IOP Publishing Quantum Sci. Technol. 9 (2024) 035010 https://doi.org/10.1088/2058-9565/ad3d7f

Quantum Science and Technology

OPEN ACCESS

RECEIVED
2 November 2023

REVISED
18 March 2024

ACCEPTED FOR PUBLICATION
11 April 2024

PUBLISHED
22 April 2024

PAPER

Efficient quantum algorithm for all quantum wavelet transforms

Mohsen Bagherimehrab[1,2,*] and Alán Aspuru-Guzik[1,2,3,4,5,6]

[1] Department of Chemistry, Chemical Physics Theory Group, University of Toronto, Toronto, Ontario, Canada
[2] Department of Computer Science, University of Toronto, Toronto, Ontario, Canada
[3] Vector Institute for Artificial Intelligence, Toronto, Ontario, Canada
[4] Department of Chemical Engineering & Applied Chemistry, University of Toronto, Toronto, Ontario, Canada
[5] Department of Materials Science & Engineering, University of Toronto, Toronto, Ontario, Canada
[6] Lebovic Fellow, Canadian Institute for Advanced Research, Toronto, Ontario, Canada
[*] Author to whom any correspondence should be addressed.

E-mail: mohsen.bagherimehrab@utoronto.ca

Keywords: quantum wavelet transform, quantum algorithms, quantum computational complexity

Abstract

Wavelet transforms are widely used in various fields of science and engineering as a mathematical tool with features that reveal information ignored by the Fourier transform. Unlike the Fourier transform, which is unique, a wavelet transform is specified by a sequence of numbers associated with the type of wavelet used and an order parameter specifying the length of the sequence. While the quantum Fourier transform, a quantum analog of the classical Fourier transform, has been pivotal in quantum computing, prior works on quantum wavelet transforms (QWTs) were limited to the second and fourth order of a particular wavelet, the Daubechies wavelet. Here we develop a simple yet efficient quantum algorithm for executing any wavelet transform on a quantum computer. Our approach is to decompose the kernel matrix of a wavelet transform as a linear combination of unitaries (LCU) that are compilable by easy-to-implement modular quantum arithmetic operations and use the LCU technique to construct a probabilistic procedure to implement a QWT with a *known* success probability. We then use properties of wavelets to make this approach deterministic by a few executions of the amplitude amplification strategy. We extend our approach to a multilevel wavelet transform and a generalized version, the packet wavelet transform, establishing computational complexities in terms of three parameters: the wavelet order M, the dimension N of the transformation matrix, and the transformation level d. We show the cost is logarithmic in N, linear in d and superlinear in M. Moreover, we show the cost is independent of M for practical applications. Our proposed QWTs could be used in quantum computing algorithms in a similar manner to their well-established counterpart, the quantum Fourier transform.

1. Introduction

As a solid alternative to the Fourier transform, wavelet transforms are a relatively new mathematical tool with diverse utility that has generated much interest in various fields of science and engineering over the past four decades. Although wavelet-like functions have existed for over a century, a prominent example is what is now known as the Haar wavelet. The interest is due to the attractive features of wavelets [1–4]. Such functions are differentiable, up to a particular order, and are local in both the real and dual spaces. They provide an exact representation for polynomials up to a certain order, and a simple yet optimal preconditioner for a large class of differential operators. Crucially, wavelets provide structured and sparse representations for vectors, functions, or operators, enabling data compression and constructing faster algorithms. These appealing features of wavelets and their associated transforms make them advantageous for numerous applications in classical computing over their established counterpart, the Fourier transform.

With the wavelet transforms' diverse utility and extensive use in classical computing, a natural expectation is that a quantum analog of such transforms will find applications in quantum computing, especially for developing faster quantum algorithms and quantum data compression. Wavelets have already been used in quantum physics and computation [5–12]. However, prior works on developing a quantum analog for wavelet transforms are limited to a few representative cases [13–17]. In contrast, the quantum Fourier transform, a quantum analog of the classical Fourier transform, has been extensively used in quantum computing as a critical subroutine for many quantum algorithms.

Unlike the Fourier transform, a wavelet transform is not unique and is specified by the type of wavelet used and an order parameter. In particular, a wavelet transform is defined by a sequence of numbers, known as the filter coefficients, associated with the type of wavelet used and an even number known as the order of the wavelet that specifies the length of the sequence. Given the sequence, a unitary matrix known as the kernel matrix of the wavelet transform is constructed, the application of which on a vector yields the single-level wavelet transform of the vector. Such a transform partitions the vector into two components: a low-frequency or average component and high-frequency or difference component (see figure 1). To expose the multi-scale structure of the vector, or a function for that matter, the wavelet transform is recursively applied to the low-frequency component, yielding the multi-level wavelet transform of the vector. The wavelet packet transform is a generalization of the multi-level wavelet transform, in which the wavelet transform is recursively applied to both the low- and high-frequency components. We refer to a quantum analog of the (single-) multi-level and packet wavelet transforms as the (single-) multi-level and packet quantum wavelet transforms (QWTs), respectively.

This paper proposes and analyzes a conceptually simple and computationally efficient quantum algorithm for executing single-level, multi-level, and packet QWTs associated with any wavelet and any order on a quantum computer. Our approach is based on decomposing a unitary associated with a wavelet transform in terms of a linear combination of a finite number of simple-to-implement unitaries and using the linear combination of unitaries (LCU) technique [18] to implement the original unitary. Specifically, we decompose the kernel matrix of the wavelet transform, associated with a wavelet of order M, as a linear combination of M simple-to-implement unitaries and, by the LCU technique, construct a probabilistic procedure for implementing the single-level QWT. The success probability of this approach is a known constant by properties of the wavelet filters. We use this known success probability to make the implementation deterministic using a single ancilla qubit and a few rounds of amplitude amplification.

Having an implementation for the single-level QWT and recursive formulae describing the multi-level and packet wavelet transforms based on single-level transforms, we construct quantum algorithms for multi-level and packet QWTs. We establish the computational complexity of these transformations in terms of three parameters: the wavelet order M, the dimension of the wavelet-transform matrix N, and the level of the wavelet transform d. Without loss of generality, we assume that our main parameter of interest N is a power of two, as $N = 2^n$, and report the computational costs with respect to n, the number of qubits that the wavelet transforms act on.

We summarize our main results on computational costs of the described transformations in the following three theorems. We establish these theorems in subsequent sections after providing a detailed description of our algorithms.

Theorem 1 (Single-level QWT with logarithmic gate cost). *A single-level QWT on n qubits, associated with a wavelet of order M, can be implemented using $\lceil \log_2 M \rceil + 1$ ancilla qubits and $\mathcal{O}(n) + \mathcal{O}(M^{3/2})$ TOFFOLI and elementary one- and two-qubit gates.*

Theorem 2 (Multi-level QWT with multiplicative gate cost). *A d-level QWT on n qubits can be achieved using $\lceil \log_2 M \rceil + 2$ ancilla qubits and $\mathcal{O}(dn) + \mathcal{O}(dM^{3/2})$ TOFFOLI and elementary one- and two-qubit gates.*

Theorem 3 (Packet QWT). *A d-level packet QWT on n qubits can be achieved using $\lceil \log_2 M \rceil + 1$ ancilla qubits and $\mathcal{O}(dn - d^2/2) + \mathcal{O}(dM^{3/2})$ TOFFOLI and elementary one- and two-qubit gates.*

We remark that the number of levels for the multi-level or packet QWTs is upper bounded by n, i.e. $d \leqslant n$. Hence, as a corollary of theorems 2 and 3 , the gate cost for these transformations is at most quadratic in $n = \log_2 N$. We show that the gate costs reported in the above theorems are independent of M for practical applications and only a few number of ancilla qubits suffice to implement the multi-level and packet QWTs. We discuss allowable range for the order parameter M versus the values used in practical applications in the discussion section.

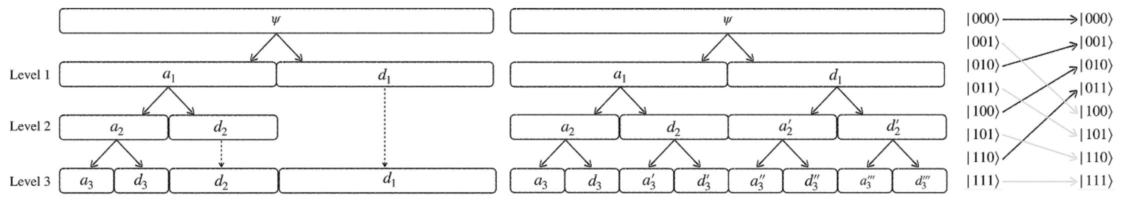


Figure 1. Visualization for (Left) multi-level wavelet transform, (Middle) the packet wavelet transform, and (Right) action of quantum perfect shuffle transform in equation (32) on three-qubit basis states. Three-level wavelet transforms are shown for simplicity. In the first level, the size-N vector ψ is partitioned into two size-$N/2$ vectors: an average vector $a_1 = H\psi$ and a difference vector $d_1 = G\psi$ with H and G defined in equation (2). For the quantum (packet) wavelet transform, the components of ψ are amplitudes of a quantum state. The wavelet transform is recursively applied to the average vector in the multi-level wavelet transform. In contrast, the packet transform applies the wavelet transform to both the average and difference vectors.

The rest of this paper proceeds as follows. We begin by describing the notation we use throughout the paper. Then we detail our approach for implementing a single-level QWT by simple modular arithmetic operations in section 2. We describe the multi-level and packet QWT in section 3, followed by detailed complexity analysis for our algorithms in section 4. Finally, we discuss our results and conclude in section 5.

Notation: We refer to $A \in \mathbb{C}^{2^n \times 2^n}$ as n-qubit matrix and denote the n-qubit identity by $\mathbb{1}_n$. Throughout the paper, we use the symbol M for the wavelet order and $m = \lceil \log_2 M \rceil$. The wavelet order is an even positive number as $M = 2\mathcal{K}$ with $\mathcal{K}$ a positive integer called the wavelet index; the symbol $\mathcal{K}$ is used for $M/2$. We use zero indexing for iterable mathematical objects such as vectors and matrices. Qubits of an n-qubit register is ordered from right to left, i.e. the rightmost (leftmost) qubit in $|q_{n-1},\dots,q_1,q_0\rangle$ representing the state of an n-qubit register that encodes the binary representation of an integer q is the first (last) qubit. The first and last qubits are also referred to as the least-significant bit (LSB) and the most-significant bit (MSB). Qubits in a quantum circuit are ordered from bottom to top: the bottom qubit is the LSB and the top qubit is the MSB.

2. Single-level QWT

This section describes our algorithm for executing a single-level wavelet transform on a quantum computer. Such a transformation is specified by a kernel matrix. We describe this matrix in section 2.1 and decompose it as a linear combination of a finite number of unitaries. The decomposition enables a prepare-select-unprepare-style procedure for probabilistic implementation of the desired transformation that we cover in section 2.2. In section 2.3, we describe how purposefully reducing the success probability yields a perfect amplitude amplification. Finally, in sections 2.4 and 2.5, we provide a compilation for the select and prepare operations based on simple-to-implement modular arithmetic operations.

2.1. The wavelet kernel matrix as a LCU

We begin this subsection by briefly describing the kernel matrix associated with a wavelet transform. We refer to [4, chapter 2.1] for a review of wavelet formalism and how this matrix is constructed. The kernel matrix W of a wavelet transform is specified by the wavelet filter coefficients: a sequence of numbers $(h_0, h_1, \dots, h_M)$ that depend on the type of wavelet and satisfy

$$\sum_{\ell=0}^{M-1} h_\ell = \sqrt{2}, \qquad \sum_{\ell=0}^{M-1} h_\ell^2 = 1, \tag{1}$$

where the even number M is the wavelet order. Specifically, the $2^n \times 2^n$ kernel matrix W is comprised of $2^{n-1} \times 2^n$ matrices H and G as

$$W = \begin{bmatrix} H \\ G \end{bmatrix}, \quad H_{ij} = h_{j-2i\,(\bmod 2^n)}, \quad G_{ij} = g_{j-2i\,(\bmod 2^n)}, \quad g_\ell = (-)^\ell h_{M-1-\ell}; \tag{2}$$

an example of the kernel matrix W for a forth-order wavelet ($M=4$) is as follows

$$W=\begin{bmatrix}
h_0 & h_1 & h_2 & h_3 & 0 & 0 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & h_0 & h_1 & h_2 & h_3 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & h_0 & h_1 & \cdots & 0 & 0 & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & h_0 & h_1 & h_2 & h_3\\
h_2 & h_3 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & h_0 & h_1\\
h_3 & -h_2 & h_1 & -h_0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & h_3 & -h_2 & h_1 & -h_0 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & h_3 & -h_2 & \cdots & 0 & 0 & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & h_3 & -h_2 & h_1 & -h_0\\
h_1 & -h_0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & h_3 & -h_2
\end{bmatrix},$$

$$U=\begin{bmatrix}
h_0 & h_1 & h_2 & h_3 & 0 & 0 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & h_0 & h_1 & h_2 & h_3 & \cdots & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & h_0 & h_1 & \cdots & 0 & 0 & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & h_0 & h_1 & h_2 & h_3\\
h_2 & h_3 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & h_0 & h_1\\
h_1 & -h_0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & h_3 & -h_2\\
h_3 & -h_2 & h_1 & -h_0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
0 & 0 & 0 & 0 & \cdots & h_1 & -h_0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & \cdots & h_3 & -h_2 & h_1 & -h_0 & 0 & 0\\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & h_3 & -h_2 & h_1 & -h_0
\end{bmatrix} \tag{3}$$

The unitary matrix U here is a modification of the unitary W that we use for decomposing W as a LCU. To this end, let us first define the circular downshift and upshift permutation operations as

$$S_n^{\downarrow} := \sum_{j=0}^{2^n-1} |j+1 \bmod 2^n\rangle\langle j| = \begin{bmatrix} & & & 1\\ 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 & \end{bmatrix}, \quad S_n^{\uparrow} := \sum_{j=0}^{2^n-1} |j-1 \bmod 2^n\rangle\langle j| = \begin{bmatrix} & 1 & & \\ & & 1 & \\ & & & \ddots \\ & & & & 1\\ 1 & & & \end{bmatrix}, \tag{4}$$

where the matrix size is $2^n \times 2^n$. Note that these operations are inverse of each other and their action on n-qubit basis state $|j\rangle$ is

$$S_n^{\downarrow}|j\rangle = |j+1 \bmod 2^n\rangle, \quad S_n^{\uparrow}|j\rangle = |j-1 \bmod 2^n\rangle. \tag{5}$$

Upon acting on a vector with 2^n components, $S_n^{\downarrow}/S_n^{\uparrow}$ shifts the vector's components one place downward/upward with wraparound. Similarly, when acting on a matrix with 2^n rows from the left side, $S_n^{\downarrow}/S_n^{\uparrow}$ shifts the rows of the matrix one place downward/upward with wraparound.

To construct an LCU decomposition for the n-qubit unitary W, the kernel matrix associated with a wavelet of order $M=2\mathcal{K}$, first we transform it into another unitary U by $\mathcal{K}-1$ downshift permutations of the rows in the lower half of W. Specifically, we transform W as

$$W=\begin{bmatrix} H\\ G\end{bmatrix} \to U=\begin{bmatrix} H\\ G'\end{bmatrix}, \quad G' := \left(S_{n-1}^{\downarrow}\right)^{\mathcal{K}-1} G, \tag{6}$$

where G' is obtained by $\mathcal{K}-1$ downshift permutations of the rows of G and its elements are

$$G'_{i,j} = (-1)^{2i+2-j} h_{2i+2-j}. \tag{7}$$

Let us now represent $\mathcal{K}-1$ upnshift permutations on n qubits by $\textsc{ushift}_n$ with the action

$$\textsc{ushift}_n|j\rangle := |j+\mathcal{K}-1 \bmod 2^n\rangle \tag{8}$$

on n-qubit basis state $|j\rangle$. Then we have

$$W = (|0\rangle\langle 0| \otimes \mathbb{1}_{n-1} + |1\rangle\langle 1| \otimes \textsc{ushift}_{n-1})\, U := \Lambda_1\left(\textsc{ushift}_{n-1}\right) U, \tag{9}$$

i.e. W is obtained by $\mathcal{K}-1$ upshift permutations of the rows in the lower half of U.

We now decompose the unitary U as a linear combination of M unitaries as

$$U = \sum_{\ell=0}^{M-1} h_\ell U_\ell, \quad U_\ell := \begin{cases} P_\ell & \text{if}\,\ell\,\text{is odd,} \\ (Z \otimes \mathbb{1}_{n-1})\, P_\ell & \text{if}\,\ell\,\text{is even,} \end{cases} \tag{10}$$

where Z is the Pauli-Z operator and the unitary

$$P_\ell := \sum_{j=0}^{N/2-1} |j\rangle\langle 2j+\ell \bmod N| + |N/2+j\rangle\langle 2j+1-\ell \bmod N| \tag{11}$$

is a permutation matrix that is obtained from U as follows: all entries of U with value $\pm h_\ell$ are replaced with 1 and all other nonzero entries are replaced with 0. Because W is unitarily equivalent to U by equation (9), the LCU decomposition in equation (10) provides a similar LCU decomposition for W.

2.2. Probabilistic implementation for the single-level QWT

The decomposition in equation (10) enables a prepare-select-unprepare-style method [18] for probabilistic implementation of U. To this end, let

$$\textsc{prep}|0^m\rangle := \frac{1}{\sqrt{h}} \sum_{\ell=0}^{M-1} \sqrt{|h_\ell|}|\ell\rangle, \quad \textsc{unprep}^\dagger|0^m\rangle := \frac{1}{\sqrt{h}} \sum_{\ell=0}^{M-1} \text{sign}\left(h_\ell\right) \sqrt{|h_\ell|}|\ell\rangle, \quad h := \sum_{\ell=0}^{M-1} |h_\ell|, \tag{12}$$

where $m = \lceil \log_2 M \rceil$ is the number of ancilla qubits, and let $\textsc{select}$ be an operation such that

$$\textsc{select}|\ell\rangle|j\rangle := |\ell\rangle U_\ell |j\rangle \tag{13}$$

with U_ℓ defined in equation (10). Then for any n-qubit state we have

$$\left(\textsc{unprep} \otimes \mathbb{1}_n\right) \textsc{select} \left(\textsc{prep} \otimes \mathbb{1}_n\right) |0^m\rangle|\psi\rangle = \frac{1}{h}|0^m\rangle U|\psi\rangle + \sqrt{1-\frac{1}{h^2}}|\perp\rangle, \tag{14}$$

where $|\perp\rangle$ is an $(m+n)$-qubit state such that $(\langle 0^m| \otimes \mathbb{1}_n)|\perp\rangle = 0$. This equation follows as

$$|0^m\rangle|\psi\rangle \xrightarrow{\text{PREP}\otimes\mathbb{1}_n} \frac{1}{\sqrt{h}} \sum_\ell \sqrt{|h_\ell|}|\ell\rangle|\psi\rangle \tag{15}$$

$$\xrightarrow{\text{SELECT}} \frac{1}{\sqrt{h}} \sum_\ell \sqrt{|h_\ell|}|\ell\rangle U_\ell|\psi\rangle \tag{16}$$

$$\xrightarrow{\text{UNPREP}\otimes\mathbb{1}_n} \frac{1}{h}|0^m\rangle U|\psi\rangle + \sqrt{1-\frac{1}{h^2}}|\perp\rangle, \tag{17}$$

where the last line follows by projecting the ancilla qubits to $|0^m\rangle$ state, i.e.

$$\left(\langle 0^m| \otimes \mathbb{1}_n\right)\left(\textsc{unprep} \otimes \mathbb{1}_n\right) \frac{1}{\sqrt{h}} \sum_\ell \sqrt{|h_\ell|}|\ell\rangle U_\ell|\psi\rangle = \frac{1}{h}\sum_{\ell'\ell} \text{sign}\left(h_{\ell'}\right) \sqrt{|h_{\ell'} h_\ell|}\langle \ell'|\ell\rangle U_\ell|\psi\rangle \tag{18}$$

$$= \frac{1}{h}\sum_\ell h_\ell U_\ell|\psi\rangle = \frac{1}{h}U|\psi\rangle. \tag{19}$$

Equation (14) yields a probabilistic implementation for U. Because U and W are unitarily equivalent, by equation (9), we also have a probabilistic implementation for W with the same success probability. In particular, let us define a probabilistic QWT as

$$\textsc{pqwt} := \left(\mathbb{1}_m \otimes \Lambda_1\left(\textsc{ushift}_{n-1}\right)\right)\left(\textsc{unprep} \otimes \mathbb{1}_n\right) \textsc{select} \left(\textsc{prep} \otimes \mathbb{1}_n\right), \tag{20}$$

then we have

$$\textsc{pqwt}|0^m\rangle|\psi\rangle = \sin(\alpha)\,|0^m\rangle W|\psi\rangle + \cos(\alpha)\,|\perp'\rangle, \quad \sin(\alpha) := 1/h, \tag{21}$$

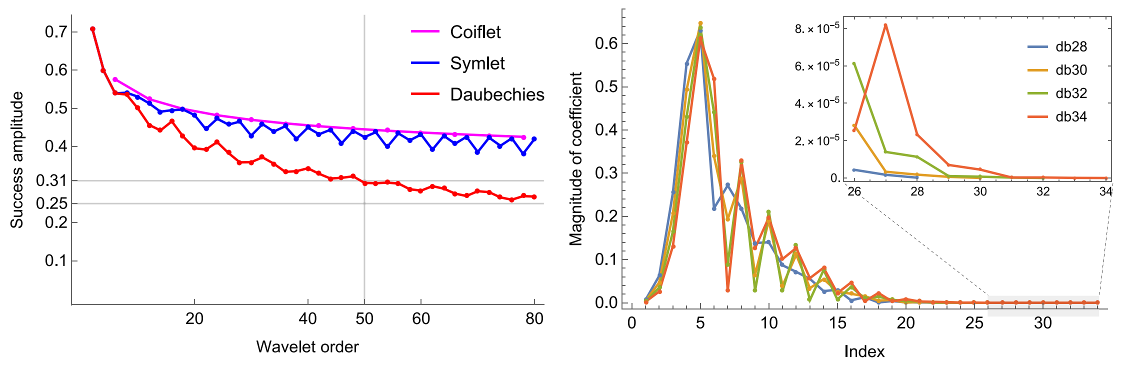


Figure 2. (Left) The success amplitude of the probabilistic implementation in equation (21) for a single-level QWT for commonly used wavelets. The success amplitude is known and is greater than 1/4 for a wide range of wavelet order; it is greater than 0.31 for the range of wavelet order used in practical applications. For perfect amplitude amplification, the former (latter) value needs three (two) rounds of amplitude amplification. (Right) The magnitude of wavelet coefficient $|h_\ell|$ as a function of the index ℓ for the Debauchees wavelet with order $M = 28, 30, 32, 34$. The zoomed-in part shows the wavelet coefficients with higher indexes are negligibly small, and the number of small coefficients increases by increasing the wavelet order. The magnitude of wavelet coefficients for other wavelets has a similar pattern.

with the $(m+n)$-qubit state $|\perp'\rangle := \mathbb{1}_m \otimes \Lambda_1(\textsc{ushift}_{n-1})|\perp\rangle$ and the $|1\rangle$-controlled unitary $\Lambda_1(\textsc{ushift})$ defined in equation (9). The success amplitude of this approach is known and its value is $1/h$. As shown in figure 2(Left), the success amplitude is greater than 1/4 for a wide range of wavelet order.

We now present an alternative approach for a probabilistic implementation of the single-level QWT. The state-preparation of this approach is simpler and could be preferred in practical applications. Instead of preparing the state with square-root coefficients by PREP in equation (12), in this approach we use the operation LINPREP defined as

$$\textsc{linprep}|0^m\rangle := \sum_{\ell=0}^{M-1} h_\ell|\ell\rangle, \tag{22}$$

which prepares the state with linear coefficients. For any n-qubit state $|\psi\rangle$ we then have

$$\left(H^{\otimes m}\otimes\mathbb{1}_n\right)\textsc{select}\left(\textsc{linprep}\otimes\mathbb{1}_n\right)|0^m\rangle|\psi\rangle = \sin(\alpha)|0^m\rangle U|\psi\rangle + \cos(\alpha)|\perp\rangle, \quad \sin(\alpha) := 1/\sqrt{M}, \tag{23}$$

where $|\perp\rangle$ and SELECT are as those in equation (14). This equation follows as

$$|0^m\rangle|\psi\rangle \xrightarrow{\text{LINPREP}\otimes\mathbb{1}_n} \sum_\ell h_\ell|\ell\rangle|\psi\rangle \xrightarrow{\text{SELECT}} \sum_\ell h_\ell|\ell\rangle U_\ell|\psi\rangle \xrightarrow{H^{\otimes m}\otimes\mathbb{1}_n} \sin(\alpha)|0^m\rangle U|\psi\rangle + \cos(\alpha)|\perp\rangle, \tag{24}$$

where the last step is obtained by projecting the ancilla qubits to $|0^m\rangle$ state, i.e.

$$\left(\langle 0^m|\otimes\mathbb{1}_n\right)\left(H^{\otimes m}\otimes\mathbb{1}_n\right)\sum_\ell h_\ell|\ell\rangle U_\ell|\psi\rangle = \frac{1}{\sqrt{M}}\sum_{\ell'\ell} h_\ell\langle\ell'|\ell\rangle U_\ell|\psi\rangle = \frac{1}{\sqrt{M}}\sum_\ell h_\ell U_\ell|\psi\rangle = \sin(\alpha)\,U|\psi\rangle. \tag{25}$$

The success amplitude of this approach is $1/\sqrt{M}$. As shown in figure 2(Right), the magnitude of wavelet coefficients h_ℓ with high index ℓ are negligibly small. Consequently, the success amplitude becomes effectively independent of M for practical applications.

2.3. Reduction of success amplitude for perfect amplitude amplification

The success amplitude of the described probabilistic approaches for implementing the single-level QWT is a known constant value. For perfect amplitude amplification, we purposefully reduce the success amplitude using one extra ancilla qubit. This end is achieved by applying a rotation gate on the extra qubit initialized in $|0\rangle$. A few rounds of amplitude amplification then yields the success state with unit probability.

The success amplitude of the probabilistic implementation PQWT in equation (21) is $\sin\,\alpha$. This amplitude is known and has a value greater than 1/4 as discussed in section 2.2. Let $\theta < \alpha$ be the angle defined in the equation below and let $R(\theta)$ be the rotation gate defined as $R(\theta)|0\rangle := \cos\theta|0\rangle + \sin\theta|1\rangle$, then by equation (21) and for any n-qubit state $|\psi\rangle$ we have

$$\left(R(\theta)\otimes\textsc{pqwt}\right)|0\rangle|0^m\rangle|\psi\rangle = \sin(\pi/14)|0^{m+1}\rangle W|\psi\rangle + \cos(\pi/14)|\perp''\rangle, \quad \cos\theta := \sin(\pi/14)/\sin\alpha, \tag{26}$$

where $|\perp''\rangle$ is an $(m+n+1)$-qubit state that satisfies $(\langle 0^{m+1}|\otimes \mathbb{1}_n)|\perp''\rangle = 0$. The success amplitude is now $\sin(\pi/14)$, enabling perfect amplitude amplification. Indeed, by only three rounds of amplitude amplification, W is applied on $|\psi\rangle$ and all $m+1$ ancilla qubits end up in the all-zero state.

We remark that the success amplitude is grater than 0.31 for the range of wavelet order used in practical applications; see figure 2(Left). In this case, we reduce the success amplitude to $\sin(\pi/10) < 0.31$ by setting $\cos\theta = \sin(\pi/10)/\sin\alpha$ and achieve the perfect amplitude amplification by only two rounds of amplitude amplification.

We use the oblivious amplitude amplification because the input state $|\psi\rangle$ is unknown. To this end, let $R_n = 2|0^n\rangle\langle 0^n| - \mathbb{1}_n$ be the n-qubit reflection operator with respect to the n-qubit zero state $|0^n\rangle$ and let

$$\mathcal{A} := -\left(R(\theta)\otimes \textsc{pqwt}\right)\left(R_{m+1}\otimes \mathbb{1}_n\right)\left(R(\theta)\otimes \textsc{pqwt}\right)^\dagger \left(R_{m+1}\otimes \mathbb{1}_n\right), \tag{27}$$

be the amplitude amplification operator. Then the following holds [19, lemma 2.2]

$$\mathcal{A}^t\left(R(\theta)\otimes \textsc{pqwt}\right)|0\rangle|0^m\rangle|\psi\rangle = \sin\left((2t+1)\pi/14\right)|0^{m+1}\rangle W|\psi\rangle + \cos\left((2t+1)\pi/14\right)|\perp''\rangle. \tag{28}$$

Therefore, the unit success probability is achieved by three executions of amplitude amplification ($t=3$).

The success amplitude of the second approach based on equation (23) is $\sin\alpha = 1/\sqrt{M}$. In this case, we reduce the success amplitude to $\sin(\pi/2(2t+1))$ by applying the rotation gate $R(\theta)$ on the extra qubit initialized in $|0\rangle$ state, with t and θ defined as

$$t := \left\lceil \frac{1}{2}\left(\frac{\pi}{2\alpha} - 1\right)\right\rceil, \quad \cos\theta := \frac{\sin\left(\pi/2\left(2t+1\right)\right)}{\sin\alpha} = \sqrt{M}\sin\left(\frac{\pi}{2}\frac{1}{2t+1}\right). \tag{29}$$

Then we achieve the desired state with unit success probability by t rounds of amplitude amplification, i.e. W is applied on $|\psi\rangle$ and all $m+1$ ancilla qubits end up in the all-zero state.

2.4. Implementing SELECT by modular quantum arithmetic

Here we describe our approach for implementing the SELECT operation by simple modular arithmetic operations on a quantum computer. As per equation (13), SELECT applies U_ℓ on the second register $|j\rangle$ based on the value of ℓ encoded in the first register $|\ell\rangle$. If ℓ is odd, then $U_\ell = P_\ell$ by equation (10). Otherwise, U_ℓ is a product of P_ℓ and a single Pauli-Z on the first qubit of the second register. That is to say that U_ℓ and P_ℓ are equivalent up to a $|0\rangle$-controlled-Z operation; control qubit is the qubit representing the LSB of ℓ and target qubit is the one representing the MSB of j. Implementing SELECT is therefore achieved by an implementation for P_ℓ.

The n-qubit permutation P_ℓ in equation (11) transforms the n-qubit basis state $|j\rangle$ as

$$P_\ell : |j\rangle \mapsto \begin{cases} |\frac{j-\ell \bmod N}{2}\rangle & \text{if } j \text{ and } \ell \text{ have same parity,} \\ |\frac{N}{2} + \frac{j+\ell-1 \bmod N}{2}\rangle & \text{if } j \text{ and } \ell \text{ have opposite parity.} \end{cases} \tag{30}$$

This transformation can be implemented by modular quantum addition ADD and subtraction SUB defined as

$$\textsc{add}|\ell\rangle|j\rangle := |\ell\rangle|j+\ell \bmod N\rangle, \quad \textsc{sub}|\ell\rangle|j\rangle := |\ell\rangle|j-\ell \bmod N\rangle, \tag{31}$$

and by the quantum perfect shuffle transformation defined as

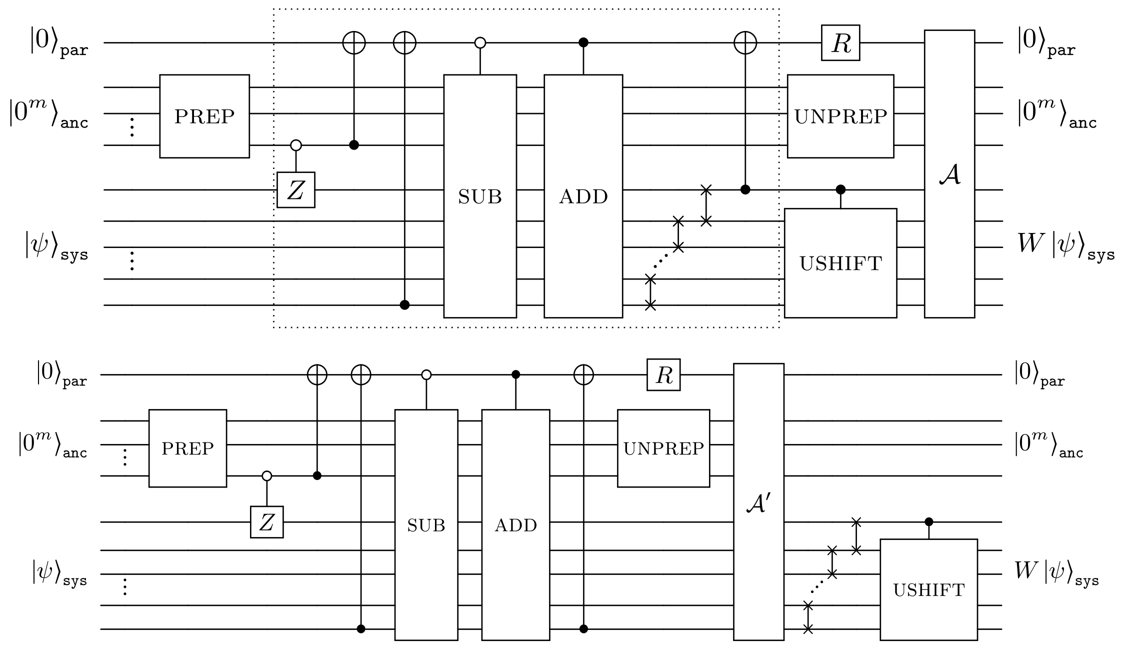
$$\textsc{shuffle}|q_{n-1}\dots q_1 q_0\rangle := |q_0 q_{n-1}\dots q_1\rangle, \tag{32}$$

which performs the transformation $|q\rangle \mapsto |q/2\rangle$ if q is an even number and $|q\rangle \mapsto |N/2 + (q-1)/2\rangle$ if q is odd (see figure 1). For clarity, we remark that here and in the following $|\ell\rangle$ is an m-qubit basis state with $m < n$ and $|j\rangle$ is an n-qubit basis state.

To implement P_ℓ by these operations, we use a single ancilla qubit called parity qubit and define the parity operation PAR as

$$\textsc{par}|0\rangle|\ell\rangle|j\rangle := \begin{cases} |0\rangle|\ell\rangle|j\rangle & \text{if } j \text{ and } \ell \text{ have same parity,} \\ |1\rangle|\ell\rangle|j\rangle & \text{if } j \text{ and } \ell \text{ have opposite parity,} \end{cases} \tag{33}$$

which flips the parity qubit based on the parity of ℓ and j; parity of a number is 0 if its even and is 1 otherwise. This operation can be implemented using two CNOT gates, one controlled on the LSQ of the register encoding ℓ and the other controlled on the LSQ of the register encoding j. The target qubit for each CNOT is the parity qubit.



$|0\rangle_{\texttt{par}}$
$|0^m\rangle_{\texttt{anc}}$
$|\psi\rangle_{\texttt{sys}}$
PREP
Z
SUB
ADD
R
UNPREP
USHIFT
$\mathcal{A}$
$|0\rangle_{\texttt{par}}$
$|0^m\rangle_{\texttt{anc}}$
$W|\psi\rangle_{\texttt{sys}}$
$|0\rangle_{\texttt{par}}$
$|0^m\rangle_{\texttt{anc}}$
$|\psi\rangle_{\texttt{sys}}$
PREP
Z
SUB
ADD
R
UNPREP
$\mathcal{A}'$
USHIFT
$|0\rangle_{\texttt{par}}$
$|0^m\rangle_{\texttt{anc}}$
$W|\psi\rangle_{\texttt{sys}}$

Figure 3. Equivalent quantum circuits for executing a single-level QWT comprised of high-level operations. Three registers are used: the parity register `par` (one qubit), the ancilla register `anc` (m qubits), and system register `sys` (n qubits). The state of `sys` register is in a superposition of $|j\rangle$ states for different values of j and the state of `anc` register, after applying PREP with action given in equation (12), is in a superposition of $|\ell\rangle$ states for different values of ℓ. The gates inside the dotted-line box implement the SELECT operation in equation (13) as follows. The $|0\rangle$-controlled Z is applied as per equation (10). The first two CNOTs compute the parity of j and ℓ by their LSB. Then controlled on the parity qubit, we apply SUB (parity zero) or ADD (parity one). The sequence of SWAP gates implement the SHUFFLE operation in equation (32). The subsequent CNOT resets the parity qubit to $|0\rangle$ because the state of `par` is filliped to $|1\rangle$ only if j and ℓ have opposite parity as per equation (33); otherwise it stays $|0\rangle$. If `par` is $|1\rangle$, the MSB of `sys` register is in the state $|1\rangle$ as the value encoded in `sys` is greater than $N/2$ by equation (30), so the last CNOT resets the parity qubit. The CNOT has no action if `par` is $|0\rangle$. This is because the value encoded in the system register is less than $N/2$ by equation (30) when j and ℓ have same parity. Consequently, the MSB of `sys` is $|0\rangle$, making the last CNOT inactive. The controlled-USHIFT operation, with USHIFT given in equation (8), maps the implemented unitary by SELECT to the single-level QWT W as in equation (9). The rotation gate R is used for amplitude amplification $\mathcal{A}$ given in equation (27). The bottom circuit follows from the top circuit. The amplitude amplification $\mathcal{A}'$ is unitarily equivalent to $\mathcal{A}$.

Having computed the parity by PAR, we then apply SUB to the last two registers if the parity qubit is $|0\rangle$ and apply ADD to these registers if the parity is $|1\rangle$, followed by the shuffle operation in equation (32) on the last register. By these operations, the state of the parity qubit, the m-qubit register encoding ℓ, and the n-qubit register encoding j transform as

$$\begin{aligned}\textsc{par}|0\rangle|\ell\rangle|j\rangle &\xrightarrow{|0\rangle\langle 0|\otimes\text{SUB}+|1\rangle\langle 1|\otimes\text{ADD}} |0\rangle|\ell\rangle|j-\ell \bmod N\rangle + |1\rangle|\ell\rangle|j+\ell \bmod N\rangle \qquad (34)\\ &\xrightarrow{\mathbb{1}_1\otimes\mathbb{1}_m\otimes\text{SHUFFLE}} |0\rangle|\ell\rangle|(j-\ell \bmod N)/2\rangle + |1\rangle|\ell\rangle|N/2+(j+\ell-1 \bmod N)/2\rangle, \qquad (35)\end{aligned}$$

where $N=2^n$. We finally erase the parity qubit to achieve an implementation for P_ℓ. To this end, we note that the parity qubit is $|1\rangle$ only if the value encoded in the last register is greater than $N/2$; see equation (30). Hence a CNOT from the qubit representing the MSB of the value encoded in the system register to the parity qubit would erase this qubit.

The quantum circuit in the dotted-line box in figure 3 gives an implementation for the SELECT operation based on the described approach. The sequence of SWAP gates in this circuit gives a gate-level implementation for SHUFFLE in equation (32)

2.5. A compilation for state-preparation operations

Here we provide procedures for implementing the LINPREP and PREP operations that prepare states with linear and square-root coefficients, respectively. We begin with an implementation for LINPREP in equation (22) using the rotation gate, defined as

$$R(\theta_\ell) := \begin{bmatrix}\cos\theta_\ell & \sin\theta_\ell\\ -\sin\theta_\ell & \cos\theta_\ell\end{bmatrix} \qquad (36)$$

for some known angle θ_ℓ, and the increment gate that preforms the map $|\ell\rangle\mapsto|\ell+1\rangle$ for $|\ell\rangle$ an m-qubit basis state. Notice that the increment gate is indeed the downshift permutation $S_m^{\downarrow}$ defined in equation (4) and its inverse is the upshift permutation $S_m^{\uparrow}$.

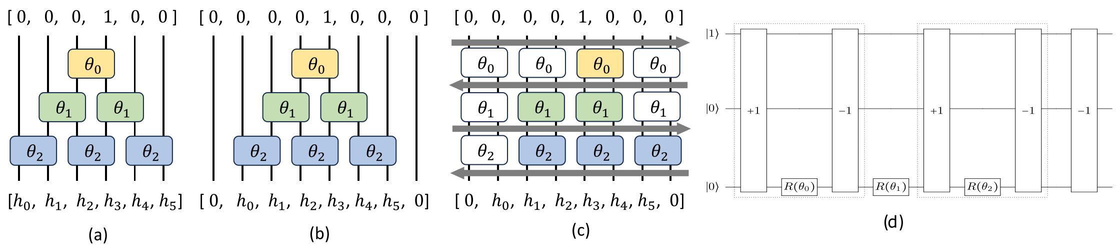


Figure 4. (a) Diagrammatic representation of the procedure producing the wavelet filters for a wavelet of order M from a particular initial vector by a set of $M/2$ rotations; $M=6$ is illustrated here. (b) Zero padding for cases that M is not a power of two. (c) Rotations can be applied in parallel. The right (left) arrow represents shifting elements of the vector one place to the right (left). As the initial vector is a particular vector, the rotations represented by white boxes do not affect the vector.
(d) Quantum circuit for LINPREP using rotation gates, the increment gate denoted by $+1$ and its inverse denoted by -1. The gate $+1$ (-1) is applied before (after) each rotation gate R_ℓ with even ℓ, as in dotted boxes.

The LINPREP operation prepares a quantum state with amplitudes given by the wavelet filter $\boldsymbol{h}=(h_0,\dots,h_{M-1})^\top$, a column vector of M real numbers that satisfy equation (1). By the procedure given in [20], the wavelet filter vector $\boldsymbol{h}$ of length $M=2\mathcal{K}$ can be achieved by a sequence of $\mathcal{K}$ unitaries U_ℓ as $\boldsymbol{h}=U_{\mathcal{K}-1}\cdots U_1U_0\,\mathrm{e}_{\mathcal{K}}$, where e_ℓ is the ℓth column of the M-by-M identity matrix and the unitary U_ℓ is constructed from rotation gates $R(\theta_\ell)$ as illustrated in figure 4(a). As an example, for $M=6$ we have

$$\begin{bmatrix}h_0\\h_1\\h_2\\h_3\\h_4\\h_5\end{bmatrix}=\begin{bmatrix}c_2&s_2&&&&\\-s_2&c_2&&&&\\&&c_2&s_2&&\\&&-s_2&c_2&&\\&&&&c_2&s_2\\&&&&-s_2&c_2\end{bmatrix}\begin{bmatrix}1&&&&&\\&c_1&s_1&&&\\&-s_1&c_1&&&\\&&&c_1&s_1&\\&&&-s_1&c_1&\\&&&&&1\end{bmatrix}\begin{bmatrix}1&&&&&\\&1&&&&\\&&c_0&s_0&&\\&&-s_0&c_0&&\\&&&&1&\\&&&&&1\end{bmatrix}\begin{bmatrix}0\\0\\0\\1\\0\\0\end{bmatrix}\tag{37}$$

where $c_\ell:=\cos\theta_\ell$ and $s_\ell:=\sin\theta_\ell$.

Having classically precomputed the rotation angles $(\theta_0,\theta_1,\dots,\theta_{\mathcal{K}-1})$ by the procedure in [20], we construct a quantum circuit for LINPREP as follows. Let $m=\lceil\log_2 M\rceil$. For M that is not a power of 2, we pad $(2^m-M)/2$ zeros from left and right to the wavelet filter vector $\boldsymbol{h}$ to have a vector as $(0,\dots,0,h_0,\dots,h_{M-1},0\dots,0)^\top$. Then unitaries U_ℓ are modified accordingly so that $U_{\mathcal{K}-1}\cdots U_1U_0\,\mathrm{e}_{2^{m-1}}$ yields the modified wavelet filter vector. A diagrammatic representation of this approach is shown in figure 4(b) for $M=6$. For each θ_ℓ with even ℓ, first we shift elements of the vector one place to the right, shown in figure 4(c) by the right arrow, to be able to apply the rotations in parallel on consequent pairs of the vector elements and then shift the vector elements one place to the left. Because the rotations are in parallel, we can decompose the associated unitary as a tensor product of an identity and a rotation gate as $\mathbb{1}_{m-1}\otimes R_\ell$. Shifting to the right (left) is implemented by the increment gate (inverse of the increment gate) on a quantum computer. The inverse of the increment gate is applied $(2^m-M)/2$ times at the end to achieve the desired amplitudes as $(h_0,\dots,h_{M-1},0\dots,0)^\top$. The quantum circuit in figure 4(d) illustrates the case where $M=6$.

We now describe an approach for implementing the PREP operation in equation (12). This operation prepares the state with square-root coefficients, i.e. the state $|\psi\rangle:=\sum_\ell\sqrt{p_\ell}|\ell\rangle$ with $p_\ell:=|h_\ell|/h$. To prepare this state, first we prepare the uniform superposition state $(1/\sqrt{M})\sum_\ell|\ell\rangle$ and then apply the uniformly controlled rotation [21] that performs the map

$$|0\rangle|\ell\rangle\mapsto(\cos(\theta_\ell)|0\rangle+\sin(\theta_\ell)|1\rangle)|\ell\rangle,\quad\cos(\theta_\ell):=\sqrt{|h_\ell|/h}.\tag{38}$$

The output state after this operation is $\sin\alpha|0\rangle|\psi\rangle+\cos\alpha|\perp\rangle$ with the success amplitude $\sin\alpha:=1/\sqrt{M}$. As per the discussion in section 2.3, the state $|\psi\rangle$ is achieved using one extra qubit and $\Theta(\sqrt{M})$ rounds of amplitude amplification. We remark that the same approach can be used to implement UNPREP in equation (12).

3. Multi-level and packet QWT

We now use our implementation for the single-level QWT as a subroutine and construct quantum algorithms for multi-level and packet QWTs. To this end, let $W_n^{(d)}$ denote the d-level wavelet transform of size $2^n\times2^n$ and let $P_n^{(d)}$ denote the d-level wavelet packet transform of the same size. Also let $W_n^{(1)}=W_n$ for notation simplicity.

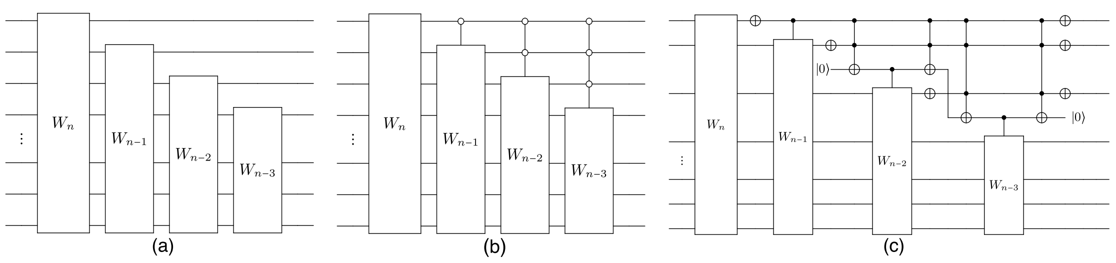


Figure 5. Quantum circuits for (a) the d-level packet QWT and (b) the d-level QWT using single-level QWTs; $d = 4$ is illustrated here. (c) An implementation of multi-controlled single-level QWTs needed for the multi-level QWT in (b) using multi-bit Toffoli gates, controlled single-level QWT and one ancilla qubit that starts and ends in the $|0\rangle$ state.

The d-level wavelet transform can be recursively decomposed as [7, appendix A]

$$W_n^{(d)} = \left(W_{n-1}^{(d-1)} \oplus \mathbb{1}_{n-1}\right) W_n. \tag{39}$$

This decomposition follows from the notion of multi-level wavelet transform: at each level, the transformation is only applied on the low-frequency component (i.e. the top part) of the column vector it acts on. The wavelet packet transform, however, acts on both the low- and high-frequency components, so we have the decomposition

$$P_n^{(d)} = \left(P_{n-1}^{(d-1)} \oplus P_{n-1}^{(d-1)}\right) W_n = \left(\mathbb{1}_1 \otimes P_{n-1}^{(d-1)}\right) W_n \tag{40}$$

for the wavelet packet transform. Equation (39) yields the decomposition

$$W_n^{(d)} = \Lambda_0^{d-1}\left(W_{n-d+1}\right) \cdots \Lambda_0^2\left(W_{n-2}\right) \Lambda_0^1\left(W_{n-1}\right) W_n, \tag{41}$$

where

$$\Lambda_0^s\left(W_{n-s}\right) := |0^s\rangle\langle 0^s| \otimes W_{n-s} + \left(\mathbb{1}_s - |0^s\rangle\langle 0^s|\right) \otimes \mathbb{1}_{n-s} \tag{42}$$

is the $|0^s\rangle$-controlled unitary operation, for any $s \in \{1, \ldots, d-1\}$. Similarity, equation (40) yields the decomposition

$$P_n^{(d)} = \left(\mathbb{1}_{d-1} \otimes W_{n-d+1}\right) \cdots \left(\mathbb{1}_2 \otimes W_{n-2}\right) \left(\mathbb{1}_1 \otimes W_{n-1}\right) W_n \tag{43}$$

for the d-level wavelet packet transform. These decompositions give a simple procedure for implementing a multi-level and packet QWT shown by the quantum circuits in figure 5

The multi-level packet QWT is construed from single-level QWTs that can be implemented by the method described in section 2. In contrast, the multi-level QWT is constructed from multi-controlled single-level QWTs. As in figure 5(c), we break down these multi-controlled operations in terms of multi-bit Toffoli gates and controlled single-level QWTs. We discuss an implementation of a multi-bit Toffoli gate in section 4.1 and a controlled single-level QWT in section 4.3, where we analyze the complexities of these operations.

4. Complexity analysis

In this section, we analyze the computational cost of executing single-level, multi-level and packet QWTs, thereby establishing theorems 1–3. We begin by analyzing the computational cost of key subroutines in our algorithms in section 4.1. We then build upon them and provide cost analysis for the single-level QWT in section 4.2 and for the multi-level and packet QWTs in section 4.3.

In our cost analysis and in implementing the key operations, we use ancilla and 'borrowed' qubits. In contrast to an ancilla qubit that starts from $|0\rangle$ and returns to $|0\rangle$, a borrowed qubit can start from any state and will return to its original state. The purpose of using borrowed qubits is that they enable simple implementation for complex multi-qubit operations. The availability of a sufficient number of qubits in our algorithm on which the key operations do not act on them allows us to use them as borrowed qubits in implementing such operations.

4.1. Complexity of key subroutines

Here we analyze the cost of key subroutines used in our algorithm for a single-level QWT: PREP, SELECT and USHIFT, the latter of which adds a classically known constant value to the value encoded in a quantum register. We also analyze the cost of implementing a multi-qubit reflection, an operation used in the amplitude amplification part of our algorithm.

For simplicity of cost analysis, we state the cost of each key subroutine in a lemma and proceed with analyzing the cost in the poof. We begin with a lemma stating the cost of executing a multi-bit Toffoli gate, an operation frequently used in our algorithm and provides an implementation for the multi-qubit reflection about the all-zero state.

Lemma 1. *The $(m+1)$-bit Toffoli gate with $m \geqslant 3$, defined as $\Lambda_1^m(X) := |1^m\rangle\langle 1^m| \otimes X + (\mathbb{1}_m - |1^m\rangle\langle 1^m|) \otimes \mathbb{1}_1$, can be implemented by either of the following computational resources:*

(I) *$m-2$ borrowed qubits and $\mathcal{O}(m)$ TOFFOLI gates, or*
(II) *one borrowed qubit and $\mathcal{O}(m)$ TOFFOLI and elementary one- or two-qubit gate.*

The implementation based on $m-2$ borrowed qubits follows from Gidney's method [22] for implementing a multi-bit Toffoli gate and the one using one borrowed qubit follows by the method given in [23, corollary 7.4] and also in [24]. Notice that the gate cost of the two methods scales similarly, but one uses only a single borrowed qubit. However, we sometimes use the method with $m-2$ borrowed qubits due to its simplicity in implementing a multi-bit Toffoli and the availability of a sufficient number of qubits in our algorithm that can be borrowed.

We proceed with the cost of SELECT in the following lemma.

Lemma 2. *SELECT in equation (13) can be executed using one ancilla and one borrowed qubit, two Hadamard and $\mathcal{O}(n)$ NOT, CNOT and TOFFOLI gates.*

Proof. By figure 3, SELECT is composed of one controlled-Z gate, three CNOT gates, one controlled-SUB, one controlled-ADD and $n-1$ SWAP gates. The controlled-Z gate can be executed using two Hadamard gates and one CNOT, and each SWAP can be executed using three CNOTs. By the compilation given in [25], the ADD itself can be implemented using one ancilla qubit and $\mathcal{O}(n)$ NOT, CNOT and TOFFOLI gates. Hence the controlled-ADD can be compiled using $\mathcal{O}(n)$ CNOT, TOFFOLI and four-bit Toffoli gates, the latter of which can be implemented using one borrowed qubit and four TOFFOLI gates by lemma 1. □

In the next lemma, we show that the m-qubit reflection R_m about the all-zero state $|0^m\rangle$ can be implemented using $m-2$ borrowed qubits.

Lemma 3. *The m-qubit reflection $R_m := |0^m\rangle\langle 0^m| - \mathbb{1}_m$ can be executed using one ancilla and $m-2$ borrowed qubits along with two Hadamard, $2m+2$ NOT and $\mathcal{O}(m)$ TOFFOLI gates.*

Proof. Using the phase kickback trick and one ancilla qubit, we can implement R_m up to an irrelevant global -1 phase factor as

$$(R_m \otimes \mathbb{1}_1)|\psi\rangle|0\rangle = -X^{\otimes m+1}(\mathbb{1}_m \otimes H)\Lambda_1^m(X)(\mathbb{1}_m \otimes H)X^{\otimes m+1}|\psi\rangle|0\rangle, \tag{44}$$

where $|\psi\rangle$ is any m-qubit state and $\Lambda_1^m(X)$ is the $(m+1)$-bit Toffoli gate. The lemma then follows by Gidney's method [22] for implementing the $(m+1)$-bit Toffoli using $\mathcal{O}(m)$ TOFFOLI gates and $m-2$ borrowed qubits. □

We remark that the $(m+1)$-bit Toffoli can be implemented using only one borrowed qubit and $\mathcal{O}(m)$ TOFFOLI and elementary one- or two-qubit gate by lemma 1. However, we use the method with $m-2$ borrowed qubits due to its simplicity in implementing a multi-bit Toffoli and the availability of a sufficient number of qubits in our algorithm that can be borrowed.

The following lemma states the cost of adding a known classical value to a quantum register. We use a controlled version of this operation in our algorithm, the cost of which is stated in the following corollary.

Lemma 4. *Adding a classically known m-bit constant to an n-qubit register with $m < n$ can be achieved using $m+1$ ancilla qubits and $\mathcal{O}(m)$ NOT, CNOT and TOFFOLI gates.*

Proof. First, prepare m ancillae in the computational state that encodes the m-bit constant. This preparation can be achieved by applying at most m NOT gates. Then add this state to the state of the n-qubit register by ADD operation in equation (31). By $m < n$ and the compilation given in [25], ADD can be implemented by one ancilla qubit and $\mathcal{O}(m)$ NOT, CNOT and TOFFOLI gates. □

The computational cost reported in lemma 4 is indeed the cost of executing USHIFT in equation (8). We use a controlled version of this operation as in the circuit shown in figure 3. Because of the TOFFOLI gate in lemma 4, the controlled-USHIFT requires implementing a four-bit Toffoli gate, an operation that can be implemented using one borrowed qubit and four TOFFOLI gates by lemma 1. Therefore, we have the following cost for the controlled USHIFT as a corollary of lemmas 1 and 4.

Corollary 4. *The controlled-USHIFT operation can be executed by one borrowed qubit, $m+1$ ancilla qubits, and $\mathcal{O}(m)$ CNOT and TOFFOLI gates.*

The final lemma states the cost of the PREP operation. We remark that the cost of this operation is independent of n as PREP generates a quantum state on a number of ancilla qubits that depends on the wavelet order M.

Lemma 5. *LINPREP in equation (22) can be executed using $\mathcal{O}(M\log_2 M)$ elementary gates and $\lceil\log_2 M\rceil$ borrowed qubits. PREP and UNPREP in equation (12) can be executed using $\mathcal{O}(M^{3/2})$ elementary gates and one ancilla qubit.*

Proof. The LINPREP can be implemented using $\mathcal{O}(M)$ rotation gates and $\mathcal{O}(M)$ increment and inverse of increment gates by the procedure given in section 2.5. The increment gate on $m=\lceil\log_2 M\rceil$ qubits can be implemented using m borrowed qubits and $\mathcal{O}(m)$ elementary gates [26], so the overall gate cost of LINPREP is $\mathcal{O}(M\log_2 M)$. As per section 2.5, PREP and UNPREP operations can be implemented by preparing the uniform superposition state on m qubits, applying the uniformly controlled rotation in equation (38), and $\mathcal{O}(\sqrt{M})$ rounds of amplitude amplification. The uniform superposition state is prepared by m Hadamard gates, and the uniformly controlled rotation can be implemented by $\mathcal{O}(M)$ CNOT and rotation gates [21]. Therefore, the overall gate cost of PREP and UNPREP is $\mathcal{O}(M^{3/2})$. □

4.2. Complexity of single-level QWT

We now build upon the computational cost of the key subroutines analyzed in the previous section to obtain the computational cost of executing a single-level QWT. To this end, we mainly use equations (27) and (28). By these equations, a single-level QWT is achieved by performing three rotation gates and

- Two PQWT and one PQWT†, which by equation (20) needs performing two SELECT and one SELECT†; two PREP and one PREP†; two UNPREP and one UNPREP†; and one controlled-USHIFT;
- Two $(m+1)$-qubit reflection R_{m+1}.

Therefore, by lemmas 2, 3, 5 and corollary 4 , the gate cost $\mathcal{G}(1\text{UNPREP})$ for executing a single-level QWT is

$$\mathcal{G}\left(1\text{QWT}\right)=3\mathcal{G}\left(\text{SELECT}\right)+6\mathcal{G}\left(\text{PREP}\right)+\mathcal{G}\left(\text{controlled-USHIFT}\right)+2\mathcal{G}\left(R_{m+1}\right)+3\in\mathcal{O}\left(n\right)+\mathcal{O}\left(M^{3/2}\right) \tag{45}$$

where $m=\lceil\log_2 M\rceil$ in our application; M is the wavelet order. The number of ancilla qubits used is $m+1$: m ancillae are used for the state-preparation step, and one extra ancilla is the parity qubit `par`, which is also used in the amplitude amplification step.

We remark that the borrowed qubits in executing PREP, SELECT, controlled-USHIFT and reflection operations, in lemmas 2–5, are borrowed from the portion of quantum registers that these operations do not act on them. For instance, the $m-2$ borrowed qubits in lemma 3 for executing the m-qubit reflection R_m could be any $m-2$ qubits of the n qubit register that R_m does not act on them. For SELECT, the borrowed qubit is needed to implement the four-bit Toffoli gate, see proof of lemma 2, and this qubit could be any qubit in the circuit that the four-bit Toffoli gate does not act on it. We also remark that the $m+1$ ancilla qubits in corollary 4 needed for controlled-USHIFT are qubits of the single-qubit `par` register and m-qubit `anc` register. This operation is executed after the amplitude amplification, see figure 3, when `par` and `anc` are in the all-zero state.

Putting all together, the overall gate cost for implementing the single-level QWT is $\mathcal{O}(n)+\mathcal{O}(M^{3/2})$ and the number of ancilla qubits is $\lceil\log_2 M\rceil+1$. This is the computational cost reported in theorem 1.

4.3. Complexity of multi-level and packet QWTs

Here we analyze the complexity of implementing a d-level and packet QWTs, thereby establishing theorems 2 and 3. By figure 5, implementing a multi-level QWT is achieved by implementing multiply-controlled single-level QWTs. Our strategy is to break down each multiply-controlled unitaries in terms of multi-bit Toffoli gates and single-controlled unitary. We then use a compilation for a controlled single-level QWT and an ancilla-friendly compilation for multi-bit Toffoli gates to achieve an efficient yet ancilla-friendly implementation for a multi-level QWT. The packet QWT, however, is achieved by a sequence of single-level QWTs without controlled qubits, as shown in figure 5.

Before describing the specifics of our implementation strategy, we first state the complexity of the $|1\rangle$-controlled single-level QWT in the following lemma. We then build upon this complexity to establish the complexity of multi-level QWT.

Lemma 6. *The controlled single-level QWT on n qubits, associated with a wavelet of order M, can be achieved using* $\lceil \log_2 M \rceil + 2$ *ancilla qubits and* $\mathcal{O}(n) + \mathcal{O}(M^{3/2})$ *elementary gates.*

Proof. By the circuit in figure 3, a controlled single-level QWT needs preforming double-controlled-SUB, -ADD and -USHIFT operations, and single-controlled PREP and UNPREP operations. Each CNOT is transformed to a TOFFOLI, each SWAP is transformed to three TOFFOLI gates and R is transformed to controlled-R. A double-controlled operation can be reduced to a single-controlled operation using two TOFFOLI gates and one ancilla qubit. By the discussion in the proof of lemma 2, the controlled-ADD (-SUB) can be compiled using $\mathcal{O}(n)$ CNOT, TOFFOLI gates. The other ancilla qubits are the m qubits used for state preparation and the parity qubit. Altogether with corollary 4 prove the lemma. □

We now proceed with the complexity of d-level QWT. Let the integer s, with $1 \leqslant s \leqslant d$, represent the level of a QWT. Then for the level $s = r + 1$ we need to implement $|0^r\rangle$-controlled-W_{n-r}, where W_{n-r} is the single-level QWT on $n - r$ qubits. For simplicity of cost analysis, we map all $|0\rangle$-controlled operations in figure 5(b) to $|1\rangle$-controlled operations; this can be achieved by $2(d-1)$ NOT gates for d-level QWT as in figure 5(c). For $r \geqslant 2$, we implement $|1^r\rangle$-controlled-W_{n-r} by a single ancilla qubit, two $(r+1)$-bit Toffoli gates and one controlled-W_{n-r} as shown in figure 5(c). Notice that $s = 1$ corresponds to a single-level QWT on n qubits and $s = 2$ corresponds to a controlled single-level QWT on $n - 1$ qubits.

The gate cost for the controlled single-level QWT on $n - r$ qubits is $\mathcal{O}(n - r)$ by lemma 6, disregarding the cost with respect to M, and the gate cost for the $(r+1)$-bit Toffoli gate is $\mathcal{O}(r)$ by lemma 1. Hence the gate cost for each level, including the first and second levels, is $\mathcal{O}(n)$. We also have an additional gate cost of $\mathcal{O}(M^{3/2})$ for each level associated with the cost of implementing PREP and UNPREP. We remark that only a single ancilla qubit is used for all levels; the ancilla qubit starts and ends in $|0\rangle$ for each level to be reused in the next level, as illustrated in figure 5(c). Putting all together, we arrive at the computational cost stated in theorem 2 for a d-level QWT.

Because the packet QWT does not have multi-controlled operations (see figure 5(a)), its gate cost simply follows from the cost of the single-level QWT. The single-level QWT acts on $n - r$ qubits at level $s = r + 1$ and has the gate cost $\mathcal{O}(n - r)$ by theorem 1. The gate cost for all levels $1 \leqslant s \leqslant d$ is therefore $\mathcal{O}(dn - d(d-1)/2)$. We also have an additional gate cost of $\mathcal{O}(M^{3/2})$ for each level associated with the cost of implementing PREP and UNPREP, yielding the overall gate cost stated in theorem 3. We note that the packet QWT does not need the extra ancilla qubit used in multi-level QWT for implementing the multi-controlled operations.

5. Discussion and conclusion

Wavelets and their associated transforms have been extensively used in classical computing. The basis functions of wavelet transforms have features that make such transforms advantageous for numerous applications over their established counterpart, the Fourier transform. However, prior works on developing a quantum analog for wavelet transforms were limited to a few representative cases. In this paper, we presented quantum algorithms for executing any wavelet transform and a generalized version, the wavelet packet transform, on a quantum computer; the algorithms work for any wavelet of any order. We have established the computational complexity of our algorithms in terms of three parameters involved in wavelet transforms: the wavelet order M, the level d of wavelet transform, and the number of qubits $n = \log_2 N$ the QWT acts on, with N the dimension of the kernel matrix associated with the wavelet transform.

The core idea of our approach is to express the kernel matrix as a linear combination of M unitary operations that are simple to implement on a quantum computer and use the LCU technique to construct a probabilistic procedure for implementing the desired QWT. We then make the implementation deterministic using the known success probability of the probabilistic procedure by only a few (two or three) rounds of amplitude amplification. The gate cost of our algorithm for single-level QWT scales optimally with n, the number of qubits, for the case that the wavelet order M is constant. Indeed, the order parameter used in practical applications is constant, typically in the range of $2 \leqslant M \leqslant 20$ [3, 27, 28]. We also demonstrated that the wavelet filter coefficients become negligibly small for larger values of the wavelet order, making the cost of our algorithms effectively independent of M for practical applications. In contrast, the transformation level d scales linearly with the number of qubits, or $\log_2 N$, for practical applications. Because the value of d is upper-bounded by n, the gate cost of multi-level and packet QWTs scales as $\mathcal{O}(n^2)$ in the worst case. Even for the worst case, our algorithm improves the gate cost of prior works on the second- and fourth-order Daubechies QWT from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$.

We remark that our approach requires a number of ancilla qubits that scales as $\log_2 M$ with the wavelet order. The number of ancilla qubits would be a small constant number considering the range of wavelet order or the magnitude of wavelet coefficients in practical applications. A potential area for further exploration is constructing ancilla-free quantum algorithms for all QWTs. Constructing such algorithms would be valuable for early fault-tolerant quantum computers with limited qubits and is plausible because QWTs are unitary transformations. More importantly, a primary area for future research is exploring the opportunities offered by QWTs in quantum algorithms, particularly in simulating quantum systems [6, 7, 12] and image processing [29–31] where wavelet transforms could be advantageous over the established Fourier transform.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

We acknowledge the generous support and funding of this project by the Defense Advanced Research Projects Agency (DARPA) under the Grant Number HR0011-23-3-0021. We also acknowledge support from the Canada 150 Research Chairs program and NSERC-IRC. A A G also acknowledges the generous support of Anders G Frøseth.

ORCID iDs

Mohsen Bagherimehrab https://orcid.org/0000-0001-8564-446X
Alán Aspuru-Guzik https://orcid.org/0000-0002-8277-4434

References

[1] Mallat S 2009 *A Wavelet Tour of Signal Processing: The Sparse Way* 3rd edn (Academic)
[2] Daubechies I 1992 *Ten Lectures on Wavelets* (Society for Industrial and Applied Mathematics)
[3] Beylkin G 1992 On the representation of operators in bases of compactly supported wavelets *SIAM J. Numer. Anal.* **29** 1716
[4] Bagherimehrab M 2022 Algorithmic quantum-state generation for simulating quantum field theories on a quantum computer *PhD Thesis* University of Calgary, Calgary, AB (https://doi.org/10.11575/PRISM/39545)
[5] Bulut F and Polyzou W N 2013 Wavelets in field theory *Phys. Rev.* D **87** 116011
[6] Brennen G K, Rohde P, Sanders B C and Singh S 2015 Multiscale quantum simulation of quantum field theory using wavelets *Phys. Rev.* A **92** 032315
[7] Bagherimehrab M, Sanders Y R, Berry D W, Brennen G K and Sanders B C 2022 Nearly optimal quantum algorithm for generating the ground state of a free quantum field theory *PRX Quantum* **3** 020364
[8] Hong C-L *et al* 2022 Accurate and efficient quantum computations of molecular properties using Daubechies wavelet molecular orbitals: a benchmark study against experimental data *PRX Quantum* **3** 020360
[9] Evenbly G and White S R 2016 Entanglement renormalization and wavelets *Phys. Rev. Lett.* **116** 140403
[10] Polyzou W N 2020 Wavelet representation of light-front quantum field theory *Phys. Rev.* D **101** 096004
[11] George D J, Sanders Y R, Bagherimehrab M, Sanders B C and Brennen G K 2022 Entanglement in quantum field theory via wavelet representations *Phys. Rev.* D **106** 036025
[12] Bagherimehrab M, Nakaji K, Wiebe N and Aspuru-Guzik A 2023 Fast quantum algorithm for differential equations (arXiv:2306.11802)
[13] Fijany A and Williams C P 1999 Quantum wavelet transforms: fast algorithms and complete circuits *Quantum Computing and Quantum Communications* ed C P Williams (Springer) pp 10–33
[14] Hoyer P 1997 Efficient quantum transforms (arXiv:quant-ph/9702028)
[15] Li H-S, Fan P, Xia H-Y, Song S and He X 2018 The multi-level and multi-dimensional quantum wavelet packet transforms *Sci. Rep.* **8** 13884
[16] Li H-S, Fan P, Xia H-Y and Song S 2019 Quantum multi-level wavelet transforms *Inf. Sci.* **504** 113
[17] Li H, Li G and Xia H 2023 Three-dimensional quantum wavelet transforms *Front. Comput. Sci.* **17** 175905
[18] Childs A M and Wiebe N 2012 Hamiltonian simulation using linear combinations of unitary operations *Quantum Inf. Comput.* **12** 901
[19] Kothari R 2014 Efficient algorithms in quantum query complexity *PhD Thesis* University of Waterloo
[20] Evenbly G and White S R 2018 Representation and design of wavelets using unitary circuits *Phys. Rev.* A **97** 052314
[21] Möttönen M, Vartiainen J J, Bergholm V and Salomaa M M 2004 Quantum circuits for general multiqubit gates *Phys. Rev. Lett.* **93** 130502
[22] Gidney C 2015 Constructing large controlled nots (available at: https://algassert.com/circuits/2015/06/05/Constructing-Large-Controlled-Nots.html) (Accessed 25 August 2023)
[23] Barenco A, Bennett C H, Cleve R, DiVincenzo D P, Margolus N, Shor P, Sleator T, Smolin J A and Weinfurter H 1995 Elementary gates for quantum computation *Phys. Rev.* A **52** 3457
[24] He Y, Luo M-X, Zhang E, Wang H-K and Wang X-F 2017 Decompositions of n-qubit Toffoli gates with linear circuit complexity *Int. J. Theor. Phys.* **56** 2350
[25] Cuccaro S A, Draper T G, Kutin S A and Moulton D P 2004 A new quantum ripple-carry addition circuit (arXiv:quant-ph/0410184)
[26] Gidney C 2015 Constructing large increment gates (available at: https://algassert.com/circuits/2015/06/12/Constructing-Large-Increment-Gates.html) (Accessed 25 August 2023)

[27] Urban K 2008 *Wavelet Methods for Elliptic Partial Differential Equations* (Oxford University Press)
[28] Beylkin G, Coifman R and Rokhlin V 1991 Fast wavelet transforms and numerical algorithms I *Commun. Pure Appl. Math.* **44** 141
[29] McCord J C and Evenbly G 2022 Improved wavelets for image compression from unitary circuits (arXiv:2203.02556)
[30] Tian C, Zheng M, Zuo W, Zhang B, Zhang Y and Zhang D 2023 Multi-stage image denoising with the wavelet transform *Pattern Recognit.* **134** 109050
[31] Zhou N-R, Hu L-L, Huang Z-W, Wang M-M and Luo G-S 2024 Novel multiple color images encryption and decryption scheme based on a bit-level extension algorithm *Expert Syst. Appl.* **238** 122052