Double-Logarithmic Depth Block-Encodings of Simple Finite Difference Method's Matrices

Sunheang Ty, Renaud Vilmart, Axel Tahmasebimoradi, Chetra Mang

DOUBLE-LOGARITHMIC DEPTH BLOCK-ENCODINGS OF SIMPLE FINITE DIFFERENCE METHOD'S MATRICES

Sunheang TY
IRT SystemX
Gif-sur-Yvette, France
sunheang.ty@irt-systemx.fr

Renaud Vilmart
Université Paris-Saclay
ENS Paris-Saclay, CNRS, Inria, LMF
Gif-sur-Yvette, France
renaud.vilmart@inria.fr

Axel TahmasebiMoradi
IRT SystemX
Gif-sur-Yvette, France
a.tahmasebimoradi@irt-systemx.fr

Chetra Mang
IRT SystemX
Gif-sur-Yvette, France
chetra.mang@irt-systemx.fr

October 3, 2024

ABSTRACT

Solving differential equations is one of the most computationally expensive problems in classical computing, occupying the vast majority of high-performance computing resources devoted towards practical applications in various fields of science and engineering. Despite recent progress made in the field of quantum computing and quantum algorithms, its end-to-end application towards practical realization still remains unattainable. In this article, we tackle one of the primary obstacles towards this ultimate objective, specifically the encoding of matrices derived via finite difference method solving Poisson partial differential equations in simple boundary-value problems. To that end, we propose a novel methodology called *block-diagonalization*, which provides a common decomposition form for our matrices, and similarly a common procedure for block-encoding these matrices inside a unitary operator of a quantum circuit. The depth of these circuits is double-logarithmic in the matrix size, which is an exponential improvement over existing quantum methods and a superexponential improvement over existing classical methods. These improvements come at the price of a constant multiplicative overhead on the number of qubits and the number of gates. Combined with quantum linear solver algorithms, we can utilize these quantum circuit to produce a quantum state representation of the solution to the Poisson partial differential equations and their boundary-value problems.

Keywords Double-Logarithmic Depth Block-Encodings, Quantum Linear Solver, Finite Difference Method

1 Introduction

Differential equations are ubiquitous in the vast majority of science and engineering fields, with an extensive number of practical applications such that a significant research-and-development in high-performance computing field are devoted towards solving these problems. Unsurprisingly, there have been several proposals for the uses of quantum computing to solve such important problems. There are several different categories and classes of differential equations depending on the fields and applications of the problem. In this article, we are interested in linear partial differential equations [1], which constitute the majority of practical applications. More specifically, we are interested in the boundary-value problems comprising a second-order linear partial differential equation called Poisson's equation within hypercube domains and a number of different boundary conditions. And although, such problems might be solved via analytical method, we are interested in the solution via a numerical method called finite difference [2–4], as it can serve as the foundation for other numerical methods such as finite element method and finite volume method. Basically, finite difference method transforms the boundary-value problem, a continuous problem, into a system of linear equations, a

discrete problem, by discretizing the continuous domain into a collection of grid points, and approximating the partial derivative of the solution locally at each grid point. The derived system of linear equations is generally sparse and can be solved using classical linear solver algorithms [5, 6], such as conjugate gradient method. Its time complexity is linear in the number of grid points or the matrix size, and logarithmic in the inverse of solution error. Note that the number of grid points, and therefore the matrix size, is related to the discretization error introduced by the method; thus, the time complexity is dependent on both errors. The solution from the linear solver is the discretized solution of the differential equation at each grid point. Generally, for practical applications, the number of grid points required are significant, such that high-performance computing resources are required; thus, providing the motivation for further research.

Recently, with the advances of quantum computing [7], and quantum algorithms [8] in particular, there have been a surge of research-and-development focusing on the application of these advances towards some of the most computationally expensive problems in classical computing. These include problems in the field of condensed matter physics, nuclear and particle physics, quantum chemistry, differential equations, combinatorial optimization, continuous optimization, and machine learning. Solving a single problem in a particular field generally requires a combination of several quantum algorithmic primitives to compose an overall quantum algorithm, and incorporate into it quantum error correction and fault tolerance in order to correctly implement it on a quantum computer. In this article, we are interested in the quantum algorithmic primitives for solving differential equations, in particular those by means of quantum linear solver algorithms [9–22], i.e., those that transformed differential equations into a system of linear equations, such as finite difference method. These algorithms [9–22] have been improved over the years using various different techniques, from quantum phase estimation to quantum singular value transformation. Currently, the best known quantum linear solver is [18] using the adiabatic theorem, whose time complexity is linear in the matrix's condition number and logarithmic in the inverse of solution error. Denote $\mathcal{L}\left|u\right\rangle = \left|f\right\rangle$ as the system of linear equations. It requires two quantum primitive inputs: a block-encoding quantum circuit of $\mathcal{L}$, and a quantum state preparation of $\left|f\right\rangle$, and it produces a quantum state representation of $\left|u\right\rangle$ as an output. To take full advantage of this algorithm, five efficient quantum algorithmic primitives are required [23]: a quantum linear solver itself, a quantum preconditioner, an encoding of $\mathcal{L}$, an encoding of $\left|f\right\rangle$, and a decoding of quantum state representation of $\left|u\right\rangle$. Each of these five primitives presents its own challenges, and can depend upon the intended applications. In this article, we are tackling the third obstacle: the encoding of matrices derived via finite difference method solving Poisson partial differential equations in boundary-value problems, with the hope that it can play an important role in the ultimate objective of an end-to-end application of quantum computing.

Main Contributions The main contribution of this article is the *block-diagonalization* methodology, which describes a common decomposition form for the matrices derived by the discretization via finite difference method of boundary-value problems described by a Poisson partial differential equation in a hypercube domain of an arbitrary dimension, and four different boundary conditions: periodic, Dirichlet, Neumann, and Robin, which are typical study cases in both academic and industry. The primary benefit of this common decomposition form is that it also leads to a common procedure for the construction of quantum circuits block-encoding each of these matrices inside a unitary operator. Thus, avoiding the need to construct a quantum oracle which can compute the entries of the matrices given its row and column position; such oracle is much harder to construct. Another main benefit of block-diagonalization is that their block-encoding quantum circuits are simple. In fact, they are simple enough such that we are able to derive their simplifications directly without using any sophisticated circuit optimization software. Moreover, the simplified quantum circuits comprise only a few ancilla qubit, elementary quantum gates and circuits. Using existing quantum gate and circuit implementation techniques, especially the families of classical reversible circuits including arithmetic circuits, we are able to show that these elementary gates and circuits can be implemented in logarithmic depth in the number of qubit. These come at the price of a constant multiplicative overhead on the number of qubits and gates. Therefore, the implementations of our block-encoding quantum circuits also have double-logarithmic depth in the matrix size.

Unfortunately, existing quantum methods, such as the aforementioned quantum oracle, have polylogarithmic depth, while existing classical methods have linear depth. Combine these with quantum and classical linear solver algorithms, we find our method's dependencies to be exponentially better than existing quantum methods, and superexponentially better than existing classical methods in terms of matrix size, for solving our boundary-value problems. However, note that the matrix size has a dependency in the discretization error. Assuming this error is of the same order as the solution error of the quantum linear solver, our method is still polynomially and exponentially better than existing quantum and classical methods, respectively. These improvements are based on the assumption that the quantum state preparation subroutine has at most the same complexity as our block-encoding. Another caveat is that like all quantum methods, we keep the solution as a quantum state, rather than a classical state; otherwise, most of the quantum speedup are lost.

Content Outline This article is divided into six sections. Section 1 introduces the state-of-the-art in the uses of quantum computing for solving differential equations, by presenting the advantages of quantum computing, and identifying the challenges of its application to this problem domain. It also describes the main contribution of the article, outlines its structure, and reviews related works. Section 2 provides background information about the boundary-value

problems discussed in this article. It comprises the Poisson partial differential equation, the four boundary conditions, i.e., periodic, Dirichlet, Neumann, and Robin, and their extensions to higher dimensional problems. For each problem, its discretization as a system of linear equations via finite difference method is presented. Section 3 presents the main contribution of the article. It begins with a definition of a *block-diagonalizable* matrix, and shows that the discretized boundary-value problems' matrices presented in the preceding section are all block-diagonalizable, i.e., they can all be put into a common form. Section 4 takes advantages of this common form, to construct quantum circuits which can block-encode each of the aforementioned matrices inside a unitary operator. For each quantum circuit, a decomposition into elementary quantum gates and circuits is also provided. Section 5 analyses the computational resources required to construct such block-encoding quantum circuits, and the time complexity of their applications with quantum linear solver algorithms to solve our boundary-value problems. Section 6 completes the article by reiterating the main contribution of the article, discussing its overall impact on the quantum computing domain for solving differential equations and beyond, and outlining some future research directions regarding the block-diagonalization methodology.

Related Works Since the inception of the first quantum linear solver algorithm, there have been several proposals regarding their application towards finite difference method and other numerical methods in general. [24–27] are some proposals for finite difference method and spectral method solving Poisson equations, using various quantum linear solver algorithms, discretization schemes, and matrix encoding techniques. [28] proposes a quantum preconditioner and demonstrates its application for finite element method, while [29] provides the first end-to-end time complexity analysis by including the dependencies between the matrix size and the discretization error for finite element method. Other works focus on the differential equations themselves, including wave equation [30], heat equation [31], linear and non-linear ordinary differential equations [32–39], and non-linear partial differential equations [40–42]. And finally, there are also works related to the block-encoding of both dense and sparse matrices with special structures [43–47].

In this work, we are particularly interested in works related to the encoding of finite difference method's matrices of our boundary-value problems using (1) the conventional quantum oracles [8, superexponential], capable of computing the entries of the matrix given their row and column indices, (2) sparse matrices derived by adaptive central-difference approximation scheme [25], and (3) sparse matrices derived by 3-points central-difference approximation scheme [27], similar to our matrices in this article. For each of these encoding techniques, we provide analyses of these encoding techniques, using either the prescribed quantum linear solver algorithm provided by each of the article, or the quantum linear solver from [18] which is used in this article, to ensure fair comparisons. Other encoding techniques of finite difference method matrices include [24, 26]; however, they can be subsumed by [25, 27] due to their similarities.

2 Preliminaries

A boundary-value problem comprises two components: a differential equation and a boundary condition. In this article, we study a typical second-order linear partial differential equation called Poisson's equation as given in eq. (1), and a periodic boundary condition as given in eq. (2), where $u, f : [0, 1] \to \mathbb{R}$ are solution and data function, respectively.

$$-\frac{d^2}{dx^2}u(x) = f(x) \quad \text{for all} \quad x \in (0, 1) \tag{1}$$

$$u(0) = u(1) \tag{2}$$

Such problem can be solved numerically by finite difference method, whereby the solutions are the discretized values of u at some particular grid points. In this article, we use uniform discretization of N grid points and define $h := 1/N$ as the grid size. Then, using a 3-points central-difference approximation scheme, we derive a system of linear equations eq. (3), where $(u_j, f_j) := (u(jh), f(jh))$ for all $j \in \{0, \ldots, N-1\}$, are the discretized values of u and f at grid point j, respectively. Finally, the discretized solutions u_j are obtained, by solving the system of linear equations eq. (3).

$$\underbrace{\frac{1}{h^2}\begin{pmatrix} 2 & -1 & & & & & -1 \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 2 & -1 & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ -1 & & & & & -1 & 2 \end{pmatrix}}_{\mathbf{L}_p} \cdot \underbrace{\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-3} \\ u_{N-2} \\ u_{N-1} \end{pmatrix}}_{|u\rangle_p} = \underbrace{\begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-3} \\ f_{N-2} \\ f_{N-1} \end{pmatrix}}_{|f\rangle_p} \tag{3}$$

2.1 Boundary Conditions

Similarly, we can form different boundary-value problems by replacing the periodic boundary condition in eq. (2) by other boundary conditions including Dirichlet, Neumann, and Robin as given in eqs. (4) to (6) respectively, with $a, b, c, d, A, B \in \mathbb{R}$ such that the problems are stable, consistent, and thus converged to the actual solution for large N. Their corresponding systems of linear equations are given in eqs. (7) to (9) using the same discretization grid and approximation scheme, except in the Dirichlet case, which use $N + 2$ gird points and $h := 1/(N + 2)$ grid size instead.

$$u(0) = a \quad \text{and} \quad u(1) = b \tag{4}$$

$$u'(0) = a \quad \text{and} \quad u'(1) = b \tag{5}$$

$$au(0) + bu'(0) = A \quad \text{and} \quad cu(1) + du'(1) = B \tag{6}$$

$$\frac{1}{h^2}\underbrace{\begin{pmatrix} 2 & -1 & & & & 0 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & 2 \end{pmatrix}}_{\mathbf{L}_D} \cdot \underbrace{\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ u_N \end{pmatrix}}_{|u\rangle_D} = \underbrace{\begin{pmatrix} f_1 - a/h^2 \\ f_2 \\ f_3 \\ f_4 \\ \vdots \\ f_{N-2} \\ f_{N-1} \\ f_N - b/h^2 \end{pmatrix}}_{|f\rangle_D} \tag{7}$$

$$\frac{1}{h^2}\underbrace{\begin{pmatrix} 1 & -1 & & & & 0 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & 1 \end{pmatrix}}_{\mathbf{L}_N} \cdot \underbrace{\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-3} \\ u_{N-2} \\ u_{N-1} \end{pmatrix}}_{|u\rangle_N} = \underbrace{\begin{pmatrix} -a/h \\ f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-3} \\ f_{N-2} \\ b/h \end{pmatrix}}_{|f\rangle_N} \tag{8}$$

$$\frac{1}{h^2}\underbrace{\begin{pmatrix} C & -1 & & & & 0 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & D \end{pmatrix}}_{\mathbf{L}_R} \cdot \underbrace{\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-3} \\ u_{N-2} \\ u_{N-1} \end{pmatrix}}_{|u\rangle_R} = \underbrace{\begin{pmatrix} -A/b \\ f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-3} \\ f_{N-2} \\ B/d \end{pmatrix}}_{|f\rangle_R} \tag{9}$$

where

$$C := 1 + ah/b \quad \text{and} \quad D := 1 + ch/d \tag{10}$$

and the subscripts p, D, N, and R stand for periodic, Dirichlet, Neumann, and Robin, respectively.

2.2 Higher Dimensions

Additionally, we can also extend the boundary-value problem to an arbitrary dimension $d \in \mathbb{N}$, by replacing eq. (1) with a generalized Poisson's equation in eq. (11) where $u, f : [0, 1]^d \to \mathbb{R}$, and imposing the same boundary condition in each dimension. Note that we also assume that the boundary condition in each dimension is separable from each other.

Using the same discretization grid and approximation scheme in each dimension, we derive a system of linear equations $\mathcal{L}\,|u\rangle = |f\rangle$, where $|u\rangle$ and $|f\rangle$ comprises the discretized values of u and f at grid point j for all $j \in \{0, \ldots, N - 1\}^d$.

Equations (12) and (13) give the definition of $\mathcal{L}$, as a function of a one-dimensional matrix $\mathbf{L} \in \{\mathbf{L}_p, \mathbf{L}_D, \mathbf{L}_N, \mathbf{L}_R\}$.

$$-\sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2} u(\mathbf{x}) = f(\mathbf{x}) \quad \text{for all} \quad \mathbf{x} \in (0,1)^d \tag{11}$$

$$\mathcal{L} := \sum_{k=1}^{d} \mathcal{L}_k \tag{12}$$

where

$$\mathcal{L}_k := \mathbf{I}_N^{\otimes(k-1)} \otimes \mathbf{L} \otimes \mathbf{I}_N^{\otimes(d-k)} \quad \text{and} \quad \mathbf{L} \in \{\mathbf{L}_p, \mathbf{L}_D, \mathbf{L}_N, \mathbf{L}_R\} \tag{13}$$

It is also possible to have a mixed boundary condition, e.g., a Dirichlet boundary condition on one side of the boundary in a particular dimension, and a Neumann boundary condition on the other side of the boundary in the same dimension or a different dimension. However, the domain may become hyper-rectangular instead of the hypercube that we have.

Additionally, note that all the matrices $\mathbf{L}_p$, $\mathbf{L}_D$, $\mathbf{L}_N$, $\mathbf{L}_R$ and $\mathcal{L}$ are all Hermitian and positive definite, which means that they are invertible, and their corresponding systems of linear equations have a unique solution.

3 Block-Diagonalization

The primary objective of this article is to show that the matrices $\mathbf{L}_p$, $\mathbf{L}_D$, $\mathbf{L}_N$ and $\mathbf{L}_R$ which are derived using finite difference method for some boundary-value problems, can be block-encoded inside a unitary operator with a double-logarithmic depth $\mathcal{O}(\log(\log(N)))$ in their matrix size N. To achieve such objective, we introduce the concept of a *block-diagonalizable* matrix in definition 3.1, and show that all the aforementioned matrices are block-diagonalizable.

Definition 3.1 (Block-Diagonalizable Matrix).
An $N \times N$ matrix $\mathbf{L}$ is *block-diagonalizable* if

$$\mathbf{L} = \sum_{c \in \{0,1\}} \sum_{\hat{\sigma} \in \{\mathbf{I},\mathbf{X},\mathbf{Y},\mathbf{Z}\}} \chi_{\hat{\sigma}} \cdot \mathbf{L}_{c,\hat{\sigma}} \tag{14}$$

for some

$$\chi_{\hat{\sigma}} \in \mathbb{R} \tag{15}$$

$$\mathbf{L}_{c,\hat{\sigma}} := \mathbf{P}_c \cdot \mathbf{D}_{\hat{\sigma}} \cdot \mathbf{P}_c^\dagger \tag{16}$$

$$\mathbf{D}_{\hat{\sigma}} := \mathrm{diag}(\alpha_0, \ldots, \alpha_{N/2-1})_{N/2} \otimes \hat{\sigma} \tag{17}$$

$$\mathbf{P}_c := \sum_{i=0}^{N-1} |i + c \bmod N\rangle\langle i| \tag{18}$$

where $\mathbf{D}_{\hat{\sigma}}$ and $\mathbf{P}_c$ are called Pauli-block-diagonal matrix and permutation matrix, respectively. ■

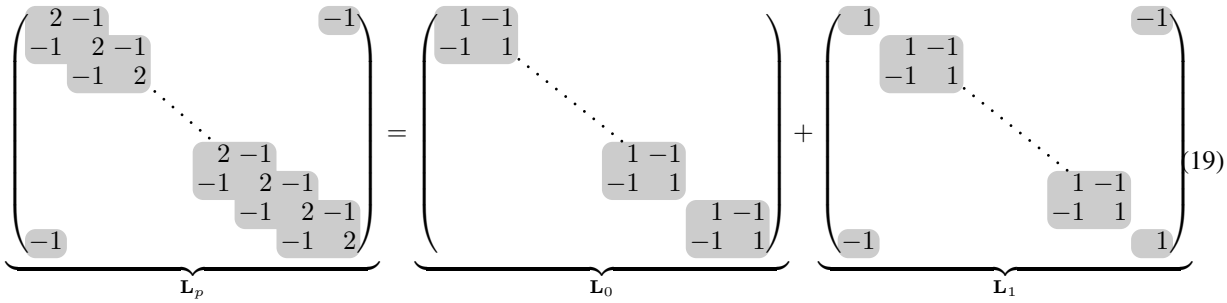
In summary, a block-diagonalizable matrix $\mathbf{L}$ can be decomposed as a linear combination of linear operators $\mathbf{L}_{c,\hat{\sigma}}$, for some $c \in \{0,1\}$ and Pauli matrix $\hat{\sigma}$; each of which is composed of a Pauli-block-diagonal matrix $\mathbf{D}_{\hat{\sigma}}$ that is left and right multiplied by a permutation matrix $\mathbf{P}_c$ and its conjugate transpose $\mathbf{P}_c^\dagger$, respectively. Such a matrix decomposition process is called *block-diagonalization*; from which a procedure for constructing a block-encoding matrix is derived.

This methodology is inspired by the combinatorially block-diagonal matrix definition of [48], since each of our linear operators $\mathbf{L}_{c,\hat{\sigma}}$ is in fact a 2×2 combinatorially block-diagonal matrix, where the permutation matrices $\mathbf{P}_c$ and $\mathbf{P}_c^\dagger$ are the combinatorial permutations. Our block-diagonalizable matrix definition is basically a linear combination of 2×2 combinatorially block-diagonal matrices; each of which is a tensor-product of a diagonal matrix and a Pauli matrix.

This section is divided into five subsections, which describe the block-diagonalization of $\mathbf{L}_p$ with periodic boundary condition in section 3.1, $\mathbf{L}_D$ with Dirichlet boundary condition in section 3.2, $\mathbf{L}_N$ with Neumann boundary condition in section 3.3, $\mathbf{L}_R$ with Robin boundary condition in section 3.4, and $\mathcal{L}$ from higher dimensional problems in section 3.5.

3.1 Periodic

We begin by decomposing $\mathbf{L}_p$ as a sum of two matrices $\mathbf{L}_0$ and $\mathbf{L}_1$ as shown in eq. (19).



$$\underbrace{\begin{pmatrix} 2 & -1 & & & & -1 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ -1 & & & & -1 & 2 \end{pmatrix}}_{\mathbf{L}_p} = \underbrace{\begin{pmatrix} 1 & -1 & & & & \\ -1 & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & & 1 \end{pmatrix}}_{\mathbf{L}_0} + \underbrace{\begin{pmatrix} 1 & & & & & -1 \\ & 1 & -1 & & & \\ & -1 & 1 & \ddots & & \\ & & \ddots & 1 & -1 & \\ & & & -1 & 1 & \\ -1 & & & & & 1 \end{pmatrix}}_{\mathbf{L}_1} \tag{19}$$

We define a 2×2 block-diagonal matrix $\mathbf{B}$ as shown in eq. (20). Then, we show that it can be transformed into $\mathbf{L}_0$ and $\mathbf{L}_1$ as in eq. (21), where $\mathbf{P}_0$ and $\mathbf{P}_1$ are permutation matrices as defined in eq. (22). As in [48], we say that $\mathbf{L}_0$ and $\mathbf{L}_1$ are 2×2 combinatorially block-diagonal matrices, i.e., block-diagonal given some row and column rearrangements.

$$\mathbf{B} := \begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & & & \\ & & \ddots & & \\ & & & 1 & -1 \\ & & & -1 & 1 \end{pmatrix} \tag{20}$$

$$\mathbf{L}_0 = \mathbf{P}_0 \cdot \mathbf{B} \cdot \mathbf{P}_0^\dagger \quad \text{and} \quad \mathbf{L}_1 = \mathbf{P}_1 \cdot \mathbf{B} \cdot \mathbf{P}_1^\dagger \tag{21}$$

$$\mathbf{P}_0 := \sum_{i=0}^{N-1} |i\rangle\langle i| \quad \text{and} \quad \mathbf{P}_1 := \sum_{i=0}^{N-1} |i+1 \bmod N\rangle\langle i| \tag{22}$$

We decompose the 2×2 block-matrix $((1,-1),(-1,1))$ as a sum of Pauli matrices $\mathbf{I}$ and $\mathbf{X}$ as in eq. (23). Substituting that into eq. (20), we can rewrite $\mathbf{B}$ as the sum of two Pauli-block-diagonal matrices $\mathbf{D_I}$ and $\mathbf{D_X}$ as in eq. (24).

$$\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{I}} - \underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}_{\mathbf{X}} \tag{23}$$

$$\mathbf{B} = \underbrace{\operatorname{diag}(1,\ldots,1)_{N/2} \otimes \mathbf{I}}_{\mathbf{D_I}} - \underbrace{\operatorname{diag}(1,\ldots,1)_{N/2} \otimes \mathbf{X}}_{\mathbf{D_X}} \tag{24}$$

Finally, substituting eq. (24) into eq. (21), and again into eq. (19), we obtain a block-diagonalization of $\mathbf{L}_p$ in theorem 3.1.

Theorem 3.1 (Block-Diagonalization, Periodic).

$$\mathbf{L}_p = \sum_{c \in \{0,1\}} \sum_{\hat{\sigma} \in \{\mathbf{I},\mathbf{X}\}} \chi_{\hat{\sigma}} \cdot \mathbf{L}_{c,\hat{\sigma}} \tag{25}$$

where

$$\chi_{\hat{\sigma}} := \begin{cases} 1, & \hat{\sigma} = \mathbf{I} \\ -1, & \hat{\sigma} = \mathbf{X} \end{cases} \tag{26}$$

$$\mathbf{L}_{c,\hat{\sigma}} := \mathbf{P}_c \cdot \mathbf{D}_{\hat{\sigma}} \cdot \mathbf{P}_c^\dagger \tag{27}$$

$$\mathbf{D}_{\hat{\sigma}} := \operatorname{diag}(1,\ldots,1)_{N/2} \otimes \hat{\sigma} \tag{28}$$

$$\mathbf{P}_c := \sum_{i=0}^{N-1} |i + c \bmod N\rangle\langle i| \tag{29}$$

■

3.2 Dirichlet

The following derives block-diagonalization of $\mathbf{L}_D$ as in theorem 3.2, using similar procedures as those of the periodic case. However, instead of defining a single 2×2 block-diagonal matrix $\mathbf{B}$, we use two such matrices $\mathbf{B}_0$ and $\mathbf{B}_1$ instead. Additionally, one of the 2×2 block matrices $((1,0),(0,1))$ is different, and also has a different Pauli decomposition.

$$\underbrace{\begin{pmatrix} 2 & -1 & & & & 0 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & 2 \end{pmatrix}}_{\mathbf{L}_D} = \underbrace{\begin{pmatrix} 1 & -1 & & & & \\ -1 & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & & 1 \end{pmatrix}}_{\mathbf{L}_0} + \underbrace{\begin{pmatrix} 1 & & & & & 0 \\ & 1 & -1 & & & \\ & -1 & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & -1 \\ 0 & & & & -1 & 1 \end{pmatrix}}_{\mathbf{L}_1} \tag{30}$$

$$\mathbf{B}_0 := \begin{pmatrix} 1 & -1 & & & & \\ -1 & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & & 1 \end{pmatrix} \quad \mathbf{B}_1 := \begin{pmatrix} 1 & -1 & & & & \\ -1 & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & & 0 \end{pmatrix} \tag{31}$$

$$\mathbf{L}_0 = \mathbf{P}_0 \cdot \mathbf{B}_0 \cdot \mathbf{P}_0^\dagger \quad \text{and} \quad \mathbf{L}_1 = \mathbf{P}_1 \cdot \mathbf{B}_1 \cdot \mathbf{P}_1^\dagger \tag{32}$$

$$\begin{aligned} \mathbf{B}_0 &= \underbrace{\mathrm{diag}(1,\ldots,1,1)_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{0,\mathbf{I}}} - \underbrace{\mathrm{diag}(1,\ldots,1,1)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{0,\mathbf{X}}} \\ \mathbf{B}_1 &= \underbrace{\mathrm{diag}(1,\ldots,1,1)_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{1,\mathbf{I}}} - \underbrace{\mathrm{diag}(1,\ldots,1,0)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{1,\mathbf{X}}} \end{aligned} \tag{33}$$

Theorem 3.2 (Block-Diagonalization, Dirichlet).

$$\mathbf{L}_D = \sum_{c \in \{0,1\}} \sum_{\hat{\sigma} \in \{\mathbf{I},\mathbf{X}\}} \chi_{\hat{\sigma}} \cdot \mathbf{L}_{c,\hat{\sigma}} \tag{34}$$

where

$$\mathbf{L}_{c,\hat{\sigma}} := \mathbf{P}_c \cdot \mathbf{D}_{c,\hat{\sigma}} \cdot \mathbf{P}_c^\dagger \tag{35}$$

$$\mathbf{D}_{c,\hat{\sigma}} := \begin{cases} \mathrm{diag}(1,\ldots,1,1)_{N/2} \otimes \hat{\sigma}, & (c,\hat{\sigma}) \neq (1,\mathbf{X}) \\ \mathrm{diag}(1,\ldots,1,0)_{N/2} \otimes \hat{\sigma}, & (c,\hat{\sigma}) = (1,\mathbf{X}) \end{cases} \tag{36}$$

■

3.3 Neumann

The following derives block-diagonalization of $\mathbf{L}_N$ as in theorem 3.3, using similar procedures as those of the periodic case. However, instead of defining a single 2×2 block-diagonal matrix $\mathbf{B}$, we use two such matrices $\mathbf{B}_0$ and $\mathbf{B}_1$ instead.

Additionally, one of the 2×2 block matrix $((0,0),(0,0))$ is different, and also has a different Pauli decomposition.

$$\underbrace{\begin{pmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & 1 \end{pmatrix}}_{\mathbf{L}_N} = \underbrace{\begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & \ddots & & \\ & \ddots & 1 & -1 & \\ & & -1 & 1 & \\ & & & & 1 & -1 \\ & & & & -1 & 1 \end{pmatrix}}_{\mathbf{L}_0} + \underbrace{\begin{pmatrix} 0 & & & & & \\ & 1 & -1 & & & \\ & -1 & 1 & \ddots & & \\ & & \ddots & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & & 0 \end{pmatrix}}_{\mathbf{L}_1} \tag{37}$$

$$\mathbf{B}_0 := \begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & \ddots & & \\ & \ddots & 1 & -1 & \\ & & -1 & 1 & \\ & & & & 1 & -1 \\ & & & & -1 & 1 \end{pmatrix} \quad \mathbf{B}_1 := \begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & \ddots & & \\ & \ddots & 1 & -1 & \\ & & -1 & 1 & \\ & & & & 0 & 0 \\ & & & & 0 & 0 \end{pmatrix} \tag{38}$$

$$\mathbf{L}_0 = \mathbf{P}_0 \cdot \mathbf{B}_0 \cdot \mathbf{P}_0^\dagger \quad \text{and} \quad \mathbf{L}_1 = \mathbf{P}_1 \cdot \mathbf{B}_1 \cdot \mathbf{P}_1^\dagger \tag{39}$$

$$\begin{aligned} \mathbf{B}_0 &= \underbrace{\operatorname{diag}(1,\ldots,1,1)_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{0,\mathbf{I}}} - \underbrace{\operatorname{diag}(1,\ldots,1,1)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{0,\mathbf{X}}} \\ \mathbf{B}_1 &= \underbrace{\operatorname{diag}(1,\ldots,1,0)_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{1,\mathbf{I}}} - \underbrace{\operatorname{diag}(1,\ldots,1,0)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{1,\mathbf{X}}} \end{aligned} \tag{40}$$

Theorem 3.3 (Block-Diagonalization, Neumann).

$$\mathbf{L}_D = \sum_{c \in \{0,1\}} \sum_{\hat{\sigma} \in \{\mathbf{I},\mathbf{X}\}} \chi_{\hat{\sigma}} \cdot \mathbf{L}_{c,\hat{\sigma}} \tag{41}$$

where

$$\mathbf{L}_{c,\hat{\sigma}} := \mathbf{P}_c \cdot \mathbf{D}_{c,\hat{\sigma}} \cdot \mathbf{P}_c^\dagger \tag{42}$$

$$\mathbf{D}_{c,\hat{\sigma}} := \begin{cases} \operatorname{diag}(1,\ldots,1,1)_{N/2} \otimes \hat{\sigma}, & c = 0 \\ \operatorname{diag}(1,\ldots,1,0)_{N/2} \otimes \hat{\sigma}, & c = 1 \end{cases} \tag{43}$$

■

3.4 Robin

The following derives block-diagonalization of $\mathbf{L}_R$ as in theorem 3.4, using similar procedures as those of the periodic case. However, instead of defining a single 2×2 block-diagonal matrix $\mathbf{B}$, we use two such matrices $\mathbf{B}_0$ and $\mathbf{B}_1$ instead. Additionally, two of the 2×2 block matrices are different, and thus have different Pauli decompositions, as in eq. (48).

$$\underbrace{\begin{pmatrix} C & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 2 & -1 \\ 0 & & & & -1 & D \end{pmatrix}}_{\mathbf{L}_R} = \underbrace{\begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & \ddots & & \\ & \ddots & 1 & -1 & \\ & & -1 & 1 & \\ & & & & 2 & -1 \\ & & & & -1 & D' \end{pmatrix}}_{\mathbf{L}_0} + \underbrace{\begin{pmatrix} C' & & & & & \\ & 1 & -1 & & & \\ & -1 & 1 & \ddots & & \\ & & \ddots & 1 & -1 & \\ & & & -1 & 1 & \\ & & & & 0 & 1 \end{pmatrix}}_{\mathbf{L}_1} \tag{44}$$

where

$$C' := C - 1 \quad \text{and} \quad D' := D - 1 \tag{45}$$

$$\mathbf{B}_0 := \begin{pmatrix} \begin{matrix} 1 & -1 \\ -1 & 1 \end{matrix} & & & \\ & \ddots & & \\ & & \begin{matrix} 1 & -1 \\ -1 & 1 \end{matrix} & \\ & & & \begin{matrix} 1 & -1 \\ -1 & D' \end{matrix} \end{pmatrix} \quad \mathbf{B}_1 := \begin{pmatrix} \begin{matrix} 1 & -1 \\ -1 & 1 \end{matrix} & & & \\ & \ddots & & \\ & & \begin{matrix} 1 & -1 \\ -1 & 1 \end{matrix} & \\ & & & \begin{matrix} 1 & 0 \\ 0 & C' \end{matrix} \end{pmatrix} \tag{46}$$

$$\mathbf{L}_0 = \mathbf{P}_0 \cdot \mathbf{B}_0 \cdot \mathbf{P}_0^\dagger \quad \text{and} \quad \mathbf{L}_1 = \mathbf{P}_1 \cdot \mathbf{B}_1 \cdot \mathbf{P}_1^\dagger \tag{47}$$

$$\begin{aligned} \begin{pmatrix} 1 & -1 \\ -1 & D' \end{pmatrix} &= \frac{D}{2} \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{I}} + \left(1 - \frac{D}{2}\right) \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}}_{\mathbf{Z}} - \underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}}_{\mathbf{X}} \\ \begin{pmatrix} 1 & 0 \\ 0 & C' \end{pmatrix} &= \frac{C}{2} \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{I}} + \left(1 - \frac{C}{2}\right) \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}}_{\mathbf{Z}} \end{aligned} \tag{48}$$

$$\begin{aligned} \mathbf{B}_0 &= \underbrace{\mathrm{diag}(1, \ldots, 1, \frac{D}{2})_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{0,\mathbf{I}}} + \underbrace{\mathrm{diag}(0, \ldots, 0, 1 - \frac{D}{2})_{N/2} \otimes \mathbf{Z}}_{\mathbf{D}_{0,\mathbf{Z}}} - \underbrace{\mathrm{diag}(1, \ldots, 1, 1)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{0,\mathbf{X}}} \\ \mathbf{B}_1 &= \underbrace{\mathrm{diag}(1, \ldots, 1, \frac{C}{2})_{N/2} \otimes \mathbf{I}}_{\mathbf{D}_{1,\mathbf{I}}} + \underbrace{\mathrm{diag}(0, \ldots, 0, 1 - \frac{C}{2})_{N/2} \otimes \mathbf{Z}}_{\mathbf{D}_{1,\mathbf{Z}}} - \underbrace{\mathrm{diag}(1, \ldots, 1, 0)_{N/2} \otimes \mathbf{X}}_{\mathbf{D}_{1,\mathbf{X}}} \end{aligned} \tag{49}$$

Theorem 3.4 (Block-Diagonalization, Robin).

$$\mathbf{L}_R = \sum_{c \in \{0,1\}} \sum_{\hat{\sigma} \in \{\mathbf{I}, \mathbf{X}, \mathbf{Z}\}} \chi_{\hat{\sigma}} \cdot \mathbf{L}_{c,\hat{\sigma}} \tag{50}$$

where

$$\chi_{\hat{\sigma}} := \begin{cases} 1, & \hat{\sigma} \in \{\mathbf{I}, \mathbf{Z}\} \\ -1, & \hat{\sigma} = \mathbf{X} \end{cases} \tag{51}$$

$$\mathbf{L}_{c,\hat{\sigma}} := \mathbf{P}_c \cdot \mathbf{D}_{c,\hat{\sigma}} \cdot \mathbf{P}_c^\dagger \tag{52}$$

$$\mathbf{D}_{c,\hat{\sigma}} := \begin{cases} \mathrm{diag}(1, \ldots, 1, D/2) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (0, \mathbf{I}) \\ \mathrm{diag}(1, \ldots, 1, C/2) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (1, \mathbf{I}) \\ \mathrm{diag}(0, \ldots, 0, 1 - D/2) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (0, \mathbf{Z}) \\ \mathrm{diag}(0, \ldots, 0, 1 - C/2) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (1, \mathbf{Z}) \\ \mathrm{diag}(1, \ldots, 1, 1) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (0, \mathbf{X}) \\ \mathrm{diag}(1, \ldots, 1, 0) \otimes \hat{\sigma}, & (c, \hat{\sigma}) = (1, \mathbf{X}) \end{cases} \tag{53}$$

■

3.5 Higher Dimensions

For higher dimensional problems, $\mathcal{L}$ is strictly speaking not block-diagonalizable by definition 3.1. However, $\mathbf{L}$ which is a dependency of $\mathcal{L}$ is block-diagonalizable, as in theorems 3.1 to 3.4. Therefore, we substitute its block-diagonalization into eq. (13), then into eq. (12), and called the resulting form a block-diagonalization of $\mathcal{L}$, by abuse of definition.

4 Block-Encoding

With $\mathbf{L}_p$, $\mathbf{L}_D$, $\mathbf{L}_N$, $\mathbf{L}_R$ and $\mathcal{L}$ in block-diagonalization forms as in theorems 3.1 to 3.4 and section 3.5, we are ready to construct their respective block-encoding unitary operators using these forms. Several variations of block-encoding definition exist in the literature. In this article, we define block-encoding as in definition 4.1, where $\mathcal{L}(\mathcal{H}^{\otimes n})$ is a space of linear operator mapping a Hilbert space $\mathcal{H}^{\otimes n} := (\mathbb{C}^2)^{\otimes n}$ onto itself, for some $n \in \mathbb{N}$. By isomorphism and abuse of notation, we say that all the aforementioned $N \times N$ matrices are also linear operators of $\mathcal{L}(\mathcal{H}^{\otimes n})$, where $n := \log(N)$.

Definition 4.1 (Block-Encoding).
A unitary operator $\bar{\mathbf{L}} \in \mathcal{L}(\mathcal{H}^{\otimes n+m})$ is a block-encoding of $\mathbf{L} \in \mathcal{L}(\mathcal{H}^{\otimes n})$, if there exists $\eta \in \mathbb{R}^+$ such that

$$\mathbf{L} = \eta \cdot \mathbf{\Pi} \cdot \bar{\mathbf{L}} \cdot \mathbf{\Pi}^\dagger \quad \text{where} \quad \mathbf{\Pi} := \langle 0|^{\otimes m} \otimes \mathbf{I}^{\otimes n} \tag{54}$$

More succinctly, we say that $\bar{\mathbf{L}}$ block-encode $\mathbf{L}/\eta$ as its upper-left block-matrix as in

$$\bar{\mathbf{L}} := \begin{matrix} \\ \mathbf{\Pi}^\dagger \\ \tilde{\mathbf{\Pi}}^\dagger \end{matrix} \begin{matrix} \begin{matrix} \mathbf{\Pi} & \tilde{\mathbf{\Pi}} \end{matrix} \\ \begin{pmatrix} \mathbf{L}/\eta & \cdot \\ \cdot & \cdot \end{pmatrix} \end{matrix} \tag{55}$$

where $\tilde{\mathbf{\Pi}}$ is a projection operator, which is orthogonal complement to $\mathbf{\Pi}$. ■

In this work, all the block-encoding contain an exact subnormalized matrix $\mathbf{L}/\eta$, rather than an approximation by some precision error ε. Moreover, all the matrices are not only square, but also Hermitian and positive definite. And finally, they can all be decomposed as a linear combination of unitary (LCU) operators with efficient implementations, which become evident later. Therefore, the majority of block-encodings used in this article is via LCU technique as in [11].

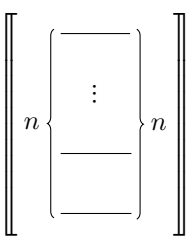
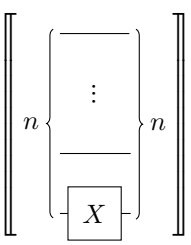
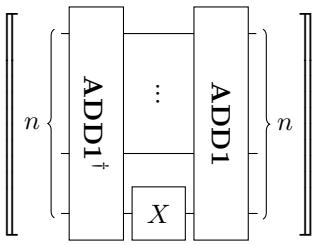
This section is also divided into five subsections, which describe the block-encoding of $\mathbf{L}_p$ with periodic boundary condition in section 3.1, $\mathbf{L}_D$ with Dirichlet boundary condition in section 4.2, $\mathbf{L}_N$ with Neumann boundary condition in section 4.3, $\mathbf{L}_R$ with Robin boundary condition in section 4.4, and $\mathcal{L}$ from higher dimensional problems in section 4.5.

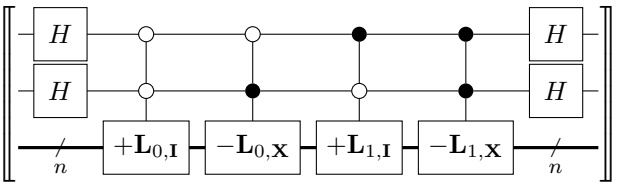
4.1 Periodic

We begin by rewriting each linear operator $\mathbf{L}_{c,\hat{\sigma}}$ from eq. (27), in terms of elementary quantum gates and circuits, as in eq. (56), where $\mathbf{ADD1}$ is a modulo-2^n incrementation operation, which is one of the quantum arithmetic operations.

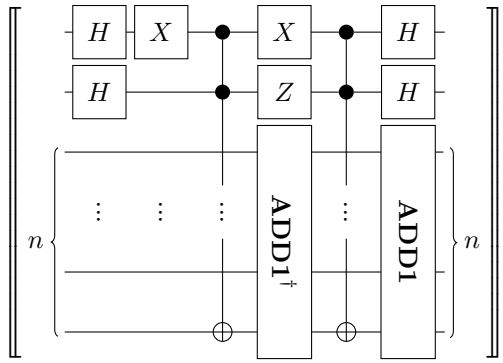
$$\begin{aligned} \mathbf{L}_{0,\mathbf{I}} = \mathbf{L}_{1,\mathbf{I}} = \mathbf{I}^{\otimes n}, \quad \mathbf{L}_{0,\mathbf{X}} &= \mathbf{I}^{\otimes n-1} \otimes \mathbf{X} \\ \mathbf{L}_{1,\mathbf{X}} &= \mathbf{ADD1} \cdot \left(\mathbf{I}^{\otimes n-1} \otimes \mathbf{X}\right) \cdot \mathbf{ADD1}^\dagger \\ \mathbf{ADD1} &:= \sum_{i=0}^{2^n-1} |i+1 \bmod 2^n\rangle\langle i| \end{aligned} \tag{56}$$

Since each $\mathbf{L}_{c,\hat{\sigma}}$ is a unitary operator, we can directly represent it using quantum circuit notation as in eq. (57). Then, we construct a unitary operator $\bar{\mathbf{L}}_p$ as in eq. (58), which block-encode $\mathbf{L}_p/4$ via LCU technique using 2 ancilla qubits. Equation (59) provides a simplified version of eq. (58), in terms of elementary quantum gates and quantum circuits.

$\mathbf{L}_{0,\mathbf{I}} := \mathbf{L}_{1,\mathbf{I}} :=$  , $\mathbf{L}_{0,\mathbf{X}} :=$  , $\mathbf{L}_{1,\mathbf{X}} :=$  (57)

$\bar{\mathbf{L}}_p :=$  (58)

$$\bar{\mathbf{L}}_p = \left[\!\!\left[\right.$$



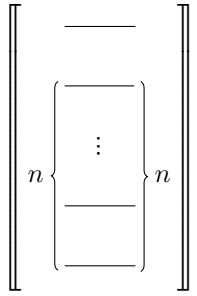
$$\left.\!\!\right]\!\!\right] \tag{59}$$

4.2 Dirichlet

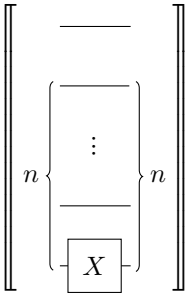
The following block-encodes $\mathbf{L}_D$ using similar procedures to those of the periodic case. However, not all $\mathbf{L}_{c,\hat{\sigma}}$ terms from eq. (35) are unitary operators as in periodic case. Hence, an additional ancilla qubit is required to block-encode such term via LCU technique, while unitary terms can be block-encoded trivially via a single ancilla qubit concatenation. The unitary operator which block-encodes each $\mathbf{L}_{c,\hat{\sigma}}$ is denoted $\bar{\mathbf{L}}_{c,\hat{\sigma}}$ to differentiate between the two operators.

$$\begin{aligned} \mathbf{L}_{0,\mathbf{I}} = \mathbf{L}_{1,\mathbf{I}} = \mathbf{I}^{\otimes n}, \quad \mathbf{L}_{0,\mathbf{X}} = \mathbf{I}^{\otimes n-1} \otimes \mathbf{X} \\ \mathbf{L}_{1,\mathbf{X}} = \mathbf{ADD1} \cdot \left(\left(\frac{1}{2} \cdot \mathbf{I}^{\otimes n-1} + \frac{1}{2} \cdot \mathbf{C}^{n-2}\mathbf{Z} \right) \otimes \mathbf{X} \right) \cdot \mathbf{ADD1}^\dagger \end{aligned} \tag{60}$$

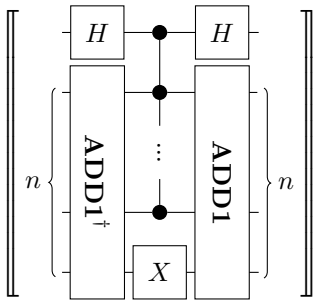
$$\bar{\mathbf{L}}_{0,\mathbf{I}} := \bar{\mathbf{L}}_{1,\mathbf{I}} := \left[\!\!\left[\right.$$



$$\left.\!\!\right]\!\!\right], \ \bar{\mathbf{L}}_{0,\mathbf{X}} := \left[\!\!\left[\right.$$

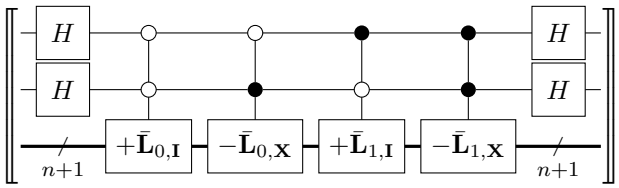


$$\left.\!\!\right]\!\!\right], \ \bar{\mathbf{L}}_{1,\mathbf{X}} := \left[\!\!\left[\right.$$



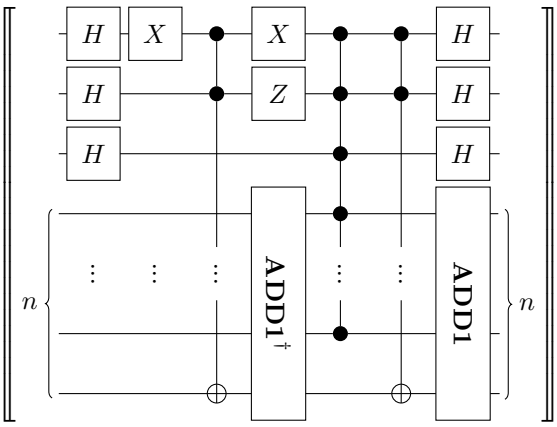
$$\left.\!\!\right]\!\!\right] \tag{61}$$

$$\bar{\mathbf{L}}_D := \left[\!\!\left[\right.$$



$$\left.\!\!\right]\!\!\right] \tag{62}$$

$$\bar{\mathbf{L}}_D = \left[\!\!\left[\right.$$



$$\left.\!\!\right]\!\!\right] \tag{63}$$

4.3 Neumann

The following block-encodes $\mathbf{L}_N$ using similar procedures to those of the periodic case. However, not all $\mathbf{L}_{c,\hat{\sigma}}$ terms from eq. (42) are unitary operators as in periodic case. Hence, an additional ancilla qubit is required to block-encode such term via LCU technique, while unitary terms can be block-encoded trivially via a single ancilla qubit concatenation. The unitary operator which block-encodes each $\mathbf{L}_{c,\hat{\sigma}}$ is denoted $\bar{\mathbf{L}}_{c,\hat{\sigma}}$ to differentiate between the two operators.

$$
\begin{aligned}
\mathbf{L}_{0,\mathbf{I}} &= \mathbf{I}^{\otimes n}, \quad \mathbf{L}_{0,\mathbf{X}} = \mathbf{I}^{\otimes n-1} \otimes \mathbf{X} \\
\mathbf{L}_{1,\mathbf{I}} &= \mathbf{ADD1} \cdot \left(\left(\frac{1}{2} \cdot \mathbf{I}^{\otimes n-1} + \frac{1}{2} \cdot \mathbf{C}^{n-2}\mathbf{Z} \right) \otimes \mathbf{I} \right) \cdot \mathbf{ADD1}^{\dagger} \\
\mathbf{L}_{1,\mathbf{X}} &= \mathbf{ADD1} \cdot \left(\left(\frac{1}{2} \cdot \mathbf{I}^{\otimes n-1} + \frac{1}{2} \cdot \mathbf{C}^{n-2}\mathbf{Z} \right) \otimes \mathbf{X} \right) \cdot \mathbf{ADD1}^{\dagger}
\end{aligned}
\tag{64}
$$

$$
\bar{\mathbf{L}}_{0,\mathbf{I}} :=
\left[\!\!\left[
\begin{array}{c}
\\
n \left\{ \begin{array}{c} \\ \vdots \\ \\ \end{array} \right. n \\
\\
\end{array}
\right]\!\!\right],
\bar{\mathbf{L}}_{0,\mathbf{X}} :=
\left[\!\!\left[
\begin{array}{c}
\\
n \left\{ \begin{array}{c} \\ \vdots \\ \\ \end{array} \right. n \\
X \\
\end{array}
\right]\!\!\right]
\tag{65}
$$

$$
\bar{\mathbf{L}}_{1,\mathbf{I}} :=
\left[\!\!\left[
\begin{array}{ccc}
H & & H \\
n \left\{ \begin{array}{c} \mathbf{ADD1}^{\dagger} \end{array} \right. & \vdots & \left. \begin{array}{c} \mathbf{ADD1} \end{array} \right\} n \\
\end{array}
\right]\!\!\right],
\bar{\mathbf{L}}_{1,\mathbf{X}} :=
\left[\!\!\left[
\begin{array}{ccc}
H & & H \\
n \left\{ \begin{array}{c} \mathbf{ADD1}^{\dagger} \end{array} \right. & \vdots & \left. \begin{array}{c} \mathbf{ADD1} \end{array} \right\} n \\
& X & \\
\end{array}
\right]\!\!\right]
$$

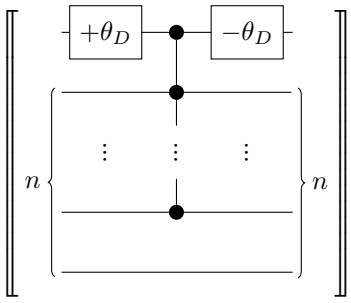
$$
\bar{\mathbf{L}}_N :=
\left[\!\!\left[
\begin{array}{ccccccc}
H & & & & & & H \\
H & & & & & & H \\
\overset{n+1}{/} & +\bar{\mathbf{L}}_{0,\mathbf{I}} & -\bar{\mathbf{L}}_{0,\mathbf{X}} & +\bar{\mathbf{L}}_{1,\mathbf{I}} & -\bar{\mathbf{L}}_{1,\mathbf{X}} & \overset{n+1}{/} & \\
\end{array}
\right]\!\!\right]
\tag{66}
$$

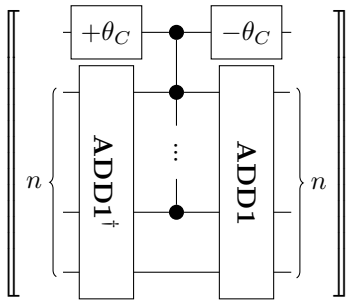
$$
\bar{\mathbf{L}}_N =
\left[\!\!\left[
\begin{array}{ccccccc}
H & X & & X & & & H \\
H & & & Z & & & H \\
H & & & & & & H \\
n \left\{ \begin{array}{c} \vdots \end{array} \right. & \vdots & \vdots & \mathbf{ADD1}^{\dagger} & \vdots & \vdots & \left. \begin{array}{c} \mathbf{ADD1} \end{array} \right\} n \\
& & \oplus & & & \oplus & \\
\end{array}
\right]\!\!\right]
\tag{67}
$$

4.4 Robin

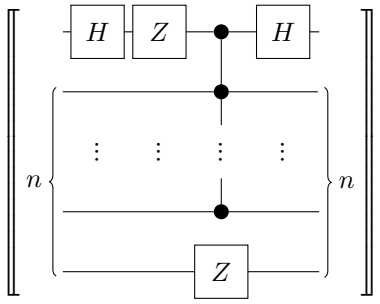
The following block-encodes $\mathbf{L}_R$ using similar procedures to those of the periodic case. However, not all $\mathbf{L}_{c,\hat{\sigma}}$ terms from eq. (52) are unitary operators as in periodic case. Hence, an additional ancilla qubit is required to block-encode such term via LCU technique, while unitary terms can be block-encoded trivially via a single ancilla qubit concatenation. The unitary operator which block-encodes each $\mathbf{L}_{c,\hat{\sigma}}$ is denoted $\bar{\mathbf{L}}_{c,\hat{\sigma}}$ to differentiate between the two operators.

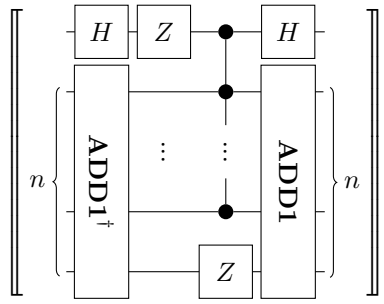
$$
\begin{aligned}
\mathbf{L}_{0,\mathbf{I}} &= \left(\frac{1}{2}(1+\frac{D}{2})\cdot\mathbf{I}^{\otimes n-1} + \frac{1}{2}(1-\frac{D}{2})\cdot\mathbf{C}^{n-2}\mathbf{Z}\right)\otimes\mathbf{I} \\
\mathbf{L}_{0,\mathbf{Z}} &= \left(\frac{1}{2}(1-\frac{D}{2})\cdot\mathbf{I}^{\otimes n-1} - \frac{1}{2}(1-\frac{D}{2})\cdot\mathbf{C}^{n-2}\mathbf{Z}\right)\otimes\mathbf{Z} \\
\mathbf{L}_{0,\mathbf{X}} &= \mathbf{I}^{\otimes n-1}\otimes\mathbf{X} \\
\mathbf{L}_{1,\mathbf{I}} &= \mathbf{ADD1}\cdot\left(\left(\frac{1}{2}(1+\frac{C}{2})\cdot\mathbf{I}^{\otimes n-1} + \frac{1}{2}(1-\frac{C}{2})\cdot\mathbf{C}^{n-2}\mathbf{Z}\right)\otimes\mathbf{I}\right)\cdot\mathbf{ADD1}^\dagger \\
\mathbf{L}_{1,\mathbf{Z}} &= \mathbf{ADD1}\cdot\left(\left(\frac{1}{2}(1-\frac{C}{2})\cdot\mathbf{I}^{\otimes n-1} - \frac{1}{2}(1-\frac{C}{2})\cdot\mathbf{C}^{n-2}\mathbf{Z}\right)\otimes\mathbf{Z}\right)\cdot\mathbf{ADD1}^\dagger \\
\mathbf{L}_{1,\mathbf{X}} &= \mathbf{ADD1}\cdot\left(\left(\frac{1}{2}\cdot\mathbf{I}^{\otimes n-1} + \frac{1}{2}\cdot\mathbf{C}^{n-2}\mathbf{Z}\right)\otimes\mathbf{X}\right)\cdot\mathbf{ADD1}^\dagger
\end{aligned}
\tag{68}
$$

$\bar{\mathbf{L}}_{0,\mathbf{I}} :=$ 

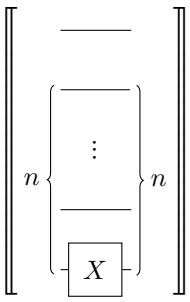
, $\bar{\mathbf{L}}_{1,\mathbf{I}} :=$ 

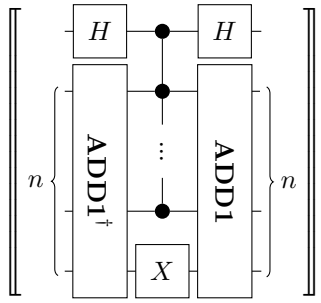
,

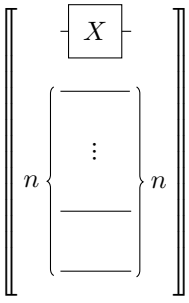
$\bar{\mathbf{L}}_{0,\mathbf{Z}} :=$ 

, $\bar{\mathbf{L}}_{1,\mathbf{Z}} :=$ 

, (69)

$\bar{\mathbf{L}}_{0,\mathbf{X}} :=$ 

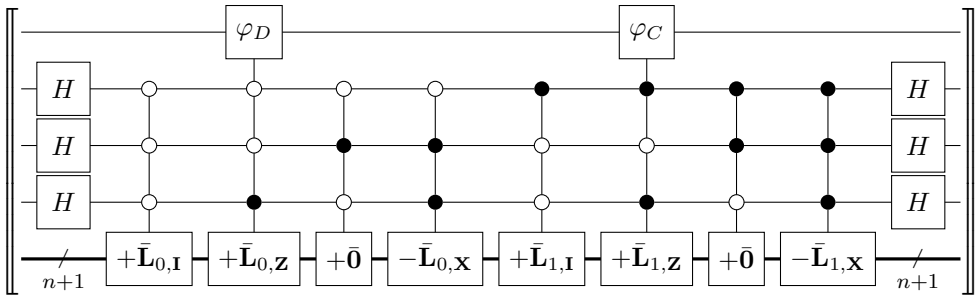
, $\bar{\mathbf{L}}_{1,\mathbf{X}} :=$ 

, $\bar{\mathbf{0}} :=$ 

where

$$
\mathbf{R}_Y(\theta) := \left[\!\left[-\boxed{\theta}- \right]\!\right], \quad \theta_D := 2\cdot\arccos\left(\sqrt{\frac{1}{2}(1+\frac{D}{2})}\right) \quad \text{and} \quad \theta_C := 2\cdot\arccos\left(\sqrt{\frac{1}{2}(1+\frac{C}{2})}\right) \tag{70}
$$

$$\bar{\mathbf{L}}_R := \left[\!\!\left[\right.$$

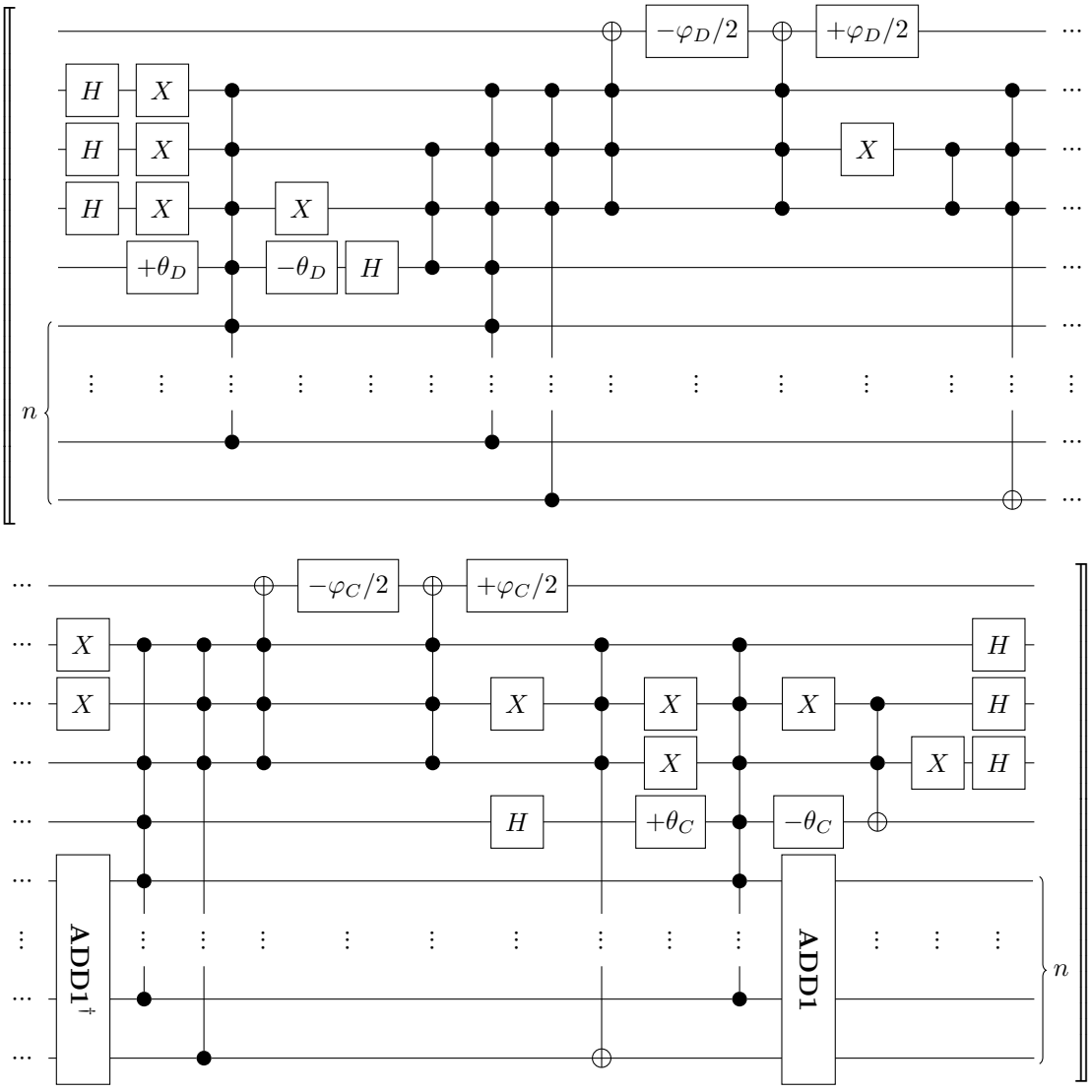


$$\left.\right]\!\!\right] \tag{71}$$

where

$$\varphi_D := 2 \cdot \arccos\left(1 - \frac{D}{2}\right) \quad \text{and} \quad \varphi_C := 2 \cdot \arccos\left(1 - \frac{C}{2}\right) \tag{72}$$

$$\bar{\mathbf{L}}_R = \left[\!\!\left[\right.$$



$$\left.\right]\!\!\right] \tag{73}$$

One caveat regarding the Robin boundary condition in comparison to previous cases, is that the number of $\mathbf{L}_{c,\hat{\sigma}}$ terms is 6 rather than 4, which is not a power of 2. To circumvent this issue, we construct a block-encoding of a zero operator $\mathbf{0}$, denoted $\bar{\mathbf{0}}$ in eq. (69), and think of $\mathbf{L}_R$ as a linear combination of 8 terms: six $\mathbf{L}_{c,\hat{\sigma}}$ and two $\mathbf{0}$ terms of coefficient 1.

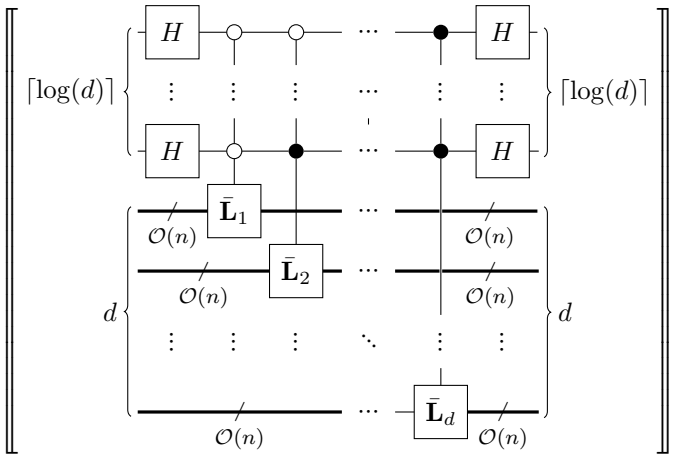
Another caveat is that the block-encoding quantum circuits of $\mathbf{L}_{0,\mathbf{I}}$, $\mathbf{L}_{0,\mathbf{Z}}$ and $\mathbf{L}_{1,\mathbf{I}}$, $\mathbf{L}_{1,\mathbf{Z}}$ can change, depending on the values of D and C, respectively. In this work, we choose $D, C \in [0, 2)$ so that the coefficients $1 \pm D/2$ and $1 \pm C/2$ are

positive. This leads to block-encodings with subnormalization constant 1 for $\mathbf{L}_{0,\mathbf{I}}$, $\mathbf{L}_{1,\mathbf{I}}$, and $1 - D/2$ and $1 - C/2$ for $\mathbf{L}_{0,\mathbf{Z}}$ and $\mathbf{L}_{1,\mathbf{Z}}$, respectively. Conventionally, to further block-encode $\mathbf{L}_{0,\mathbf{Z}}$ and $\mathbf{L}_{1,\mathbf{Z}}$ as a linear combination terms of $\mathbf{L}_R$ with coefficient 1 as in eq. (50), we need a quantum state preparation subroutine that encode $1 - D/2$ and $1 - C/2$. In this work, we keep the previous uniform quantum state preparation, and instead use controlled-$\mathbf{R}_Y$ gates to encode $1 - D/2$ and $1 - C/2$ on another ancilla, as in eq. (71). This results in a quantum circuit $\bar{\mathbf{L}}_R$ block-encoding $\mathbf{L}_R/8$. A similar block-encoding can be constructed for other D, C values, using at most a constant number of additional gates.

4.5 Higher Dimensions

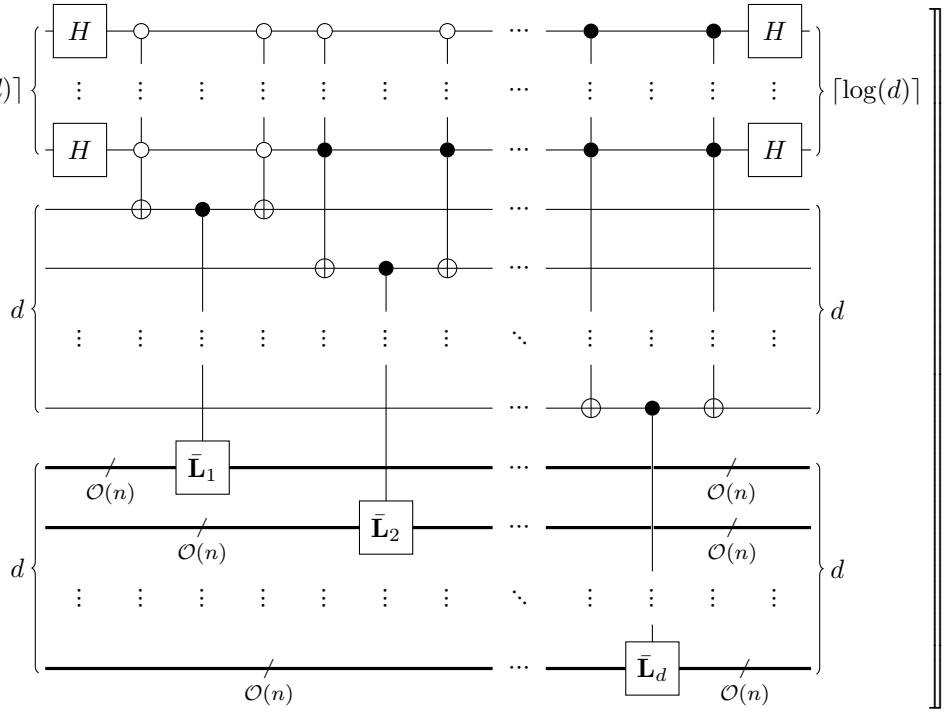
For higher-dimensional cases, i.e., $d > 1$, we provide two alternative block-encoding schemes. The first scheme is given by a unitary operator $\bar{\mathcal{L}}$ as in eq. (74). It block-enocdes $\mathcal{L}/d$ via LCU technique using $\lceil \log(d) \rceil$ additional ancilla qubits. The original $\mathcal{O}(1)$ ancilla qubits used by block-encoding $\bar{\mathbf{L}}_1, \ldots, \bar{\mathbf{L}}_d$ can be shared together. In case that $\log(d) \notin \mathbb{N}$, we need to construct additional block-encodings of a zero operator $\mathbf{0}$ for the remaining controls, as in the Robin case.

$$\bar{\mathcal{L}} := \left[\!\!\left[\quad \right]\!\!\right] \tag{74}$$



The second scheme is given by a unitary operator $\bar{\mathcal{L}}'$ as in eq. (75). It block-encodes $\mathcal{L}/d$ via LCU technique using $\lceil \log(d) \rceil + d$ additional ancilla qubits. Unlike previous scheme, the original $\mathcal{O}(1)$ ancilla qubits from block-encodings $\bar{\mathbf{L}}_1, \ldots, \bar{\mathbf{L}}_d$ cannot be shared together. Therefore, the total number of ancillae is $\lceil \log(n) \rceil + \mathcal{O}(d)$. In case $\log(d) \notin \mathbb{N}$, additional block-encodings of a zero operator $\mathbf{0}$ can be similarly added, as in the previous scheme and the Robin case.

$$\bar{\mathcal{L}}' := \left[\!\!\left[\quad \right]\!\!\right] \tag{75}$$



5 Analyses

This section is divided into two parts. Section 5.1 analyses the computational resources, which include the number of ancilla qubits, gate count, and gate depth, required for the block-encoding quantum circuit of our block-diagonalization method. Section 5.2 analyses the time complexity for solving boundary-value problems, by utilizing the block-encoding quantum circuits and quantum linear solver algorithms to prepare the quantum state representation of the solution.

5.1 Block-Encoding

Table 1a shows the computational resources required for the block-encoding quantum circuits $\bar{\mathbf{L}}_p$, $\bar{\mathbf{L}}_D$, $\bar{\mathbf{L}}_N$ and $\bar{\mathbf{L}}_R$, respectively. These resources include the number of ancilla qubits, quantum gates, and quantum circuits. For brevity, we count $\mathbf{ADD1}^\dagger$ as the same quantum circuit as $\mathbf{ADD1}$. Table 1b provides the description of each quantum gate. As observed, the computational complexities are dominated by quantum gate $\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$ and quantum circuit $\mathbf{ADD1}$, whose complexities depend on the number of qubit $n := \log(N)$ and their implementations in terms of simple quantum gates.

		Single-Qubit Gates			**Multiple-Qubit Gates**		
	Ancillae	$\mathbf{H}$	$\hat{\sigma}$	$\mathbf{R}_{\hat{\sigma}}$	$\mathbf{C}^{\mathcal{O}(1)}\hat{\sigma}$	$\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$	$\mathbf{ADD1}$
Periodic $\bar{\mathbf{L}}_p$	2	4	3	0	2	0	2
Dirichlet $\bar{\mathbf{L}}_D$	3	6	3	0	2	1	2
Neumann $\bar{\mathbf{L}}_N$	3	6	3	0	2	1	2
Robin $\bar{\mathbf{L}}_R$	5	8	12	8	11	4	2

(a) Ancilla qubit, quantum gates, and quantum circuits count

Gate	**Description**
$\mathbf{H}$	Hadamard Gate
$\hat{\sigma}$	Pauli-$\hat{\sigma}$ Gate
$\mathbf{R}_{\hat{\sigma}}$	Pauli-$\hat{\sigma}$ Rotation Gate
$\mathbf{C}^{\mathcal{O}(1)}\hat{\sigma}$	$\mathcal{O}(1)$-Controlled Pauli-$\hat{\sigma}$ Gate
$\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$	$\mathcal{O}(n)$-Controlled Pauli-$\hat{\sigma}$ Gate

(b) Quantum gates description

Table 1: Computational resources required for the block-encoding quantum circuits $\bar{\mathbf{L}}_p$, $\bar{\mathbf{L}}_D$, $\bar{\mathbf{L}}_N$ and $\bar{\mathbf{L}}_R$

Tables 2a and 2b show the computational resource required to implement quantum gate $\mathbf{C}^{\mathcal{O}(n)\hat{\sigma}}$ and quantum circuit $\mathbf{ADD1}$, respectively. Because these implementations consist solely of classical reversible gates, they are given by the depth and count of Toffoli gate used; note that, the depth and count of smaller gates cannot exceed those of Toffoli gate.

We present two implementations of $\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$. The first implementation [14] has $\mathcal{O}(n)$ depth and count. This is achieved by borrowing one qubit which is not part of the quantum gate from the circuit, and thus does not require any additional ancilla qubit. Unfortunately, the second implementation [49] does not provide their complexities. However, our analysis, denoted by * in table 2a, suggests that they have $\mathcal{O}(\log(n))$ depth and $\mathcal{O}(n)$ count, using some $\mathcal{O}(n)$ additional ancillae.

For $\mathbf{ADD1}$, we present 8 alternative implementations. The first [14] once again has $\mathcal{O}(n)$ depth and count, and introduce no additional ancilla qubit by borrowing one existing qubit from the circuit. The rest [50–55] are implementations of quantum addition circuit, which are specialized as a quantum modulo incrementation circuit instead. Unfortunately, except [50], the complexities are given for addition instead of modular incrementation. We denote this by * in table 2b. They have $\mathcal{O}(\log(n))$ depth, and $\mathcal{O}(n)$ or $\mathcal{O}(n\log(n))$ count, using $\mathcal{O}(n)$, $\mathcal{O}(n\log(n))$, or $\mathcal{O}(n/\log(n))$ ancillae. The best overall implementations are [50, 53, 54], whose complexities are $\mathcal{O}(\log(n))$ depth, $\mathcal{O}(n)$ count, and $\mathcal{O}(n)$ ancillae.

Substituting n by $\log(N)$, we can achieve a double-logarithmic depth $\mathcal{O}(\log(\log(N)))$ and a logarithmic $\mathcal{O}(\log(N))$ size implementation of our block-encoding quantum circuit, using implementations [49, 50, 53, 54] and $\mathcal{O}(\log(N))$ ancilla qubits. Together with the original qubits, the total number of qubits is $c\log(N) + \mathcal{O}(1)$ for some $c \leq 12$.

References	Additional Ancillae	Toffoli Depth	Toffoli Count
[14]	0	$\mathcal{O}(n)$	$\mathcal{O}(n)$
[49]*	$\mathcal{O}(n)$	$\mathcal{O}(\log(n))$	$\mathcal{O}(n)$

(a) $\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$

References	Additional Ancillae	Toffoli Depth	Toffoli Count
[14]	0	$\mathcal{O}(n)$	$\mathcal{O}(n)$
[50]	$2n - 2\log(n)$	$2\log(n) + 1$	$5n - 6\log(n) - 3$
[51]*	$3n/\log(n) + n$	$30\log(n)$	$28n$
[52]*	$3n/\log(n) + n$	$18\log(n)$	$7n$
[53]*	$6n - \log(n) + \mathcal{O}(1)$	$4\log(n) + \mathcal{O}(1)$	$8n - 3\log(n) + \mathcal{O}(1)$
[54]*	$12n - 6\log(n) + \mathcal{O}(1)$	$4\log(n) + \mathcal{O}(1)$	$13n - 6\log(n) + \mathcal{O}(1)$
[55]*	$n\log(n) + n + \log(n) + 2$	$2\log(n) + 1$	$1.5n\log(n) - n - 6\log(n)$
[55]*	$n\log(n) + n + \log(n) + 2$	$\log(n) + 1$	$0.5n\log(n)$

(b) **ADD1**

Table 2: Computational resources required for the implementation of $\mathbf{C}^{\mathcal{O}(n)}\hat{\sigma}$ and **ADD1**

Table 3 shows the circuit depth, size, and number of ancillae of the block-encoding $\bar{\mathcal{L}}$ and $\bar{\mathcal{L}}'$ in higher dimensional cases. Both have roughly the same number of gate. However, $\bar{\mathcal{L}}'$ has lower depth and requires more ancilla qubits.

	Total Ancillae	Toffoli Depth	Toffoli Count
Higher Dimensions $\bar{\mathcal{L}}$	$\mathcal{O}(\log(d))$	$\mathcal{O}(d\log(\log(N)))$	$\mathcal{O}(d\log(dN))$
Higher Dimensions $\bar{\mathcal{L}}'$	$\mathcal{O}(d + \log(dN))$	$\mathcal{O}(d + \log(\log(N)))$	$\mathcal{O}(d\log(dN))$

Table 3: Computational resources required for the block-encoding quantum circuit $\bar{\mathcal{L}}$ and $\bar{\mathcal{L}}'$

5.2 Boundary-Value Problems

With our block-encoding quantum circuits, we are ready to solve our boundary-value problems. This is achieved by solving a system of linear equations $\mathcal{L}\left|u\right\rangle = \left|f\right\rangle$ via quantum linear solver algorithms [9–22]. In this article, we use the quantum linear solver from [18], which at present has the best time complexity. It requires two quantum oracles: one block-encoding $\mathcal{L}$, and one preparing a quantum state encoding of $\left|f\right\rangle$. It produces a quantum state encoding of $\mathcal{L}^{-1}\left|f\right\rangle$ to within ϵ error, using $\mathcal{O}(\kappa\log(1/\epsilon))$ calls to both oracles, where the condition number $\kappa := ||\mathcal{L}^{-1}||$ and $||\mathcal{L}|| = 1$.

Table 4 shows the time complexities of quantum linear solver [18], when applied to our boundary-value problems using our block-diagonalization block-encoding $\bar{\mathcal{L}}$ and $\bar{\mathcal{L}}'$. We compare our analyses against those of classical linear solvers, i.e., the conjugate gradient method [5, 6], and quantum linear solvers using other matrix encoding methods [8, 25, 27].

Basically, [8, §7 solving differential equations] refers to the conventional quantum oracle which computes the entries of the matrix given the row and column index, which we use together with quantum linear solver [18]. While, [25] refers to the adaptive finite difference method, with quantum linear solver [10]. And, [27] refers to the block-encoding technique using matrix decomposition similar to our block-diagonalization, with quantum linear solver [18]. There are also several other quantum methods [24, 26] as well; however, their techniques are similar to the aforementioned.

Table 4a shows the time complexities with dependencies in κ the matrix's condition number, the number of grid point in one-dimension N, the dimension d, and the linear solver's solution error ϵ. The primary observation is the dependency in N, where our method is exponentially better than existing quantum methods [8, 25, 27] and superexponentially better

than classical method [5, 6]. For other dependencies, the classical method is quadratically better than all quantum methods in κ, the quantum method [25] is polynomially worse than others in d, and dependency in ϵ is the same for all.

Table 4b shows the time complexities with dependencies in $d, \alpha, \delta, \epsilon$, where δ is the grid discretization error of the domain $[0, 1]^d$, and $\alpha = 0.5$ represents a second-order approximation of the 3-points central difference approximation scheme. They are derived by substituting $\kappa = \mathcal{O}(d)$ and $N = h^{-1} = \mathcal{O}((1/\delta)^{\alpha d})$ into the complexities of table 4a.

The primary dependency is δ, where our methods are exponentially better than existing quantum methods [8, 25, 27], and superexponentially better than existing classical methods [5, 6]. The classical method is polynomially better than other methods by a small degree in d. Our methods are exponentially better in α and slightly better in d than other quantum methods. The dependency in ϵ is once again the same for all. In the end-to-end analysis, we take $\delta = \mathcal{O}(\epsilon)$ [8], i.e., we assume that both errors are of the same order; in which case, our dependencies in $1/\epsilon$ are still polynomially better than [25], almost quadratically better than [27], and exponentially better than existing classical methods [5, 6].

Despite the quantum advantages, there are several caveats in these analyses. First, we assume there exists a quantum state preparation of $|f\rangle$ with at worst $\mathcal{O}(\log(\log(N)))$ time complexity. As far as we know, except for some trivial cases, such quantum subroutine does not exist for general $|f\rangle$ [8]. Secondly, we produce the quantum state representation of $|u\rangle$, rather than the classical state representation, which requires quantum state tomography with $\mathcal{O}(N)$ or $\mathcal{O}(\sqrt{N})$ multiplicative overhead [8]. Similarly, estimating a linear functional of $|u\rangle$ to within ϵ error, requires quantum amplitude estimation with $\mathcal{O}(||u||/\epsilon)$ multiplicative overhead [8]. In either case, any gained quantum speedup diminishes.

References	**Dependencies in** κ, N, d, ϵ
Classical [5, 6]	$\mathcal{O}(\sqrt{\kappa} d N^d \log(1/\epsilon))$
Quantum [8, 25, 27]	$\mathcal{O}(\kappa \operatorname{poly}(d, \log(N)) \log(1/\epsilon))$
Block-Diagonalization $\bar{\mathcal{L}}$	$\mathcal{O}(\kappa d \log(\log(N)) \log(1/\epsilon))$
Block-Diagonalization $\bar{\mathcal{L}}'$	$\mathcal{O}(\kappa \log(\log(N)) \log(1/\epsilon) + \kappa d \log(1/\epsilon))$

(a) κ, N, d, ϵ

References	**Dependencies in** $d, \alpha, \delta, \epsilon$
Classical [5, 6]	$\mathcal{O}(d^{1.5} (1/\delta)^{\alpha d} \log(1/\epsilon))$
Quantum [8, 25, 27]	$\mathcal{O}(d \operatorname{poly}(\alpha d, \log(1/\delta)) \log(1/\epsilon))$
Block-Diagonalization $\bar{\mathcal{L}}$	$\mathcal{O}(d^2 \log(\alpha d \log(1/\delta)) \log(1/\epsilon))$
Block-Diagonalization $\bar{\mathcal{L}}'$	$\mathcal{O}(d \log(\alpha d \log(1/\delta)) \log(1/\epsilon) + d^2 \log(1/\epsilon))$

(b) $d, \alpha, \delta, \epsilon$

Table 4: Time complexity of boundary-value problems solved using classical and quantum linear solver algorithms

6 Discussion

Solving differential equations is one of the primary applications of quantum computing; however, in spite of great advances towards a practical realization of an end-to-end application, several critical obstacles still remain. In this work, we address one of these obstacles, namely the encoding of the matrices representing the discretized forms of a Poisson partial differential equation within a quantum computer as quantum circuits. More specifically, this equation is given within the setting of boundary-value problems with hypercube domains and a number of different boundary conditions, i.e., periodic, Dirichlet, Neumann, and Robin, while the matrices are derived via finite difference method. The primary contribution of our work is the *block-diagonalization* methodology. It provides a common decomposition form for all our matrices, whereby a common procedure for constructing quantum circuits, each of which encoding one of our matrices via the block-encoding technique. Additionally, a simplification of these quantum circuits in terms of elementary quantum gates and circuits can be trivially derived without the use of a sophisticated circuit optimization software, which shows the level of simplicities of these circuits. From the analyses and existing implementation techniques, we show that the computational resources required to construct each of these circuits are very efficient in terms of matrix size. Particularly, in conjunction with quantum linear solver algorithm, we can produce a quantum state solution of our boundary-value problems in time complexity double-logarithmic in the matrix size, using no more than

a constant multiplicative overhead on the number of qubits and the number of gates. This represents an exponential and a superexponential improvement over existing quantum and classical methods, respectively. In terms of the inverse of solution error, it is at least polynomially and exponentially better than quantum and classical methods, respectively.

Our work represents an important step towards a practical realization of the uses of quantum computing for solving differential equations, by tackling one of the major obstacles in achieving this goal. We show that our methodology improves upon the existing methods in the literature, in terms of time complexity. In fact, as far as we know, this is the first method, which shows a double-logarithmic time complexity in the problem of encoding matrices from finite difference method. Several existing works are shown to have similar matrix decompositions or quantum circuits; however, our analyses are the first to show that a quantum circuit with a double-logarithmic depth in terms of matrix size, can be implemented with a small overhead in the number of ancilla qubits. Finally, this is but a first step for block-diagonalization methodology. We believe that it has the potential to be extended towards more complicated problems, or at least inspire a similar methodology to be developed, catering towards the specificities and complexities of each problem. These problems are not limited to solving differential equations; they can also include other quantum computing applications such as physics, chemistry, machine learning, continuous and combinatorial optimization.

The followings are the primary directions of future works, which are related to our block-diagonalization methodology.

Finite Difference Method The first obvious extension of block-diagonalization is to study its applications towards more complicated discretization schemes, or boundary-value problems, within the framework of finite difference method. For instance, it would be interesting to see whether it is possible to derive the block-diagonalization form for other discretization schemes, especially for higher-order schemes; or whether such derived forms, if found, also lead to a double-logarithmic depth quantum circuit implementation. Another interesting extension is for more complicated domains and domain boundaries, because the structure of their corresponding matrices are not as simple as those of hypercube domains, where we are able to derive the block-diagonalization form quite easily. And, the most important extensions are towards more complicated equations, including elliptic equations, hyperbolic equations, parabolic equations, and non-linear ordinary and partial differential equations, which are generally more difficult to solve.

General Numerical Methods Another line of extensions is towards other numerical methods for solving differential equations, such as finite element method, finite volume method, and spectral method. All these methods have a long history, and are developed to compensate for the weaknesses of finite difference method; for instance, for problems with domains of complex geometries. However, the majority of them results in a system of linear equations much like finite difference method, albeit one with a completely different structure. Then, the question is again whether it is also possible to derive a similar decomposition as block-diagonalization of these matrices, or whether such decomposition has a double-logarithmic depth implementation. Generally, these matrices are more complicated than those of finite difference method, especially their higher dimensional extensions. Together, they represent the majority of use cases in all the science and engineering fields; thus, their studies would greatly benefit a significant number of applications.

Solving Differential Equations Rather than extending the methodology towards other use cases, it is also imperative to study other quantum computing problems that can greatly aid in the practical realization of a complete end-to-end application of quantum algorithms towards solving differential equations. As pointed out several times throughout this article, the encoding of problem matrices is but one of the major obstacle towards this goal. For instance, to achieve an end-to-end double-logarithmic time complexity, we also need a quantum state preparation with a double-logarithmic depth in the matrix size, or a quantum linear solver algorithm with a double-logarithmic depth in the inverse of solution error. Similarly, the study of a quantum preconditioner would also greatly benefit the quantum linear solver algorithm. However, the most difficult challenge, we believe, is the measurement of the quantum state solution to obtain either the classical state solution or one of its properties, which has a linear depth in either the matrix size or the inverse of solution error. Together, they represent the primary obstacles towards a complete end-to-end practical realization.

Acknowledgement

The authors would like to thank David Danan, Pierre-Alain Boucard, and François Jouve, for their helpful comments, suggestions and discussions. All of whom help contribute towards improving the quality of both our work and article. This project is as well supported by the French government's aid in the framework of PIA (Programme d'Investissement d'Avenir) for Institut de Recherche Technologique SystemX.

References

[1] Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Society, 2022.

[2] Gordon D Smith. *Numerical solution of partial differential equations: finite difference methods*. Oxford university press, 1985.

[3] Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, 2007.

[4] James William Thomas. *Numerical partial differential equations: finite difference methods*, volume 22. Springer Science & Business Media, 2013.

[5] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[6] Nisheeth K Vishnoi et al. Lx= b. *Foundations and Trends® in Theoretical Computer Science*, 8(1–2):1–141, 2013.

[7] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[8] Alexander M Dalzell, Sam McArdle, Mario Berta, Przemyslaw Bienias, Chi-Fang Chen, András Gilyén, Connor T Hann, Michael J Kastoryano, Emil T Khabiboulline, Aleksander Kubica, et al. Quantum algorithms: A survey of applications and end-to-end complexities. *arXiv preprint arXiv:2310.03011*, 2023.

[9] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[10] Andrew M Childs, Robin Kothari, and Rolando D Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017.

[11] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. 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*, pages 193–204, 2019.

[12] John M Martyn, Zane M Rossi, Andrew K Tan, and Isaac L Chuang. Grand unification of quantum algorithms. *PRX Quantum*, 2(4):040203, 2021.

[13] Yu Tong, Dong An, Nathan Wiebe, and Lin Lin. Fast inversion, preconditioned quantum linear system solvers, fast green's-function computation, and fast evaluation of matrix functions. *Physical Review A*, 104(3):032422, 2021.

[14] Craig Gidney. Using Quantum Gates instead of Ancilla Bits — algassert.com. https://algassert.com/circuits/2015/06/22/Using-Quantum-Gates-instead-of-Ancilla-Bits.html. [Accessed 17-07-2024].

[15] Yiğit Subaşı, Rolando D Somma, and Davide Orsucci. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Physical review letters*, 122(6):060504, 2019.

[16] Dong An and Lin Lin. Quantum linear system solver based on time-optimal adiabatic quantum computing and quantum approximate optimization algorithm. *ACM Transactions on Quantum Computing*, 3(2):1–28, 2022.

[17] Lin Lin and Yu Tong. Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems. *Quantum*, 4:361, 2020.

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

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

[20] Hsin-Yuan Huang, Kishor Bharti, and Patrick Rebentrost. Near-term quantum algorithms for linear systems of equations. *arXiv preprint arXiv:1909.07344*, 2019.

[21] Carlos Bravo-Prieto, Ryan LaRose, Marco Cerezo, Yigit Subasi, Lukasz Cincio, and Patrick J Coles. Variational quantum linear solver. *arXiv preprint arXiv:1909.05820*, 2019.

[22] Xiaosi Xu, Jinzhao Sun, Suguru Endo, Ying Li, Simon C Benjamin, and Xiao Yuan. Variational algorithms for linear algebra. *Science Bulletin*, 66(21):2181–2188, 2021.

[23] Scott Aaronson. Read the fine print. *Nature Physics*, 11(4):291–293, 2015.

[24] Yudong Cao, Anargyros Papageorgiou, Iasonas Petras, Joseph Traub, and Sabre Kais. Quantum algorithm and circuit design solving the poisson equation. *New Journal of Physics*, 15(1):013021, 2013.

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

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

[27] Tyler Kharazi, Ahmad M Alkadri, Jin-Peng Liu, Kranthi K Mandadapu, and K Birgitta Whaley. Explicit block encodings of boundary value problems for many-body elliptic operators. *arXiv preprint arXiv:2407.18347*, 2024.

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

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

[30] Pedro CS Costa, Stephen Jordan, and Aaron Ostrander. Quantum algorithm for simulating the wave equation. *Physical Review A*, 99(1):012323, 2019.

[31] Noah Linden, Ashley Montanaro, and Changpeng Shao. Quantum vs. classical algorithms for solving the heat equation. *Communications in Mathematical Physics*, 395(2):601–641, 2022.

[32] Sarah K Leyton and Tobias J Osborne. A quantum algorithm to solve nonlinear differential equations. *arXiv preprint arXiv:0812.4423*, 2008.

[33] Dominic W Berry. High-order quantum algorithm for solving linear differential equations. *Journal of Physics A: Mathematical and Theoretical*, 47(10):105301, 2014.

[34] Dominic W Berry, Andrew M Childs, Aaron Ostrander, and Guoming Wang. Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Communications in Mathematical Physics*, 356:1057–1081, 2017.

[35] Andrew M Childs and Jin-Peng Liu. Quantum spectral methods for differential equations. *Communications in Mathematical Physics*, 375(2):1427–1457, 2020.

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

[37] Dong An, Jin-Peng Liu, Daochen Wang, and Qi Zhao. A theory of quantum differential equation solvers: limitations and fast-forwarding. *arXiv preprint arXiv:2211.05246*, 2022.

[38] Hari Krovi. Improved quantum algorithms for linear and nonlinear differential equations. *Quantum*, 7:913, 2023.

[39] Jin-Peng Liu, Dong An, Di Fang, Jiasu Wang, Guang Hao Low, and Stephen Jordan. Efficient quantum algorithm for nonlinear reaction–diffusion equations and energy estimation. *Communications in Mathematical Physics*, 404(2):963–1020, 2023.

[40] Frank Gaitan. Finding flows of a navier–stokes fluid through quantum computing. *npj Quantum Information*, 6(1):61, 2020.

[41] Frank Gaitan. Finding solutions of the navier-stokes equations through quantum computing—recent progress, a generalization, and next steps forward. *Advanced Quantum Technologies*, 4(10):2100055, 2021.

[42] Furkan Oz, Rohit KSS Vuppala, Kursat Kara, and Frank Gaitan. Solving burgers' equation with quantum computing. *Quantum Information Processing*, 21(1):30, 2022.

[43] B David Clader, Alexander M Dalzell, Nikitas Stamatopoulos, Grant Salton, Mario Berta, and William J Zeng. Quantum resources required to block-encode a matrix of classical data. *IEEE Transactions on Quantum Engineering*, 3:1–23, 2022.

[44] Quynh T Nguyen, Bobak T Kiani, and Seth Lloyd. Block-encoding dense and full-rank kernels using hierarchical matrices: applications in quantum numerical linear algebra. *Quantum*, 6:876, 2022.

[45] Daan Camps, Lin Lin, Roel Van Beeumen, and Chao Yang. Explicit quantum circuits for block encodings of certain sparse matrices. *SIAM Journal on Matrix Analysis and Applications*, 45(1):801–827, 2024.

[46] Christoph Sünderhauf, Earl Campbell, and Joan Camps. Block-encoding structured matrices for data input in quantum computing. *Quantum*, 8:1226, 2024.

[47] Haoya Li, Hongkang Ni, and Lexing Ying. On efficient quantum block encoding of pseudo-differential operators. *Quantum*, 7:1031, 2023.

[48] Dorit Aharonov and Amnon Ta-Shma. Adiabatic quantum state generation and statistical zero knowledge. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 20–29, 2003.

[49] Craig Gidney. Halving the cost of quantum addition. *Quantum*, 2:74, 2018.

[50] Thomas G Draper, Samuel A Kutin, Eric M Rains, and Krysta M Svore. A logarithmic-depth quantum carry-lookahead adder. *arXiv preprint quant-ph/0406142*, 2004.

[51] Yasuhiro Takahashi and Noboru Kunihiro. A fast quantum circuit for addition with few qubits. *Quantum Information & Computation*, 8(6):636–649, 2008.

[52] Yasuhiro Takahashi, Seiichiro Tani, and Noboru Kunihiro. Quantum addition circuits and unbounded fan-out. *arXiv preprint arXiv:0910.2530*, 2009.

[53] Siyi Wang, Anubhab Baksi, and Anupam Chattopadhyay. A higher radix architecture for quantum carry-lookahead adder. *Scientific Reports*, 13(1):16338, 2023.

[54] Siyi Wang and Anupam Chattopadhyay. Reducing depth of quantum adder using ling structure. In *2023 IFIP/IEEE 31st International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 1–6. IEEE, 2023.

[55] Siyi Wang, Suman Deb, Ankit Mondal, and Anupam Chattopadhyay. Optimal toffoli-depth quantum adder. *arXiv preprint arXiv:2405.02523*, 2024.