t}\left\|(\hat{\alpha}\bar{\alpha}_j - \alpha_j)\langle 0|^a U^j |0\rangle^a\right\| \\
\leq&2\varepsilon.
\end{aligned}$$

We do not include b in the notation of the block-encoding because this register can actually be taken out of the system right after the uncomputation step. □

In the above lemma, if the normalization factors α_j are the same for all blocks, then the operator W is already an $(\hat{\alpha}, a, \varepsilon)$-block-encoding of the desired block-diagonal matrix.

Next, we apply the proof techniques above to generalize to the block-sparse case. The following lemma is the same as Lemma 3.5.

Lemma B.11 (Block-encoding of block-sparse matrices)**.** *Let $A = \sum_{i,j=0}^{2^t-1} |i\rangle\langle j| \otimes A^{ij}$ be a d_r-row-block-sparse and d_c-column-block-sparse matrix, where each A^{ij} is an s-qubit*

[3]The subscripts indicate the registers the identity act on.

operator. Let U^{ij} be an $(\alpha_{ij}, a, \varepsilon)$-block-encoding of A^{ij}. Suppose that we have the access to the following $2(t+1)$-qubit oracles

$$\begin{aligned} \mathcal{O}_r &: |i\rangle |k\rangle \to |i\rangle |r_{ik}\rangle \qquad \forall i \in \{0,\dots,2^t-1\}, k \in \{0,\dots,d_r-1\},\\ \mathcal{O}_c &: |l\rangle |j\rangle \to |c_{lj}\rangle |j\rangle \qquad \forall j \in \{0,\dots,2^t-1\}, l \in \{0,\dots,d_c-1\}, \end{aligned}$$

where r_{ik} is the index for the k-th non-zero block of the i-th block-row of A, or if there are less than k non-zero blocks, then $r_{ik} = k + 2^t$, and similarly c_{lj} is the index for the l-th non-zero block of the j-th block-column of A, or if there are less than l non-zero blocks, then $c_{lj} = l + 2^t$. Additionally, suppose the following oracle is provided:

$$\mathcal{O}_\alpha : |i\rangle |j\rangle |z\rangle \to |i\rangle |j\rangle |z \oplus \tilde{\alpha}_{ij}\rangle, \forall i, j \in \{0,\dots,2^t-1\}$$

where $\tilde{\alpha}_{ij}$ is the b-qubit description of α_{ij} (if i or j is out of range then $\tilde{\alpha}_{ij} = 0$), and let the $(2t+2+s+a)$-qubit unitary $W = \sum_{i,j:A^{ij}\neq 0}(|i\rangle|j\rangle\langle i|\langle j|)\otimes U^{ij} + \Big(I - \sum_{i,j:A^{ij}\neq 0}(|i\rangle|j\rangle\langle i|\langle j|)\Big)\otimes I_{s+a}$. Let $\hat{\alpha} = \max_{i,j}\alpha_{ij}$. Then one can implement an $(\hat{\alpha}\sqrt{d_r d_c}, t+a+3, 2\sqrt{d_r d_c}\varepsilon)$-block-encoding of A, with one use of each of $\mathcal{O}_r, \mathcal{O}_c$, and W, two uses of $\mathcal{O}_\alpha$, $O(t + \mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ additional one- and two-qubit gates, and $O(b, \mathrm{polylog}(\frac{\hat{\alpha}}{\varepsilon}))$ extra ancilla qubits (which are discarded before the post-selection step).

Proof. Let CR be the controlled rotation operator: $\mathrm{CR}\,|\tilde{\alpha}_{ij}\rangle|0\rangle = |\tilde{\alpha}_{ij}\rangle\Big(\bar{\alpha}_{ij}|0\rangle + \sqrt{1-|\bar{\alpha}_{ij}|^2}\,|1\rangle\Big)$, where $|\bar{\alpha}_{ij} - \frac{\alpha_{ij}}{\hat{\alpha}}| \leq \frac{\varepsilon}{\hat{\alpha}}$. Let G_d be a $(t+1)$-qubit unitary such that $G_d|0^{t+1}\rangle = \sum_{k=0}^{d-1}\frac{|k\rangle}{\sqrt{d}}$, where $d \leq 2^t$. Additionally, let $V_R = \mathcal{O}_c(G_{d_c}\otimes I_{t+1})$ and $V_L = \mathcal{O}_r(I_{t+1}\otimes G_{d_r})\,\mathrm{SWAP}_{t+1}$. For any $0 \leq i, j < 2^t$, observe that

$$\begin{aligned} \langle 0^{t+1}|\langle i| V_L^\dagger V_R |0^{t+1}\rangle|j\rangle &= \left(\sum_{k=0}^{d_r-1}\langle i|\frac{\langle r_{ik}|}{\sqrt{d_r}}\right)\left(\sum_{k=0}^{d_c-1}|c_{lj}\rangle\frac{|j\rangle}{\sqrt{d_c}}\right)\\ &= \frac{1}{\sqrt{d_r d_c}} \text{ if } A^{ij}\neq 0 \text{ and } 0 \text{ otherwise.} \end{aligned} \tag{30}$$

Let the $(2t+2+s+a)$-qubit unitary $W = \sum_{i,j:A^{ij}\neq 0}(|i\rangle|j\rangle\langle i|\langle j|)\otimes U^{ij} + (I - \sum_{i,j:A^{ij}\neq 0}(|i\rangle|j\rangle\langle i|\langle j|))\otimes I_{s+a}$ and, similarly to Lemma B.10, $\widetilde{W} = (\mathcal{O}_\alpha\otimes I_{s+a+1})(\mathrm{CR}\otimes I_{s+2t+2+a})(\mathcal{O}_\alpha\otimes I_{s+a+1})(I_{b+1}\otimes W)$ be a $(2t+s+a+b+3)$-qubit unitary [4]. Then $(V_L^\dagger\otimes I_{s+a+b+1})\widetilde{W}(V_R\otimes I_{s+a+b+1})$ is the desired block-encoding. Indeed, this follows directly from Equation 30 and a similar proof to that of Lemma B.10, noting that $\widetilde{W}$ leaves the $|c_{lj}\rangle$ and $|j\rangle$ registers unchanged, and

$$\begin{aligned} &\left\|\sum_{i,j:A^{ij}\neq 0}|i\rangle\langle j|\otimes(\hat{\alpha}\bar{\alpha}_{ij}\langle 0|^a U^{ij}|0\rangle^a - A^{ij})\right\|\\ \leq&\left\|\sum_{i,j:A^{ij}\neq 0}|i\rangle\langle j|\otimes(\alpha_{ij}\langle 0|^a U^{ij}|0\rangle^a - A^{ij})\right\| + \left\|\sum_{i,j:A^{ij}\neq 0}|i\rangle\langle j|\otimes(\hat{\alpha}\bar{\alpha}_{ij} - \alpha_{ij})\langle 0|^a U^{ij}|0\rangle^a)\right\|\\ \leq&\sqrt{d_r d_c}\cdot\left(\max_{i,j:A^{ij}\neq 0}\left\|\alpha_{ij}\langle 0|^a U^{ij}|0\rangle^a - A^{ij}\right\| + \max_{i,j:A^{ij}\neq 0}\left\|(\hat{\alpha}\bar{\alpha}_{ij} - \alpha_{ij})\langle 0|^a U^{ij}|0\rangle^a\right\|\right)\\ \leq&2\sqrt{d_r d_c}\varepsilon. \end{aligned}$$

Implementing G_d and the controlled rotation operator requires the indicated additional gate and ancilla qubit complexities (see explanation in the proof of Lemma B.5). □

[4] The subscripts indicate the registers the identity act on.

This lemma does not require sparsity to work. It generalizes Lemma B.5 (also Lemma 48 of [3]) for block-encoding matrices with oracle access to entries. If α_{ij} are the same for all blocks, we can omit the oracle $\mathcal{O}_\alpha$ and the controlled rotation step, hence only $t+a+2$ ancilla qubits are needed.

C Detailed analysis of block-encodings of kernel matrices

C.1 Circuit complexity analysis of Section 4.1

In this section, we provide detailed analysis on circuit and query complexities of the block-encoding procedure presented in Section 4.1. Our problem is to optimally block-encode the kernel matrix $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$, where $x_i = i$, $N = 2^L$, and $k(x, x') = |x - x'|^{-p}$ is a polynomially decaying kernel, given the oracle

$$\mathcal{O}_k : |i\rangle\,|j\rangle\,|0\rangle^{\otimes b} \to |i\rangle\,|j\rangle\,|\tilde{k}(x_i, x_j)\rangle\,, \tag{31}$$

where $\tilde{k}(x_i, x_j)$ is the (exact) b-qubit description of $k(x_i, x_j)$.

As a reminder, our procedure uses the hierarchical splitting in Figure 1. Specifically, at level ℓ of the hierarchy, each admissible block has size $2^{L-\ell}$. In addition, the maximum entry of an admissible block in $\mathbf{K}^{(\ell)}$ is bounded by $d_{\min}^{-p} = 2^{-(L-\ell)p}$. Thus, an admissible block at level ℓ can be $(\alpha_\ell, L-\ell+1, \frac{\varepsilon}{3})$-block-encoded via the naive procedure for dense matrices (Lemma 3.2 or Remark B.6), where $\alpha_\ell = 2^{(L-\ell)(1-p)}$. From here, each $\mathbf{K}^{(\ell)}$, which is a block-sparse matrix of sparsity 3, can be $(3\alpha_\ell, L+3, \varepsilon)$-block-encoded via the procedure for block-sparse matrices (Lemma 3.5). The adjacent interaction part $\mathbf{K}_{ad}$, which is a tridiagonal matrix, can be $(3, L+3, \varepsilon)$-block-encoded with Lemma B.5. Finally, we use the procedure for linear combination of block-encoded matrices (Lemma B.3) to obtain a block-encoding of $\mathbf{K} = \sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}$ as $U = \sum_{\ell=2}^{L} 3\alpha_\ell U^{(\ell)} + 3U_{ad}$, where the U's denote the corresponding block-encodings. This step requires a $(\log L)$-qubit state preparation pair (Definition 3.3) which prepares the coefficients $[3, 3\alpha_2, \ldots, 3\alpha_L]$ in the linear combination. This entire procedure results in a $(\alpha, L+\log L+3, \alpha\varepsilon)$-block-encoding of the *exact* kernel matrix $\mathbf{K}$, where,

$$\alpha = \begin{cases} 3\left(1 + \frac{2^{1-p} - N^{1-p}}{2^{1-p} - 4^{1-p}}\right) \le O(1) & \text{if } p > 1 \\ 3\log N & \text{if } p = 1 \\ 3\left(1 + \frac{N^{1-p} - 2^{1-p}}{4^{1-p} - 2^{1-p}}\right) & \text{if } p < 1 \end{cases}. \tag{32}$$

In the main text, we have shown that this procedure results in an optimal block-encoding. That is, in the obtained $(\alpha, \log N + \log\log N + 3, \varepsilon)$-block-encoding, the normalization factor α is exactly $\Theta(\|\mathbf{K}\|)$. Here, we show why exactly two queries to $\mathcal{O}_k$ and $O(\text{polylog}(\frac{N}{\varepsilon}))$ extra one- and two-qubit gates are needed by carefully combining Lemma 3.5, Lemma B.3, and Lemma 3.2.

Resources We use 4 registers A ($\log L$ qubits), B ($L+1$ qubits), C ($L+1$ qubits), D (single qubit). For simplicity, assume $\log L$ is an integer. We want to construct a $(2L + \log L + 3)$-qubit unitary U such that, for any $0 \le m, n \le 2^L - 1$, we have

$$\langle 0|_A \langle 0|_B \langle m|_C \langle 0|_D\, U\, |0\rangle_A |0\rangle_B |n\rangle_C |0\rangle_D = \frac{k_{mn}}{\alpha}. \tag{33}$$

Here, although $0 \le m, n \le 2^L - 1$, we allocate $(L+1)$ qubits to the registers B, C for later use.

State preparation pair Observe that Lemma B.3 requires a state preparation pair (P_L, P_R) (Definition 3.3) that prepares the coefficient vector $\boldsymbol{\beta} = [3, 3\alpha_2, \ldots, 3\alpha_L]$ (the first entry being α_{ad} of $\mathbf{K}_{ad}$). Note that $\|\boldsymbol{\beta}\|_1 = \alpha$, the overall normalization factor. These operators are $\log L = \log\log N$ qubits, hence we assume they are provided for now (later we will show how to efficiently construct them). In particular, we choose $P_L = P_R$ and

$$P_R \left|0\right\rangle_A = \frac{1}{\sqrt{\alpha}} \sum_{\ell=0}^{L-1} \sqrt{\beta_\ell} \left|\ell\right\rangle_A . \tag{34}$$

Implementing the "right" part Suppose the registers are in the state

$$\left|0\right\rangle_A \left|0\right\rangle_B \left|n\right\rangle_C \left|0\right\rangle_D . \tag{35}$$

We apply P_R to get (omitting the register names for ease of notation)

$$\frac{1}{\sqrt{\alpha}} \sum_{\ell=1}^{L} \sqrt{\beta_{\ell-1}} \left|\ell-1\right\rangle \left|0\right\rangle \left|n\right\rangle \left|0\right\rangle . \tag{36}$$

Next, we take care of Lemma 3.5 for the block-sparse levels $\mathbf{K}^{(\ell)}$. Conditioned on register A being $\ell - 1$, where $2 \leq \ell \leq L$, we focus on a particular ℓ. Define the operator $D_3 : \left|0^{L+1}\right\rangle \to \frac{1}{\sqrt{3}} \sum_{c=1}^{3} \left|0\right\rangle \left|c\right\rangle \left|0^{L-2}\right\rangle$. Apply D_3 on the register B we get

$$\frac{1}{\sqrt{3}} \sum_{c=1}^{3} \left|\ell-1\right\rangle \underbrace{\left|0\right\rangle \left|c\right\rangle \left|0^{L-2}\right\rangle}_{\text{register } B} \left|n\right\rangle \left|0\right\rangle . \tag{37}$$

Next, we apply the controlled block-row index oracle $\mathcal{O}_r$, which, based on ℓ, c, n, computes the block-row index m_b of the c-th non-zero admissible block in $\mathbf{K}^{(\ell)}$ that intersects column n (there are at most 3 such blocks since $\mathbf{K}^{(\ell)}$ is 3-block-sparse) and writes the result m_b to the first $\ell + 1$ qubits of register B. Note that this computation is easy and reversible. We denote this set of non-zero block-row index m_b as $I(\ell, n)$. If there are less than c such blocks, $\mathcal{O}_r$ writes $c + 2^\ell$ instead (this choice ensures the oracle is reversible and also the first qubit of $\left|m_b\right\rangle$ acts as a flag that will be useful later).

$$\frac{1}{\sqrt{3}} \sum_{m_b \in I(\ell,n)} \left|\ell-1\right\rangle \underbrace{\left|m_b\right\rangle \left|0^{L-\ell}\right\rangle}_{\text{register } B} \left|n\right\rangle \left|0\right\rangle . \tag{38}$$

Next, we take care of the steps in Lemma 3.2. Also conditioned on register A being $\ell - 1$, where $2 \leq \ell \leq L$, we apply Hadamard gates on the last $L - \ell$ qubits of register B

$$\frac{1}{\sqrt{3}} \frac{1}{\sqrt{2^{L-\ell}}} \sum_{m_b \in I(\ell,n)} \sum_{m_i=0}^{2^{L-\ell}-1} \left|\ell-1\right\rangle \underbrace{\left|m_b\right\rangle \left|m_i\right\rangle}_{\text{register } B} \left|n\right\rangle \left|0\right\rangle . \tag{39}$$

Observe that when the first qubit of $\left|m_b\right\rangle$ is zero, $\left|m_b\right\rangle \left|m_i\right\rangle$ together form a valid row index $m < 2^L$. If this qubit is one, then $\left|m_b\right\rangle \left|m_i\right\rangle$ is not a valid row index and we can simply take $k_{mn} = 0$ when querying the matrix entry oracle $\mathcal{O}_k$. In particular, we query the oracle $\mathcal{O}_k$ on registers B, C to obtain $|\tilde{k}_{mn}\rangle$ in another appended b-qubit register. Then, we perform the controlled rotation to rotate register D as: $\left|0\right\rangle \to \left(\frac{k_{mn}}{k_{\max}^{(\ell)}} \left|0\right\rangle + \sqrt{1 - \left|\frac{k_{mn}}{k_{\max}^{(\ell)}}\right|^2} \left|1\right\rangle \right)$, where $k_{\max}^{(\ell)}$ is the maximum entry in the admissible blocks at level ℓ, which we have shown to be $2^{-(L-\ell)p}$. Finally, we call $\mathcal{O}_k$ one more time to uncompute and discard the appended register. This gives

$$\frac{1}{\sqrt{3}}\frac{1}{\sqrt{2^{L-\ell}}}\sum_{m_b\in I(\ell,n)}\sum_{m_i=0}^{2^{L-\ell}-1}|\ell-1\rangle\underbrace{|m_b\rangle|m_i\rangle}_{\text{register } B}|n\rangle\left(\frac{k_{mn}}{k_{\max}^{(\ell)}}|0\rangle+\sqrt{1-\left|\frac{k_{mn}}{k_{\max}^{(\ell)}}\right|^2}|1\rangle\right) \tag{40}$$

for $2\leq\ell\leq L$.

On the other hand, conditioned on $\ell=1$ (register A being $|0\rangle$), we take care of the adjacent part $\mathbf{K}_{ad}$ rather simply. Apply the $(2L+2)$-qubit operator $\mathcal{O}_{ad}$ which performs the map $|0\rangle|n\rangle\to\frac{1}{\sqrt{3}}\sum_{i=-1}^{1}|n+i\rangle|n\rangle$ for $0\leq n<2^L$ (if $n+i=-1$ then it writes to the first register any value that is not a valid row index, e.g., $2^{L+1}-1$) on registers B, C. Then, we perform the oracle query, controlled rotation, and uncomputation similarly as done above to obtain

$$\frac{1}{\sqrt{3}}\sum_{i=-1}^{1}|0\rangle_A|n+i\rangle_B|n\rangle_C\left(k_{n+i,n}|0\rangle+\sqrt{1-|k_{n+i,n}|^2}|1\rangle\right), \tag{41}$$

where $k_{n+i,n}=0$ if $n+i$ is an invalid row index.

To summarize, the entire procedure from Equation 36 to Equation 40 (and Equation 41), denoted by U_R, performs the following

$$\begin{aligned}U_RP_R|0\rangle|0\rangle|n\rangle|0\rangle=&\frac{1}{\sqrt{3\alpha}}\Bigg[\sqrt{\beta_0}\sum_{i=-1}^{1}|0\rangle|n+i\rangle|n\rangle\left(k_{n+i,n}|0\rangle+\sqrt{1-|k_{n+i,n}|^2}|1\rangle\right)\\&+\sum_{\ell=2}^{L}\frac{\sqrt{\beta_{\ell-1}}}{\sqrt{2^{L-\ell}}}\sum_{m_b\in I(\ell,n)}\sum_{m_i=0}^{2^{L-\ell}-1}|\ell-1\rangle\underbrace{|m_b\rangle|m_i\rangle}_{\text{equiv. to some } m}|n\rangle\left(\frac{k_{mn}}{k_{\max}^{(\ell)}}|0\rangle+\sqrt{1-\left|\frac{k_{mn}}{k_{\max}^{(\ell)}}\right|^2}|1\rangle\right)\Bigg],\end{aligned} \tag{42}$$

where $\beta_0=3$ is the normalization factor of the adjacent level $\mathbf{K}_{ad}$ and $\beta_{\ell-1}=3\alpha_\ell$ is the normalization factor of the hierarchical level $\mathbf{K}^{(\ell)}$. Importantly, notice that register B has full support on all 2^L possible values of row index m in this superposition.

Implementing the "left" part We would like to transform the state

$$|0\rangle|0\rangle|m\rangle|0\rangle \tag{43}$$

to obtain similar result to that of the "right" procedure above. The procedure follows the same steps, except

- P_R needs to be replaced by P_L.
- $\mathcal{O}_r$ needs to be replaced by $\mathcal{O}_c$, the controlled block-column index oracle which, based on ℓ, m, computes the block-column indices n_b of the non-zero admissible blocks in $\mathbf{K}^{(\ell)}$ that intersect row m (there are at most 3 such blocks since $\mathbf{K}^{(\ell)}$ is 3-block-sparse) and writes the result n_b to the first $\ell+1$ qubits of register B. We denote this set of non-zero block-column indices n_b as $J(\ell,m)$.
- There is no need to query the entry oracle $\mathcal{O}_k$ (nor the accompanied controlled rotation).
- Registers B and C are swapped at the end.

Using similar notation to that of Equation 42, the above modifications give

$$\mathrm{SWAP}_{B,C}\, U_L P_L \left|0\right\rangle \left|0\right\rangle \left|m\right\rangle \left|0\right\rangle = \frac{1}{\sqrt{3\alpha}} \left[\sqrt{\beta_0} \sum_{i=-1}^{1} \left|0\right\rangle \left|m\right\rangle \left|m+i\right\rangle \left|0\right\rangle \right. \\ \left. + \sum_{\ell=2}^{L} \frac{\sqrt{\beta_{\ell-1}}}{\sqrt{2^{L-\ell}}} \sum_{n_b \in J(\ell,m)} \sum_{n_j=0}^{2^{L-\ell}-1} \left|\ell-1\right\rangle \left|m\right\rangle \underbrace{\left|n_b\right\rangle \left|n_j\right\rangle}_{\text{equiv. to some } n} \left|0\right\rangle \right]. \tag{44}$$

Combining both parts Combining Equation 42 with Equation 44 and noting that $\beta_0 = 3, \beta_{\ell-1} = 3 \cdot 2^{(L-\ell)(1-p)}$ and $k_{\max}^{(\ell)} = 2^{-(L-\ell)p}$, we get

$$\left\langle 0\right| \left\langle 0\right| \left\langle m\right| \left\langle 0\right| P_L^\dagger U_L^\dagger\, \mathrm{SWAP}_{B,C}\, U_R P_R \left|0\right\rangle \left|0\right\rangle \left|n\right\rangle \left|0\right\rangle = \frac{k_{mn}}{\alpha}, \tag{45}$$

which is exactly what we set out to prove (Equation 33).

Circuit complexity As seen above, this block-encoding only uses two calls to $\mathcal{O}_k$, and one call to each of $\mathcal{O}_r$, $\mathcal{O}_c$ and P_R, P_L. To achieve an overall error bound of ε, it suffices for the controlled rotation on $|\tilde{k}_{mn}\rangle \left|0\right\rangle$ to have error $O(\frac{\varepsilon}{N})$. This can be done with $O(b, \mathrm{polylog}(\frac{N}{\varepsilon}))$ extra qubits and one- and two-qubit gates (see explanation in proof of Lemma B.5).

Constructing state preparation pair P_L and P_R act on only $\log\log N$ qubits, hence were assumed given as unitaries. If this is not the case, we can still construct them (Equation 34) as block-encoded operators with only an extra normalization factor of $\log N$ as follows. Let

$$\begin{aligned} \mathcal{P} : |0^{\log L}\rangle \left|0\right\rangle &\to \frac{1}{\sqrt{L}} \sum_{\ell=0}^{L-1} \left|\ell\right\rangle \left|0\right\rangle \\ &\to \frac{1}{\sqrt{L}} \sum_{\ell=0}^{L-1} \left|\ell\right\rangle |\tilde{\beta}_\ell\rangle \left|0\right\rangle \qquad \text{(query } \beta_\ell \text{ into an appended register)} \\ &\to \frac{1}{\sqrt{L}} \sum_{\ell=0}^{L-1} \left|\ell\right\rangle \left(\sqrt{\frac{\beta_\ell}{\hat{\beta}}} \left|0\right\rangle + \sqrt{1 - \left|\frac{\beta_\ell}{\hat{\beta}}\right|} \left|1\right\rangle \right), \end{aligned} \tag{46}$$

where $\beta_0 = 3$ and $\beta_{\ell-1} = 3\alpha_\ell = 3 \cdot 2^{(L-\ell)(1-p)}$ for $2 \le \ell \le L$ and $\hat{\beta} = \max_\ell |\beta_\ell|$. As explained in Lemma B.5, the controlled rotation can be implemented with error bound ε using $O(\mathrm{polylog}(\frac{L}{\varepsilon}))$ extra one- and two-qubit gates. Comparing with Equation 34 we see that $\mathcal{P}$ is a $(\sqrt{\frac{\hat{\beta}L}{\alpha}}, 1, \varepsilon)$-block-encoding of P_R. Thus, using Lemma B.4 (multiplying block-encoded matrices) we can achieve Equation 45 with just two more ancilla qubits. The RHS of Equation 45 then becomes $\frac{k_{mn}}{\alpha} \left(\frac{\hat{\beta}L}{\alpha}\right)^{-1}$. In other words, the normalization factor will become a factor of $\frac{\hat{\beta}L}{\alpha}$ larger than the optimal block-encoding. We can bound this extra factor rather easily. As a reminder, α is given in Equation 32.

- For $p > 1$: $\hat{\beta} = 3$ and $\alpha \ge 3$, thus $\frac{\hat{\beta}L}{\alpha} \le L = \log N$.
- For $p = 1$: $\hat{\beta} = 3$ and $\alpha = 3\log N$, thus $\frac{\hat{\beta}L}{\alpha} = 1$.

- For $p < 1$: $\hat{\beta} = 3 \cdot 2^{(L-2)(1-p)} = O(N^{1-p})$ and $\alpha = \Omega(N^{1-p})$, thus $\frac{\hat{\beta} L}{\alpha} \leq O(L) = O(\log N)$.

Therefore, this extra factor only increases the runtimes of Lemma 5.1 (applying block-encoded matrices) and Lemma 5.2 (applying inverse of block-encoded matrices) by a factor of $O(\log N)$.

C.2 Variants of hierarchical splitting

The form of the hierarchical splitting can be slightly modified for other types of decaying kernels. For example, consider kernels of the form $k(x, x') = \frac{1}{|(x-x')+c|^p}$ on the domain $\Omega = [0, N-1)$ and $-N < c < N$ is an integer. In this case, we can use a "shifted" hierarchical splitting to obtain an optimal block-encoding of the kernel matrix $\mathbf{K} = (|i-j+c|^{-p})_{i,j=0}^{N-1}$, in which the hierarchy is shifted in the horizontal direction (along the rows, see Figure 3.) Similarly, for kernels of the form $k(x, x') = \frac{1}{(|x-x'|-c)^p}$ $(0 \leq c < N)$, we can apply a hierarchical splitting shifted along the skew-diagonal direction, such that the upper and lower triangular halves of the matrix are shifted in opposite directions.

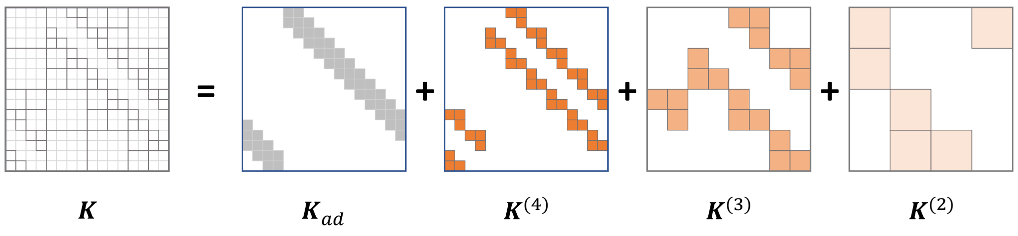

Figure 3: Hierarchical splitting corresponding to the kernel $k(x, x') = \frac{1}{|x-x'+4|^p}$ with uniform distribution of particles, which is obtained by shifting the splitting in Figure 1C 4 units to the right. Note that the fine splittings in the first 4 columns are actually only needed when the sum $x - x' + 4$ is modulo N.

C.3 Adaptive algorithm for non-uniform particle distribution

In particle simulation tasks, if the distribution of particles is not approximately uniform, an adaptive hierarchical splitting can be employed. Let the smallest interparticle spacing be N^{-c}, where N is the number of particles. Then, we can scale the mesh such that the interparticle distance is at least 1 and the mesh size is N^c. The added mesh sites are treated as zero-mass particles. The kernel matrix has size $N^c \times N^c$ and the hierarchical splitting constructed on this mesh has $c \log N$ levels. The entire block-encoding procedure presented in Section 4 applies, with only an extra factor of $\text{poly}(c)$ in the normalization factor and the required resources of the block-encoding. The optimality of the normalization factor of this adaptive block-encoding is harder to state precisely, since it depends on the distribution of particles which is not determined a priori.

C.4 Generalization of quantum hierarchical block-encoding

Here, we provide a block-encoding procedures in which one has direct access to the structure of the entry magnitudes, thus generalizing hierarchical block-encoding procedure presented in Section 4.1 and Appendix C.1.

First, we present a procedure for preparing a quantum state $|x\rangle \in \mathbb{C}^N$, where $N = 2^L$, with high probability of success. For any $0 \le \ell < L-1$, let $I_\ell = \{i : 2^{-(\ell+1)} < |x_i| \le 2^{-\ell}\}$ and $I_{L-1} = \{i : |x_i| \le 2^{-(L-1)}\}$. Also, let $n_\ell = |I_\ell|$. Apart from the oracle $\mathcal{O}_x : |i\rangle |0\rangle \to |i\rangle |\tilde{x}_i\rangle$, our procedure assumes the following oracle is provided:

$$\mathcal{O}_{index} : |\ell\rangle |j\rangle \to |0\rangle |f(\ell, j)\rangle, \qquad 0 \le j < n_\ell, \tag{47}$$

where the first register is $\log L$ qubits and the second register is $L = \log N$ qubits, and $f(\ell, j)$ is an efficiently computable and *reversible* function that computes the index of the j-th element in the set I_ℓ (according to some fixed ordering, e.g. top to bottom if $|x\rangle$ is treated as a column vector) if $0 \le j < n_\ell$, or else the oracle writes $N + (j - n_\ell + 1) + \sum_{l=0}^{\ell-1}(N - n_l)$ as a bit string on the concatenation of the two registers (so that the first register will be non-zero). This ensures the oracle is reversible. An example of such efficient and reversible $f(\ell, j)$ is in regular $\mathcal{H}$-matrix splitting. Our procedure is as follows:

$$\begin{aligned}
|0^{\log L}\rangle |0\rangle &\to \sum_{\ell=0}^{L-1} \frac{\sqrt{n_\ell} 2^{-\ell}}{\beta} |\ell\rangle |0\rangle, \qquad \text{where } \beta = \left(\sum_\ell n_\ell 2^{-2\ell}\right)^{1/2} \\
&\to \sum_{\ell=0}^{L-1} \frac{\sqrt{n_\ell} 2^{-\ell}}{\beta} |\ell\rangle \sum_{j=1}^{n_\ell} \frac{|j\rangle}{\sqrt{n_\ell}} \qquad \text{(Hadamards controlled on } \ell) \\
&\to \sum_{\ell=0}^{L-1} \sum_{j=1}^{n_\ell} \frac{2^{-\ell}}{\beta} |0\rangle |f(\ell, j)\rangle \qquad \text{(call oracle } \mathcal{O}_{index}) \\
&\to \sum_{\ell,j} \frac{2^{-\ell}}{\beta} |0\rangle |f(\ell, j)\rangle \left(\frac{x_{f(\ell,j)}}{2^{-\ell}} |0\rangle + \sqrt{1 - \left|\frac{x_{f(\ell,j)}}{2^{-\ell}}\right|^2} |1\rangle \right) \\
&\qquad \text{(query } x_{f(\ell,j)} \text{ and rotate ancilla qubit)}
\end{aligned}$$

Above, we have ignored the terms when j is invalid ($j \ge n_\ell$) in the sum because we can simply set $x_{f(\ell,j)} = 0$ for those cases. Furthermore, observe that the superposition has support on all indices. Thus, conditioning on the ancilla being zero, we obtain the desire state $|x\rangle$. The probability of successfully post-selecting is $\frac{1}{\beta^2}$, where

$$\beta^2 = \sum_{\ell=0}^{L-1} n_\ell 2^{-2\ell} = \sum_{\ell=0}^{L-2} n_\ell 2^{-2\ell} + n_{L-1} 2^{-2(L-1)} \le 4 \sum_i |x_i|^2 + \frac{4}{N} \le 8. \tag{48}$$

We also assumed that the first operation in the procedure, which maps $|0^{\log L}\rangle$ to a weighted superposition over $|\ell\rangle$, can be done perfectly. In fact, it can only be block-encoded with an extra normalization factor of $\sqrt{L}$: for a vector $|y\rangle \in \mathbb{C}^L$ one can simply do

$$\begin{aligned}
|0^{\log L}\rangle |0\rangle &\to \sum_{\ell=0}^{L-1} \frac{1}{\sqrt{L}} |\ell\rangle |0\rangle, \\
&\to \sum_{\ell=0}^{L-1} \frac{1}{\sqrt{L}} |\ell\rangle \left(y_\ell |0\rangle + \sqrt{1 - |y_\ell|^2} |1\rangle \right) \qquad \text{(query } y_\ell \text{ and rotate ancilla qubit)}
\end{aligned}$$

When including this step into the procedure above, the probability of successfully post-selecting $|00\rangle$ and obtaining $|x\rangle$ is $\frac{1}{L\beta^2} \ge \frac{1}{8 \log N}$.

Intuitively, our procedure exploits the structure of the amplitudes such that in the controlled rotation step, the ancilla has a large component in the good subspace $|0\rangle$.

The procedure to block-encode a matrix A follows similarly. For simplicity, assume A is Hermitian and $N^{-1} \le |A_{ij}| \le 1$. Let $I_\ell(j)$ be the set of row indices of the entries in column j whose values are in the range $[2^{-(\ell+1)}, 2^{-\ell})$. Let $n_\ell(j) = |I_\ell(j)|$, let $n_\ell = \max_j n_\ell(j)$. Let γ be such that $\min_j n_\ell(j) \ge \gamma n_\ell$. Our procedure, generalizing Appendix C.1, proceeds as follows:

The right part For any column index j:

$$\begin{aligned}
|0^{\log L}\rangle |0\rangle |j\rangle |0\rangle &\to \sum_{\ell=0}^{L-1} \frac{\sqrt{n_\ell 2^{-\ell}}}{\beta} |\ell\rangle |0\rangle |j\rangle |0\rangle, \qquad \text{where } \beta = \left(\sum_\ell n_\ell 2^{-\ell}\right)^{1/2} \\
&\to \sum_{\ell=0}^{L-1} \frac{\sqrt{n_\ell 2^{-\ell}}}{\beta} |\ell\rangle \sum_{i=1}^{n_\ell} \frac{|i\rangle}{\sqrt{n_\ell}} |j\rangle |0\rangle \qquad \text{(Hadamards controlled on } \ell) \\
&\to \sum_{\ell=0}^{L-1} \sum_{i=1}^{n_\ell} \frac{\sqrt{2^{-\ell}}}{\beta} |0\rangle |f_j(\ell,i)\rangle |j\rangle |0\rangle \qquad \text{(call oracle } \mathcal{O}_{index}) \\
&\to \sum_{\ell; i \in [n_\ell(j)]} \frac{\sqrt{2^{-\ell}}}{\beta} |0\rangle |f_j(\ell,i)\rangle |j\rangle \left(\frac{A_{f_j(\ell,i),j}}{2^{-\ell}} |0\rangle + \sqrt{1 - \left| \frac{A_{f_j(\ell,i),j}}{2^{-\ell}} \right|^2} |1\rangle \right) \\
&\qquad \text{(query } A_{f_j(\ell,i),j} \text{ and rotate ancilla qubit)}
\end{aligned} \tag{49}$$

Here, $\mathcal{O}_{index}$ is similar to that in Equation 47; it returns the row index, $f_j(\ell, i)$, of the i-th element in $I_\ell(j)$. We ignore the terms when i is invalid ($i \ge n_\ell(j)$) as we can set $A_{f_j(\ell,i),j} = 0$ for those terms. Importantly, the above superposition has support over all row indices.

The left part Similarly, we can construct a unitary that queries all the column indices

$$|0^{\log L}\rangle |0\rangle |i\rangle |0\rangle \to \sum_{\ell; j \in [n_\ell(i)]} \frac{\sqrt{2^{-\ell}}}{\beta} |0\rangle |i\rangle |g_i(\ell, j)\rangle |0\rangle, \tag{50}$$

where $g_i(\ell, j)$ is the column index of the j-th element in $I_\ell(i)$.

Combining Equation 49 and Equation 50 we obtain a unitary U that is block-encodes A with a normalization factor of β^2. If we take into consideration of first step Equation 49 (i.e. preparing $\sum_{\ell=0}^{L-1} \frac{\sqrt{n_\ell 2^{-\ell}}}{\beta} |\ell\rangle$) as did in the quantum state preparation procedure above, then the normalization factor is $\beta^2 \log N$. We now compare β^2 with the operator norm $\|A\|$. If the entries of A are real and non-negative, we have that

$$\|A\| \ge \|A\,|\mathbf{1}\rangle\,\| \ge \sum_\ell \min_j n_\ell(j) 2^{-(\ell+1)} \ge \frac{\gamma}{2}\beta^2. \tag{51}$$

Thus, compared to the optimal block-encoding (one that has normalization factor equal to $\|A\|$), the presented block-encoding for A is worse by at most a factor of $2(\log N)/\gamma$. In other words, if $1/\gamma = O(\mathrm{polylog}(N))$, then the normalization factor is only a factor of $\mathrm{polylog}(N)$ worse than that of the optimal block-encoding. As before (appendix C.1), the numbers of elementary gates for the controlled rotation step and ancilla qubits are $O(\mathrm{polylog}(\frac{N}{\varepsilon}))$.

C.5 Condition number of polynomially and exponentially decaying kernels

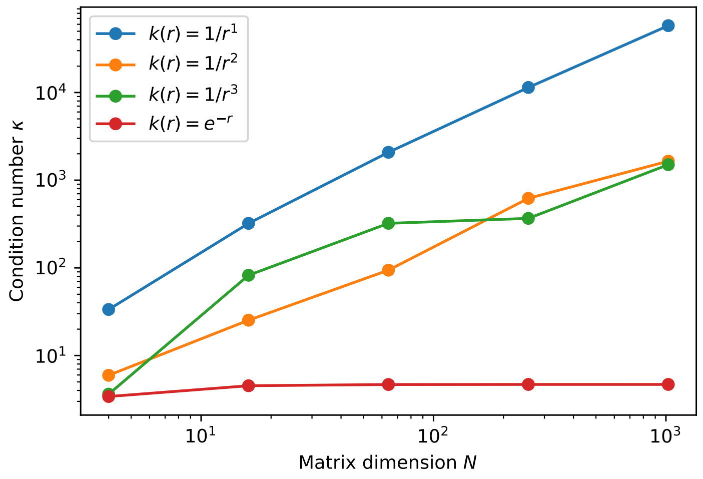

Figure 4: The condition number of polynomially decaying kernels grow polynomially with the matrix dimension. Whereas, the condition number of the exponentially decaying kernel converges. Here, the kernel matrices take the form $\mathbf{K} = (k(|x_i - x_j|))_{i,j=0}^{N-1}$, where $x_i = i$, $k(|x - x'|) = k(r)$, and $k(0) = 0$ (no self-interactions). We remark that, the condition number can be dramatically improved if the self-interacting terms are sufficiently large. For example, for $k(0) = 2$ we found that the condition number for $k(r) = 1/r$ grows only logarithmically in N, while those of the other kernels scale as $O(1)$. This can be explained using similar calculations to Equation 26.

C.6 Kernels for which the naive block-encoding approach (Lemma 3.2) works optimally

We provide examples of kernels that are studied in classical $\mathcal{H}$-matrix or fast multipole method literature where the naive block-encoding approach of Lemma 3.2 (or Remark B.6) can be directly used to produce an optimal block-encoding of their kernel matrices.

The log kernel $k(x, x') = \log |x - x'|$ (chapter 1, [54]). We use the same setup as in the main text. Let the kernel matrix be $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$, where $x_i = i$, $N = 2^L$, and $k(x, x) = 0$. Directly applying Lemma 3.2 yields a $(N \log N, \log N + 1, \varepsilon)$-block-encoding using only $O(\operatorname{polylog} \frac{N}{\varepsilon})$ additional resources. To show that this block-encoding is optimal, we bound the operator norm from below: $\|\mathbf{K}\| \geq \|\mathbf{K}\,|\mathbf{1}\rangle\,\| \geq \int_1^{N/2} \log x dx = \Omega(N \log N)$.

The multiquadric kernel $k(x, x') = \sqrt{c^2 + |x - x'|^2}$ (chapter 2.1, [13]). For this, let the kernel matrix be $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$, where $x_i = i/N$, $N = 2^L$. Directly applying Lemma 3.2 yields a $(2N, \log N + 1, \varepsilon)$ using only $O(\operatorname{polylog} \frac{N}{\varepsilon})$ additional resources. The operator norm of $\mathbf{K}$ is bounded as $\|\mathbf{K}\| \geq \|\mathbf{K}\,|\mathbf{1}\rangle\,\| \geq N \int_0^{1/2} \sqrt{c^2 + x^2} dx = \Omega(N)$, implying that the block-encoding is optimal.

We also list here, omitting the proofs, some typical kernels (which are not considered in classical $\mathcal{H}$-matrix) that can be optimally block-encoded by the naive approach of Lemma 3.2. For example, these include polynomially growing kernels $k(x, x') = |x - x'|^p$ and polyharmonic radial basis functions $k(x, x') = |x - x'|^p \log |x - x'|$. Our numerical experiments indicate that these kernels are numerically low-rank, as shown in Figure 5. Furthermore, the first principal component (singular vector corresponding to the largest

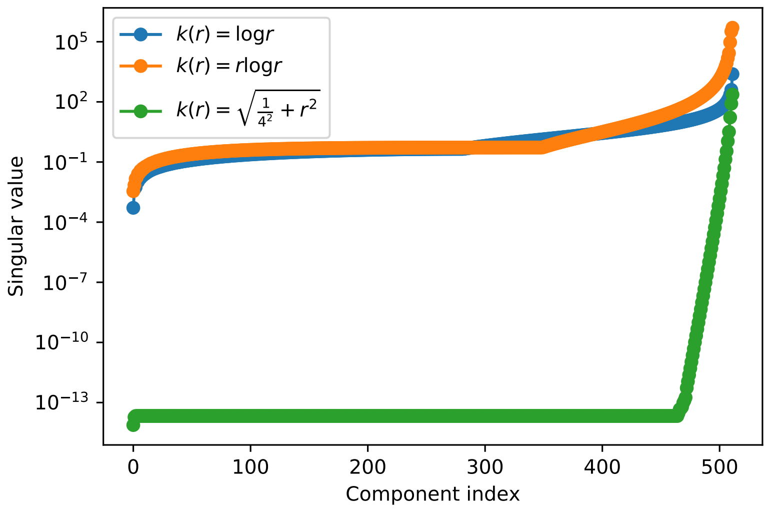

Figure 5: Singular values of the 512 by 512 kernel matrices $\mathbf{K} = (k(x_i, x_j))_{i,j=0}^{N-1}$ of the log, polyharmonic, and multiquadric kernels. Here $r = |x - x'|$, $x_j = j/N$ for the multiquadric kernel ($k(r) = \sqrt{1/4^2 + r^2}$), and $x_j = j$ for the other kernels. The plots indicate that all kernels are numerically low-rank.

singular value) is close to uniform (the normalized all-ones vector). This explains why these kernels can be block-encoded by the naive block-encoding.

D Classical $\mathcal{H}$-matrices

D.1 Low-rank approximation using asymptotic smoothness

In this section, we show how the asymptotic smoothness condition (Equation 6) can be used to approximate an admissible block as a low-rank matrix. As a reminder, the asymptotic smoothness condition is

$$|\partial_{x'}^p k(x, x')| \leq Cp!|x - x'|^{-p} \qquad (\forall p \in \mathbb{N}).$$

For instance, the log kernel $k(x, x') = \log(|x - x'|)$ directly holds this property.

Consider the admissible block $\mathbf{K}^{\sigma,\rho} = (k(x_i, x_j))_{i\in\sigma, j\in\rho}$, where $\sigma, \rho \in \mathcal{P}^{(\ell)}$. Let c_ρ be the center of ρ, we first obtain the p-th order Taylor expansion of $k(x, x')$ around $x' = c_\rho$

$$k(x, x') = \sum_{q=0}^{p-1} \frac{(x' - c_\rho)^q}{q!} \partial_{x'}^q k(x, c_\rho) + R(x, x'), \tag{52}$$

where the remainder $R(x, x')$ can be bounded using the asymptotic smoothness property as follows

$$\begin{aligned} k(x, x') &= \sum_{q=p}^{\infty} \frac{(x' - c_\rho)^q}{q!} \partial_{x'}^q k(x, c_\rho) \leq C \sum_{q=p}^{\infty} \left(\frac{x' - c_\rho}{|x - c_\rho|} \right)^q \\ &\leq C \sum_{q=p}^{\infty} \left(\frac{r_\rho}{\operatorname{dist}(\sigma, \rho)} \right)^q \leq C \sum_{q=p}^{\infty} \left(\frac{r_\rho}{2r_\rho} \right)^q = O(2^{-p}). \end{aligned} \tag{53}$$

Here, we have used the admissibility criteria (Definition 2.1) in the third inequality and that $r_\sigma = r_\rho$. Thus, $\mathbf{K}^{\sigma,\rho}$ can be approximated with exponentially small error by a rank-p

matrix

$$\mathbf{K}^{\sigma,\rho} \approx \widetilde{\mathbf{K}}^{\sigma,\rho} = \mathbf{\Psi}^{\sigma,\rho}\mathbf{D}\left(\mathbf{\Phi}^{\rho}\right)^{\dagger}, \tag{54}$$

where $\mathbf{D} = \mathrm{Diag}(1/q!)_{q\in P} \in \mathbb{R}^{p\times p}$, $P = \{0, 1, \ldots p-1\}$ and

$$\mathbf{\Psi}^{\sigma,\rho} = (\partial^q_{x'} k(x_i, c_\rho))_{i\in\sigma, q\in P} \in \mathbb{R}^{|\sigma|\times p}, \tag{55}$$

$$\mathbf{\Phi}^{\rho} = ((x'_j - c_\rho)^q)_{j\in\rho, q\in P} \in \mathbb{R}^{|\rho|\times p}. \tag{56}$$

Next, we give an example of a kernel that does not directly satisfy the asymptotic smoothness condition (Equation 6), but can still be manipulated to apply the low-rank approximation. Consider the kernel $k(x, x') = |x - x'|^{-2}$. This kernel does not directly satisfy Equation 6. Nevertheless, for an admissible block $\mathbf{K}^{\sigma,\rho} = (k(x_i, x_j))_{i\in\sigma, j\in\rho}$ at level ℓ, the kernel can be rewritten as

$$\begin{aligned} k\left(x - x'\right) &= k\left(x - c_\rho + c_\rho - x'\right) = k\left((x - c_\rho)\left(1 + \frac{c_\rho - x'}{x - c_\rho}\right)\right) \\ &= k\left(x - c_\rho\right) k\left(1 + \frac{c_\rho - x'}{x - c_\rho}\right) = k\left(x - c_\rho\right) \sum_{q=0}^{\infty} \frac{k^{(q)}(1)}{q!}\left(\frac{c_\rho - x'}{x - c_\rho}\right)^q, \end{aligned} \tag{57}$$

where c_ρ is the center of ρ. In our setting (see Section 2), c_ρ is simply the middle point of the corresponding segment on the domain Ω. Furthermore, noting that $k^{(q)}(1) = (-1)^q(q+1)!$, let

$$\begin{aligned} A_q\left(x - c_\rho\right) &= \frac{k\left(x - c_\rho\right)}{\left(x - c_\rho\right)^q}, \\ S_q\left(c_\rho - x'\right) &= (-1)^q(q+1)\left(c_\rho - x'\right)^q, \end{aligned} \tag{58}$$

and

$$k_p(x - x') = \sum_{q=0}^{p} A_q\left(x - c_\rho\right) S_q\left(c_\rho - x'\right). \tag{59}$$

Since $\mathbf{K}^{\sigma,\rho}$ is an admissible block (Definition 2.1), we have that $\left|\frac{c_\rho - x'_j}{x_i - c_\rho}\right| \leq \frac{1}{2}, \forall i \in \sigma, j \in \rho$. Therefore,

$$\begin{aligned} |k(x_i, x_j) - k_p(x_i, x_j)| &\leq (\sup_{x,x'\in\Omega} |k(x, x')|)\left(\sum_{q=p+1}^{\infty} (q+1)\left|\frac{c_\rho - x'}{x - c_\rho}\right|^q\right) \\ &\leq \sum_{q=p+1}^{\infty} (q+1)2^{-q} = 2^{-p}(2p+4). \end{aligned} \tag{60}$$

Thus, $k(x_i, x_j)$ can be approximated with error ε/N by the truncated sum in Equation 59 of order $p = O(\mathrm{polylog}\, N/\varepsilon)$. Accordingly, the admissible block $\mathbf{K}^{\sigma,\rho}$ can be approximated to error ε by a rank-p matrix.

D.2 Classical fast $\mathcal{H}$-matrix-vector multiplication

We count the number of operations needed to compute the matrix-vector multiplication, where $\mathbf{K} \in \mathbb{R}^{N\times N}$ and $\mathbf{v} \in \mathbb{R}^N$,

$$\mathbf{K}\mathbf{v} = \left(\sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}\right)\mathbf{v} = \sum_{\ell=2}^{L} \mathbf{u}_\ell + \mathbf{u}_{ad}. \tag{61}$$

The adjacent term $\mathbf{K}_{ad}$ is 3-sparse, hence requires only $O(N)$ operations. In level ℓ, each block-row (row of blocks) requires computing 3 matrix-vector multiplications on dense matrices of size $N \cdot 2^{-\ell}$, which takes $O(pN2^{-\ell})$ operations (where p is the rank of the admissible blocks). There are 2^ℓ block-rows, thus computing each $\mathbf{K}^{(\ell)}$ requires $O(pN)$ operations. Since there are $L-1 = O(\log N)$ levels, the total operation complexity to compute all $\mathbf{u}_\ell$ is $O(pN \log N)$. Adding up $\mathbf{u}_\ell$ and $\mathbf{u}_{ad}$ costs another $O(N \log N)$. As we have shown above, it is required that $p = O(\text{polylog}\,\frac{N}{\varepsilon})$ to achieve an error bound of ε on $\mathbf{K}$. Thus, the overall runtime of matrix-vector multiplication on $\mathcal{H}$-matrices is $O(N\,\text{polylog}\,\frac{N}{\varepsilon})$. We refer to [16] for operation complexities of other tasks on $\mathcal{H}$-matrices, such as matrix-matrix multiplication and matrix inversion, which are also $O(N\,\text{polylog}\,\frac{N}{\varepsilon})$ when $p = O(\text{polylog}\,\frac{N}{\varepsilon})$.

D.3 High dimensional hierarchical splittings

In the main text, we have seen that two key properties of the 1D hierarchical splitting that enables optimal quantum block-encodings were the logarithmic number of hierarchical levels and the block sparseness of each level (the third key property was that the off-diagonal entries were decaying or sufficiently smooth). Here, we show that these properties still hold for the 2D case and make a generalization argument for higher dimensional cases.

The 2D and 3D hierarchical splittings are well-studied in classical literature [17]. We will follow the indexing rules and conventions in the original work [17]. For an overview of hierarchical matrices and splittings, we suggest interested readers to read [13]–which covers the fast multipole algorithm (a special case of hierarchical splittings), [16]–which is the original paper that introduced the 1D hieararchical splitting, and [17]–the follow-up paper that generalized the hierarchical splittings to 2D and 3D problems.

Consider the uniform 2D grid of size $[1,N] \times [1,N]$ and $N = 2^L$, where unit-mass particles are placed at sites (i,j) (hence there are N^2 particles). We define the hierarchical partitions for this grid in the same spirit as Equation 3. For level $\ell < L$, define the partitioning to be

$$\mathcal{P}^{(\ell)} = \{\sigma^{(\ell)}_{I,J} | I, J \in \{0, \dots, 2^\ell - 1\}]\}, \tag{62}$$

where the clusters $\sigma^{(\ell)}_{I,J}$ are

$$\sigma^{(\ell)}_{I,J} = \{(i,j) | i \in [2^{L-\ell} I, 2^{L-\ell}(I+1)), j \in [2^{L-\ell} J, 2^{L-\ell}(J+1))\}. \tag{63}$$

Before we proceed to construct the hierarchical splitting of the kernel matrix, we first have to decide how to number particles in a 2D grid. We follow the "canonical" choice in Section 4 of [17] (see Figure 6). In this numbering rule, the conversion from a grid site (i,j) to the corresponding vectorized index m is given by

$$(i_1 \dots i_L, j_1 \dots j_L) \to m(i,j) = \sum_{\ell=1}^{L} f(i_\ell, j_\ell) 4^{L-\ell}, \tag{64}$$

where $i_1 \dots i_L$ $(j_1 \dots j_L)$ is the binary representation of the row (column) index which ranges from 0 to $2^L - 1$ and f is the CNOT function whose target bit is the second: $f(0,0) = 0$, $f(0,1) = 1$, $f(1,0) = 3$, $f(1,1) = 2$. The conversion from the vectorized index K to grid site (i,j) can be done by first writing m in base-4 representation, then inverting the function f. In summary, the entries of the kernel matrix are of the form

$$\mathbf{K}_{m(i_1,j_1),m(i_2,j_2)} = k((i_1,j_1),(i_2,j_2)). \tag{65}$$

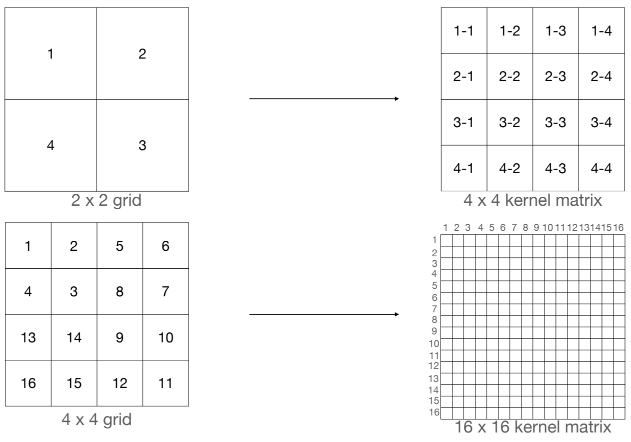

Figure 6: Left: 2D grid site numbering rule for vectorizing input. Right: the associated kernel matrix representing pairwise interactions based on this numbering system.

Next, we also need to define the admissibility criteria (Definition 2.1) for now 2D clusters. For two clusters in the same level, *e.g.*, $\sigma_{I,J}^{(\ell)}$ and $\sigma_{I',J'}^{(\ell)}$, wee define their diameters to be the length of their diagonal, which are easily seen to be $\operatorname{diam}(\sigma_{I,J}^{(\ell)}) = \sqrt{2}\cdot 2^{L-\ell}$. Define the distance between these two clusters to be $\operatorname{dist}(\sigma_{I,J}^{(\ell)}, \sigma_{I',J'}^{(\ell)}) = 2^{L-\ell}\cdot \sqrt{\max\{0,|I-I'|-1\}^2+\max\{0,|J-J'|-1\}^2}$. We say the interactions between these two clusters are admissible if and only if

$$\operatorname{dist}(\sigma_{I,J}^{(\ell)}, \sigma_{I',J'}^{(\ell)}) \geq \eta \max\{\operatorname{diam}(\sigma_{I,J}^{(\ell)})\, \operatorname{diam}(\sigma_{I',J'}^{(\ell)})\}, \tag{66}$$

where we choose $\eta = \frac{1}{\sqrt{2}}$.

In other words, two clusters (hence the block in the kernel matrix $\mathbf{K}$ that contains the interactions between them) are admissible when they are not neighboring along a column, row, or diagonal.

Having established the admissibility criteria, we proceed similarly to Section 2 to obtain the following decomposition of the kernel matrix $\mathbf{K}$

$$\mathbf{K} = \sum_{\ell=2}^{L} \mathbf{K}^{(\ell)} + \mathbf{K}_{ad}, \tag{67}$$

where $\mathbf{K}^{(\ell)}$ contains the admissible interactions in level ℓ that have *not* been included in $\mathbf{K}^{(\ell-1)}$.

Next, we calculate the block-sparsity of $\mathbf{K}^{(\ell)}$ by counting, among the 4^ℓ clusters in level ℓ, how many clusters contribute to the potential at the particles inside a particular cluster $\sigma_{I,J}^{(\ell)}$. In fact, at most $6^2-3^2=27$ level-ℓ clusters need to be included. This can be most easily observed in Figure 7. Thus, the row-block-sparsity of $\mathbf{K}^{(\ell)}$ is 27. And since $\mathbf{K}^{(\ell)}$ is symmetric, the column-block-sparsity is also 27.

In summary, we still have $\log N$ hierarchical levels, each of which is block-sparse in the 2D case. Therefore, we can apply the entire block-encoding procedure in Section 4 to block-encode kernel matrices of a 2D grid. Below, we provide a similar analysis to that of Section 4.1 for polynomially decaying kernels ($k(\mathbf{x},\mathbf{x}') = \|\mathbf{x}-\mathbf{x}'\|^{-p}$), showing that the hierarchical block-encoding is optimal for the 2D case.

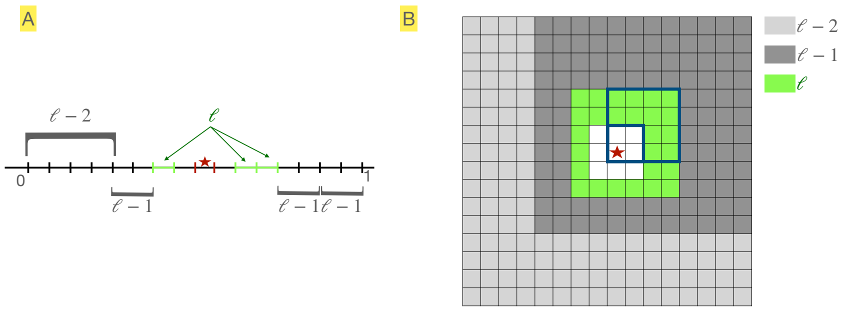

Figure 7: (A) shows the structure of the clusters in level ℓ (here $\ell = 4$) of a 1D hierarchical splitting. Since most clusters have been taken into account in previous levels of the hierarchy, at most $6-3=3$ of them (green) will contribute to the potential at the target (red star). Similarly, (B) shows the structure of the clusters in level ℓ ($\ell = 4$) of a 2D hierarchical splitting. Since most clusters have been included in previous levels of the hierarchy, at most $6^2-3^2=27$ of them (green) will contribute to the potential at the target.

At level ℓ ($2 \leq \ell \leq L$), an admissible matrix block $\mathbf{K}^{\sigma,\rho}$, which contains the interactions between two clusters σ, ρ at level ℓ, has size $4^{L-\ell} \times 4^{L-\ell}$; and the minimum pairwise distance between particles within this block is $2^{L-\ell}$ by the definition of admissibility (Equation 66). It follows that the maximum entry in $\mathbf{K}^{\sigma,\rho}$ is $2^{-(L-\ell)p}$. Thus, $\mathbf{K}^{\sigma,\rho}$ can be block-encoded by Lemma 3.2 with a normalization factor of $\alpha_\ell = 2^{(L-\ell)(2-p)}$. Then, $\mathbf{K}^{(\ell)}$ can be block-encoded by Lemma 3.5 with a normalization factor of $27\alpha_\ell$. The adjacent interaction part, $\mathbf{K}_{ad}$, is a sparse matrix with sparsity 9 and the largest entry equal to 1. Thus, $\mathbf{K}_{ad}$ can be block-encoded with a normalization factor of 9 using Lemma B.5. Finally, the entire kernel matrix $\mathbf{K}$ is obtained by summing over all levels using Lemma B.3. The overall normalization factor is

$$\alpha = 9 + \sum_{\ell=2}^{L} 27\alpha_\ell = \begin{cases} 9 + 27\frac{2^{2-p}-N^{2-p}}{2^{2-p}-4^{2-p}} \leq O(1) & \text{if } p > 2 \\ 9 + 27(\log N - 1) & \text{if } p = 2 \\ 9 + 27\frac{N^{2-p}-2^{2-p}}{4^{2-p}-2^{2-p}} & \text{if } p < 2 \end{cases}. \tag{68}$$

This block-encoding can be shown to be optimal by a similar method to Remark 4.1. We have that $\|\mathbf{K}\| \geq \|\mathbf{K}\,|\mathbf{1}\rangle\|$, where $|\mathbf{1}\rangle$ is the normalized all-ones vector. Furthermore, observe that

$$\begin{aligned} \|\mathbf{K}\,|\mathbf{1}\rangle\| &\geq \int_1^N \int_1^N (x^2+y^2)^{-p/2} dx dy \\ &\geq \int_0^{\pi/2} \int_{\sqrt{2}}^{\sqrt{N^2+1}} r^{-p} r dr d\theta - 2\int_1^{\sqrt{N^2+1}} (x^2+1)^{-p/2} dx. \end{aligned} \tag{69}$$

In the limit $N \to \infty$, it can be easily verified that the above integral is $O(N^{2-p})$ when $p \neq 2$ and $O(\log N)$ when $p = 2$. Therefore, $\alpha = \Theta(\|\mathbf{K}\|)$ and the block-encoding procedure using hierarchical splitting is optimal.

Finally, the analysis in this section can be generalized to any d-dimensional setting, where the block-sparsity of each hierarchical level $\mathbf{K}^{(\ell)}$ is $6^d - 3^d$ and the sparsity of the adjacent part $\mathbf{K}_{ad}$ is 3^d. For a visualization of 2D and 3D hierarchical splittings, see Figure 6 of [55], which has a smaller block-sparsity than the aforementioned value since diagonally neighboring clusters are considered admissible by the authors of [55], as opposed to our treatment in this section.

E Quantum state preparation for smooth functions

We provide an efficient procedure to prepare a quantum state whose entries are sampled from a smooth function. Consider a smooth function g bounded by $|g(x)| \leq 1$ in the domain $\Omega = [0, 2\pi]$ which can be approximated by a low-order Fourier series

$$g(x_j) = \sum_{n=-p}^{p} c_n e^{2\pi i n j/N}, \tag{70}$$

where $p = O(1)$ and the coefficients c_n are $\Theta(1)$.

We want to prepare the quantum state $|\mathbf{g}\rangle$ where the vector $\mathbf{g}$ is

$$\mathbf{g}_j = g(2\pi j/N) \tag{71}$$

Let the QFT matrix be

$$F_N = \sum_{k,j=0}^{N-1} \frac{1}{\sqrt{N}} e^{-2\pi i jk/N} |k\rangle \langle j| \,. \tag{72}$$

Let $\widetilde{\mathbf{g}} = \begin{bmatrix} c_0 \\ \vdots \\ c_{N-1} \end{bmatrix}$, where at most $d = 2p+1$ terms c_n are non-zero, and $c_{N-k} = c_{-k}$ for $1 \leq k \leq p$.

Observe that

$$\mathbf{g} = \frac{F_N}{\sqrt{N}} \widetilde{\mathbf{g}}. \tag{73}$$

Since $\widetilde{\mathbf{g}}$ is sparse and the values and locations of the non-zero entries are known, one can easily prepare its quantum version $|\widetilde{\mathbf{g}}\rangle$ with high probability using the following procedure.

1. Prepare $|0\rangle \to \frac{1}{\sqrt{d}} \sum_{k=0}^{d-1} |k\rangle \to \frac{1}{\sqrt{d}} \sum_{j:c_j \neq 0} |j\rangle$
2. Call the oracle $\mathcal{O}_c : |j\rangle |0\rangle \to |j\rangle |\tilde{c}_j\rangle$ (where $\tilde{c}_j$ is a qubit string description of c_j)
3. Apply the conditioned rotation $|j\rangle |\tilde{c}_j\rangle |0\rangle \to \frac{c_j}{\max_j c_j} |j\rangle |\tilde{c}_j\rangle |0\rangle + \sqrt{1 - \left|\frac{c_j}{\max_j c_j}\right|^2} |j\rangle |\tilde{c}_j\rangle |1\rangle$
4. Uncompute and post-select on the subspace $|0\rangle$ of the third register to obtain $|\widetilde{\mathbf{g}}\rangle$

Upon successfully obtaining $|\widetilde{\mathbf{g}}\rangle$ (which happens with probability $\Omega(1)$), one applies the QFT to obtain $|\mathbf{g}\rangle = F_N |\widetilde{\mathbf{g}}\rangle$. This procedure clearly applies for higher dimensional smooth functions. In these cases, one uses a tensor product of the conventional QFTs to transform the state back and forth between the real regime and the Fourier regime.