

Quantum Wave Atom Transforms

Marianna Podzorova^{1,3} and Yi-Kai Liu^{2,3,*}

¹Department of Computer Science, University of Maryland, College Park, MD

²Applied and Computational Mathematics Division, National Institute of Standards and Technology (NIST), Gaithersburg, MD

³Joint Center for Quantum Information and Computer Science (QuICS), NIST/University of Maryland, College Park, MD

*Corresponding author: yi-kai.liu@nist.gov

Abstract

This paper constructs the first efficient implementation of a quantum wavelet packet transform with a “parabolic scaling” tree structure, sometimes called a quantum wave atom transform. Classically, wave atom transforms are used to construct sparse representations of differential operators, which enable fast classical algorithms for solving wave equations. Compared to previous work on quantum wavelet transforms, our quantum algorithm can implement a larger class of wavelet and wave atom transforms, by using an efficient representation for a larger class of possible tree structures. Our quantum implementation has $O(\text{poly}(n))$ gate complexity for applying a transform of dimension 2^n , while classical implementations use $O(n2^n)$ floating point operations. This is potentially useful for designing quantum algorithms for solving wave equations that achieve an exponential speedup over classical algorithms.

1 Introduction

The quantum Fourier transform plays an important role in many quantum algorithms that achieve an exponential speedup over classical algorithms [1, 2]. Thus it is natural to ask whether quantum wavelet transforms could be similarly useful? Here, a *quantum wavelet transform* is a unitary operation that can be performed on a quantum computer with n qubits.¹ It acts on quantum superposition states:

$$\sum_{x=0}^{2^n-1} f(x) |x\rangle \mapsto \sum_{j,m,n \in \Gamma} c_{j,m,n} |j,m,n\rangle, \quad (1)$$

i.e., given a superposition state whose complex amplitudes encode some complex-valued function f on the spatial domain $\{0, 1, \dots, 2^n - 1\} \subset \mathbb{R}$, the quantum wavelet transform returns a superposition state whose complex amplitudes encode the coefficients $c_{j,m,n} \in \mathbb{C}$ that appear when f is expanded in some wavelet basis. Typically these coefficients $c_{j,m,n}$ are indexed by variables (j, m, n) that range over some set Γ , and have an interpretation in terms of scale, frequency and position. Here we have assumed that f is defined on a 1-dimensional spatial domain, but generalizations to more dimensions are also possible.

There are two natural questions that arise, when using wavelets to design quantum algorithms:

¹In principle, one can also consider more general kinds of operations, such as isometries or quantum channels. These can be implemented using a quantum computer with ancilla qubits, or an open quantum system.

Question 1. *Can a quantum wavelet transform be implemented efficiently on a quantum computer, that is, can it be implemented using a family of quantum circuits whose size grows at most polynomially with the number of qubits n ?*

This is different from the implementation of classical wavelet transforms on classical computers, in two respects: the number of gates in the quantum circuit is only allowed to grow polylogarithmically (not polynomially) with the dimension of the vector $(f(0), f(1), \dots, f(2^n - 1))$, and the individual gates in the quantum circuit can act simultaneously on the entire vector “in superposition” (though the action of each gate can only depend on a constant-sized subset of the qubits).

Question 2. *Can one use a quantum wavelet transform to design a quantum algorithm that solves a useful computational problem, exponentially faster than any classical algorithm?*

Here, the adjective *useful* expresses the idea that one ultimately wants to solve a computational problem that has a broader motivation, beyond the mathematical theory of wavelets. This requires combining the quantum wavelet transform in Eq. (1) with other techniques that are relevant to the problem being solved.

The answers to Questions 1 and 2 both depend crucially on the choice of a particular wavelet transform. In this paper, we consider a family of wavelet transforms, known as *wave atom* transforms [3–5], which are used in classical algorithms for solving wave equations and simulating wave propagation [6, 7], but (to the best of our knowledge) have not been applied to the design of quantum algorithms.

There is a particularly strong motivation for studying *quantum* wave atom transforms: if such an operation can be implemented efficiently on a quantum computer, then there is a promising (albeit heuristic) approach for using this operation to obtain fast quantum algorithms for solving wave equations. (We will describe this in more detail below.) If these steps could be made rigorous, they would provide positive answers to both Questions 1 and 2, which would have practical applications in optics, seismology and other fields.

Yet for wave atoms, both of the above questions have remained open, due to the complexity of the wave atom construction. For example, wave atoms are constructed using a tree structure, which is not handled in previous works on efficient implementation of quantum wavelet transforms [8–15]. Furthermore, in order to use a wave atom transform to solve a wave equation, one constructs sparse approximations of the solution operator, which are quite delicate. These sparse approximations have been studied in the context of classical algorithms [3, 4, 16, 17], but not in the context of quantum algorithms, which depend on different notions of sparsity and approximation error.

Our main contribution in this paper is to overcome the first difficulty (involving the tree structure), leading to the first efficient implementation of a quantum wave atom transform. This provides a positive answer to Question 1, and is a step towards addressing the technical issues involved in Question 2. In the following sections, we will describe the background and motivation for wave atoms, the strengths and weaknesses of our results, and how this relates to previous work, in more detail.

1.1 Wavelets and wave atoms

In classical signal processing, wavelets are a powerful tool for constructing *sparse representations* of complicated signals, using specially-tailored basis functions that are well-concentrated in both time and frequency. This has motivated the study of a vast variety of different wavelet families, characterized by various choices of vanishing moments, size of wave packet support, symmetry, smoothness, and other properties: Shannon, Meyer, Haar, Battle-Lemarié, or Daubechies compactly supported wavelets [18, 19], Coiflets [20], Ricker hats [21], Gabor atoms [22], wave atoms [3–6], and curvelets [3, 16].

Among these families of wavelets, wave atoms and curvelets are known to provide especially sparse representations for certain differential operators that arise in the study of wave equations [3–6, 16, 17]. This allows their application as a potentially effective preconditioner for partial differential equations [3, 7]. So, some wave packet bases can be valuable for creating sparse representations of linear operators.

These sparse representations depend on a specific geometric property of the wave atom basis functions: the basis functions obey a so-called *parabolic scaling law*,

$$(\text{wavelength}) \propto (\text{diameter})^2, \quad (2)$$

where the wavelength (of each wave packet) is proportional to the square of the diameter (of the set on which the wave packet is supported). This is in contrast to conventional wavelets, which obey a linear scaling law (wavelength $\propto$ diameter), and Gabor wavelets, where the diameter is fixed while the wavelength is allowed to vary [4].

In order to satisfy this scaling law, the wave atom basis functions are constructed by a recursive procedure, which can be viewed as building a *wave packet tree* [19], where the branching pattern of the tree is determined by the scaling law. This branching pattern is one of the main features that distinguish wave atoms from other wavelet bases. For most wavelet bases, the branching pattern is simple, e.g., it corresponds to a complete binary tree, or a dyadic tree, shown in Fig. 6. Wave atoms require a branching pattern that is less regular, and lies somewhere in between a complete binary tree and a dyadic tree, as shown in Fig. 7.

1.2 Our results

In this paper, we address Question 1, by constructing efficient quantum circuits for implementing a large family of quantum wavelet transforms that are characterized by wave packet trees (Theorems 3.1 and 4.5). This construction is quite general, subject to some mild technical conditions (e.g., the existence of efficiently-computable encodings of the branching patterns of the trees, and the existence of efficient quantum circuits for performing certain 2-dimensional unitary rotations that are related to the shape of the mother wavelet).

By making appropriate choices for the mother wavelet and the branching pattern of the tree, we obtain the first efficient implementation a quantum wave atom transform that satisfies a parabolic scaling law (Corollary 4.6). As mentioned above, this parabolic scaling law is crucial for constructing sparse representations of linear operators that appear, e.g., in solving wave equations [4].

Note that there is an issue of approximation error, when running the quantum circuits described in this paper. These quantum circuits provide an *exact* implementation of the desired quantum wave atom transform, assuming the ability to perform arbitrary 1- and 2-qubit gates. In practice, however, one only has the ability to perform 1- and 2-qubit gates that are chosen from a fixed, finite set G , which generates a dense subgroup of $SU(2)$. Thus, one must use the gates in G to approximate the arbitrary 1- and 2-qubit gates that appear in our quantum circuits for the quantum wave atom transform. This will increase the size of the quantum circuits, and produce some nonzero error in their output. However, these effects can be controlled, while preserving the efficiency of the implementation, using standard techniques. For example, using the Solovay-Kitaev theorem [23], for any $\epsilon > 0$, one can ensure that the error in the output of the quantum circuit is at most ϵ in operator norm, while the size of the quantum circuit is increased by at most a multiplicative factor that is polylogarithmic in n and $1/\epsilon$.

1.3 Related work

There is a large literature on quantum wavelet transforms [8–15]. Many of these works are complementary to ours: they solve technical problems related to generating quantum superposition states that have more complicated shapes (compared to the ones used in our work), but they do not handle wave packet trees with complicated branching patterns (which is the main focus of our construction). As discussed in Section 1.1, the use of wave packet trees with complex branching patterns is crucial for applications that involve solving wave equations.

Among these previous works, the work of Ni et al [15] is particularly relevant, because it provides efficient implementations of quantum transforms using wavelets and Gabor atoms, using a “blending” technique, which we adapt to our setting in Section 4.3. Our main contribution, compared to [15], is to handle a much larger class of wave packet trees, namely “monotonic trees,” defined in Section 2.3; these include parabolic-scaling trees as a special case.

Our results are a first demonstration that a quantum wave atom transform can be implemented efficiently. This raises an interesting question: can one extend these techniques to work with different choices of the mother wavelet? Recent work by Bagherimehrab et al [14], on implementing quantum wavelet transforms using a powerful technique called linear combinations of unitaries, may be helpful here. (Note that the constructions in [14] are quite flexible, and can already handle some wavelet transforms that have simple tree structures. The hope is that these techniques can be extended to handle wave atom transforms, where the tree structure is more elaborate.)

Finally, we expect that our results can be extended to handle wave atoms defined on $\mathbb{R}^2$, by adapting ideas from the existing literature on classical wave atom transforms [3, 4].

1.4 Potential applications of our results

Turning to Question 2, we now describe how our results are potentially useful for quantum algorithms for solving wave equations. At a high level, we argue that the special properties of wave atom bases (namely, the existence of sparse representations of wave propagators using wave atom bases) can be used to improve the performance of quantum algorithms for solving wave equations and differential equations, such as those developed in Costa et al [24] and Bagherimehrab et al [25].

Suppose one wants to solve a linear wave equation, describing wave propagation in an inhomogeneous medium in some finite region of space $\Omega \subset \mathbb{R}^d$, with appropriate boundary conditions. One can use standard techniques to encode this problem into a linear system of equations $Ax = b$ in dimension N , where N is proportional to the number of points used to discretize the domain Ω . (One can also use similar techniques to encode the problem into the dynamics of a quantum Hamiltonian [24].)

One can then run a quantum algorithm, such as the HHL (Harrow-Hassidim-Lloyd) algorithm [26, 27], to solve this system of equations, with running time that scales polylogarithmically with N — an exponential speedup over classical algorithms. (A large body of work has shown similar speedups for solving differential equations, and simulating Hamiltonian time evolution; see, e.g., [28–30].)

However, in order to use the HHL algorithm, certain conditions must be satisfied: the linear system must be *well-conditioned*, and it must have an extremely compact *description*, which is efficiently computable, using space and time polylogarithmic in N . Furthermore, there must be ways to *prepare* the quantum state that encodes b , and *read-out* interesting properties of the solution x , which are similarly efficient (i.e., running in time polylogarithmic in N). While these conditions are quite restrictive, they can be satisfied in certain cases, particularly using techniques that involve sparsity [31, 32].

The hope is that the above conditions can be satisfied by transforming the system of equations $Ax = b$ into a wave atom basis, using a quantum wave atom transform. That is, one computes matrix elements of A with respect to a wave atom basis, and one performs state preparation and read-out in a wave atom basis. Note that this necessarily requires a *quantum* wave atom transform, rather than a classical transform, since we are seeking an algorithm that runs in time polylogarithmic in N .

There are heuristic reasons why this approach might satisfy the conditions needed by the HHL algorithm. First, in cases where the solution to the wave equation satisfies a conservation law (e.g., conservation of energy), one may expect the system of equations $Ax = b$ to be well-conditioned. (A similar intuition was used in the quantum algorithm of [24].)

Second, in light of classical results on sparse representations of wave propagators [3, 4, 16, 17], one may hope that the system of equations $Ax = b$ will be sufficiently sparse in the wave atom basis, so that the computation of A , the preparation of the state encoding b , and the read-out of the solution x , can all be performed in time polylogarithmic in N . (Some aspects of this idea were investigated previously in [25], which showed that a quantum algorithm for solving differential equations can be sped up by preconditioning using a wavelet basis. The new elements in our work are the choice of a wave atom basis, and the role of sparsity.)

Our construction of an efficient quantum wave atom transform is an important step towards realizing this approach to solving wave equations on a quantum computer. However, significant obstacles remain. One obstacle is that this approach requires sparse approximations of wave propagators, where the approximation error is small in operator norm. It is an interesting question whether these kinds of bounds can be proved by adapting the known techniques for bounding the approximation error, which often use the vector ℓ_∞ norm.

If this approach is successful, it would provide a new quantum algorithm for solving wave equations, which performs well when the solution of the wave equation is sparse in the wave atom basis. In such cases, this quantum algorithm might achieve an exponential speedup over classical algorithms, and it might also outperform existing quantum algorithms that use discretization on a grid [24] or a wavelet basis [25] (rather than a wave atom basis).

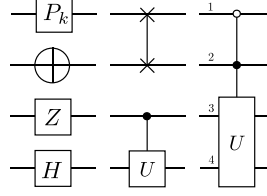
Finally, we mention a few other ways in which our efficient implementation of a quantum wave atom transform might be useful for quantum algorithms. One possibility is to use a quantum wave atom transform to *post-process* the quantum state that is obtained from a quantum differential equation solver, as studied in [33] (using wavelet bases). Another (more speculative) possibility is to use a quantum wave

Fig. 1 Circuit representations of standard quantum gates. Left column (from top to bottom): phase shift P_k -, X -, Z - and H -gates.

Center-top: SWAP gate.

Center-bottom: controlled- U gate, which is equivalent to $|1\rangle\langle 1| \otimes U + |0\rangle\langle 0| \otimes I_2$

Right: example of controlled- U gate, when applied to qubits 3 and 4 if qubit 1 is $|0\rangle$ and qubit 2 is $|1\rangle$. In this case controlled- U is equivalent to $|01\rangle\langle 01| \otimes U + (I_4 - |01\rangle\langle 01|) \otimes I_4$.



atom transform to solve certain geometric problems in $\mathbb{R}^n$, or to prepare quantum superpositions that are potentially useful for solving high-dimensional lattice problems, as studied in [34, 35] (using curvelet bases).

1.5 Organization of this paper

Section 2 discusses methods, including notation and background information about wave packets and wavelet packet trees. In particular, in Section 2.4 we discuss the encoding of wave packets and frequencies, along with an important discussion of the reverse procedure, decoding. This section establishes a foundation to build wave packet transform generated by multilevel wavelet packet trees. In Section 3 we demonstrate the use of encoding/decoding procedures on the example of the Shannon wavelet transform. Section 4 provides a quantum algorithm for the wave atom transform, which is the main result of the paper. Finally, Section 5 discusses directions for further research. Additional details of the algorithms can be found in the Supplementary Material.

2 Methods

2.1 Notations and assumptions

A universal gate set, such as Clifford+ T , enables the approximation of any unitary transformation with arbitrary precision ε [23, 36]. In analyzing gate complexity, this work assumes unrestricted access to all single-qubit and two-qubit gates. Consequently, the gate complexity is expressed in terms of the number of single-qubit and two-qubit gates, while disregarding the $O(1/\varepsilon)$ factor associated with the approximation. This paper implicitly assumes that a given quantum circuit and its controlled version, with a constant number of control qubits, exhibit the same asymptotic gate complexity. However, the gate complexity may increase when accounting for the quantum device topology, particularly in scenarios where qubits are not fully connected [37].

Throughout the paper, we use some standard quantum gates given in Eq. (3) with circuit representations described in Figure 1. X , Z and H represent X -Pauli, Z -Pauli, and Hadamard gates, respectively. The SWAP gate swaps the state of two qubits. Phase shift gates P_k map basis states as $P_k|0\rangle = |0\rangle$ and $P_k|1\rangle = e^{2\pi i/2^k}|1\rangle$, where i is the imaginary unit and $|\cdot\rangle$ are vectors in the Dirac notation such that $|0\rangle = (1, 0)^T$ and $|1\rangle = (0, 1)^T$, $\langle \cdot |$ is a conjugate transpose of the vector. $U^\dagger$ is a conjugate transpose of matrix U and $\bar{z}$ is a conjugate of a complex number z . $\otimes$ denotes the tensor product of matrices, and $\oplus$ represents a direct sum of subspaces in one case and a modulo 2 sum in another, with the meaning clear from the context. Finally, $I(\cdot)$ is an indicator function with some condition, for example, $I(x \leq 0) = 1$ if $x \leq 0$, and 0 otherwise, while I_M is an identity matrix of size $M \times M$.

$$\begin{aligned} P_k &= \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix} & X &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} & \text{SWAP} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ Z &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} & H &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{aligned} \quad (3)$$

We denote a subroutine of integer comparison on j qubits to some value k as $\text{Comparator}_j(k)$, formally,

$$\text{Comparator}_j(k) |x\rangle |b\rangle \mapsto |x\rangle |b + I(x \geq k) \bmod 2\rangle, \quad (4)$$

which can be implemented by adding two's complement of $-k$ with carry bit representing the result, which in turn requires $O(j)$ gates. We denote a subroutine of cyclic shift of j qubits by some value k as

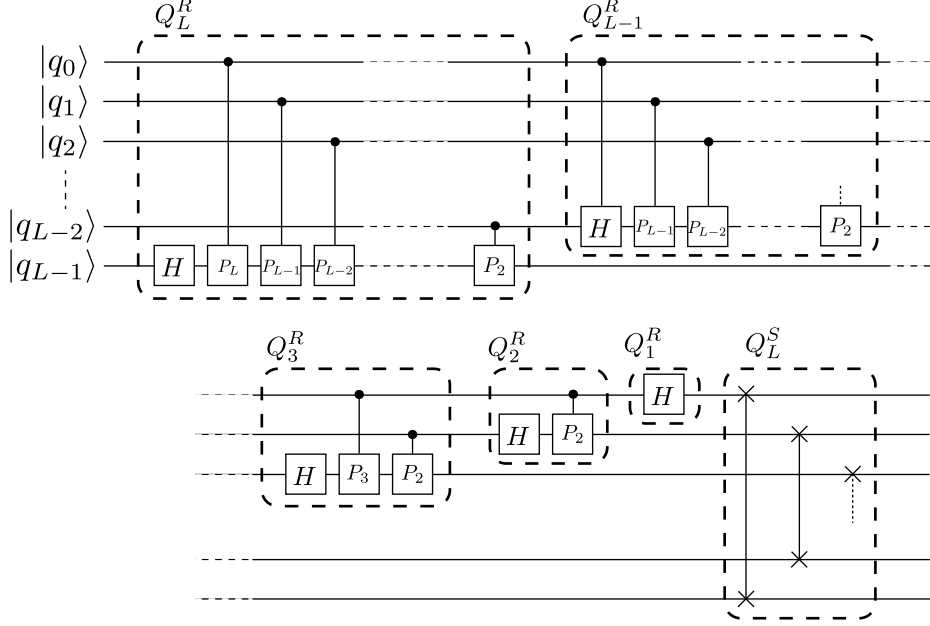


Fig. 2 The circuit of the QFT on L qubits, consisting of QFT rotations Q_i^R , followed by a QFT swap Q_L^S (see Eq. (6)).

Adder $_j(k)$, formally,

$$\text{Adder}_j(k) |x\rangle \mapsto |x + k \bmod 2^j\rangle. \quad (5)$$

There are various implementations of this operation [38–40] with various numbers of ancilla qubits required. [38, 39] require $O(j)$ gates and $O(j)$ ancilla qubits. [40] implements the cyclic shift with the use of the quantum Fourier transform (QFT) [41] that requires at most $O(j^2)$ gates and no ancilla qubits.

We extensively use QFT and its subroutines. Recall the transform of size $N = 2^L$,

$$QFT : |x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x k / N} |k\rangle,$$

where $|x\rangle$ and $|k\rangle$ are basis states of L qubits. Moreover,

$$|k\rangle = |k_{L-1} \dots k_0\rangle = |k_{L-1}\rangle \otimes |k_{L-2}\rangle \otimes \dots \otimes |k_0\rangle$$

where $k = \sum_{j=0}^{L-1} 2^j k_j$ or $k_{L-1} \dots k_0$ is a binary representation of k . This also implies that we follow the big-endian convention.

The implementation of the QFT of L qubits, Q_L , has a decomposition:

$$Q_L = Q_L^S Q_1^R \dots Q_L^R = Q_L^S \left(\prod_{i=1}^L Q_i^R \right), \quad (6)$$

where Q_i^R is a QFT rotation of i qubits and Q_L^S reverses the order of L qubits. QFT rotation Q_i^R comprises of H -gate applied to qubit $i-1$ followed by phase gates P_{i-j+1} applied to qubit $i-1$ controlled by j for $j = 0, \dots, i-2$. Q_L^S reverses the order by swapping each pair of qubits i and $L-i-1$. Figure 2 depicts a circuit of the QFT of L qubits and highlights its subroutines such as QFT rotations Q_i^R and QFT swap Q_L^S .

The classical counterpart of the QFT is the discrete Fourier transform (DFT) which has a different sign of the exponent. Specifically, the DFT of size N in Dirac's notation is

$$DFT : |x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-2\pi i x k / N} |k\rangle.$$

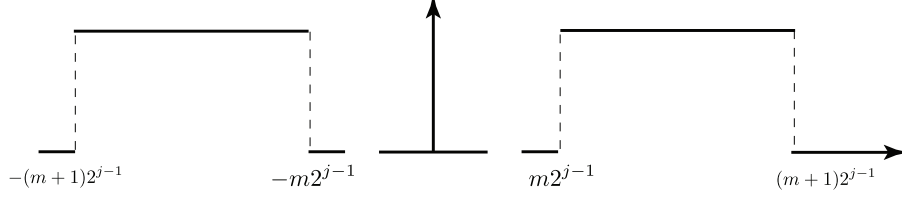


Fig. 3 Shannon wavelet $\hat{\varphi}_m^j$: one bump per each side of the axis in the frequency domain.

Therefore, matrix of the DFT of size 2^L is $Q_L^\dagger$. The DFT, in turn, is a counterpart of the continuous Fourier transform. The Fourier transform of a measurable function $f \in L_2(\mathbb{R})$ is given by the following equation

$$\hat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-i2\pi x \xi} dx. \quad (7)$$

We refer to f as a function in the spatial domain, and $f(x - x_0)$ is a shift by x_0 in the spatial domain by x_0 . The functions with the circumflex, for example $\hat{f}$, are considered in the (ordinary) frequency domain, and $\hat{f}(\xi - \xi_0)$ is a shift by ξ_0 in the frequency domain.

When we consider $f : [0, 1] \mapsto \mathbb{R}$ and uniform discretization in the interval $[0, 1]$, $x_k = k/N$ for $k = 0, \dots, N-1$ where $N = 2^L$ for some integer $L \geq 1$. We use brackets to denote the discretization of a function and let $f[k] = f(k/N)$. In the (ordinary) frequency domain, we emphasize that $k \in \mathbb{Z}$ by the square brackets and $\hat{f}[k] = \hat{f}(k)$. Then in the discrete case,

$$(\hat{f}[0], \dots, \hat{f}[N-1])^T = Q_L^\dagger (f[0], \dots, f[N-1])^T.$$

Notice that we use a different range $\{-N/2, \dots, N/2-1\}$ in the discretization of $\hat{f}$, however, $\hat{f}[-N/2+k] = \hat{f}[N/2+k]$.

2.2 Wave packets

In the paper, we design algorithms that implement wave packet transforms for wave packets with compact frequency support. Moreover, we define wave packets exclusively in the frequency domain, although the wave packets in the spatial domain can be obtained via inverse Fourier transform. We index wave packets $\varphi_{m,n}^j$ by three indices: their scale $j \in \mathbb{Z}_+ \setminus \{0\}$, frequency $m \in \mathbb{Z}_+$, and position in space $n \in \mathbb{Z}$. Here, $\mathbb{Z}_+$ denotes nonnegative integer numbers.

First, we consider wave packets with sharp frequency windows such that generating functions are based on perfect filters:

$$\begin{aligned} \hat{\varphi}_m^0(\xi) &= I \{2\xi \in [-m-1, -m) \cup [m, m+1)\}, \\ \hat{\varphi}_m^j(\xi) &= 2^{-j/2} \hat{\varphi}_m^0(2^{-j}\xi), \\ \hat{\varphi}_{m,n}^j(\xi) &= e^{-i2\pi n 2^{-j}\xi} \hat{\varphi}_m^j(\xi). \end{aligned} \quad (8)$$

(See Figure 3.) The support of $\hat{\varphi}_{m,n}^j$ is centered in frequency around $\pm \xi_{j,m} = \pm 2^{j-1}m$ and in space around $x_{j,n} = 2^{-j}n$. The resulting wave packets are *Shannon wavelets*. The classical indexation arising from multipass filters differs from the above and requires a simple permutation (see Chapter 8 of [19] for details).

The *1D wave atoms* introduced in [4] are based on non-standard wavelets of [5], and are similarly centered in frequency around $\pm \xi_{j,m} = \pm 2^{j-1}m$ and centered in space around $x_{j,n} = 2^{-j}n$:

$$\begin{aligned} \hat{\psi}_m^j(\xi) &= 2^{-j/2} \hat{\psi}_m^0(2^{-j}\xi), \\ \hat{\psi}_{m,n}^j(\xi) &= e^{-i2\pi n 2^{-j}\xi} \hat{\psi}_m^j(\xi). \end{aligned} \quad (9)$$

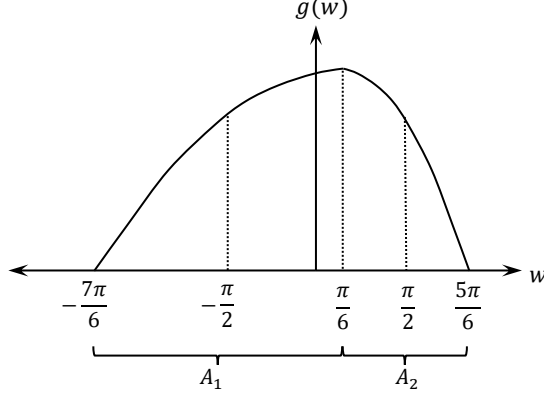


Fig. 4 The function g , satisfying equations (11) and (12)

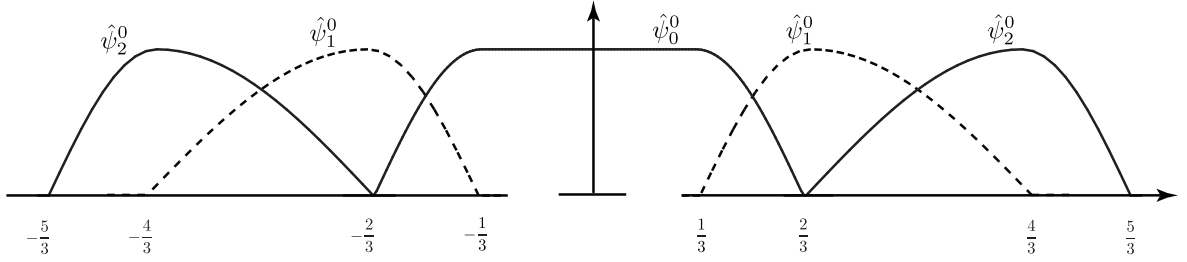


Fig. 5 Wave atoms. $\hat{\psi}_0^0$ is unique, $\hat{\psi}_1^0$ is an example of odd m 's, $\hat{\psi}_2^0$ represents even m 's.

Its mother wave packets demonstrate an elegant way to deal with the problem of frequency localization [4, 5] and involve symmetric pairs of compactly supported bumps in the frequency domain:

$$\hat{\psi}_m^0(\xi) = e^{-i\pi\xi} [e^{i\alpha_m} g((-1)^m(2\pi\xi - 2\alpha_m)) + e^{-i\alpha_m} g((-1)^{m+1}(2\pi\xi + 2\alpha_m))], \quad (10)$$

where $\alpha_m = \frac{\pi}{2}(m + \frac{1}{2})$ and g is a continuous real-valued function compactly supported on an open interval $(-\frac{7\pi}{6}, \frac{5\pi}{6})$. Furthermore, g must satisfy the following requirements: for any $w \in [-\frac{\pi}{3}, \frac{\pi}{3}]$,

$$g^2(\frac{\pi}{2} - w) + g^2(\frac{\pi}{2} + w) = 1, \quad (11)$$

$$g(-2w - \frac{\pi}{2}) = g(\frac{\pi}{2} + w). \quad (12)$$

The above construction can be understood qualitatively in the following way. First, one can write the support of g as the union of two intervals:

$$A_1 = [-\frac{7\pi}{6}, \frac{1\pi}{6}], \quad A_2 = [\frac{1\pi}{6}, \frac{5\pi}{6}]. \quad (13)$$

(See Figure 4.) Typically g is chosen to be monotonically increasing on A_1 , and monotonically decreasing on A_2 . In Eq. (11), $g(\frac{\pi}{2} - w)$ and $g(\frac{\pi}{2} + w)$ can be understood as mirror images of each other, reflected around the midpoint of the interval A_2 . In the wave atom transform, these mirror images are associated with neighboring wavepackets, and Eq. (11) says that their squares sum to 1. Eq. (12) says that the increasing part of g (on the interval A_1 , centered at $-\pi/2$) is a mirror image of the decreasing part of g (on the interval A_2 , centered at $\pi/2$), stretched by a factor of 2. This asymmetric property (12) of profile g is important to preserve the orthogonality of the $\hat{\psi}_{m,n}^j$. In contrast to Shannon wavelets, wave atoms have overlapping support in the frequency domain.

One can describe the wavepackets $\hat{\psi}_m^0(\xi)$ qualitatively as follows. (See Figure 5.) In Eq. (10), $\hat{\psi}_m^0(\xi)$ consists of a complex phase factor of $e^{-i\pi\xi}$, times a linear combination of two bumps, which are supported

near $\pm\alpha_m/\pi = \pm\frac{1}{2}(m + \frac{1}{2})$. These two bumps are mirror images of each other, with a relative phase difference of $e^{2i\alpha_m} = i(-1)^m$. Furthermore, when comparing wavepackets $\hat{\psi}_m^0(\xi)$ and $\hat{\psi}_{m+1}^0(\xi)$ that are adjacent to each other in the frequency domain, one sees that the bumps corresponding to m overlap with, and are mirror images of, the bumps corresponding to $m + 1$. In other words, the shapes of the wavepackets $\hat{\psi}_m^0(\xi)$ change according to the parity of m .

Because of the above properties, these wavepackets form an orthonormal basis so that for the square-integrable function $f \in L^2(\mathbb{R})$,

$$f = \sum_{j,m} \sum_n \langle f, \psi_{m,n}^j \rangle \psi_{m,n}^j = \sum_{j,m} \sum_n c_{j,m,n} \psi_{m,n}^j. \quad (14)$$

For all (j, m) , the coefficients $c_{j,m,n}$ can be interpreted as a convolution at scale 2^{-j} and are defined by Plancherel's theorem as follows:

$$\begin{aligned} c_{j,m,n} &= \langle f, \psi_{m,n}^j \rangle = \int \psi_{m,n}^j(x) f(x) dx \\ &= \int \psi_m^j(x - 2^{-j}n) f(x) dx = \frac{1}{2\pi} \int e^{i2\pi 2^{-j}n\xi} \overline{\hat{\psi}_m^j(\xi)} \hat{f}(\xi) d\xi. \end{aligned} \quad (15)$$

2.2.1 Approximation and Discretization

Since we are interested in numerical computations (both classical and quantum), the question naturally arises: is there a way to implement these wavelet transforms approximately, while preserving the orthogonality of the wavepackets? The answer is yes. Intuitively, the reason is that the orthogonality only depends on the overlaps between *pairs* of neighboring wavepackets $\hat{\psi}_m^j$ and $\hat{\psi}_{m'}^{j'}$, i.e., it does not depend on relationships involving more than two wavepackets at a time. Even if both of the wavepackets $\hat{\psi}_m^j$ and $\hat{\psi}_{m'}^{j'}$ are computed approximately on the region where they overlap, one ensure that the approximation errors “cancel out,” so that the wavepackets are still orthogonal.

The above argument can be made precise, by noticing that the orthogonality of the wavepackets follows from Eqs. (11) and (12), involving the function g . We can ensure that these equations hold exactly, even if g is only computed approximately, in the following way. Recall that the support of g consists of the set $A_1 \cup A_2$ defined in Eq. (13). We can split this set into four subsets: the left and right halves of the interval A_1 , and the left and right halves of the interval A_2 . Then we can compute the function g approximately on one of these subsets, and then use Eqs. (11) and (12) to determine the values of g on the rest of the set $A_1 \cup A_2$. In this way, even though g is only computed approximately, Eqs. (11) and (12) are still satisfied with high precision, hence the wavepackets are still orthogonal.

Finally, we introduce the discrete version of the wavelet transform. For simplicity, we make some fairly strong assumptions about the function f , in order to motivate the definition of the discrete transform. We consider $f : [0, 1) \rightarrow \mathbb{R}$, and we extend its domain to all of $\mathbb{R}$ while imposing periodic boundary conditions, i.e., $f(x + 1) = f(x)$. Then f can be represented by a Fourier series. We assume that f is band-limited, so it can be recovered from the first N terms in its Fourier series, where $N = 2^L$ for some integer $L \geq 1$. By the Shannon sampling theorem, f can also be recovered from its uniform discretization on the interval, $f[k] = f(k/N)$ for $k = 0, \dots, N - 1$. In this setting, it is natural to define a discrete version of the wavelet transform:

$$c_{j,m,n}^D = \sum_{k=-N/2, \dots, N/2-1} \overline{\hat{\psi}_{m,n}^j(k)} \hat{f}[k]. \quad (16)$$

Therefore, the operation of our interest is

$$(f[0], \dots, f[N - 1]) \mapsto \{c_{j,m,n}^D\}_{(j,m,n) \in \Gamma}, \quad (17)$$

where $\{\hat{\psi}_{m,n}^j\}_{(j,m,n) \in \Gamma}$ organize an orthogonal basis for some set of index triplets Γ . We discuss the selection of orthogonal bases in the next section.

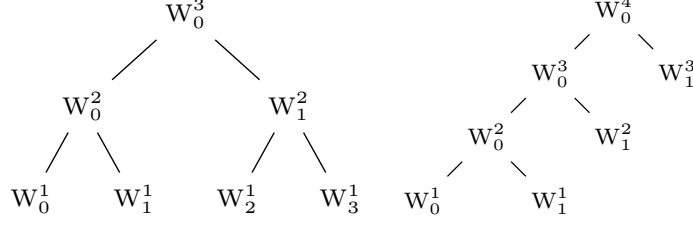


Fig. 6 Examples of trees that generate wavelet packet bases. Left: a perfect binary tree, which generates a uniform wavelet packet basis. Right: a dyadic tree, which generates a dyadic wavelet packet basis.

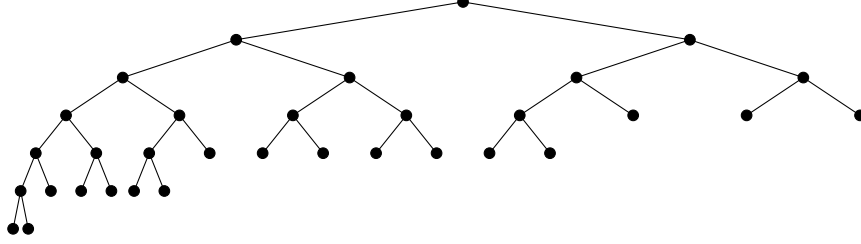


Fig. 7 Example of a tree that generates a wave atom basis, due to [4]. The branching pattern of the tree is chosen so that the wave atom basis functions obey the parabolic scaling law in Eq. (2). To simplify the picture, the labels on the nodes of the tree are not shown.

2.3 Wavelet packet trees

Coifman, Meyer, and Wickerhauser introduced wavelet packets by linking multiresolution approximations (a sequence of embedded vector spaces $(V_j)_{j \in \mathbb{Z}}$ for approximating $L_2(\mathbb{R})$ functions) with wavelets [42]. They decompose a multiresolution approximation space V_j into a lower-resolution space V_{j+1} and a detail space W_{j+1} by splitting the orthogonal basis into two new orthogonal bases. Intuitively, when wavelets are used for image processing, we may lose some details when reducing the resolution. [42] also show that, instead of just dividing the approximation spaces to create the so-called detail spaces and wavelet bases, we can split the detail spaces to generate new bases. This process of dividing vector spaces can be represented in a structure of a binary tree (see Chapter 8 of [19]). Any node in this binary tree can be labeled by (j, m) , where j is the level of the tree, and m is the number of nodes on its left at the same level. To each node we associate a space W_m^j , which admits an orthogonal basis $\{\varphi_{m,n}^j\}_{n \in \mathbb{Z}}$.

We adopt the notion of wavelet packet trees and associate W_m^j with an orthogonal basis $\{\hat{\psi}_{m,n}^j\}_{n \in \mathbb{Z}}$ which in the discrete case becomes $\{\hat{\psi}_{m,n}^j\}_{0 \leq n < 2^j}$ for wave atoms or $\{\hat{\varphi}_{m,n}^j\}_{0 \leq n < 2^j}$ for the Shannon wavelet. Then, all W_m^j can be organized into a binary wavelet packet tree. We will abuse the notation and refer to W_m^j as both the space spanned by the basis $\{\hat{\psi}_{m,n}^j\}_{0 \leq n < 2^j}$ (or $\{\hat{\varphi}_{m,n}^j\}_{0 \leq n < 2^j}$) and the node in the tree.

Definition 2.1 (Admissible wavelet packet binary tree, adapted [19]). *A binary tree of height $L - 1$ with nodes $\{W_m^j\}$ is admissible if*

1. The root of the tree is W_0^L ,
2. Every non-leaf node W_m^j has two children W_{2m}^{j-1} and W_{2m+1}^{j-1} , and $j \geq 2$.

Using Definition 2.1, one can construct wavelet bases $\{\varphi_{m,n}^j \mid (j, m, n) \in \Gamma_T\}$, where Γ_T is the set of index triplets generated by the wave packet admissible tree T :

$$\Gamma_T = \{(j, m, n) \mid W_m^j \in \Lambda_T \text{ and } 0 \leq n < 2^j\}, \quad (18)$$

$$\Lambda_T = \{W_m^j \mid W_m^j \text{ is a leaf-node in } T\}. \quad (19)$$

Two of the more popular bases generated by wavelet packet trees [19] are the following:

- *Uniform* $\{\varphi_{m,n}^j\}$, where j 's are equal to some fixed value (call it j_0), and $\Gamma = \{(j, m, n) \mid 0 \leq m < 2^{L-j}, 0 \leq n < 2^j, j = j_0\}$. This type of basis is generated by perfect binary trees.

- *Dyadic* $\{\varphi_{m,n}^j\}$. In this case, most of the basis function has $m = 1$, and $\Gamma = \{(1, 0, 0), (1, 0, 1)\} \cup \{(j, 1, n) \mid 1 \leq j < L, 0 \leq n < 2^j\}$. This type of basis is generated by dyadic trees (such that every right node is a leaf while every left node is nonleaf unless the node is on the minimal level).

The examples of wavelet packet trees generating orthogonal bases discussed above can be found in Figure 6. One can verify that both approaches render an orthogonal basis for Shannon wavelets and wave atoms.

In addition, we will consider another class of trees that generate orthogonal bases for the wave atom transform, and have a special property, “parabolic scaling,” defined in Eq. (2), which is desirable for many applications of the wave atom transform [4]. We briefly describe how Eq. (2) constrains the structure of the wavepacket tree. Recall that Eq. (2) says that, for each wavelet basis function, wavelength $\propto$ (diameter)². From the discussion in Section 2.2, the basis function ψ_m^j has wavelength $\propto (1/m)2^{-j}$ and diameter $\propto 2^{-j}$. Hence Eq. (2) implies that:

$$m \propto 2^j. \quad (20)$$

This condition must be satisfied at every leaf of the wavepacket tree. Hence, the tree resembles a complete binary tree whose lower-right branches have been cut away at the points specified by Eq. (20). An example of such a tree is shown in Fig. 7.

Finally, we define a class of “monotonic” trees that is easier to work with, and contains the “parabolic scaling” trees described above:

Definition 2.2. An admissible wave packet tree T is called *monotonic* if, when the tree’s leaf nodes W_m^j are visited from left to right, the sequence of j values is nondecreasing.

Proposition 2.1. If T is an admissible wave packet tree that satisfies the parabolic scaling relation (20), then T is monotonic.

Proof. Suppose T is not monotonic. Then there are leaf nodes W_m^j and $W_{m'}^{j'}$ such that W_m^j appears to the left of $W_{m'}^{j'}$ and $j > j'$. Since both of these leaf nodes satisfy Eq. (20), we must have $m > m'$. Since T is admissible, we have $m \leq m'$, leading to a contradiction. $\square$

We will develop algorithms for monotonic trees (which will apply to parabolic scaling trees, as a special case) in Sections 2.6 and 4.2.

We would also like to state a few properties of admissible wave packet trees that will be helpful in the later discussion.

Proposition 2.2 (enumeration properties of admissible wave packet tree). For wave packet admissible tree T (by definition 2.1) with a root W_0^L and K leaf nodes $\{W_{m_i}^{j_i} \mid i = 1, \dots, K\}$ traversed from left to right,

1. $m_1 = 0$,
2. For $i = 1, \dots, K - 1$, $(m_i + 1)2^{j_i} = m_{i+1}2^{j_{i+1}}$,
3. $(m_K + 1)2^{j_K} = 2^L$.

Proof. We traverse the tree as follows:

1. We start from the root node W_0^L and descend the tree while always moving down to a left child, so at each level $W_0^j \mapsto W_0^{j-1}$. Suppose that we stopped at node $W_0^{j'}$, then $W_0^{j'}$ is a leaf node; otherwise, it should have left and right children. Therefore, the first node in the traverse is $W_0^{j'}$, i.e. $m_1 = 0$.
2. Imagine that we will add nodes to the tree T until it becomes a perfect binary tree of height of L , that is, for each leaf node W_m^j we append a perfect binary subtree of height j at that node. Then the leftmost leaf node in the subtree with root $W_{m_i}^{j_i}$ is $W_{m_i 2^{j_i}}^0$, and the rightmost leaf node is $W_{m'}^0$ where $m' = (m_i + 1)2^{j_i} - 1$. The leaf node next to $W_{m'}^0$ is the leftmost leaf node in the subtree with root $W_{m_{i+1}}^{j_{i+1}}$, which is $W_{m_{i+1} 2^{j_{i+1}}}^0$. Therefore, $(m_i + 1)2^{j_i} = m_{i+1}2^{j_{i+1}}$.
3. Observe that the rightmost node in the expanded perfect binary tree (constructed in 2) is $W_{2^L-1}^0$ and is also the rightmost node of the subtree with root $W_{m_K}^{j_K}$. Therefore, $(m_K + 1)2^{j_K} = 2^L$.

$\square$

2.4 Encoding

The transform of interest in equation (17) does not define an order of the coefficients $c_{j,m,n}^D$. In other words, $\hat{\varphi}_{m,n}^j$ and $\hat{\psi}_{m,n}^j$ need to be indexed. Moreover, in equation (16) the sum goes over negative and positive integer frequencies. We would like to encode both the wave packet indices (j, m, n) and the (integer) frequencies k such that each is represented by L bits.

2.5 Frequency encoding

This section introduces frequency encoding and decoding procedures, where encoded frequencies are represented by L bits. It also covers the concept of retaining decoding, allowing a subset of bits to be decoded without affecting others, and involves constructing corresponding quantum circuits.

Given that both Shannon wavelets and wave atoms are symmetric around zero, we would like to map the frequencies of the same absolute value with different sign in such a way that they would have consecutive indices when encoded. Moreover, it is natural to expect that the nonnegative frequency k would have a smaller encoded index than the frequency $k + 1$.

In order to accomplish this, we introduce a *frequency encoding procedure* as a bijection $e : \{-N/2, \dots, N/2 - 1\} \rightarrow \{0, \dots, N - 1\}$ given by

$$e(k) = \begin{cases} 2k, & k \geq 0, \\ 2|k| - 1, & k < 0. \end{cases} \quad (21)$$

We call the reverse procedure of (21) *frequency decoding procedure* and define it by the inverse mapping $d : \{0, \dots, N - 1\} \rightarrow \{-N/2, \dots, N/2 - 1\}$ given by

$$d(i) = \begin{cases} \frac{1}{2}i, & i \text{ even}, \\ -\frac{1}{2}(i + 1), & i \text{ odd}. \end{cases} \quad (22)$$

We also introduce a *retaining decoding procedure* $\tilde{d}(\cdot, j, m) : \{0, \dots, 2^j - 1\} \rightarrow \{0, \dots, 2^j - 1\}$ such that for $c \in \{0, \dots, 2^j - 1\}$

$$\tilde{d}(c, j, m) = d(m2^j + c) \bmod 2^j. \quad (23)$$

Note that this is function $\tilde{d}$ is *not* the inverse of the encoding function introduced above. Rather, this function will be used in our implementations of quantum wavelet transforms, later in this article. The word “retaining” refers to the idea of staying in range of size 2^j (which will be fully explored in section 3) and the argument c corresponds to the indices within the range of desired size. We can compute $\tilde{d}$ as follows

$$\tilde{d}(c, j, m) = \begin{cases} 2^{j-1}I(m \text{ is odd}) + \frac{1}{2}c, & \text{if } c \text{ is even,} \\ 2^j - 2^{j-1}I(m \text{ is odd}) - \frac{1}{2}(c + 1), & \text{if } c \text{ is odd.} \end{cases} \quad (24)$$

2.5.1 Frequency encoding circuit

Both Shannon wavelet and wave atom transforms include DFT (inverse QFT) as the initial step to transform the spatial domain into the frequency domain. However, the result of DFT is defined on a grid $\{0, \dots, N - 1\}$, but our transforms are defined on a grid $\{-N/2, \dots, N/2 - 1\}$. Due to the periodicity of Fourier transform $\hat{f}[k] = \hat{f}[k - N]$, we have the initial stage as follows

$$Q_L^\dagger \left(\sum_{k=0}^{N-1} f[k] |k\rangle \right) = \sum_{k=0}^{N-1} \hat{f}[k] |k\rangle = \sum_{k=0}^{N/2-1} \hat{f}[k] |k\rangle + \sum_{k=-N/2}^{-1} \hat{f}[k] |k + N\rangle.$$

We transform $|k\rangle$ into $|e(k)\rangle$ when $k \geq 0$ and $|k + N\rangle$ into $|e(k)\rangle$ when $k < 0$ as described in (21). The further steps of the transforms are based on

$$\sum_{k=-N/2}^{N/2-1} \hat{f}[k] |e(k)\rangle = \sum_{k=0}^{N-1} \hat{f}[d(k)] |k\rangle. \quad (25)$$

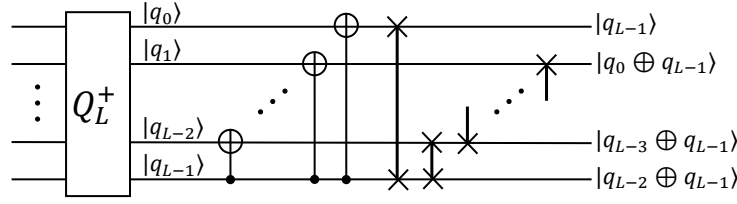


Fig. 8 Frequency encoding circuit which implements function $e(\cdot)$ combined with the inverse QFT $Q_L^\dagger$

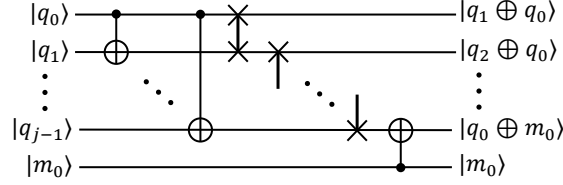


Fig. 9 Retaining decoding circuit $\tilde{D}_j$ which implements function $\tilde{d}(\cdot, j, m)$

The two initial steps of the transforms, the inverse QFT and frequency encoding, are carried out by the circuit shown in Figure 8 (see Supplementary Material for details). The gate complexity of the algorithm is $O(L^2)$ due to the QFT step.

2.5.2 Retaining decoding circuit

Recall the retaining decoding procedure defined by (23). Notice that $\tilde{d}(c, j, m) = \tilde{d}(c, j, m \bmod 2)$ meaning that the retaining decoding procedure depends on the parity of m rather than the value. Considering the parity of m we may construct the quantum circuit of $\tilde{d}(\cdot, j, m)$ that operates on $j + 1$ qubits: j qubits in representation of c and 1 qubit of m_0 where m_0 is the least significant qubit of m . See Figure 9. (Additional details are presented in the Supplementary Material.) We refer to this circuit as $\tilde{D}_j$. The implementation of $\tilde{D}_j$ has gate complexity of $O(j)$.

In the special case of $j = L$, we do not pass the parity of m to the algorithm and assume that $m_0 = 0$.

2.6 Wave packet encoding

The wavelet packet transform (17) can be written as

$$\sum_{k=0}^{N-1} f[k] |k\rangle \mapsto \sum_{(j,m,n) \in \Gamma_T} c_{j,m,n}^D |(j, m, n)\rangle,$$

where $N = 2^L$, each $c_{j,m,n}^D \approx \langle f, \psi_{m,n}^j \rangle$ and $|(j, m, n)\rangle$ is a representation of the triplet (j, m, n) . In this section, we discuss the encoding of such triplets using L bits. To do this, we will introduce data structures for representing wave packet trees.

Consider an admissible wave packet tree T (in the sense of Definition 2.1), described by the set of its leaf nodes $\Lambda_T = \{W_m^j\}$. By Proposition 2.2, the mapping

$$(j, m, n) \mapsto p = m2^j + n \tag{26}$$

is a bijection between the set of index triplets $\Gamma_T = \{(j, m, n) | W_m^j \in \Lambda_T, 0 \leq n < 2^j\}$ and the range of $\{0, \dots, N - 1\}$. Observe that j least significant bits of the encoded index p represent n in binary form, while the remaining bits represent m . Therefore, to decode the index, we need to know j . This requires information about the structure of the tree T .

2.6.1 Encodings of wave packet trees

We will construct algorithms for implementing wave packet transforms, given access to information about the wave packet tree T through the following mechanism. Let us define a function $\tilde{h} : \{0, \dots, N-1\} \mapsto \{1, \dots, L-1\}$ such that

$$\forall (j, m, n) \in \Gamma_T, \quad \tilde{h}(m2^j + n) = j. \quad (27)$$

In other words, the function $\tilde{h}$ maps the encoded index p back to j .

Furthermore, we define a family of *helper decoding boolean functions* $h_j : \{0, \dots, N-1\} \mapsto \{0, 1\}$ for $1 \leq j < L$ such that

$$h_j(p) = 1 \text{ if and only if } \tilde{h}(p) \geq j. \quad (28)$$

Note that the function $\tilde{h}(\cdot)$ and the family of functions $h_j(\cdot)$ allows us to decode p 's.

In this paper, we assume that there is an efficient implementation of functions h_j , specifically

$$|p\rangle |b_1 \dots b_{L-1}\rangle \mapsto |p\rangle |b_1 \oplus h_1(p), \dots b_{L-1} \oplus h_{L-1}(p)\rangle$$

can be implemented using $O(L^2)$ gates. We demonstrate that this is possible for specific kinds of wave packet trees, including those described in Section 2.3:

- *Tree generating uniform basis.* All j 's of leaf nodes in the wave packet tree are equal, say $j = j'$, so $h_j(p) = I(j \geq j')$, with the implementation using one X -gate.
- *Dyadic tree.* Observe that $h_{L-1}(p) = p_{L-1}$, $h_{L-2}(p) = h_{L-1}(p) \vee p_{L-2} = p_{L-1} \vee p_{L-2}$, in general,

$$h_i(p) = \bigvee_{i'=i}^{L-1} p_{i'} = h_{i+1}(p) \vee p_i \text{ for } i = 2, \dots, L-2,$$

and $h_1(p) = 1$, where $\vee$ is a logical OR operator. The procedure requires $O(L)$ gates.

- *Monotonic tree.* Denote by m_j^* , index m of the leftmost leaf node on the level j . If no leaf nodes are on the level, m_j^* is the index of the node which could be next to the most right node on level j , that is, $m_j^* - 1$ is the index of the most right node. If there are no nodes on the level, let $m_j^* = 0$. Then

$$h_j(p) = I(p \geq m_j^* 2^j).$$

Given that p is an L -bit value, the comparison requires $O(L)$ classical bit operations or quantum gates. The total number of gates for the procedure to compute all $h_j(p)$, $1 \leq j < L$, is $O(L^2)$.

Recall from Section 2.3 that the wave atom transform requires a parabolic tree, which is covered by the case of monotonic trees. In the paper, we focus on monotonic trees and assume that all trees of interest are encoded by the indices of the leftmost leaf nodes at each level, m_j^* 's. This tree representation requires $O(L^2)$ bits and, therefore, does not increase the complexity of circuit construction.

3 Results: Quantum Shannon wavelet transform

In this section, we demonstrate how the retaining decoding function $\tilde{d}$ and the helper decoding functions h_j can be used to construct an efficient implementation of the quantum Shannon wavelet transform for a monotonic tree (see Theorem 3.1).

Recall the definition of Shannon wavelets $\hat{\varphi}_m^j$ by equation (8),

$$\hat{\varphi}_m^j(\xi) = 2^{-j/2} I \{ 2^{-j+1} \xi \in [-m-1, -m) \cup [m, m+1) \}.$$

(See also Figure 3.) It is straightforward to see that for any wave packet admissible tree T generating a transform of size $N = 2^L$, each frequency $k \in [-N/2, N/2)$ is in the support of one Shannon wavelet $\hat{\varphi}_m^j$ corresponding to a leaf node $W_m^j \in \Lambda_T$. With the inclusion of shifts in the spatial domain,

$$\hat{\varphi}_{m,n}^j(\xi) = e^{-i2\pi n 2^{-j} \cdot \xi} \hat{\varphi}_m^j(\xi), \quad (29)$$

and following the definition of the discrete wave packets transform (16), we can formally define the quantum Shannon wavelet transform.

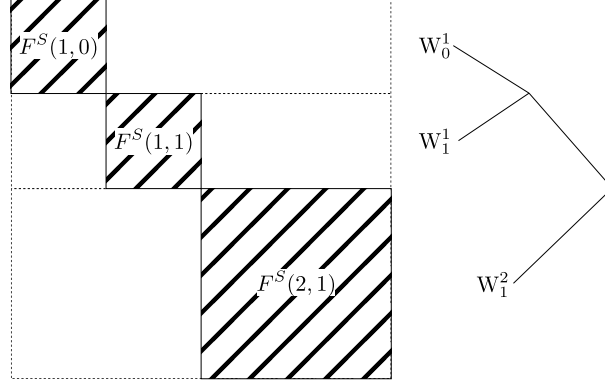


Fig. 10 Block-diagonal structure of matrix C^S with the blocks $F^S(1,0)$, $F^S(1,1)$ and $F^S(2,1)$ corresponding to leaf nodes of $T = \{W_0^1, W_1^1, W_1^2\}$.

Definition 3.1. Consider an admissible wave packet tree T with a root node W_0^L . Define a matrix C^S as

$$C_{m2^j+n,i}^S = \overline{\hat{\varphi}_{m,n}^j(d(i))}, \quad (30)$$

where $W_m^j \in \Lambda_T$, d is the decoding function in equation (22), and $n, i \in \mathbb{Z}$ such that $0 \leq n < 2^j$ and $0 \leq i < N$. The following transform of size $N = 2^L$, where Q_L is QFT,

$$\sum_{k=0}^{N-1} f[k] |k\rangle \mapsto \sum_{W_m^j \in \Lambda_T} \sum_{n=0}^{2^j-1} \left(\sum_{i=0}^{N-1} C_{m2^j+n,i}^S \hat{f}[d(i)] \right) |m2^j+n\rangle \quad (31)$$

where $\hat{f}[k] = \begin{cases} \sum_{i=0}^{N-1} (Q_L^\dagger)_{ki} f[i], & \text{if } k \geq 0, \\ \sum_{i=0}^{N-1} (Q_L^\dagger)_{(k+N)i} f[i], & \text{if } k < 0, \end{cases}$

is called a quantum Shannon wavelet transform.

3.1 Block structure

The Shannon wavelet transform has a block structure, which allows it to be implemented efficiently by a quantum circuit, as follows. When we consider a leaf node $W_m^j \in \Lambda_T$ and $n = 0, \dots, 2^j - 1$, $C_{m2^j+n,i}^S$ can be expressed as follows

$$\begin{aligned} C_{m2^j+n,i}^S &= \overline{\hat{\varphi}_{m,n}^j(d(i))} \\ &= 2^{-j/2} e^{i2\pi 2^{-j} n \cdot d(i)} I \{d(i) \in [-(m+1)2^{j-1}, -m2^{j-1}) \cup [m2^{j-1}, (m+1)2^{j-1})\} \\ &= 2^{-j/2} e^{i2\pi 2^{-j} n \cdot d(i)} I \{i \in [m2^j, (m+1)2^j)\}. \end{aligned}$$

Denote $c = i - m2^j$, then

$$C_{m2^j+n,m2^j+c}^S = 2^{-j/2} e^{i2\pi 2^{-j} n \cdot d(m2^j+c)} I \{0 \leq c < 2^j\}.$$

This implies that C^S is a block diagonal matrix with blocks of size $2^j \times 2^j$ for each leaf node $W_m^j \in \Lambda_T$. Let us denote the block corresponding to W_m^j by $F^S(j, m)$. See Figure 10 for an example of the block-diagonal structure of C^S . Furthermore, given the periodicity of the exponent term of $C_{m2^j+n,m2^j+c}^S$, we can replace $d(m2^j + c)$ with $\tilde{d}(j, m, c)$ by definition (23). Then for $W_m^j \in \Lambda_T$, $0 \leq n, c < 2^j$,

$$C_{m2^j+n,m2^j+c}^S = (F^S(j, m))_{n,c} = 2^{-j/2} e^{i2\pi 2^{-j} n \cdot \tilde{d}(c,j,m)}. \quad (32)$$

This can also be written more compactly as:

$$C^S = \bigoplus_{W_m^j \in \Lambda_T} F^S(j, m). \quad (33)$$

3.2 Implementation on a quantum computer

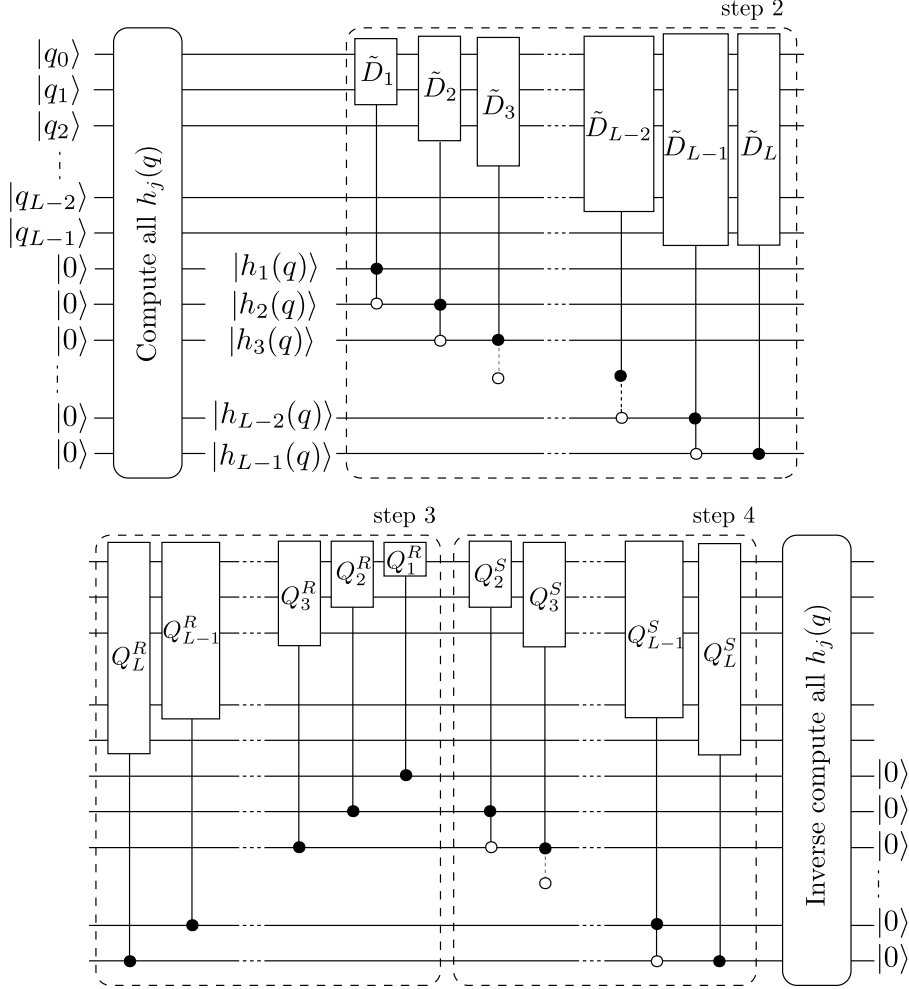


Fig. 11 Quantum circuit implementing Shannon wavelet transform C^S of size 2^L . The family of boolean functions h_j is defined in (28), the retaining decoding circuits $\tilde{D}_j$ are shown in Figure 9. Q_j^R and Q_j^S are steps of the quantum Fourier transform, see Figure 2.

As shown above, $F^S(j, m)$ consists of two actions: (1) decoding $\tilde{d}$, and (2) applying a QFT on j qubits. For the decoding $\tilde{d}$ we pass $j + 1$ qubits to the retaining decoding circuit $\tilde{D}_j$ (see Figure 9), where the most significant qubit gives parity of m . To know j we use the helper decoding functions h_j by definition (28). Finally, we apply QFT on j qubits controlled by ancilla qubits that contain the value of h_j .

Combining these steps, we get the quantum circuit shown in Figure 11. For a detailed description, see the Supplementary Material. The total gate complexity of the algorithm is $O(L^2)$. In summary, we have shown:

Theorem 3.1. *Let T be a monotonic wave packet admissible tree defined for discretization of size $O(2^L)$. If T is represented by the leftmost node on each level, then the gate complexity of quantum Shannon wavelet transform generated by the tree T is $O(L^2)$. The complexity of the circuit construction is $O(L^2)$.*

4 Results: Quantum wave atom transform

In this section, we define and implement the quantum wave atom transform in one dimension, and we prove that it is efficient, for a variety of choices of the mother wave atom and the wave atom tree structure (see Theorem 4.5 and Corollary 4.6). This is a step towards implementing two-dimensional wave atoms, which have numerous applications [4], on a quantum computer.

To do this, we address two technical issues. First, in contrast to the Shannon wavelet transform, wave atoms have supports that overlap in the frequency domain; therefore, the wave atom transform requires considering the “blending” of the elements of $\hat{f}$.

Second, for many applications, it is desirable to use wave atoms that obey a “parabolic” scaling relationship, i.e., wavelength $\sim$ (diameter of support)² [4]. In order to do this, it is necessary to use more general classes of wave packet trees (introduced in Section 2.6), combined with a specific construction of wave packets that allows neighboring wave packets to have different scales (due to Villemoes [5], and described in Section 2.2).

These techniques are quite flexible, and can be used to implement quantum wave atom transforms based on a variety of wave packet tree structures (see Theorem 4.5), including monotonic trees (see Corollary 4.6).

4.1 Wave atoms’ support over the frequency domain

Recall the definition of wave atoms $\hat{\psi}_m^j$ by equations (9) and (10),

$$\hat{\psi}_m^j(\xi) = 2^{-j/2} \hat{\psi}_m^0(2^{-j}\xi),$$

$$\hat{\psi}_m^0(\xi) = e^{-i\pi\xi} [e^{i\alpha_m} g((-1)^m(2\pi\xi - 2\alpha_m)) + e^{-i\alpha_m} g((-1)^{m+1}(2\pi\xi + 2\alpha_m))],$$

where $\alpha_m = \frac{\pi}{2}(m + \frac{1}{2})$. The structure of wave atom is dictated by the function g and its properties. Given that the support of function g is $(-\frac{7}{6}\pi, \frac{5}{6}\pi)$, we derive the support of the wave atom $\hat{\psi}_m^j$ over the frequency domain:

$$\begin{aligned} & 2^{j-1} \left(-\frac{4}{3}, \frac{2}{3}\right) \cup 2^{j-1} \left(-\frac{2}{3}, \frac{4}{3}\right) = 2^{j-1} \left(-\frac{4}{3}, \frac{4}{3}\right), \text{ if } m = 0, \\ & 2^{j-1} \left(-\left(m + \frac{5}{3}\right), -\left(m - \frac{1}{3}\right)\right) \cup 2^{j-1} \left(\left(m - \frac{1}{3}\right), \left(m + \frac{5}{3}\right)\right), \text{ if } m \text{ is odd,} \\ & 2^{j-1} \left(-\left(m + \frac{4}{3}\right), -\left(m - \frac{2}{3}\right)\right) \cup 2^{j-1} \left(\left(m - \frac{2}{3}\right), \left(m + \frac{4}{3}\right)\right), \text{ if } m \text{ is even.} \end{aligned} \quad (34)$$

(See Figure 5.) The support of each wave atom consists of two intervals, one for negative and one for positive frequencies. This can be rewritten as a union of four intervals with centers $\pm 2^{j-1}m$ and $\pm 2^{j-1}(m+1)$, as follows:

$$\begin{aligned} & \pm 2^{j-1} \left(-\frac{2}{3}, \frac{2}{3}\right) \cup \pm 2^{j-1} \left(1 - \frac{1}{3}, 1 + \frac{1}{3}\right), \text{ if } m = 0, \\ & \pm 2^{j-1} \left(m - \frac{1}{3}, m + \frac{1}{3}\right) \cup \pm 2^{j-1} \left((m+1) - \frac{2}{3}, (m+1) + \frac{2}{3}\right), \text{ if } m \text{ is odd,} \\ & \pm 2^{j-1} \left(m - \frac{2}{3}, m + \frac{2}{3}\right) \cup \pm 2^{j-1} \left((m+1) - \frac{1}{3}, (m+1) + \frac{1}{3}\right), \text{ if } m \text{ is even.} \end{aligned} \quad (35)$$

Let us denote those intervals centered around $\pm 2^{j-1}m$ and $\pm 2^{j-1}(m+1)$ as $\chi_{\pm 2^{j-1}m}$ and $\chi_{\pm 2^{j-1}(m+1)}$ correspondingly. Then (34) can be rewritten in general case as

$$\chi_{-2^{j-1}(m+1)} \cup \chi_{-2^{j-1}m} \cup \chi_{2^{j-1}m} \cup \chi_{2^{j-1}(m+1)} \quad (36)$$

We will implement a discrete wave atom transform, by restricting to integer frequencies only. For this purpose, let us denote the integer part of the half interval lengths of $\chi_{\pm 2^{j-1}m}$ and $\chi_{\pm 2^{j-1}(m+1)}$ by $\mu_0(j, m)$ and $\mu_1(j, m)$ respectively. Observe that they are equal to

$$\mu_0(j, m) = \lfloor 2^{j-I(m \text{ is odd})}/3 \rfloor, \quad \mu_1(j, m) = \lfloor 2^{j-I(m \text{ is even})}/3 \rfloor. \quad (37)$$

Proposition 4.1 (Integer support of wave atoms). *An integer frequency $k \in \mathbb{Z}$ is in the support of $\hat{\psi}_m^j$ if and only if*

$$||k| - 2^{j-1}m| \leq \mu_0(j, m) \text{ or } ||k| - 2^{j-1}(m+1)| \leq \mu_1(j, m).$$

Proof. Given that the support is symmetric around zero. Without loss of generality, we consider only the non-negative frequencies. For odd m , the support of $\hat{\psi}_m^j$ is

$$\begin{aligned} & (2^{j-1}(m - \frac{1}{3}), 2^{j-1}(m + 1 + \frac{2}{3})) \\ &= (2^{j-1}(m - \frac{1}{3}), 2^{j-1}(m + \frac{1}{3})) \cup \{2^{j-1}(m + \frac{1}{3})\} \cup (2^{j-1}(m + 1 - \frac{2}{3}), 2^{j-1}(m + 1 + \frac{2}{3})). \end{aligned}$$

Given that $2^{j-1}(m + \frac{1}{3})$ is never an integer, we can ignore this point. Further, as $2^{j-1}m$ and $2^{j-1}(m + 1)$ are integers, the integer support of $\hat{\psi}_m^j$ becomes

$$[2^{j-1}m - \lfloor 2^{j-1}/3 \rfloor, 2^{j-1}m + \lfloor 2^{j-1}/3 \rfloor] \cup [2^{j-1}(m + 1) - \lfloor 2^j/3 \rfloor, 2^{j-1}(m + 1) + \lfloor 2^j/3 \rfloor].$$

Given that for odd m , $\mu_0(j, m) = \lfloor 2^{j-1}/3 \rfloor$ and $\mu_1(j, m) = \lfloor 2^j/3 \rfloor$, this concludes the proof for the case of odd m . Similarly for even m . $\square$

One can notice that the intersection of the intervals consists of one point $\chi_{2^{j-1}(m+1)} \cap \chi_{2^{j-1}m} = \{2^{j-1}m + 2^{j-1}(m \text{ is odd})/3\}$ and this value is always fractional, so there is no $k \in \mathbb{Z}$ such that

$$||k| - 2^{j-1}m| \leq \mu_0(j, m) \text{ and } ||k| - 2^{j-1}(m + 1)| \leq \mu_1(j, m).$$

4.2 Wave atom admissible trees

We would like to construct a (discrete) wave atom transform based on an admissible wave packet tree T . Here, we state some conditions on T that are sufficient to ensure that the resulting wave atom basis is orthogonal.

We would like every integer $k \in \{-N/2, \dots, N/2 - 1\}$ to lie in the support of two wave atoms, unless k is close to zero (which is covered by a single wave atom) or close to $\pm N$ (which is treated as a special case). As such when we consider two neighboring leaf nodes $W_m^j, W_{m'}^{j'} \in \Lambda_T$, where W_m^j is the left neighbor, we require that the intervals centered around $\pm(m + 1)2^j$ related to the two nodes are of equal length,

$$\mu_1(j, m) = \mu_0(j', m'). \quad (38)$$

We define the subset of all wave packet admissible trees that satisfy the condition in equation (38).

Definition 4.1. Consider an admissible wave packet tree with a root W_0^L and K leaf nodes $\{W_{m_i}^{j_i} \mid i = 1, \dots, K\}$ traversed from left to right. If the following conditions hold for all $i \in \{1, \dots, K - 1\}$:

1. $|j_i - j_{i+1}| \leq 1$,
2. if $j_{i+1} = j_i + 1$, then both m_i and m_{i+1} are odd,
3. if $j_{i+1} = j_i - 1$, then both m_i and m_{i+1} are even,

we call such a tree a wave atom admissible tree.

It is straightforward to see that other admissible wave packet trees do not satisfy the condition (38) for some pair of neighboring leaf nodes of the tree. Given the Definition 4.1 and its reliance on the condition (38), we state an evident property of the wave atom admissible trees.

Proposition 4.2. Consider a wave atom admissible tree T with a root node W_0^L . Denote by $W_0^{j_l}$ and $W_{m_r}^{j_r}$ the leftmost and rightmost leaf nodes respectively. For any $k \in \{-N/2, \dots, N/2 - 1\}$, where $N = 2^L$, the following holds:

1. If $|k| \leq \mu_0(j_l, 0)$, then k is in the support of only $\hat{\psi}_0^{j_l}$;
2. If $|k| > m_r 2^{j_r-1} + \mu_0(j_r, m_r)$, then k is in the support of only $\hat{\psi}_{m_r}^{j_r}$;
3. Otherwise, k is in the support of exactly two wave atoms.

Proof. Follows directly from condition (38), observation that $\mu_0(j, m) + \mu_1(j, m) = 2^{j-1} - 1$ and $\chi_{\pm 2^{j-1}(m+1)} \cap \chi_{\pm 2^{j-1}m}$ has no integer points. $\square$

In addition, we would like the discrete wave atom transform to be a unitary operator. This requires special treatment of the lowest and highest frequencies, corresponding to the left-most and right-most wave packets in the tree T . So we define a modified basis $\{\hat{\psi}_m^j\}$ as follows: we straighten the right-most wave packet corresponding to $W_{m_r}^{j_r} \in \Lambda_T$ as its support extends beyond the interval of interest

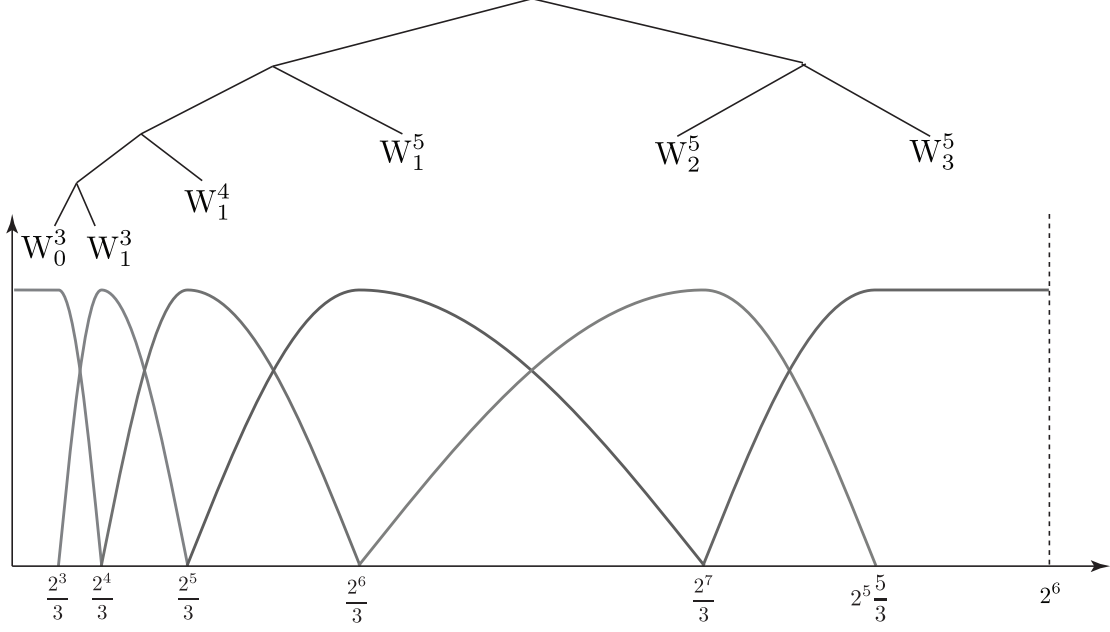


Fig. 12 Wave atoms $\tilde{\psi}_m^j$ corresponding to the wave atom admissible tree on the top (absolute values). The figure shows only the positive side of the frequency domain. The negative side of the frequency domain is symmetric.

$\{-N/2, \dots, N/2 - 1\}$. Moreover, we also modify the wave packet corresponding to the leftmost leaf node $W_0^j \in \Lambda_T$ as its support approaches the origin, as follows:

$$\tilde{\psi}_m^j(k) = \begin{cases} 2^{-j/2} e^{-i\pi 2^{-j} k} e^{(-1)^{k < 0} i \alpha_0} & \text{if } |k| \leq \mu_0(j_l, 0), W_m^j = W_0^j, \\ 2^{-j/2} e^{-i\pi 2^{-j} k} e^{(-1)^{k < 0} i \alpha_{m_r}} & \text{if } |k| > m_r 2^{j_r} + \mu_0(j_r, m_r), W_m^j = W_{m_r}^j, \\ \tilde{\psi}_m^j(k) & \text{otherwise.} \end{cases} \quad (39)$$

Similarly to equation (9), we then define wave atoms with the shift of $2^{-j}n$ in the spatial domain to be

$$\tilde{\psi}_{m,n}^j(k) = e^{-i2\pi n 2^{-j} k} \tilde{\psi}_m^j(k). \quad (40)$$

An example of this construction is shown on Figure 12. Now, we can formally define the quantum wave atom transform.

Definition 4.2. Consider a wave atom admissible tree T with a root node W_0^L . Define a matrix C^A as

$$C_{m2^j+n,i}^A = \overline{\tilde{\psi}_{m,n}^j(d(i))}, \quad (41)$$

where $W_m^j \in \Lambda_T$, d is the decoding function in equation (22), and $n, i \in \mathbb{Z}$ such that $0 \leq n < 2^j$ and $0 \leq i < N$. The following transform of size $N = 2^L$, where Q_L is QFT,

$$\sum_{k=0}^{N-1} f[k] |k\rangle \mapsto \sum_{W_m^j \in \Lambda_T} \sum_{n=0}^{2^j-1} \left(\sum_{i=0}^{N-1} C_{m2^j+n,i}^A \hat{f}[d(i)] \right) |m2^j + n\rangle \quad (42)$$

$$\text{where } \hat{f}[k] = \begin{cases} \sum_{i=0}^{N-1} (Q_L^\dagger)_{ki} f[i], & \text{if } k \geq 0, \\ \sum_{i=0}^{N-1} (Q_L^\dagger)_{(k+N)i} f[i], & \text{if } k < 0, \end{cases}$$

is called a quantum wave atom transform.

4.3 Decomposition of C^A

Recall that the Shannon wavelet transform was implemented using a Fourier-like transform that had a block-diagonal structure determined by the wave packet tree T (see Section 3). To implement the wave

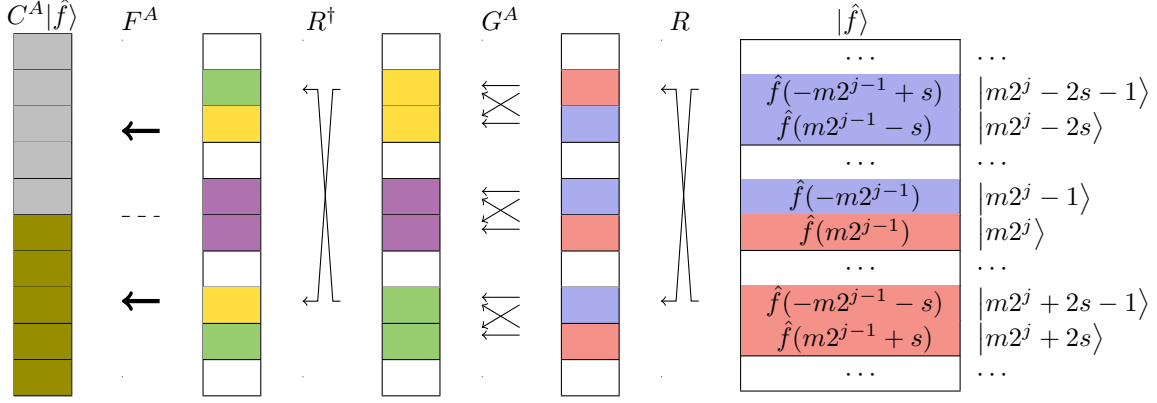


Fig. 13 The implementation of the operation C^A , defined in Eq. (41). This represents a wave atom transform, omitting the initial step of applying a discrete Fourier transform (DFT). In other words, one starts from the state $|\hat{f}\rangle = \sum_{k=-N/2}^{N/2-1} \hat{f}[k] |e(k)\rangle = \sum_{k=0}^{N-1} \hat{f}[d(k)] |k\rangle$ from Eq. (25). Each box in the diagram represents one component of the state-vector, when the quantum state is written as a superposition of standard basis vectors (denoted by $|\dots\rangle$ symbols on the right side of the diagram). First the permutation ρ from Eq. (46) is applied, which rearranges the components of the state-vector. This is followed by the “blending matrix” G^A from Eq. (48), which forms linear combinations of pairs of components of the state-vector, with coefficients given by $g_+[j, m, s]$ and $g_-[j, m, s]$ in Eq. (47). To illustrate this, the diagram shows the intervals $\chi_{\pm m2^j}$ (see Eq. (36)) that represent the intersection of supports of two wave atoms: $W_{m'}^j$ on the left side and W_m^j on the right side. Finally the inverse permutation ρ^{-1} is applied, followed by the operation F^A , which implements a Fourier transform adapted to the tree structure T , defined in Eq. (43). On the parts of the state-vector that are shown here, the action of F^A consists of two blocks: $F^A(j, m')$ (acting on the top half of the vector) and $F^A(j, m)$ (acting on the bottom half of the vector). This produces the desired state $C^A|\hat{f}\rangle$.

atom transform, one can use the same idea, combined with a new technique: redistribution of weights between elements of the vector $\hat{f}$, sometimes called “blending” [15]. This is needed because of the overlaps between the supports of neighboring wave atoms over the frequency domain (which did not occur in the case of Shannon wavelets).

In this section, we will decompose the matrix C^A (from Definition 4.2) into a product of a blending operation, and a Fourier-like transform.

Classically, for a given node W_m^j in the tree T , the wave atom transform blends 2^j pairs of elements of the vector $\hat{f}$ into a vector of size 2^j and applies the Fourier transform of size 2^j to obtain the vector of wave atom coefficients $\{c_{j,m,n}^D\}_{n=0}^{2^j-1}$ [4]. Unless we consider border cases of W_m^j , the transform blends $\hat{f}[k]$ and $\hat{f}[k']$, for all pairs (k, k') of the form:

$$\begin{aligned} &(-m2^{j-1} + s, m2^{j-1} + s), \quad \forall |s| \leq \mu_0(j, m), \\ &(-(m+1)2^{j-1} + s, (m+1)2^{j-1} + s), \quad \forall |s| \leq \mu_1(j, m). \end{aligned}$$

(This can be seen by recalling Definition 4.2, equations (39)-(40), and the fact that $\hat{\psi}_m^j$ is supported on two intervals in the frequency domain, as in equation (34).)

In order to implement the blending efficiently, we permute elements of the vector $\hat{f}$ such that the pairs involved in the blending become adjacent to each other. Given that the encoding procedure (21) orders frequencies by increasing absolute values, for some W_m^j and $|s| \leq \mu_0(j, m)$, the pair of frequencies $-m2^{j-1} + s$ and $m2^{j-1} + s$ are not neighboring indices when encoded unless $s = 0$. Therefore, the blending procedure becomes a three-stage process.

1. Permute elements of the vector $\hat{f}$ such that $-m2^{j-1} + s$ and $m2^{j-1} + s$ become neighbors by swapping $-m2^{j-1} + s$ with $-m2^{j-1} - s$.
2. Redistribute the weights between adjacent elements according to the function g .
3. Reverse the permutation in Step 1.

Figure 13 schematically illustrates the decomposition of C^A by tracking the actions on pairs $\pm m2^{j-1} \pm s$. In the following, we define the building blocks that are used to construct C^A .

Definition 4.3. For each $W_m^j \in \Lambda_T$, consider matrix $F^A(j, m) \in \mathbb{C}^{2^j \times 2^j}$ defined by elements

$$(F^A(j, m))_{n,c} = 2^{-j/2} e^{i2\pi 2^{-j}(n+\frac{1}{2})d(m2^j+c)} e^{(-1)^{c+1}i\alpha_m}, \text{ for } 0 \leq n, c < 2^j. \quad (43)$$

Let us define matrix $F^A \in \mathbb{C}^{N \times N}$ of Fourier transform adapted to tree structure of T as a composition of matrices $F^A(j, m)$,

$$F^A = \bigoplus_{W_m^j \in \Lambda_T} F^A(j, m), \quad (44)$$

that is to say,

$$F_{m2^j+n, m2^j+c}^A = (F^A(j, m))_{n,c}, \text{ for } W_m^j \in \Lambda_T, 0 \leq n, c < 2^j. \quad (45)$$

Definition 4.4. Let us define permutation of indices $\rho : \{0, \dots, N-1\} \mapsto \{0, \dots, N-1\}$ by its non-trivial actions as

$$\forall W_m^j \in \Lambda_T \text{ s.t. } W_m^j \neq W_0^j, \forall |s| \leq \mu_0(j, m), \rho(m2^j + 2s - 1) = m2^j - 2s - 1. \quad (46)$$

Define its permutation matrix $R \in \mathbb{C}^{N \times N}$ such that nonzero elements of the matrix are

$$R_{\rho(i)i} = 1 \text{ for } i \in \{0, \dots, N-1\}.$$

Let us use the following shorthand for function g . For $j \geq 1, m \geq 0$ and $s \in \mathbb{Z}$,

$$\begin{aligned} g_-[j, m, s] &= g((-1)^m(2^{-j+1}\pi|s| - \frac{\pi}{2})), \\ g_+[j, m, s] &= i(-1)^{I(s<0)}g((-1)^{m+1}(2^{-j+1}\pi|s| + \frac{\pi}{2})). \end{aligned} \quad (47)$$

Definition 4.5. Let us define blending matrix $G^A \in \mathbb{C}^{N \times N}$ as follows: for any $W_m^j \in \Lambda_T$ such that $W_m^j \neq W_0^j$, and for any $s \in \mathbb{Z}$ such that $|s| \leq \mu_0(j, m)$,

$$\begin{pmatrix} G_{m2^j+2s-1, m2^j+2s-1}^A & G_{m2^j+2s-1, m2^j+2s}^A \\ G_{m2^j+2s, m2^j+2s-1}^A & G_{m2^j+2s, m2^j+2s}^A \end{pmatrix} = \begin{pmatrix} g_-[j, m, s] & g_+[j, m, s] \\ g_+[j, m, s] & g_-[j, m, s] \end{pmatrix}, \quad (48)$$

and G^A is the identity on other diagonal blocks.

Theorem 4.3. The matrix C^A (from Definition 4.2) is a product of matrices given by Definitions 4.3, 4.4, 4.5,

$$C^A = F^A R^\dagger G^A R. \quad (49)$$

Before proving Theorem 4.3, we note that it implies the following:

Corollary 4.4. The matrix C^A is unitary.

Proof. This follows from the observation that each matrix in the product of equation (49) is unitary. The blocks of G^A are unitary due to properties (11) and (12) of the function g . Note that G^A will still be unitary, even if g is only computed approximately (not exactly), when g is computed using the methods in Section 2.2.1. $\square$

We now prove Theorem 4.3. First, we state two identities due to periodicity of the complex exponential function and the definition of $\alpha_m = \frac{\pi}{2}(m + \frac{1}{2})$:

$$e^{i2\pi 2^{-j}(n+\frac{1}{2})(-m2^{j-1}+s)} e^{i\alpha_m} = i e^{i2\pi 2^{-j}(n+\frac{1}{2})(m2^{j-1}+s)} e^{-i\alpha_m} \quad (50)$$

$$e^{i2\pi 2^{-j}(n+\frac{1}{2})((m+1)2^{j-1}+s)} e^{-i\alpha_m} = i e^{i2\pi 2^{-j}(n+\frac{1}{2})(-(m+1)2^{j-1}+s)} e^{i\alpha_m} \quad (51)$$

Also we state a simple identity:

$$g((-1)^m(2^{-j+1}\pi(-s) - \frac{\pi}{2})) = g((-1)^{m+1}(2^{-j+1}\pi s + \frac{\pi}{2})) \quad (52)$$

Our goal is to implement the operation described in Definition 4.2:

$$\sum_{i=0}^{N-1} C_{m2^j+n,i}^A \hat{f}[d(i)] = \sum_{i=0}^{N-1} \overline{\hat{\psi}_{m,n}^j(d(i))} \hat{f}[d(i)]. \quad (53)$$

Recall that $\hat{\psi}_m^j$ is related to $\hat{\psi}_m^j$ via equations (39)-(40), and $\hat{\psi}_m^j$ is supported on two intervals in the frequency domain, described by equation (34) and proposition 4.1. This motivates us to state and prove the following two lemmas.

Lemma 4.1. *For any $W_m^j \neq W_0^j$,*

$$\begin{aligned} \sum_{|s| \leq \mu_0(j,m)} \left(\overline{\hat{\psi}_{m,n}^j(m2^{j-1} + s)} \hat{f}[m2^{j-1} + s] + \overline{\hat{\psi}_{m,n}^j(-m2^{j-1} + s)} \hat{f}[-m2^{j-1} + s] \right) \\ = \sum_{\substack{i_k = m2^j \\ i_k \leq m2^j}}^{m2^j + 2\mu_0(j,m)} F_{m2^j+n,i_k}^A \sum_{c=0}^{N-1} G_{\rho(i_k),c}^A \hat{f}[d(\rho(c))]. \end{aligned} \quad (54)$$

Proof. For $s \geq 0$ we use identity (50) for points on negative side.

$$\begin{aligned} \overline{\hat{\psi}_{m,n}^j(m2^{j-1} + s)} &= 2^{-j/2} e^{i2\pi 2^{-j}(n+\frac{1}{2})(m2^{j-1}+s)} e^{-i\alpha_m} g((-1)^m (2^{-j+1}\pi|s| - \frac{\pi}{2})) \\ &= F_{m2^j+n,m2^j+2s}^A G_{\rho(m2^j+2s),m2^j+2s}^A, \\ \overline{\hat{\psi}_{m,n}^j(-m2^{j-1} + s)} &= 2^{-j/2} e^{i2\pi 2^{-j}(n+\frac{1}{2})(m2^{j-1}+s)} e^{-i\alpha_m} \mathbf{i} g((-1)^{m+1} (2^{-j+1}\pi|s| + \frac{\pi}{2})) \\ &= F_{m2^j+n,m2^j+2s}^A G_{\rho(m2^j+2s),m2^j+2s-1}^A. \end{aligned}$$

For $s < 0$ we use the identity (50) for points on positive side. Moreover, we use the identity (52) for points on both sides.

$$\begin{aligned} \overline{\hat{\psi}_{m,n}^j(m2^{j-1} + s)} &= 2^{-j/2} e^{i2\pi 2^{-j}(n+\frac{1}{2})(-m2^{j-1}+s)} e^{i\alpha_m} (-\mathbf{i}) g((-1)^{m+1} (2^{-j+1}\pi|s| + \frac{\pi}{2})) \\ &= F_{m2^j+n,m2^j+2|s|-1}^A G_{\rho(m2^j+2|s|-1),m2^j-2|s|}^A, \\ \overline{\hat{\psi}_{m,n}^j(-m2^{j-1} + s)} &= 2^{-j/2} e^{i2\pi 2^{-j}(n+\frac{1}{2})(-m2^{j-1}+s)} e^{i\alpha_m} g((-1)^m (2^{-j+1}\pi|s| - \frac{\pi}{2})) \\ &= F_{m2^j+n,m2^j+2|s|-1}^A G_{\rho(m2^j+2|s|-1),m2^j-2|s|-1}^A. \end{aligned}$$

Therefore, the left-hand side of equation (54) becomes

$$\begin{aligned} \sum_{|s| \leq \mu_0(j,m)} \left(\overline{\hat{\psi}_{m,n}^j(m2^{j-1} + s)} \hat{f}[m2^{j-1} + s] + \overline{\hat{\psi}_{m,n}^j(-m2^{j-1} + s)} \hat{f}[-m2^{j-1} + s] \right) \\ = \sum_{s=0}^{\mu_0(j,m)} F_{m2^j+n,m2^j+2s}^A \left(G_{\rho(m2^j+2s),m2^j+s}^A \hat{f}[d(\rho(m2^j+2s))] \right. \\ \left. + G_{\rho(m2^j+2s),m2^j+2s-1}^A \hat{f}[d(\rho(m2^j+2s-1))] \right) \\ + \sum_{s=1}^{\mu_0(j,m)} F_{m2^j+n,m2^j+2|s|-1}^A \left(G_{\rho(m2^j+2|s|-1),m2^j-2|s|}^A \hat{f}[d(\rho(m2^j-2|s|))] \right. \\ \left. + G_{\rho(m2^j+2|s|-1),m2^j-2|s|-1}^A \hat{f}[d(\rho(m2^j-2|s|-1))] \right) \\ = \sum_{s=0}^{2\mu_0(j,m)} F_{m2^j+n,m2^j+s}^A \sum_{c=0}^{N-1} G_{\rho(m2^j+s),c}^A \hat{f}[d(\rho(c))]. \end{aligned}$$

The latter step is due to the block diagonal structure of matrix G^A with blocks of size 2×2 . $\square$

Lemma 4.2. For any $W_m^j \neq W_{m_r}^{j_r}$,

$$\begin{aligned}
& \sum_{|s| \leq \mu_1(j, m)} \left(\overline{\hat{\psi}_{m, n}^j((m+1)2^{j-1} + s)} \hat{f}[(m+1)2^{j-1} + s] \right. \\
& \quad \left. + \overline{\hat{\psi}_{m, n}^j(-(m+1)2^{j-1} + s)} \hat{f}[-(m+1)2^{j-1} + s] \right) \\
& = \sum_{\substack{i_k = m2^j + 2\mu_0(j, m) + 1 \\ (m+1)2^j - 1}} F_{m2^j + n, i_k}^A \sum_{c=0}^{N-1} G_{\rho(i_k), c}^A \hat{f}[d(\rho(c))].
\end{aligned} \tag{55}$$

Proof. Before reviewing the sum, we show that for any $W_m^j \neq W_{m_r}^{j_r}$ and $s \in \mathbb{Z}$ such that $|s| \leq \mu_1(j, m)$

$$\begin{aligned}
& \begin{pmatrix} G_{(m+1)2^j + 2s-1, (m+1)2^j + 2s-1}^A & G_{(m+1)2^j + 2s-1, (m+1)2^j + 2s}^A \\ G_{(m+1)2^j + 2s, (m+1)2^j + 2s-1}^A & G_{(m+1)2^j + 2s, (m+1)2^j + 2s}^A \end{pmatrix} \\
& = \begin{pmatrix} g_-[j, m+1, s] & g_+[j, m+1, s] \\ g_+[j, m+1, s] & g_-[j, m+1, s] \end{pmatrix}.
\end{aligned} \tag{56}$$

Denote the right neighbor of W_m^j by $W_{m'}^{j'}$, then

- If $j' = j$, then $m' = m+1$ and (56) is simply the definition the block of matrix G^A ;
- If $j' = j+1$, then both m and m' are odd and by properties of the function g .

$$\begin{aligned}
& g((-1)^{m'}(2^{-j'+1}\pi s - \frac{\pi}{2})) = g(-2^{-j'+1}\pi s + \frac{\pi}{2}) \\
& = g(-\frac{\pi}{2} + 2^{-j+1}\pi s) = g((-1)^{m+1}(2^{-j+1}\pi s - \frac{\pi}{2})),
\end{aligned} \tag{57}$$

and

$$\begin{aligned}
& g((-1)^{m'+1}(2^{-j'+1}\pi s + \frac{\pi}{2})) = g(2^{-j'+1}\pi s + \frac{\pi}{2}) \\
& = g(-\frac{\pi}{2} - 2^{-j+1}\pi s) = g((-1)^m(2^{-j+1}\pi s + \frac{\pi}{2})).
\end{aligned} \tag{58}$$

- For $j' = j-1$, then m and m' are even and the derivation is analogous to case of $j' = j+1$.

For $s \geq 0$ we use the identity (51) for points on the positive side.

$$\begin{aligned}
& \overline{\hat{\psi}_{m, n}^j((m+1)2^{j-1} + s)} \\
& = 2^{-j/2} e^{i2\pi 2^{-j}(n + \frac{1}{2})(-(m+1)2^{j-1} + s)} e^{i\alpha_m} \mathbf{i} g((-1)^m(2^{-j+1}\pi|s| + \frac{\pi}{2})) \\
& = F_{m2^j + n, (m+1)2^j - 2s-1}^A G_{\rho((m+1)2^j - 2s-1), (m+1)2^j + 2s}^A, \\
& \overline{\hat{\psi}_{m, n}^j(-(m+1)2^{j-1} + s)} \\
& = 2^{-j/2} e^{i2\pi 2^{-j}(n + \frac{1}{2})(-(m+1)2^{j-1} + s)} e^{i\alpha_m} g((-1)^{m+1}(2^{-j+1}\pi|s| - \frac{\pi}{2})) \\
& = F_{m2^j + n, (m+1)2^j - 2s-1}^A G_{\rho((m+1)2^j - 2s-1), (m+1)2^j + 2s-1}^A.
\end{aligned}$$

For $s < 0$ we use the identity (51) for points on the negative side. Moreover, we use the identity (52) for points on both sides.

$$\begin{aligned}
& \overline{\hat{\psi}_{m, n}^j((m+1)2^{j-1} + s)} \\
& = 2^{-j/2} e^{i2\pi 2^{-j}(n + \frac{1}{2})((m+1)2^{j-1} + s)} e^{-i\alpha_m} g((-1)^{m+1}(2^{-j+1}\pi|s| - \frac{\pi}{2})) \\
& = F_{m2^j + n, (m+1)2^j - 2|s|}^A G_{\rho((m+1)2^j - 2|s|), (m+1)2^j - 2|s|}^A, \\
& \overline{\hat{\psi}_{m, n}^j(-(m+1)2^{j-1} + s)} \\
& = 2^{-j/2} e^{i2\pi 2^{-j}(n + \frac{1}{2})((m+1)2^{j-1} + s)} e^{-i\alpha_m} (-\mathbf{i}) g((-1)^m(2^{-j+1}\pi|s| + \frac{\pi}{2}))
\end{aligned}$$

$$= F_{m2^j+n, (m+1)2^j-2|s|}^A G_{\rho((m+1)2^j-2|s|), (m+1)2^j-2|s|-1}^A.$$

Therefore, the left-hand side of equation (55) becomes

$$\begin{aligned} & \sum_{|s| \leq \mu_1(j, m)} \left(\overline{\hat{\psi}_{m, n}^j((m+1)2^{j-1} + s)} \hat{f}[(m+1)2^{j-1} + s] \right. \\ & \quad \left. + \overline{\hat{\psi}_{m, n}^j(-(m+1)2^{j-1} + s)} \hat{f}[-(m+1)2^{j-1} + s] \right) \\ &= \sum_{s=0}^{\mu_1(j, m)} F_{m2^j+n, (m+1)2^j-2s-1}^A \times \left(G_{\rho((m+1)2^j-2s-1), (m+1)2^j+2s}^A \hat{f}[d(\rho((m+1)2^j + 2s))] \right. \\ & \quad \left. + G_{\rho((m+1)2^j-2s-1), (m+1)2^j+2s-1}^A \hat{f}[d(\rho((m+1)2^j + 2s-1))] \right) \\ &+ \sum_{s=1}^{\mu_1(j, m)} F_{m2^j+n, (m+1)2^j-2s}^A \left(G_{\rho((m+1)2^j-2s), (m+1)2^j-2s}^A \hat{f}[d(\rho((m+1)2^j - 2s))] \right. \\ & \quad \left. + G_{\rho((m+1)2^j-2s), (m+1)2^j-2s-1}^A \hat{f}[d(\rho((m+1)2^j - 2s-1))] \right) \\ &= \sum_{s=1}^{2\mu_1(j, m)+1} F_{m2^j+n, (m+1)2^j-s}^A \sum_{c=0}^{N-1} G_{\rho((m+1)2^j-s, c)}^A \hat{f}[d(\rho(c))] \end{aligned}$$

The latter step is due to the block diagonal structure of matrix G^A with blocks of size 2×2 . $\square$

Proof (of Theorem 4.3). First, we consider two special cases. Observe that for $W_0^{j_l}$ and $i_k \leq 2\mu_0(j_l, 0)$ and $n \in \{0, \dots, 2^{j_l} - 1\}$ we have

$$C_{n, i_k}^A = F_{n, i_k}^A \text{ and } R_{i_k, i_k} = G_{i_k, i_k}^A = 1.$$

Also for $W_{m_r}^{j_r}$ and $i_k > m_r 2^{j_r} + 2\mu_0(j_r, m_r)$ and $n \in \{0, \dots, 2^{j_r} - 1\}$ we have

$$C_{m_r 2^{j_r} + n, i_k}^A = F_{m_r 2^{j_r} + n, i_k}^A \text{ and } R_{i_k, i_k} = G_{i_k, i_k}^A = 1.$$

The general case can be handled using Lemmas 4.1 and 4.2, and Proposition 4.1. Combining all of the above, we get

$$C^A \sum_c^{N-1} \hat{f}[d(c)] |c\rangle = F^A R^\dagger G^A R \sum_c^{N-1} \hat{f}[d(c)] |c\rangle. \quad (59)$$

Observe that (59) holds for any $\hat{f}$ and conclude the proof. $\square$

In the following subsections, we discuss the implementation of each of the operators, F^A , G^A and R .

4.4 Quantum circuits for F^A

In this section we will implement the operation F^A defined in (43), for a variety of wave packet trees T . This is similar to the implementation of the Shannon wavelet transform in (32), modulo some technical details.

First, equation (43) involves a complex phase factor that has a periodicity of 2^{j+1} (rather than 2^j , in the case of (32)):

$$e^{i2^{-j}2\pi(n+\frac{1}{2})d(m2^j+c)} = e^{i2^{-j}2\pi(n+\frac{1}{2})(d(m2^j+c)+p2^{j+1})}.$$

Recall that in the wave packet encoding (26), the j least significant bits represent c in binary form while the remaining bits represent m . So the $(j+1)$ -th bit represents the parity of m , which determines the shape of the wavepacket (see Section 2.2).

Recall functions d and $\tilde{d}$ defined by (22), (24) correspondingly and observe that

$$d(m2^j + c) = \tilde{d}(c, j, m) + (-1)^c \lfloor \frac{m}{2} \rfloor \cdot 2^j - I(c \text{ is odd}) \cdot 2^j. \quad (60)$$

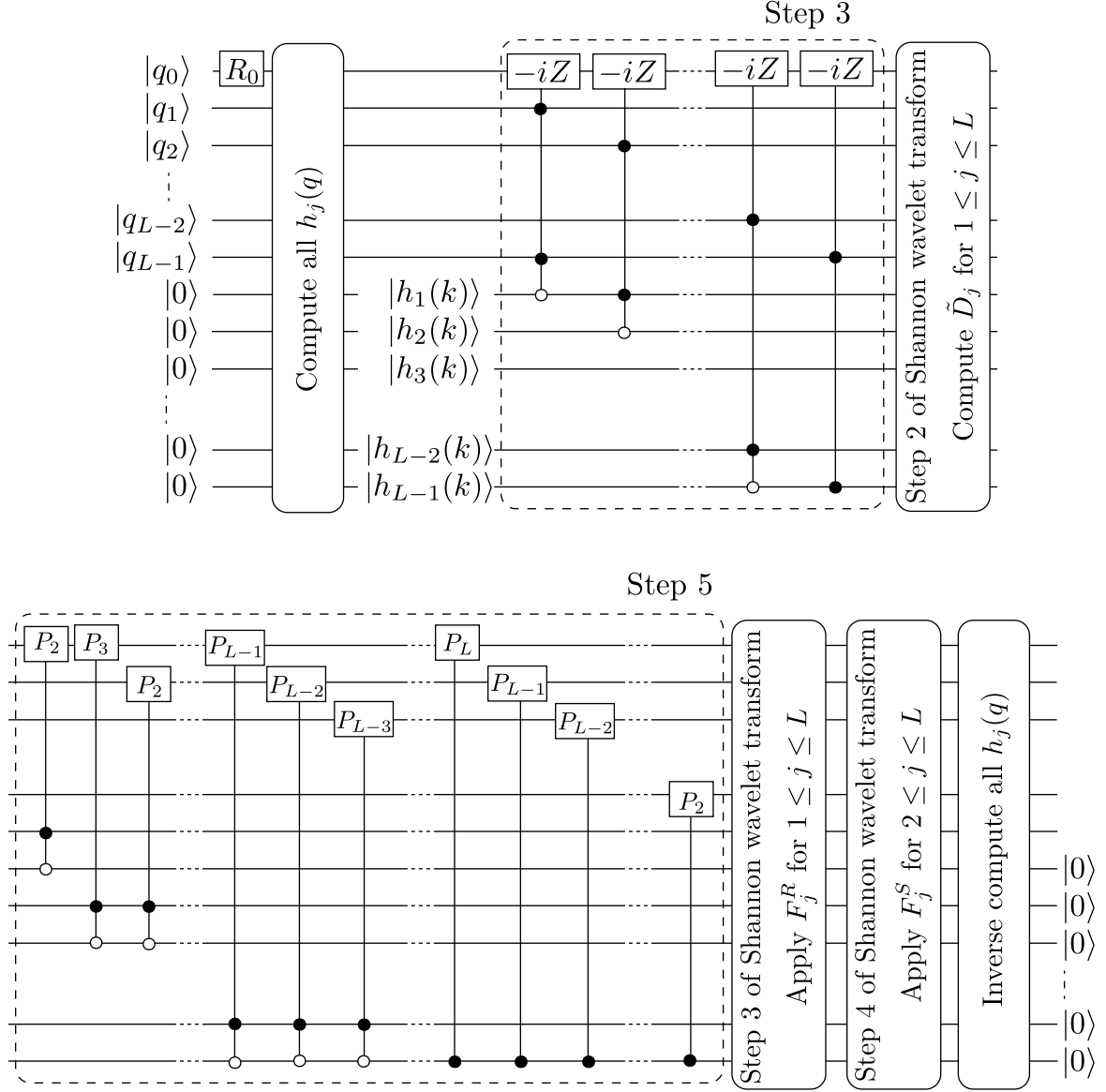


Fig. 14 Quantum circuit implementation of the operation F^A of size 2^L . The boxes labeled Step 3 and Step 5 apply the first two phase factors in Eq. (61). For more details, see the Supplementary Material. The third factor in Eq. (61) is handled using the same techniques as in the Shannon wavelet transform. Steps 2-4 of the Shannon wavelet transform can be found in Figure 11.

Like in the case of Shannon wavelets, the decoding $\tilde{d}$ keeps us within the same block of size 2^j . However, (60) shows that some adjustment is required due to the change in periodicity. Then, each block of the matrix F^A can be expressed as the product of three factors:

$$(F^A(j, m))_{n,c} = \underbrace{e^{(-1)^c i \pi \lfloor \frac{m}{2} \rfloor} e^{-I(c \text{ is odd}) i \pi}}_{r_m} \underbrace{e^{(-1)^{c+1} i \alpha_m}}_{\text{Phase}} \underbrace{e^{i \cdot \pi 2^{-j} \tilde{d}(c,j,m)} 2^{-j/2} e^{i \cdot 2 \pi 2^{-j} n \cdot \tilde{d}(c,j,m)}}_{\text{QFT}}. \quad (61)$$

The first two factors in Eq. (61) are new, compared to the analogous formula (32) for the Shannon wavelet transform.

The first factor of (61) represents the necessary adjustment. Consider combination of values of c and m :

$$r_m(c) = \begin{cases} e^{-i\pi/4} & \text{if } m = c \bmod 2, \\ -e^{i\pi/4} & \text{if } m \neq c \bmod 2. \end{cases} \quad (62)$$

For quantum realization, we would like to have a unitary corresponding different cases of m and c . We may apply the following operators to the qubit corresponding to the least significant bit of c which defines its parity.

$$R_0 = \begin{pmatrix} e^{-i\pi/4} & 0 \\ 0 & -e^{i\pi/4} \end{pmatrix}, \quad R_1 = R_0 \cdot (-iZ). \quad (63)$$

We apply R_0 if m is even and R_1 if m is odd.

The complete procedure is implemented by the quantum circuit shown in Figure 14. Additional details are presented in the Supplemental Material. The gate complexity of the algorithm is $O(L^2)$.

4.5 Quantum circuits for G^A

Recall Definition 4.5, where each block of matrix G^A is defined by equation (48). It will be convenient to shift the indices of this matrix by 1. In other words, let us define the block-diagonal matrix $\tilde{G}^A \in \mathbb{C}^{N \times N}$ as follows: for $W_m^j \neq W_0^{j_l}$ and $|s| \leq \mu_0(j, m)$,

$$\begin{pmatrix} \tilde{G}_{m2^j+2s, m2^j+2s}^A & \tilde{G}_{m2^j+2s, m2^j+2s+1}^A \\ \tilde{G}_{m2^j+2s+1, m2^j+2s}^A & \tilde{G}_{m2^j+2s+1, m2^j+2s+1}^A \end{pmatrix} = \begin{pmatrix} g_-[j, m, s] & g_+[j, m, s] \\ g_+[j, m, s] & g_-[j, m, s] \end{pmatrix}. \quad (64)$$

This satisfies

$$G^A = \text{Adder}_L(1)\tilde{G}^A\text{Adder}_L(-1),$$

where $\text{Adder}_L(\pm 1)$ is the unitary operation defined in equation (5).

Furthermore, the definition of $\tilde{G}^A$ involves both positive and negative shifts s from $m2^j$. The challenge with this approach is that when we work with wave packets in W_m^j , it requires knowing information about the left neighbor of W_m^j in the tree. It would be more practical if we could define the operation $\tilde{G}^A$ in a way that involves only positive shifts (which we denote by n), and only one leaf node W_m^j at a time. This can be done in the following way.

We can re-write the formula (64) for $\tilde{G}^A$ in terms of $n \in \{0, \dots, 2^{j-1} - 1\}$ for $W_m^j \in \Lambda_T$. Then for $W_m^j \neq W_0^{j_l}$ and $0 \leq n \leq \mu_0(j, m)$, we have the same expressions as in (64), but substituting $s \mapsto n$:

$$\begin{aligned} \tilde{g}_-^{(0)}[j, m, n] &= g\left((-1)^m \pi(2^{-j+1}n - \tfrac{1}{2})\right), \\ \tilde{g}_+^{(0)}[j, m, n] &= ig\left((-1)^{m+1} \pi(2^{-j+1}n + \tfrac{1}{2})\right), \\ \begin{pmatrix} \tilde{G}_{m2^j+2n, m2^j+2n}^A & \tilde{G}_{m2^j+2n, m2^j+2n+1}^A \\ \tilde{G}_{m2^j+2n+1, m2^j+2n}^A & \tilde{G}_{m2^j+2n+1, m2^j+2n+1}^A \end{pmatrix} &= \begin{pmatrix} \tilde{g}_-^{(0)}[j, m, n] & \tilde{g}_+^{(0)}[j, m, n] \\ \tilde{g}_+^{(0)}[j, m, n] & \tilde{g}_-^{(0)}[j, m, n] \end{pmatrix}, \end{aligned} \quad (65)$$

and for $W_m^j \neq W_{m_r}^{j_r}$ and $\mu_0(j, m) < n < 2^{j-1}$, we have the same expressions as in (64), but substituting $(j, m) \mapsto (j, m+1)$ and $s \mapsto n - 2^{j-1}$:

$$\begin{aligned} \tilde{g}_-^{(1)}[j, m, n] &= g\left((-1)^m \pi(2^{-j+1}n - \tfrac{1}{2})\right), \\ \tilde{g}_+^{(1)}[j, m, n] &= -ig\left((-1)^{m+1} \pi(2^{-j+1}n - \tfrac{3}{2})\right), \\ \begin{pmatrix} \tilde{G}_{m2^j+2n, m2^j+2n}^A & \tilde{G}_{m2^j+2n, m2^j+2n+1}^A \\ \tilde{G}_{m2^j+2n+1, m2^j+2n}^A & \tilde{G}_{m2^j+2n+1, m2^j+2n+1}^A \end{pmatrix} &= \begin{pmatrix} \tilde{g}_-^{(1)}[j, m, n] & \tilde{g}_+^{(1)}[j, m, n] \\ \tilde{g}_+^{(1)}[j, m, n] & \tilde{g}_-^{(1)}[j, m, n] \end{pmatrix}. \end{aligned} \quad (66)$$

So far, the discussion of wave atoms did not rely on the exact form of function g . Each block of matrix $\tilde{G}^A$ depends on the value of $2^{-j+1}n$, and there are $O(2^{j_m})$ distinct such values, where j_m is the maximum level of leaf nodes in the tree. Therefore there are $O(2^{j_m})$ different types of blocks in $\tilde{G}^A$. So, in the general case, the number of gates in the implementation of $\tilde{G}^A$ without any approximation is $O(2^{j_m})$. However, the complexity can be reduced for certain choices of the function g .

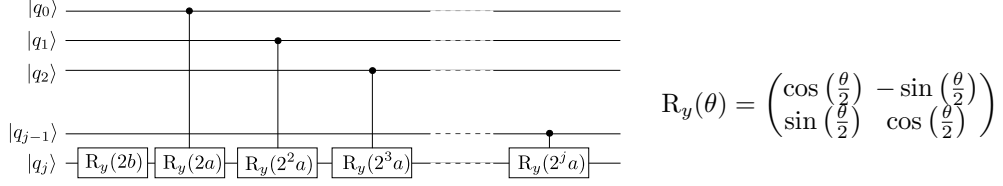


Fig. 15 Quantum circuit that implements a linear Pauli rotation with slope $2a$ and offset $2b$, defined in equation (70).

4.5.1 Efficient quantum circuits for $\tilde{G}^A$

We give an example of a function g for which $\tilde{G}^A$ can be implemented exactly and efficiently. Consider the following function g :

$$g(w) = \begin{cases} \cos(\frac{3}{8}w - \frac{\pi}{16}), & \text{if } -\frac{7}{6}\pi \leq w \leq \frac{\pi}{6}, \\ \cos(\frac{3}{4}w - \frac{\pi}{8}), & \text{if } \frac{\pi}{6} < w \leq \frac{5}{6}\pi, \\ 0, & \text{otherwise.} \end{cases} \quad (67)$$

This is a simple example of a function that resembles an asymmetric “bump” and satisfies the requirements in equation (11) and equation (12). Other types of functions satisfying these requirements also can be considered.

For this choice of g , let

$$v(j, m, n) = \begin{cases} \frac{\pi}{2^{j+2}} (3n - 2^{j-1}) \cdot 2^{I(n \leq \mu_0(j, m))}, & \text{if } m \text{ is odd,} \\ \frac{\pi}{2^{j+2}} (3n - 2^j) \cdot 2^{I(n > \mu_0(j, m))}, & \text{if } m \text{ is even,} \end{cases} \quad (68)$$

then

$$\begin{aligned} \begin{pmatrix} \tilde{G}_{m2^j+2n, m2^j+2n}^A & \tilde{G}_{m2^j+2n, m2^j+2n+1}^A \\ \tilde{G}_{m2^j+2n+1, m2^j+2n}^A & \tilde{G}_{m2^j+2n+1, m2^j+2n+1}^A \end{pmatrix} &= \begin{pmatrix} \cos(v(j, m, n)) & \mathbf{i} \sin(v(j, m, n)) \\ \mathbf{i} \sin(v(j, m, n)) & \cos(v(j, m, n)) \end{pmatrix} \\ &= S^\dagger \underbrace{\begin{pmatrix} \cos(v(j, m, n)) & \sin(v(j, m, n)) \\ -\sin(v(j, m, n)) & \cos(v(j, m, n)) \end{pmatrix}}_{\text{Linear Pauli Rotations}} S, \end{aligned} \quad (69)$$

where S denotes the single-qubit gate $S = \begin{pmatrix} 1 & 0 \\ 0 & \mathbf{i} \end{pmatrix}$.

The middle factor in (69) can be implemented, by using the formulas (68), and applying a controlled single-qubit gate of the form

$$|x\rangle |y\rangle \mapsto |x\rangle \left(\cos(ax + b) |y\rangle + (-1)^y \sin(ax + b) |y \oplus 1\rangle \right), \text{ for } y \in \{0, 1\}, \quad (70)$$

which we call a “linear Pauli rotation” with slope $2a$ and offset $2b$. This can be implemented using the quantum circuit shown in Figure 15. This procedure is described in detail in the Supplementary Material, and has gate complexity of $O(L^2)$.

Finally, to implement $\tilde{G}^A$, we need to call the above procedure controlled by the predicates $I(q > 2\mu_0(j_l, 0))$ and $I(q \leq m_r 2^{j_r} + 2\mu(j_r, m_r))$.

4.6 Quantum circuits for R

Recall that, by Definition 4.4, the action of R is described by a permutation ρ , and ρ permutes odd indices as follows. For any $W_m^j \neq W_0^j$ and $s \in \mathbb{Z}$ such that $|s| \leq \mu_0(j, m)$,

$$\rho(m2^j + 2s - 1) = m2^j - 2s - 1.$$

Let us exclude even indices from consideration. We identify the set of odd indices $\{1, 3, 5, \dots, N-1\}$ with the set of all indices up to $N/2 - 1$, that is $\{0, 1, 2, \dots, N/2 - 1\}$. We let $\hat{\rho}$ denote the permutation

acting on $\{0, 1, 2, \dots, N/2 - 1\}$ that corresponds to ρ . Note that $\dot{\rho}$ is related to ρ as follows:

$$\rho(2k - 1) = 2\dot{\rho}(k) - 1. \quad (71)$$

Given an implementation of $\dot{\rho}$, one can implement the operation R (i.e., the permutation ρ) as follows: given that

$$\rho(2k + 1) = \rho(2(k + 1) - 1) = 2\dot{\rho}(k + 1) - 1,$$

the implementation of R consists of: a shift by 2; the circuit that implements $\dot{\rho}$ on qubits $1, \dots, L - 1$, controlled by the parity of qubit 0; and a shift by -2 . This transforms the initial state of $|2k + 1\rangle$ into $|2(k + 1) + 1\rangle$; then the state after applying $\dot{\rho}$ is $|2\dot{\rho}(k + 1) + 1\rangle$, which we need to transform into $|2\dot{\rho}(k + 1) - 1\rangle$.

4.6.1 Applying the permutation $\dot{\rho}$

It remains for us to show how to apply the permutation $\dot{\rho}$. For a fixed leaf node W_m^j such that $W_m^j \neq W_0^{j_l}$, let us denote the permutation associated with leaf node W_m^j by $\dot{\rho}_m^j$. Then $\dot{\rho}_m^j$ has non-trivial action as

$$\dot{\rho}_m^j(m2^{j-1} + s) = m2^{j-1} - s \text{ for } s \in \mathbb{Z} \text{ s.t. } |s| \leq \mu_0(j, m). \quad (72)$$

Combining all permutations $\dot{\rho}_m^j$ together we get the permutation $\dot{\rho}$:

$$\dot{\rho} = \prod_{W_m^j \neq W_0^{j_l}} \dot{\rho}_m^j. \quad (73)$$

The order in the product is not important since all permutations $\dot{\rho}_m^j$ generated by wave atom admissible tree T pair-wise commute. This follows from the observation that no two such permutations have non-trivial action on the same index. One can notice that equation (71) does not work in case of $k = N/2$ or $k = 0$. However, ρ and $\dot{\rho}$ act trivially on the boundaries, so we do not consider such k 's.

Each $\dot{\rho}_m^j$ acts non-trivially on $2\mu_0(j, m)$ indices. Let us combine permutations that share the same value of $2\mu_0(j, m)$. Suppose this common value is $2\mu_0(j_*, 0)$, for some j_* . Then let $\dot{\rho}^{j_*}$ denote the combined permutation. By the definition of μ_0 in equation (37),

$$\dot{\rho}^{j_*} = \prod_{\substack{W_m^j \neq W_0^{j_l} \\ \mu_0(j, m) = \mu_0(j_*, 0)}} \dot{\rho}_m^j = \prod_{\substack{W_m^j \neq W_0^{j_l} \\ j - I(m \text{ is odd}) = j_*}} \dot{\rho}_m^j. \quad (74)$$

For permutation ρ , let us define a helper function $\tilde{h}^\rho : \{0, \dots, N/2 - 1\} \mapsto \{0, \dots, L - 2\}$ that says, for each index, which permutation $\dot{\rho}^{j_*}$ acts nontrivially on it. Formally, we have:

$$\begin{aligned} \forall W_m^j \neq W_0^{j_l} \text{ and } |s| \leq \mu_0(j, m), \quad \tilde{h}^\rho(m2^{j-1} + s) = j - I(m \text{ is odd}), \\ \text{and} \\ \tilde{h}^\rho(k) = 0 \text{ if } \left(k \leq \mu_0(j_l, 0) \text{ or } k > m_r 2^{j_r-1} + \mu_0(j_r, m_r) \right). \end{aligned} \quad (75)$$

Furthermore, for $j \in \{1, \dots, L - 1\}$, define a family of boolean functions $h_j^\rho : \{0, \dots, N/2 - 1\} \mapsto \{0, 1\}$ such that

$$h_j^\rho(k) = I(\tilde{h}^\rho(k) = j). \quad (76)$$

With such boolean helper functions, the quantum implementation of $\dot{\rho}^{j_*}$ in equation (74) becomes a three-stage process listed in Algorithm 1. One can notice that $\dot{\rho}$ keeps indices within their respective intervals centered around $m2^{j-1}$ for some W_m^j . This means that

$$\tilde{h}^\rho(k) = \tilde{h}^\rho(\dot{\rho}(k)) \text{ for any } k \in \{0, \dots, N/2 - 1\}, \quad (77)$$

which implies that the calculation of h_j^ρ 's can be reversed after $\dot{\rho}$ is applied.

Algorithm 1 Applying permutation $\dot{\rho}^{j*}$, given values of helper function $h_{j*}^\rho(\cdot)$

Input: $\sum_{q=0}^{2^{L-1}-1} \alpha_q |q\rangle |h_{j*}^\rho(q)\rangle$, that is, a superposition over indices $|q\rangle$, with labels $|h_{j*}^\rho(q)\rangle$ computed by the helper function $h_{j*}^\rho(\cdot)$.

Result: $\sum_{q=0}^{2^{L-1}-1} \alpha_q |\dot{\rho}^{j*}(q)\rangle |h_{j*}^\rho(q)\rangle$.

Notation:

Quantum register $\mathcal{R}_q$ of $L-1$ qubits; $\mathcal{R}_q[i]$ denotes the i -th qubit, $i = 0, \dots, L-1$;

Quantum register $\mathcal{R}_h$ of 1 qubit.

1. Apply $\text{Adder}_{L-1}(\mu_0(j*, 0))$ to $\mathcal{R}_q$ controlled on $\mathcal{R}_h = |1\rangle$.
 2. For each $i = 0, \dots, j_* - 2$ apply X -gate to $\mathcal{R}_q[i]$ controlled on $\mathcal{R}_h = |1\rangle$.
 3. Apply $\text{Adder}_{L-1}(-\mu_1(j*, 0))$ to $\mathcal{R}_q$ controlled on $\mathcal{R}_h = |1\rangle$.
-

Finally, we show a way to compute the helper functions $h_j^\rho(\cdot)$, using the helper functions $h_j(\cdot)$ defined in (28). We start by examining a leaf node $W_m^j \notin \{W_0^j, W_{m_r}^j\}$. Consider an index $m2^{j-1} + n$ where $n \in \{0, \dots, N/2 - 1\}$. There are two cases:

- If $n \leq \mu_0(j, m)$, then $\tilde{h}^\rho(m2^{j-1} + n) = j - I(m \text{ is odd})$ as $m2^{j-1} + n$ is affected by the permutation $\dot{\rho}_m^j$.
- If $n > \mu_0(j, m)$, then $\tilde{h}^\rho(m2^{j-1} + n) = j - I(m \text{ is even})$ as $m2^{j-1} + n$ is affected by the permutation $\dot{\rho}_{m'}^{j'}$ where $W_{m'}^{j'}$ is the right neighbor of W_m^j .

For $W_m^j \in \{W_0^j, W_{m_r}^j\}$, in one of the above cases $\tilde{h}^\rho(m2^{j-1} + n) = 0$ according to the definition in equation (75). Finally, one can evaluate these expressions, by obtaining the value of j from the functions $h_j(\cdot)$. The above procedure is described in detail in the Supplementary Material. Its gate complexity is $O(L^2)$.

Altogether, the complexity of implementation of the permutation $\dot{\rho}$ is $O(L^2)$, given that we need to call the above procedure two times to compute and uncompute h_j^ρ 's, and call $O(L)$ times Algorithm 1 (we care about $2 \leq j < L$).

4.7 Computational Complexity

The classical complexity of the 1-D wave atom transform is the same as the complexity of the fast Fourier transform $O(N \log N)$ where N is the number of points in discretization [4].

The quantum implementation of the wave atom transform that combines all the steps in decomposition

$$C^A = F^A R^\dagger G^A R,$$

depends on

1. Efficient computation of the family of boolean functions $h_j(\cdot)$, $j = \{1, \dots, L-1\}$ for tree T ,
2. Efficient computation of blocks of G^A .

Here, efficient computation means polynomial time in the number of qubits, in other words, polylogarithmic in N . The results are summarized in the following theorem.

Theorem 4.5 (Quantum complexity of 1D wave atom transform). *Take a wave atom admissible tree T defined for discretization of size $O(2^L)$. If*

- *function g of the wave atom transform allows the implementation of matrix G^A with gate complexity of $O(L^2)$,*
- *the family of functions $h_j(\cdot)$ has a quantum implementation that requires $O(L^2)$ gates,*

then the gate complexity of quantum wave atom transform generated by tree T and function g is $O(L^2)$.

In Section 4.5, we provided a concrete implementation of the matrix G^A for a specific function g . In addition, recall from Section 2.6 that the family of functions h_j can be efficiently implemented for monotonic trees. This implies the following Corollary.

Corollary 4.6 (Quantum complexity of 1D wave atom transform for monotonic trees). *Let T be a monotonic wave atom admissible tree defined for discretization of size $O(2^L)$. If T is represented by the*

leftmost node on each level, then the gate complexity of the quantum wave atom transform generated by the tree T and the function g defined in (67) is $O(L^2)$. The complexity of the construction of the quantum circuit is $O(L^2)$.

Finally, note that the above result applies to parabolic scaling trees as a special case, as shown in Section 2.3.

5 Discussion and Conclusion

The first result of this paper is a quantum implementation of Shannon wavelet transform with a hierarchical tree structure of shifts and scales (Theorem 3.1). This construction made use of helper decoding functions $h_j(\cdot)$ (equation (28)) that map indices to their respective levels in the wave packet tree. With such functions, the hierarchical quantum Shannon wavelet transform becomes a product of Fourier transforms of different scales. The type of wave packet trees dictates the circuit implementation of functions h_j . We demonstrated that for monotonic trees, the implementation of helper decoding functions is efficient and requires $O(L^2)$ gates, when implementing a wavelet transform of dimension 2^L . Together, this allows us to achieve exponential speedup with the quantum gate complexity of $O(L^2)$ for the entire procedure of Shannon wavelet transform.

The wave atom transform with a hierarchical tree structure is more intricate as it requires “blending,” similar to [15], due to the overlaps between the supports of neighboring wave packets in frequency space [4, 5]. A crucial contribution of this paper is to show how to implement blending that *also follows the tree structure*, on a quantum computer, i.e., acting on quantum superposition states. Using this technique, we showed that for monotonic admissible trees with wave atoms satisfying some technical conditions, the quantum wave atom transform also achieves an exponential speedup with the gate complexity of $O(L^2)$ (Theorem 4.5). We demonstrated this exponential speedup for wave atoms constructed from a particular function $g(\cdot)$ (Corollary 4.6). In particular, this family of wave atoms has “parabolic scaling,” which is useful for constructing sparse representations of solutions to wave equations [4].

Generalizing this construction is an interesting topic for future work. In particular, can this construction be carried out efficiently for different choices of the function $g(\cdot)$ (besides the choice described in Section 4.5.1), perhaps by adapting the techniques introduced recently by [14]?

Using tree-structured wavelet bases, one has additional flexibility to construct wavelet bases with desirable properties. This suggests considering other wave packet trees that satisfy our admissibility conditions (Definition 4.1). However, one is unlikely to find a single “universal” quantum algorithm that is efficient and applicable to all admissible trees. The general tree representation requires at least $O(2^L)$ bits as there are more than $2^{2^{L-1}}$ different admissible wave packet trees [19]. If there is a classical step, the decoding of the tree representation is already $\Omega(2^L)$.

Another potential future direction involves higher-dimensional wave packet transforms, which could help solve wave equations and systems of hyperbolic differential equations in general, as its basis offers a sparse representation of the solution operator [3, 6, 16, 17]. This paper shows a 1-dimensional wave atom transform, which can be extended to construct a multi-dimensional transform. For example, taking products of 1-D wavelet packets allows for constructing two-dimensional orthonormal basis functions with four bumps in the frequency plane. Although the construction is not a straightforward tensor product since both dimensions share the same scale [3], the results in 1-D are easily extendable to the 2-D case.

Acknowledgements: We thank the anonymous referees for several suggestions that improved the presentation of our results. MP is grateful to Radu Balan for the discussion related to this paper, which restored belief in the best.

This work is supported by a collaboration between the US DOE and other Agencies. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Accelerated Research in Quantum Computing, Fundamental Algorithmic Research toward Quantum Utility (FAR-Qu). Additional support is acknowledged from NSF and NIST.

MP also acknowledges funding and support from a NIST/UMD Joint Center for Quantum Information and Computer Science (QuICS) Lanczos Graduate Fellowship, and a MathQuantum Graduate Fellowship (NSF award DMS-2231533).

Contributions by NIST employees are not subject to US copyright protection.

References

- [1] Jozsa, R.: Quantum algorithms and the fourier transform. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* **454**(1969), 323–337 (1998) <https://doi.org/10.1098/rspa.1998.0163>
- [2] Childs, A.M., Van Dam, W.: Quantum algorithms for algebraic problems. *Reviews of Modern Physics* **82**(1), 1–52 (2010)
- [3] Demanet, L.: Curvelets, wave atoms, and wave equations. PhD thesis (2006). <https://doi.org/10.7907/1TEF-RQ51> . <https://resolver.caltech.edu/CaltechETD:etd-05262006-133555>
- [4] Demanet, L., Ying, L.: Wave atoms and sparsity of oscillatory patterns. *Applied and Computational Harmonic Analysis* **23**(3), 368–387 (2007) <https://doi.org/10.1016/j.acha.2007.03.003>
- [5] Villemoes, L.F.: Wavelet packets with uniform time-frequency localization. *Comptes Rendus Mathematique* **335**, 793–796 (2002)
- [6] Demanet, L., Ying, L.: Wave atoms and time upscaling of wave equations. *Numer. Math. (Heidelb.)* **113**(1), 1–71 (2009)
- [7] Demanet, L., Létourneau, P.-D., Boumal, N., Calandra, H., Chiu, J., Snelson, S.: Matrix probing: A randomized preconditioner for the wave-equation hessian. *Applied and Computational Harmonic Analysis* **32**, 155–168 (2012) <https://doi.org/10.1016/j.acha.2011.03.006>
- [8] Hoyer, P.: Efficient Quantum Transforms (1997). <https://arxiv.org/abs/quant-ph/9702028>
- [9] Fijany, A., Williams, C.P.: Quantum Wavelet Transforms: Fast Algorithms and Complete Circuits (1998). <https://arxiv.org/abs/quant-ph/9809004>
- [10] Klappenhecker, A.: Wavelets and wavelet packets on quantum computers. In: Unser, M.A., Aldroubi, A., Laine, A.F. (eds.) *Wavelet Applications in Signal and Image Processing VII*. SPIE, Bellingham, WA (1999)
- [11] Argüello, F.: Quantum wavelet transforms of any order. *Quantum Information & Computation* **9**, 414–422 (2009)
- [12] Li, H.-S., Fan, P., Xia, H.-Y., Song, S.: Quantum multi-level wavelet transforms. *Inf. Sci. (Ny)* **504**, 113–135 (2019)
- [13] Zhang, Z., Kon, M.A.: Wavelet matrix operations and quantum transforms. *Appl. Math. Comput.* **428**(127179), 127179 (2022)
- [14] Bagherimehrab, M., Aspuru-Guzik, A.: Efficient quantum algorithm for all quantum wavelet transforms. *Quantum Science and Technology* **9**(3), 035010 (2024) <https://doi.org/10.1088/2058-9565/ad3d7f>
- [15] Ni, H., Ying, L.: Quantum wave packet transforms with compact frequency support: Implementations for wavelets and gabor atoms. *Applied and Computational Harmonic Analysis*, 101850 (2025)
- [16] Candes, E.J., Demanet, L.: The curvelet representation of wave propagators is optimally sparse. *Communications on Pure and Applied Mathematics* **58**(11), 1472–1528 (2005)
- [17] Candès, E., Demanet, L.: Curvelets and fourier integral operators. *C. R. Math. Acad. Sci. Paris* **336**(5), 395–398 (2003)
- [18] Daubechies, I.: *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, USA

(1992)

- [19] Mallat, S.: A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way, 3rd edn. Academic Press, Inc., USA (2008)
- [20] Daubechies, I.: Orthonormal bases of compactly supported wavelets II. variations on a theme. SIAM J. Math. Anal. **24**(2), 499–519 (1993)
- [21] Ryan, H.: Ricker, ormsby, klander, butterworth – a choice of wavelets. CSEG Recorder **19**(7), 8–9 (1994)
- [22] Gabor, D.: Theory of communication. part 1: The analysis of information. Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering **93**(26), 429–441 (1946) <https://doi.org/10.1049/ji-3-2.1946.0074>
- [23] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, Cambridge, UK (2010)
- [24] Costa, P.C., Jordan, S., Ostrander, A.: Quantum algorithm for simulating the wave equation. Physical Review A **99**(1), 012323 (2019)
- [25] Bagherimehrab, M., Nakaji, K., Wiebe, N., Aspuru-Guzik, A.: Fast quantum algorithm for differential equations (2023). <https://arxiv.org/abs/2306.11802>
- [26] Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Physical Review Letters **103**(15) (2009) <https://doi.org/10.1103/physrevlett.103.150502>
- [27] Morales, M.E., Pira, L., Schleich, P., Koor, K., Costa, P., An, D., Aspuru-Guzik, A., Lin, L., Rebentrost, P., Berry, D.W.: Quantum linear system solvers: A survey of algorithms and applications. arXiv preprint arXiv:2411.02522 (2024)
- [28] Berry, D.W.: High-order quantum algorithm for solving linear differential equations. Journal of Physics A: Mathematical and Theoretical **47**(10), 105301 (2014)
- [29] Childs, A.M., Maslov, D., Nam, Y., Ross, N.J., Su, Y.: Toward the first quantum simulation with quantum speedup. Proceedings of the National Academy of Sciences **115**(38), 9456–9461 (2018)
- [30] Jennings, D., Lostaglio, M., Lowrie, R.B., Pallister, S., Sornborger, A.T.: The cost of solving linear differential equations on a quantum computer: fast-forwarding to explicit resource counts. Quantum **8**, 1553 (2024)
- [31] Clader, B.D., Jacobs, B.C., Sprouse, C.R.: Preconditioned quantum linear system algorithm. Physical review letters **110**(25), 250504 (2013)
- [32] Berry, D.W., Childs, A.M., Cleve, R., Kothari, R., Somma, R.D.: Exponential improvement in precision for simulating sparse hamiltonians. In: Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing, pp. 283–292 (2014)
- [33] Kiani, B.T., De Palma, G., Englund, D., Kaminsky, W., Marvian, M., Lloyd, S.: Quantum advantage for differential equation analysis. Physical Review A **105**(2) (2022) <https://doi.org/10.1103/physreva.105.022415>
- [34] Liu, Y.-K.: Quantum algorithms using the curvelet transform. In: Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, pp. 391–400 (2009)
- [35] Liu, Y.-K.: An Uncertainty Principle for the Curvelet Transform, and the Infeasibility of Quantum Algorithms for Finding Short Lattice Vectors (2023). <https://arxiv.org/abs/2310.03735>

- [36] Harrow, A.W., Recht, B., Chuang, I.L.: Efficient discrete approximations of quantum gates. *Journal of Mathematical Physics* **43**(9), 4445–4451 (2002) <https://doi.org/10.1063/1.1495899>
- [37] Beals, R., Brierley, S., Gray, O., Harrow, A.W., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Efficient distributed quantum computing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **469**(2153), 20120686 (2013) <https://doi.org/10.1098/rspa.2012.0686>
- [38] Vedral, V., Barenco, A., Ekert, A.: Quantum networks for elementary arithmetic operations. *Physical Review A* **54**(1), 147–153 (1996) <https://doi.org/10.1103/physreva.54.147>
- [39] Cuccaro, S.A., Draper, T.G., Kutin, S.A., Moulton, D.P.: A new quantum ripple-carry addition circuit (2004). <https://arxiv.org/abs/quant-ph/0410184>
- [40] Draper, T.G.: Addition on a Quantum Computer (2000). <https://arxiv.org/abs/quant-ph/0008033>
- [41] Coppersmith, D.: An approximate Fourier transform useful in quantum factoring (2002). <https://arxiv.org/abs/quant-ph/0201067>
- [42] Coifman, R.R., Meyer, Y., Wickerhauser, V.: Wavelet analysis and signal processing. In: Auslander, L., Kailath, T., Mitter, S.K. (eds.) *Signal Processing, Part I: Signal Processing Theory*, pp. 59–68. Springer, New York, NY (1990). <http://citeseer.nj.nec.com/coifman92wavelet.html>