Quantum Information Processing (2026) 25:41
https://doi.org/10.1007/s11128-026-05067-7

A review on quantum Fourier transform

Tomás Barros[1] · Pablo Álvarez[1] · Bárbara Vidal[1,3] · Mauricio Solar[2]

Received: 27 February 2025 / Accepted: 12 January 2026

Abstract

This survey presents a comprehensive analysis of the quantum Fourier transform (QFT), a fundamental tool in quantum computing, in comparison to its classical counterpart, the fast Fourier transform (FFT). The study begins with an introduction to the classical Fourier transform, which is widely used in different disciplines such as signal and image processing. The article introduces the QFT as a quantum version of the Fourier transform, detailing how it leverages quantum parallelism and superposition to reduce the time complexity of the operation, and highlighting its crucial role in quantum algorithms like Shor's algorithm for integer factorization. The analysis also addresses the mathematical foundations of the QFT, its implementation in quantum circuits, and the key advantages and challenges associated with its use, such as measurement precision and quantum decoherence. Finally, it concludes with an exploration of the current and potential applications of QFT in quantum computing.

Keywords Quantum Fourier transform · Fast Fourier transform · Discrete Fourier transform

Tomás Barros, Pablo Álvarez, Bárbara Vidal and Mauricio Solar have contributed equally to this work.

This work is supported by Universidad Tecnica Federico Santa Maria.

✉ Mauricio Solar
mauricio.solar@usm.cl

Tomás Barros
tomas.barros@usm.cl

Pablo Álvarez
pablo.alvarezs@usm.cl

Bárbara Vidal
bvidal@usm.cl

[1] Department of Informatics, Universidad Tecnica Federico Santa Maria, Av. España 1680, 2390123 Valparaiso, V Región, Chile

[2] Department of Informatics, Universidad Tecnica Federico Santa Maria, Av. Vicuña Mackenna 3939, 8940000 Santiago, Metropolitana, Chile

[3] Toulouse INP-N7, Ecole Nationale Supérieure d'Electrotechnique, d'Electronique, d'Informatique, d'Hydraulique et des Télécommunications (ENSEEIHT), 6 allée Emile Monso, 34038 Toulouse, Midi-Pyrenees, France

Published online: 27 January 2026

1 Introduction

At the beginning of the modern information era, the first electronic digital computer ENIAC (1945) weighed 27 tons, consumed 174 kW, and had a memory size of 20 numbers-words and a clock frequency of 100 kHz [27]. Today, the miniaturization and operation speed of digital computing devices have reached a level that makes it possible to speak about intelligent systems with fantastic capabilities, and the boundary in this direction has not yet been reached.

Despite this, not even parallelization is enough to solve problems at exponential growth of complexity. At present, the operation speed of the fastest supercomputers is close to 10^{17} FLOPS. Will humankind be restricted in calculations to only problems of polynomial complexity?

An alternative computing model is the quantum computing model, which offers a fundamentally different approach to solving complex problems. One of the core components of quantum computing is the quantum Fourier transform (QFT), a quantum analogue of the classical Fourier transform. The QFT plays a critical role in several quantum algorithms, such as Shor's algorithm for integer factorization, which exponentially speeds up the process compared to classical methods. This survey explores the theoretical foundations of the QFT and delves into its applications in fields such as cryptography, quantum simulation, and solving systems of linear equations, highlighting how the QFT can revolutionize computations that are intractable for classical systems [50]. The creation of scalable quantum computers would fully unleash the potential of QFT-based algorithms, enabling them to tackle problems that are currently beyond the reach of even the fastest classical supercomputers.

After the description of theoretical foundations of the QFT, Section 3 explains the followed methodology. Section 3.1 presents the categorization chart to describe the overall tendencies in QFT research. Section 4 shows a review of relevant research on the QFT according to the categorization of Sects. 3.1, and 5 concludes the present work.

2 Theoretical foundations

2.1 The Fourier transform

In linear algebra, a vector $\mathbf{v} \in \mathbb{R}^n$ or $\mathbb{C}^n$ can be expressed as a linear combination of basis vectors. Given a basis $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ for the vector space, any vector $\mathbf{v}$ can be written as Eq. 1, where v_i are the coordinates (or components) of $\mathbf{v}$ with respect to the basis vectors $\mathbf{e}_i$.

$$\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{e}_i \tag{1}$$

A change of basis involves re-expressing the vector $\mathbf{v}$ in terms of a new set of basis vectors $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n\}$. If this new basis is orthonormal, meaning that the basis vectors are mutually orthogonal and have unit length (i.e., $\langle \mathbf{f}_i, \mathbf{f}_j \rangle = \delta_{ij}$), then the expansion

of the vector in the new basis can be written as Eq. 2.

$$\mathbf{v} = \sum_{i=1}^{n} \hat{v}_i \mathbf{f}_i \tag{2}$$

where the coefficients $\hat{v}_i$ are given by the inner product between the vector and the basis vectors of Eq. 3.

$$\hat{v}_i = \langle \mathbf{v}, \mathbf{f}_i \rangle. \tag{3}$$

This process is called "the expansion of a vector in an orthonormal basis".
The discrete Fourier transform (DFT) can be interpreted as a special case of this change of basis, where the signal (treated as a vector) is expanded in terms of sinusoidal basis functions. These functions form an orthonormal basis in the space of discrete signals. Thus, applying the DFT to a signal is equivalent to expressing the signal as a sum of sinusoidal components (frequency components), which is analogous to expanding a vector in an orthonormal basis.

2.2 The fast Fourier transform (FFT)

For a signal f of which we know a sample $\{f[n]\}_{n=0}^{N-1}$, the direct calculation of the N coefficients of the DFT, shown in Eq. 4, requires $2N^2$ operations (complex addition and multiplication).

$$\hat{f}[k] \equiv \sum_{n=0}^{N-1} f[n] e^{-kn\frac{2i\pi}{N}} \qquad \text{for } k = 0, \ldots, N-1 \tag{4}$$

By splitting the operations, the FFT algorithm allows for a considerable reduction in the computation time by bringing it down to an order of $O(N \log(N))$. The algorithm discovered by [14] has been developed in many more sophisticated versions to adapt to different conditions (mainly the length of vectors) to obtain an even faster result. Behind a seemingly simple transformation, FFT hides a multitude of ideas of a combinatorial and algebraic nature.
The Cooley and Tukey algorithm revolutionized signal processing. It allows, when we have a "good" decomposition of the integer N, to calculate the DFT very quickly. We will see in the remainder of the presentation of other algorithms which make it possible to exploit certain less optimal decompositions of N. However, in this first approach of the FFT algorithm, we will assume that $N = 2^p$. This very simple factorization of N will facilitate the use of the "divide and conquer" philosophy by performing a split in the calculation of the DFT. To implement the split, let us group the terms of the sum of a DFT according to the parity of the indices (Eq. 5). Then, we obtain Eq. 6, for

$k \in \{0, \ldots, N-1\}$.

$$\hat{f}[k] = \sum_{n=0}^{N/2-1} f[2n] e^{-2\imath\pi k(2n)/N} + \sum_{n=0}^{N/2-1} f[2n+1] e^{-2\imath\pi k(2n+1)/N} \tag{5}$$

$$= \sum_{n=0}^{N/2-1} f[2n] e^{-2\imath\pi kn/(N/2)} + \omega_N^{-k} \sum_{n=0}^{N/2-1} f[2n+1] e^{-2\imath\pi kn/(N/2)}, \tag{6}$$

where we denote $\omega_N = e^{2\imath\pi/N}$. So, if we write f^0 and f^1 as in Eqs. 7 and 8,

$$f^0 \equiv \{f[0],\ f[2], \ldots,\ f[N-2]\} \tag{7}$$

$$f^1 \equiv \{f[1],\ f[3], \ldots,\ f[N-1]\} \tag{8}$$

the vectors of even (f^0 in Eq. 7) (and odds with f^1 in Eq. 8) indices formed from f, we notice that for the $N/2$ first indices $k \in \{0,\ 1, \ldots,\ N/2-1\}$, Eq. (6) is written as the sum of two DFTs, according to Eq. 9.

$$\hat{f}[k] = \hat{f}^0[k] + \omega_N^{-k} \hat{f}^1[k] \tag{9}$$

For the indices $k \in \{N/2, \ldots,\ N-1\}$, if we write $k' = kN/2$, using the fact that the vectors $\hat{f}^0$ and $\hat{f}^1$ represent samples of period $N/2$, and that $\omega_N^k = -\omega_N^{k'}$, this time we get the difference of two transforms from Fourier (Eq. 10):

$$\hat{f}[k] = \hat{f}^0[k'] - \omega_N^{-k'} \hat{f}^1[k'] \tag{10}$$

Equations (9) and (10) are the key to the FFT algorithm. They allow us to calculate the DFT of a vector of length N by calculating two DFTs of length $N/2$ and then combining them in a certain way. This approach is then applied recursively to the vectors of length $N/2$, until we reach vectors of length 1.

One of the most important properties of the Fourier transform is the convolution theorem. Instead of directly computing the convolution, which is computationally expensive (requiring $O(N^2)$ operations), we can instead transform both signals to the frequency domain using the DFT, multiply their Fourier transforms pointwise, and then apply the inverse DFT to obtain the result in the time domain. This approach is computationally efficient, particularly when used in conjunction with the FFT, reducing the complexity to $O(N \log N)$.

This property is extremely useful in various fields, such as digital signal processing (see Sect. 2.3.1), where convolution is a fundamental operation for filtering, noise reduction, and event detection (see also Sect. 2.3). Additionally, convolution plays a key role in image processing (see Sect. 2.3.2), where it is used for denoising, edge enhancement, and image restoration. Finally, in numerical analysis and cryptography (see Sect. 2.3.3), the FFT enables efficient polynomial multiplication and is integral to modern cryptographic algorithms such as lattice-based cryptography and homomorphic encryption. These applications highlight the significance of the Convolution

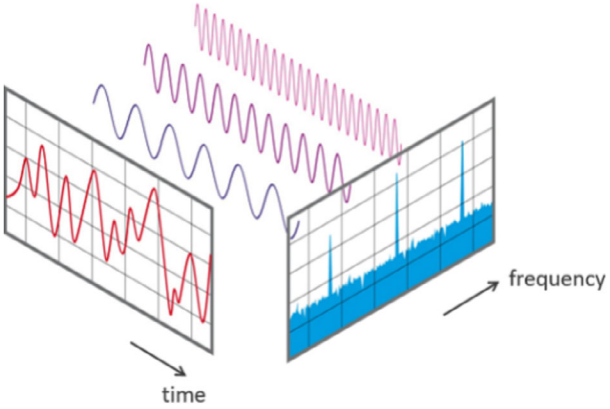

Fig. 1 View of a signal in the time and frequency domain. . Source: NTi Audio [51]

Theorem in both time and frequency domains, making it an essential tool for efficiently processing signals, images, and numerical data.

2.3 Applications of the FFT

2.3.1 Applications in signal processing

The FFT is one of the most frequently utilized methods for signal processing, as it facilitates the analysis of a time series by transforming a signal from the time domain into the frequency domain (see Fig. 1).

The FFT is used to perform some common applications by processing the resulting frequency-space representation and then returning to the time domain with the inverse transform. Some common applications are:

- Noise Reduction: Spectral subtraction is a powerful tool that leverages the FFT to remove noise from a signal by computing the spectrum of the noisy speech and subtracting the average magnitude of the noise spectrum from the noisy speech spectrum [31].
- Event Detection: The combination of the FFT and a Hilbert–Huang transform has been successfully applied to the detection of ball-bearing failures [56].
- Compression: Lossy compression of a signal can be achieved by removing high-frequency components in its frequency domain. When the signal returns to the time domain, a compressed version is obtained, as said frequencies tend to represent the noise and/or details of the signal [32].

2.3.2 Applications in image processing

In 2D, the Fourier transform retains the same properties as in 1D, except that the values in the frequency plane are complex and that it is symmetrical with respect to the center of the image. When being graphically displayed, it is common for the modulus of the image to be shown.

Observing the orientation of images: The modulus of the Fourier transform in (u, v) represents the importance of the periodic patterns in the image in the direction given by the vector $((0, 0) \rightarrow (u, v))$ with a frequency proportional to $\frac{1}{(u^2+v^2)}$. Thus, the preferred directions of the image can be seen in the modulus of its Fourier transform.

This operation has many applications, such as:

- Background/shape separation: In the frequency plane, the low frequencies are represented in the center and correspond to the flatness of colors, while the high frequencies located at the edges correspond to the rapid changes in colorimetry, and therefore to the contours. Separating high and low frequencies therefore separates the background from the shape of the image.
- Denoising, edge enhancement, image restoration: In addition, convolution calculation in the time base only corresponds to term-by-term multiplication in the frequency base. This makes conventional filtering easy to perform and understand.
- Image compression: In the same way as in 1D, the frequency vision of the image enables it to be compressed while retaining important information, such as contours.
- Magnetic resonance imaging (MRI)/tomography.

2.3.3 Applications in finite fields

The FFT has significant applications in numerical analysis and cryptography. One key area is the efficient multiplication of large integers and polynomials, which is fundamental for algorithms in both fields.

Polynomial multiplication and Schönhage–Strassen algorithm

One of the key advancements in reducing computational complexity is the use of the FFT in polynomial and large number multiplications. Normally, multiplying two numbers or polynomials of degree n requires $O(n^2)$ operations. However, the Schönhage–Strassen algorithm [60] leverages the FFT to reduce this complexity to $O(n \log n \log \log n)$, making it highly efficient for large integer multiplications. This technique is not only useful for basic arithmetic operations but also plays a vital role in algorithms used for number theory and cryptography.

Integer factorization and cryptography.

In number theory and cryptography, the efficiency of integer factorization algorithms depends heavily on fast multiplication techniques like the Schönhage–Strassen algorithm [60] and FFT-based methods. Integer factorization is central to cryptographic schemes such as RSA (Rivest–Shamir–Adleman), where the security relies on the computational difficulty of factoring large composite numbers. Fast multiplication algorithms improve the performance of these factorization methods, although even with these improvements, factoring large integers (especially in cryptography) remains computationally intensive. The application of the NTT (number theoretic transform) in cryptographic algorithms further enhances the efficiency of modular arithmetic operations, which are key in public-key encryption schemes [34].

Thus, the FFT and its variants are not only tools for signal and image processing but also powerful methods for solving complex problems in numerical analysis and cryp-

tography, contributing to both theoretical advancements and practical implementations in these fields.

2.4 The quantum Fourier transform (QFT)

The quantum Fourier transform (QFT) is a key ingredient for several quantum algorithms, such as Shor's algorithm for factorization [62] or the quantum phase estimation (QPE) algorithm for the estimation of the eigenvalues of a unitary operator [50]. The latter additionally appears as a subroutine of other algorithms, such as the Harrow–Hassidim–Lloyd (HHL) algorithm for linear systems of equations [25] or the quantum principal component analysis (PCA) algorithm [37]. The quantum version of the discrete Fourier transform (DFT) has an exponential speed-up over its classical counterpart. Although, on the classical version, it is necessary to apply $O(n2^n)$ gates, where n refers to the number of bits, on the quantum approach only $O(n^2)$ gates are needed, in this case n stands for the number of qubits [41].

QFT is the quantum implementation of the DFT over the amplitudes of a wavefunction. The QFT acts on a quantum state $|X\rangle = \sum_{\ell=0}^{q-1} x_\ell |\ell\rangle$ and maps it to the quantum state $|Y\rangle = \sum_{k=0}^{q-1} y_k |k\rangle$. Note that only the amplitudes of the state are affected by this transformation.

The DFT acts on a vector $(\ell_0, \ldots, \ell_{q-1})$ and maps it to the vector $(k_0, \ldots, k_{q-1})$ according to Eq. 11, which represents the q-dimensional Fourier transform F_q, where $0 \leq k < q$ and $\omega = e^{2\pi i/q}$.

$$F_q|k\rangle = \frac{1}{\sqrt{q}} \sum_{\ell=0}^{q-1} \omega^{k\ell} |\ell\rangle \tag{11}$$

It should be noted that the input (k, ℓ) of F_q is $(F_q)_{k\ell} = \frac{\omega^{k\ell}}{\sqrt{q}}$. So, F_q is a symmetric matrix.

To find $F_q^\dagger$, one simply takes the complex conjugate of each entry, which is $\omega^{-k\ell}/\sqrt{q}$ because the complex conjugate of ω is ω^{-1}. Then one has Eq. 12, where $0 \leq k < q$.

$$F_q^\dagger|k\rangle = \frac{1}{\sqrt{q}} \sum_{\ell=0}^{q-1} \omega^{-k\ell} |\ell\rangle \tag{12}$$

For this to work, it must be demonstrated that F_q is unitary. Using the definitions of F_q and $F_q^\dagger$, we have Eq. 13.

$$\langle k^{'}|F_q^\dagger F_q|k\rangle = \left(\frac{1}{\sqrt{q}} \sum_{\ell^{'}=0}^{q-1} \omega^{-k^{'}\ell^{'}} \langle \ell^{'}|\right) \left(\frac{1}{\sqrt{q}} \sum_{\ell=0}^{q-1} \omega^{k\ell}|\ell\rangle\right) = \frac{1}{q} \sum_{\ell=0}^{q-1} \omega^{(k-k^{'})\ell} \tag{13}$$

Using the closed-form formula for the geometric series with q terms, we obtain Eq. 14 for $s \neq 1$.

$$\sum_{k=0}^{q-1} s^k = \frac{1-s^q}{1-s} \tag{14}$$

When $s = 1$, the sum on the left is equal to q. In this case, $s = \omega^{(k-k')}$. By the definition of ω, we have $\omega^q = 1$ and then $\omega^{(k-k')q} = 1$. Combining those results, we obtain Eq. 15.

$$\frac{1}{q}\sum_{\ell=0}^{q-1} \omega^{(k-k')\ell} = \begin{cases} 1, \text{if} k = k', \\ 0, \text{otherwise} \end{cases} \tag{15}$$

This demonstrates that $\langle k'|F_q^\dagger F_q|k\rangle = \delta_{kk'}$, that is, $F_q^\dagger F_q = I$.

The first decomposition of the Fourier transform in terms of basic quantum gates is given in Coppersmith [15]. A description of this decomposition based on the classical FFT is available in Marquezino et al. [40]. The best known QFT algorithms require $O(n \log n)$ gates to achieve an efficient approximation. The circuit of F_q in terms of CNOTs and one-qubit gates is described in the next section.

2.4.1 Fourier transform circuit

The basic block of the Fourier transform circuit F_{2^n}, where n is the number of qubits, is the controlled gate $C(R_k)$ for $k \geq 0$, where R_k is given by Eq. 16.

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix} \tag{16}$$

The matrix representation of $C(R_k)$ is shown in Eq. 17.

$$C(R_k) = \begin{bmatrix} I_2 & \\ & R_k \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix} \tag{17}$$

The gate set R_k has several special subcases, as shown in Eq. 18.

$$\begin{aligned} R_0 &= I_2, \\ R_1 &= Z, \\ R_2 &= S, \\ R_3 &= T, \end{aligned} \tag{18}$$

where Z, S, and T are the Pauli gates Z, the phase gate, and the $\pi/8$ or T gate, respectively.

It should be noted that $R_{k+1} = \sqrt{R_k}$. So, the above sequence means that the next gate is the square root of the previous one. The same idea applies to $C(R_k)$, i.e.,

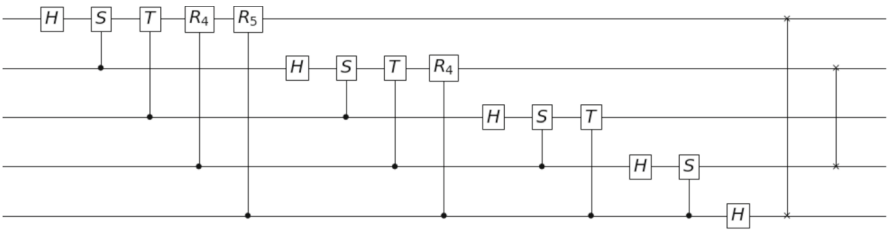

Fig. 2 Decomposition of the Fourier transform F_{2^5}

$C(R_{k+1}) = \sqrt{C(R_k)}$, but in the latter case the square root of (4×4) matrices is being computed. Figure 2 depicts the circuit of F_{2^n} in terms of the Hadamard gate, $C(R_k)$ and swap gates when $n = 5$.

The circuit structure can be easily understood from this example. The circuit has $n + 1$ blocks. From left to right, the first block has n gates, starting with H and then with $R_2, R_3, \ldots, R_n$ acting on qubit 1 and controlled by qubit $2, 3, \ldots, n$, respectively. The next block starts again with H and then with $R_2, R_3, \ldots, R_{n-1}$ acting on qubit 2 and controlled by qubit $3, 4, \ldots, n$, respectively. This continues until reaching the last qubit, on which a single (n-th) H block is applied. The last block is made of $\lfloor n/2 \rfloor$ swap gates and has a simple symmetric structure. If n is odd, the middle qubit is not swapped. The number of gates is $n + (n-1) + \cdots + 1 + \lfloor n/2 \rfloor = n(n+1)/2 + \lfloor n/2 \rfloor$.

It is shown that the circuit with the structure depicted in Fig. 2 implements the Fourier transform when having n qubits. Assume that the input is $|\ell\rangle = |\ell_1\rangle \otimes \cdots \otimes |\ell_n\rangle$. When the input is a state of the computational basis, the output is a deentanglement state $|\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$.

We start by calculating the output $|\psi_1\rangle$ of the first qubit. Since there is an exchange gate that inverts the states of the first and last qubit, we have Eq. 19.

$$|\psi_1\rangle = H|\ell_n\rangle = \frac{|0\rangle + e^{2\pi i \frac{\ell_n}{2}}|1\rangle}{\sqrt{2}} = \frac{|0\rangle + e^{2\pi i \frac{\ell}{2}}|1\rangle}{\sqrt{2}} \tag{19}$$

The second equation follows from the fact that if ℓ_n is 0, the output is $|+\rangle$, and if $\ell_n = 1$, the output is $|-\rangle$ (because $e^{\pi i}$ is -1). The last equation follows from the decomposition of $\ell = 2^{n-1}\ell_1 + 2^{n-2}\ell_{n-2} + \cdots + 2\ell_{n-1} + \ell_n$ and of $e^{2\pi i k} = 1$ if k is an integer.

The output of the second qubit is given by Eq. 20.

$$\begin{aligned} |\psi_2\rangle &= R_2^{\ell_n} H|\ell_{n-1}\rangle = R_2^{\ell_n}\left(\frac{|0\rangle + e^{2\pi i \frac{\ell_{n-1}}{2}}|1\rangle}{\sqrt{2}}\right) \\ &= \frac{|0\rangle + e^{2\pi i\left(\frac{\ell_{n-1}}{2} + \frac{\ell_n}{2^2}\right)}|1\rangle}{\sqrt{2}} = \frac{|0\rangle + e^{2\pi i \frac{\ell}{2^2}}|1\rangle}{\sqrt{2}} \end{aligned} \tag{20}$$

The first equation is obtained from Fig. 2 using $C(R_2)|\ell_n\rangle|\ell_{n-1}\rangle = |\ell_n\rangle(R_2^{\ell_n}|\ell_{n-1}\rangle)$.

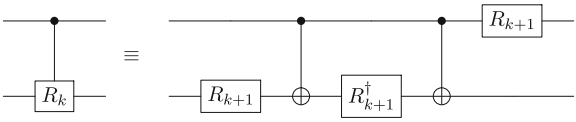

Fig. 3 $C(R_k)$ decomposition circuit

The second equation uses the same calculation described above for the first qubit. The third equation follows from $R_2^{\ell_n}|0\rangle = |0\rangle$, and $R_2^{\ell_n}|1\rangle = exp^{2\pi i \frac{\ell_n}{2^2}}|1\rangle$.

The last equation uses the same decomposition of ℓ described above. The output of the last qubit is given by Eq. 21.

$$|\psi_n\rangle = R_n^{\ell_n} \cdots R_2^{\ell_2} H|\ell_1\rangle = \frac{|0\rangle + e^{2\pi i(\frac{\ell_1}{2} + \frac{\ell_2}{2^2} + \cdots + \frac{\ell_n}{2^n})}|1\rangle}{\sqrt{2}} = \frac{|0\rangle + e^{2\pi i \frac{\ell}{2^n}}|1\rangle}{\sqrt{2}} \quad (21)$$

The first equation is obtained from Fig. 2 using the output of the last qubit which is obtained from the action of $H, R_2^{\ell_2}, \ldots, R_n^{\ell_n}$ on the first qubit. The second and third equations are obtained with the same type of calculations described above for the first and second qubit.

Then, the output $|\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$ of the circuit in Fig. 2 with n qubits is Eq. 22.

$$\frac{|0\rangle + e^{2\pi i \frac{\ell}{2}}|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + e^{2\pi i \frac{\ell}{2^2}}|1\rangle}{\sqrt{2}} \otimes \cdots \otimes \frac{|0\rangle + e^{2\pi i \frac{\ell}{2^n}}|1\rangle}{\sqrt{2}} \quad (22)$$

Converting each term into a sum gives Eq. 23.

$$\frac{1}{\sqrt{2}} \sum_{k_1=0}^{1} e^{2\pi i k_1 \frac{\ell}{2}}|k_1\rangle \otimes \frac{1}{\sqrt{2}} \sum_{k_2=0}^{1} e^{2\pi i k_2 \frac{\ell}{2^2}}|k_2\rangle \otimes \cdots \otimes \frac{1}{\sqrt{2}} \sum_{k_n=0}^{1} e^{2\pi i k_n \frac{\ell}{2^n}}|k_n\rangle \quad (23)$$

Placing all the sums to the right side and combining the exponentials, we obtain Eq. 24.

$$\frac{1}{\sqrt{2^n}} \sum_{k_1,\ldots,k_n=0}^{1} e^{2\pi i \ell \left(\frac{k_1}{2} + \cdots + \frac{k_n}{2^n}\right)}|k_1, \ldots, k_n\rangle \quad (24)$$

which is equivalent to $\frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{\frac{2\pi i \ell k}{2^n}}|k\rangle$.

Using the definition of the Fourier transform given in Sect. 2.4, the last expression is recognized to be $F_{2^n}|\ell\rangle$.

Decomposition of $C(R_k)$

The circuit providing the decomposition of $C(R_k)$ in terms of CNOTs and 1-qubit gates R_{k+1} is shown in Fig. 3.

This is not a decomposition in terms of universal gates because one needs to write R_{k+1} in terms of a finite set of 1-qubit gates. In most cases, one does not need to worry about this step as it is done automatically by the compiler of quantum computers. R_k

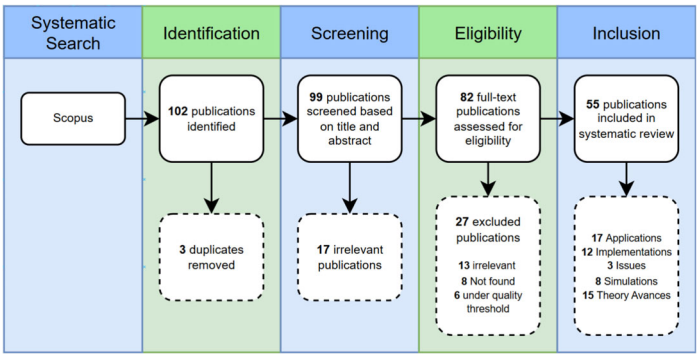

Fig. 4 The PRISMA diagram for this review

can be implemented using R_z, which is a rotation of the Bloch sphere about the z axis. Note that if k is large, errors will prevent these gates from working correctly unless error-correcting codes exist.

3 Methodology

We performed a systematic literature review to identify relevant academic papers regarding the QFT. We employed the Scopus database using the search terms:

- 'Quantum' AND ' 'Fourier' AND 'Transform' AND 'Computing'
- 'Shor' AND 'Algorithm'
- 'QFT' AND 'Algorithm'

and limited the search to articles and conference papers which include the 'Quantum Theory' keyword as to reduce the amount of false positive matches.

Figure 4 corresponds to the PRISMA diagram for this review. 102 publications were initially identified through the use of database keywords, out of which 47 were excluded for relevance, quality and accessibility reasons. The authors independently screened all records for eligibility, and 55 publications were finally included in the systematic review.

3.1 Tendencies in QFT research

Figure 5 presents the categorization chart to describe the overall tendencies in QFT research. The categorization is designed to reflect the multi-faceted nature of QFT as both a theoretical construct and a practical tool in quantum computing. Based on the works analyzed during the research process and following the methodological approach, we identified three main categories that encompass the breadth of QFT research: fundamental studies, practical aspects, and applications. This structure aims to highlight the key areas of focus and the depth of ongoing work in this field.

The diagram in Fig. 5 illustrates this categorization, providing a clear structure to analyze the research landscape of the QFT.

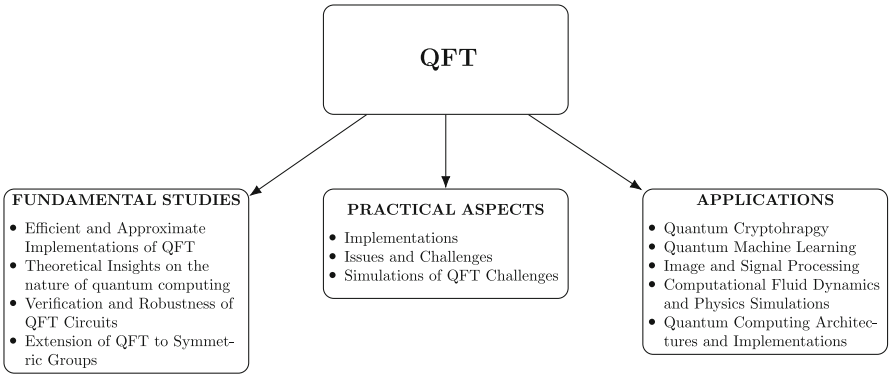

Fig. 5 Categorization chart

Fundamental studies focus on the mathematical and theoretical foundations of the QFT. This includes research into its inherent properties, its relationship with classical Fourier transforms, and its behavior under conditions such as noise and quantum errors. These studies are essential for understanding and improving the core algorithm.

Practical aspects cover the implementation and optimization of the QFT. This category includes work on quantum hardware and software, the simulation of QFT on classical systems, and algorithmic advancements that improve its efficiency and resource requirements. These efforts are critical for bridging the gap between theory and real-world usage.

Applications represent the practical use cases of the QFT in quantum computing. This includes its role in Quantum Phase Estimation (QPE), cryptographic protocols, and quantum simulations, among others. These applications demonstrate the transformative potential of the QFT in solving complex problems across various domains.

4 A brief description of relevant research on the QFT

What follows is a review of relevant research on the QFT according to the previously mentioned categorization. We present a brief description of each publication's results in order to facilitate further research.

4.1 Fundamental studies

Fundamental studies focus on theoretical advances in the understanding of the QFT. This section looks at recent research on more efficient QFT approximations, its role in quantum-classical comparisons, and insights regarding the algorithm's distributability.

4.1.1 Efficient and approximate implementations of QFT

Circuit optimization is essential for making quantum algorithms practical on near-term quantum hardware. The standard fault-tolerant implementation of the QFT approx-

imates the desired transformation by removing small-angle controlled rotations and synthesizing the remaining ones into Clifford+T gates, incurring the T-count complexity of $O(n \log_2(n))$. Nam et al. [48] developed an approximate QFT implementation that uses only $O(n \log n)$ T gates, significantly reducing the algorithms' gate complexity.

Similarly, higher-radix representations have been explored to improve the scaling and accuracy of QFT computations, enabling better approximations. Zilic and Radecka [75] propose the use of ternary quantum bits (with corresponding ternary quantum gates) to access better approximation properties than the traditional binary QFT.

Parallelism is another key factor. Research on fast parallel circuits for the QFT shows that parallel implementations can reduce execution time and improve fault tolerance [13]. Additionally, by adapting classical computing techniques, the QFT can be derived from the FFT using QR decomposition, offering a new approach for building quantum algorithms using classical ones [40].

Cleve and Watrous [13] give an upper bound $O(\log n + \log\log(1/\epsilon))$ on the circuit depth for computing an approximate QFT with respect to the modulus 2^n with error bounded by ϵ. They also prove an $\Omega(\log n)$ lower bound on the depth complexity of approximate QFT with constant error.

The circuit shown in Kahanamoku-Meyer et al. [29] has depth $O(\log(n/\epsilon))$ for tunable error parameter ϵ, uses n total qubits, i.e., no ancillas, is logarithmically local for input qubits arranged in 1D, and is measurement-free. The circuit's error is bounded by ϵ on all input states except an $O(\epsilon)$-sized fraction of the Hilbert space. According to the authors, the circuit is also rather simple and thus may be practically useful.

In Bäumer et al. [7] is shown a n-qubit QFT followed immediately by measurement, where the scaling of resource requirements is reduced from $O(n^2)$ two-qubit gates in an all-to-all connectivity in the standard unitary formulation to $O(n)$ mid-circuit measurements in its dynamic counterpart without any connectivity constraints.

Bäumer et al. [8] also presented a circuit which uses measurement and feed-forward to achieve logarithmic depth with only nearest-neighbor connectivity and $O(n)$ ancilla qubits.

While in some sense these quantum circuits are exponentially faster than the classical FFT, the task that they perform is quite different. The QFT does not explicitly produce any of the values $\beta_0, \beta_1, \ldots, \beta_{m-1}$ as output (nor does it explicitly obtain any of the values $\alpha_0, \alpha_1, \ldots, \alpha_{m-1}$ as input). Intuitively, the difference between performing a DFT and a QFT can be thought of as being analogous to the difference between computing all the probabilities that comprise a probability distribution and sampling a probability distribution—the latter task being frequently much easier [13].

4.1.2 Theoretical insights on the nature of quantum computing

The QFT plays a crucial role in highlighting the difference between quantum and classical computing, and several studies have explored the fundamental nature of this advantage. For example, the Forrelation problem provides a clear distinction between quantum and classical computing capabilities, with the QFT helping to demonstrate quantum superiority [1]. The idea of quantum parallelism has also been explored, focusing on its exact role and limitations in quantum computing models [39].

Other studies question whether quantum mechanics is truly necessary for quantum algorithms, exploring whether quantum speed-ups come from specific physical properties or if they can be replicated classically. The question "Do we need 'Quantum' for quantum computing?" [19] challenges the assumptions about quantum advantage and prompts a discussion on the principles behind QFT's computational power. Similarly, "Quantum Computation: Where Does the Speed-Up Come From?" looks at the mechanisms that allow QFT-based quantum algorithms to outperform classical methods [10] .

Kahanamoku-Meyer et al. [29] say that there exist strategies for avoiding an algorithm's worst-case behavior entirely: it may be possible to detect bad inputs early and switch to an algorithm which is more performant on them, or to solve the worst case by building off of the solution of a related average-case input—a technique known as a worst-case to average-case reduction [6].

4.1.3 Verification and robustness of QFT circuits

The QFT by itself will not provide an exponential speed-up in comparison with classical algorithms. One possibility is using atomic target qubits as discussed in Tomita and Nakamura [69], where the unitary transformation is achieved by the interaction between atoms.

Beauregard [9] used QFT for modular exponentiation and a QFT followed by measurement on the control qubit. It is fault-tolerant, so his scheme not only reduces the number of qubits but also provides robustness against decoherence.

Ensuring the correctness of QFT implementations requires formal verification techniques. One approach involves using rotational abstractions to verify QFT circuits, providing guarantees about their accuracy and functionality [21]. Their method scales up to 10,000 qubits and 50 million quantum gates, providing an enormous advance in the size of QFT circuits thus far verified using formal verification methods

Studies on the impact of noise on QFT-based quantum algorithms investigate how errors in intermediate-scale quantum systems can affect performance, offering insights into the robustness of QFT in real-world applications. Volya and Mishra [70] present a complete characterization of quantum noise and performs a case-study to investigate the impact of noise on the QFT in NISQ systems.

4.1.4 Extensions of QFT

While QFT is traditionally defined for integer-based quantum states, there has been interest in extending it to other mathematical structures. In particular, the QFT over symmetric groups looks at how QFT can be adapted for problems involving permutation symmetries. Kawano and Sekigawa [33] present an algorithm specialized for symmetric groups with $O(n^3 \log n)$ time complexity, faster than the $O(n^4 \log n)$ of the general version, which can run in $O(n^3)$ time when allowed some additional gate requirements. Furthermore, Rodriguez [58] proposes a generalization of the QFT that applies to Galois fields (a mathematical construct widely used in quantum error correction algorithms).

4.2 Practical aspects

4.2.1 Implementations of the QFT

Due to the importance of the QFT, there are several implementations optimized for different hardware and algorithmic needs. This section reviews important works that have contributed to making QFT more efficient, covering experimental, algorithmic, and distributed computing approaches.

Hardware-Specific Implementations

Many studies focus on adapting the QFT for specific quantum hardware platforms. For example, a hybrid qubit–qutrit nuclear magnetic resonance (NMR) quantum emulator has been used to implement the QFT, showing that it's feasible to run on hybrid quantum systems employing higher-radix qudits [17].

In a similar vein, digital–analog quantum computation has been proposed as an effective method for performing QFT while reducing gate overhead and minimizing decoherence. The approach found in Martin et al. [41] suggests that in the NISQ era, hybrid protocols combining digital and analog quantum computing could be a sensible approach to reach useful quantum supremacy.

It has been shown in Freedman and Wang [20] that large QFTs cannot be exactly realized using topological quantum computing, which is an alternative architecture to quantum circuits. This presents challenges for fault-tolerant quantum computation, as it means alternative methods or approximations must be used when implementing QFT in topological quantum circuits.

Other research has explored QFT optimizations through tailored processor architectures. Yung et al. [74] propose an architecture based on a processing 'core' where multiple qubits interact perpetually, and a separate 'store' where qubits exist in isolation. Said model allows the implementation of a QFT with $O(n)$ circuit depth, as well as producing potential reductions in the decoherence rate.

Vorobyov et al. [71] utilize QFT to enhance the performance of a quantum sensor using individual solid-state nuclear spins in diamond.

Specifically, the authors show the application of QFT for correlation spectroscopy, where the long correlation time benefits the use of the QFT in gaining maximum precision and dynamic range at the same time.

Algorithmic Optimizations and Error Mitigation

Efficiently implementing the QFT requires strategies to reduce circuit depth and improve resilience to errors. Nagy and Akl [47] prove that under the constraints imposed by quantum decoherence, only a parallel approach can guarantee a reliable, scalable solution. The authors also propose a parallelization scheme to minimize the effect of decoherence on the computation of the QFT.

Another study explores how genetic programming can be used to evolve novel QFT implementations that outperform traditional, hand-designed circuits. Massey et al. [42] propose Q-Pace, a software suite built to evolve quantum circuits. Their resulting designs are competitive with the known QFT algorithm, and their approach may be able to construct further quantum algorithms.

Beyond the direct implementation of the QFT, research on QPE algorithms has led to improved methods that balance computational accuracy with qubit overhead.

Chiang [12] presents an alternative QFT implementation that produces a result with exact n-bit precision by leveraging additional qubits. The algorithm also boasts a lower rotational gate complexity of $O(n \log k)$ compared to the standard $O(n^2)$

Experimental results in Mohammadbagherpoor et al. [44] also show that the accuracy of QPE is greatly constrained by current NISQ architectures' physical characteristics. The authors propose a modified solution that reduces the amount of operations to increase the accuracy in near-term quantum computers specifically.

Distributed Approaches

As quantum computing scales up, distributed implementations of the QFT have become an area of interest. For example, a distributed version of Shor's algorithm integrates QFT across multiple nodes, demonstrating how quantum networks can enable large-scale computations that would otherwise be hampered by decoherence [73].

Further research focuses on distributed QPE. Neumann et al. [49] evaluate the effects on the output fidelity of a quantum algorithm when using noisy shared entangled states. Their results show that using less noisy shared entangled states results in a higher overall fidelity.

Experimental Perspectives

Finally, practical implementations of QFT have been analyzed from an experimental perspective, where trade-offs between gate fidelity, execution time, and circuit depth are carefully considered. These studies provide benchmarks for different QFT architectures and offer valuable insights for real-world implementations [18, 72].

Some studies also explore de-quantization approaches, looking into how QFT-inspired methods can be used in classical computation to optimize certain tasks [2]. Additionally, there have been proposals to unify QFT with other fast unitary transforms, broadening its applicability beyond traditional quantum algorithms [3].

4.2.2 Issues and challenges

While the QFT is a fundamental component of quantum algorithms, its practical implementation faces several significant challenges. These challenges arise from mathematical limitations, hardware constraints, and fundamental quantum effects that impact the QFT's efficiency and reliability.

Limitations in Fourier Sampling

One of the key theoretical challenges with the QFT is its use in Fourier sampling problems. It has been shown that for particular problems the QFT is not as efficient as is expected. Particularly, Moore et al. [45] prove that the QFT cannot efficiently solve the notable hidden subgroup problem (HSP) for any group. This limitation places constraints on the types of computational problems where the QFT can deliver exponential speed-ups.

Sensitivity to Chaos and Imperfections

As with all quantum algorithms, the QFT is sensitive to quantum chaos and system imperfections. Small errors in gate operations, environmental noise, and decoherence have a cumulative effect on the accuracy of QFT-based computations. Song and Shepelyansky [66] explore how imperfections affect different aspects of quantum computing error rates, particularly how they degrade the performance of QFT-based algorithms.

4.2.3 Simulations of the QFT

Simulating the QFT on classical hardware is essential for testing quantum algorithms, improving implementations, and understanding the practical limits of near-term quantum devices. Researchers have explored a variety of methods, including high-performance computing, hardware acceleration, and algorithmic optimizations, to make QFT emulation more efficient on classical systems.

Classical Simulations of the QFT for Algorithm Testing
One of the main reasons for simulating the QFT on classical hardware is to evaluate quantum algorithms before running them on actual quantum processors. Karafyllidis [30] presents a quantum computer simulator based on the circuit model, which provides a general framework allowing researchers to study quantum algorithms without the need of quantum mechanical expertise. Pereira et al. [55] also propose an open-source alternative for the simulation of the QFT.

Optimizing Shor's Algorithm Simulations
Since Shor's factoring algorithm relies on the QFT, it has been also been a focus for classical simulation. Studies using frameworks like IBM's Qiskit, Javadi-Abhari et al. [28] have examined how to optimize and analyze Shor's algorithm, particularly in the context of near-term quantum hardware [68]. Additionally, researchers have used GPU acceleration to speed up the QFT portion of Shor's algorithm, allowing for significantly faster execution on classical hardware [59].

High-Performance Computing for QFT Simulations For large-scale QFT simulations, researchers have turned to supercomputers. Liu et al. [36] utilize the Sunway TaihuLight supercomputer, which leverages massive parallelism to handle large quantum state computations. These large-scale classical simulations are crucial for testing quantum algorithms at scales beyond what current quantum processors can handle.

Efficient Classical Computation of Unitary Transformations
Beyond direct QFT simulation, some studies focus on efficient classical computation of unitary transformations: The fundamental building blocks of quantum circuits. Researchers have proposed hardware architectures optimized for quantum algorithm emulation, offering a bridge between classical and quantum computing [38]. Additionally, research into Fourier 1-norm techniques has examined how classical systems can approximate some of the speed-ups associated with quantum Fourier-based computations [24].

4.3 Applications

The QFT is a key quantum algorithm with a wide range of applications in fields such as cryptography, machine learning (ML), image processing, computational physics, and signal processing. This section highlights some of the most impactful uses of QFT.

4.3.1 Quantum cryptography

One of the most prominent applications of the QFT is in cryptanalysis. Shor's algorithm for integer factorization and discrete logarithms, which forms the foundation

of many classical cryptographic systems, relies on QFT to efficiently find periodicity [62]. In addition to this, QFT has been employed to enhance the security of cryptographic schemes, such as Multi-Prime RSA, by limiting the potential for private key recovery [65]. QFT also plays a significant role in quantum homomorphic encryption, enabling secure computations on encrypted quantum data [11]. An updated survey on quantum cryptography is in Solar, Cisterna, Villacura and Dombrovskaia [63] and Solar, Villacura, Cisterna and Dombrovskaia [64].

4.3.2 Quantum machine learning (QML)

Quantum computing holds great promise for transforming ML, with the QFT playing a critical role, as QPE forms the backbone of many QML algorithms, Ouedrhiri et al. [52] present a few quantum implementations of classical ML algorithms, such as PCA (principal component analysis), the HHL (Harrow–Hassidim–Lloyd) algorithm and SVM (support vector machine). Additionally, QFT-based quantum neurons have been proposed to implement nonlinear functions more efficiently, which is a key step toward building quantum neural networks [54].

4.3.3 Image and signal processing

Following the FFTs widespread use in image processing shown in Sect. 2.3.2, some research has attempted to extend the processing of images to the realm of quantum computation. The application of the QFT in this domain has led to novel methods for representing images as quantum states, enabling efficient manipulation and compression [22, 23, 46]. QFT-inspired inverse algorithms have also been developed for unsupervised image segmentation, improving the robustness of feature extraction tasks [4].

In image processing literature, template matching is the problem of finding an image within another. By leveraging the QFT, quantum computers can efficiently compare and match patterns within large datasets, which is essential for applications like image recognition and computer vision [16].

The QFT also provides an efficient way to compute and analyze wave patterns, as these can be represented by signals with discrete frequencies. By transforming spatial data into frequency components, the QFT has been applied to array antenna analysis as [57], as well as signal synthesis [61]. The QFT has also been employed for quantum frequency detection, facilitating efficient signal analysis and processing [43].

Haque et al. [26] show a very complete taxonomy of the quantum image compression and representation scheme, where image representation approaches are classified into ten major categories, and one of them is QFT presented by Pang et al. [53].

4.3.4 Computational fluid dynamics and physics simulations

Quantum algorithms utilizing the QFT have shown potential in the field of computational fluid dynamics, offering parallel evaluation methods for fluid flow problems. Steijl and Barakos [67] propose the use of a hybrid architecture where select computationally intensive parts are implemented as quantum circuits. The authors show

that despite inevitable noise and uncertainties, meaningful flow simulations can be performed using their approach.

Additionally, the QFT plays a role in quantum computational mechanics, presenting new approaches for simulating and analyzing physical systems [35]. The authors' RVE solver attains exponential acceleration with respect to classical solvers, bringing concurrent multiscale computing closer to practicality.

4.3.5 Quantum computing architectures and implementations

Arsoski [5] proposes a novel implementation of a multi-controlled X gates based on the QFT. Said gates are crucial for many quantum algorithms, though they are computationally costly. Emulating said gates with the QFT allows for a decrease in gate complexity, while maintaining a constant time complexity. In cases where it is necessary to implement complex arithmetic operations for a gate, the QFT-based approach can be of great advantage.

5 Conclusion

The QFT is key to many quantum algorithms. Research has focused on making QFT more practical through efficient circuit designs, distributed implementations, and noise reduction techniques, helping to reduce the overhead that comes with running it on real quantum hardware. Advances in approximate QFTs, hardware-aware optimizations, and parallelization are making its execution more feasible for NISQ devices. Beyond its role in speeding up computations, the QFT has found applications in areas like image processing, cryptography, and signal analysis, showing its broader potential.

Still, challenges remain. Scalability, noise sensitivity, and hardware limitations make large-scale QFT implementations difficult. Finding ways to bridge classical and quantum approaches, refine error correction, and develop alternative architectures will be key to overcoming these hurdles.

Author Contributions All authors contributed equally to this work.

Data Availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no Conflict of interest.

References

1. Aaronson, S., Ambainis, A.: Forrelation: a problem that optimally separates quantum from classical computing. In: STOC'15, pp. 307–316. Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2746539.2746547
2. Abbott, A.A.: De-quantisation of the quantum Fourier transform. Appl. Math. Comput. **219**(1), 3–13 (2012). https://doi.org/10.1016/j.amc.2011.06.057
3. Agaian, S.S., Klappenecker, A.: Quantum computing and a unified approach to fast unitary transforms. In: Electronic Imaging (2002). https://api.semanticscholar.org/CorpusID:10204460
4. Akinola, T.A., Li, X., Wilkins, R., Obiomon, P.H., Qian, L.: Robust inverse quantum Fourier transform inspired algorithm for unsupervised image segmentation. IEEE Access **12**, 99029–99044 (2024). https://doi.org/10.1109/ACCESS.2024.3429413
5. Arsoski, V.V.: Implementing multi-controlled x gates using the quantum Fourier transform. Quantum Inf. Process. (2024). https://doi.org/10.1007/s11128-024-04511-w
6. Asadi, V., Golovnev, A., Gur, T., Shinkar, I., Subramanian, S.: Quantum worst-case to average-case reductions for all linear problems. In: Proceedings of the 2024 Annual ACM-SIAM Symposium On Discrete Algorithms (SODA) 2535–2567 (2024) https://doi.org/10.1137/1.9781611977912.90
7. Bäumer, E., Tripathi, V., Seif, A., Lidar, D., Wang, D.: Quantum Fourier transform using dynamic circuits. Phys. Rev. Lett. (2024). https://doi.org/10.1103/PhysRevLett.133.150602
8. Bäumer, E., Sutter, D., Woerner, S.: Approximate Quantum Fourier Transform in Logarithmic Depth on a Line (2025). arxiv:2504.20832
9. Beauregard, S.: Circuit for Shor's algorithm using 2n+3 qubits. Quantum Inf. Comput. **3**, 175–185 (2003)
10. Bub, J.: Quantum computation: Where does the speed-up come from? Philos. Quantum Inf. Entangl. (2010). https://doi.org/10.1017/CBO9780511676550.013
11. Chen, T., Lu, D.-J., Deng, Z.-M., Yao, W.-X.: Quantum homomorphic aggregate signature based on quantum Fourier transform. Quantum Inf. Process. (2024). https://doi.org/10.21203/rs.3.rs-3728263/v1 https://doi.org/10.21203/rs.3.rs-3728263/v1
12. Chiang, C.-F.: Quantum phase estimation with an arbitrary number of qubits. Int. J. Quantum Inf. (2013). https://doi.org/10.1142/S0219749913500081
13. Cleve, R., Watrous, J.: Fast parallel circuits for the quantum Fourier transform. In: Proceedings 41st Annual Symposium on Foundations of Computer Science, pp. 526–536 (2000). https://doi.org/10.1109/SFCS.2000.892140
14. Cooley, J., Tukey, J.: An algorithm for the machine calculation of complex Fourier series. Math. Comput. **19**(90), 297–301 (1965). https://doi.org/10.1090/S0025-5718-1965-0178586-1
15. Coppersmith, D.: An approximate Fourier transform useful in quantum factoring. Arxiv (2002) arxiv:quant-ph/0201067
16. Curtis, D., Meyer, D.: Towards quantum template matching. Proc. SPIE Int. Soc. Opt. Eng. (2003). https://doi.org/10.1117/12.506669
17. Dogra, S., Dorai, A., Dorai, K.: Implementation of the quantum Fourier transform on a hybrid qubit-qutrit NMR quantum emulator. Int. J. Quantum Inf. **13**, 1550059 (2015). https://doi.org/10.1142/s0219749915500598
18. Dorai, K., Suter, D.: Efficient implementations of the quantum Fourier transform: an experimental perspective. arXiv Quantum Physics (2002) https://doi.org/10.1142/S0219749905000967
19. Ferry, D.K., Akis, R., Gilbert, M.J., Knezevic, I.: In: Kish, L.B., Green, F., Iannaccone, G., Vig, J.R. (eds.) Noise and Information in Nanoelectronics, Sensors, and Standards, vol. 5115, pp. 271–280. SPIE (2003). International Society for Optics and Photonics. https://doi.org/10.1117/12.488887
20. Freedman, M.H., Wang, Z.: Large quantum Fourier transforms are never exactly realized by braiding conformal blocks. Phys. Rev. A (2006). https://doi.org/10.1103/PhysRevA.75.032322
21. Govindankutty, A., Srinivasan, S.K., Mathure, N.: Rotational abstractions for verification of quantum Fourier transform circuits. IET Quantum Commun. **4**(2), 84–92 (2023). https://doi.org/10.1049/qtc2.12055
22. Grigoryan, A., Agaian, S.: New look on quantum representation of images: Fourier transform representation. Quantum Inf. Process. (2020). https://doi.org/10.1007/s11128-020-02643-3
23. Grigoryan, A.M., Gomez, A.A., Agaian, S.S.: Methods of calculation of the 2-D quantum Fourier transform of images. In: Agaian, S.S., Asari, V.K., DelMarco, S.P. (eds.) Multimodal Image Exploitation

and Learning 2024, vol. 13033, p. 130330. SPIE, National Harbor, Maryland, United States (2024). International Society for Optics and Photonics. https://doi.org/10.1117/12.3013940
24. Grillo, S.A., De Lima Marquezino, F.: Fourier 1-norm and quantum speed-up. Quantum Inf. Process. **18**(4), 1–15 (2019). https://doi.org/10.1007/s11128-019-2208-7
25. Harrow, A., Hassidim, A., Lloyd, S.: Quantum algorithm for linear systems of equations. Phys. Rev. Lett. (2009). https://doi.org/10.1103/PhysRevLett.103.150502
26. Haque, E., Paul, M., Tohidi, F., Ulhaq, A.: An overview of quantum circuit design focusing on compression and representation. Electronics (2025) https://www.mdpi.com/2079-9292/14/1/72
27. IEEE Life: ENIAC: The World's First Computer. Accessed: 2024-10-01 (2023). https://life.ieee.org/eniac-the-worlds-first-computer/
28. Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C.J., Lishman, J., Gacon, J., Martiel, S., Nation, P.D., Bishop, L.S., Cross, A.W., Johnson, B.R., Gambetta, J.M.: Quantum computing with Qiskit (2024). arxiv:2405.08810
29. Kahanamoku-Meyer, G., Blue, J., Bergamaschi, T., Gidney, C., Chuang, I.: A log-depth in-place quantum Fourier transform that rarely needs ancillas. (2025). arxiv:2505.00701
30. Karafyllidis, I.G.: Quantum computer simulator based on the circuit model of quantum computation. IEEE Trans. Circuits Syst. I Regul. Pap. **52**(8), 1590–1596 (2005). https://doi.org/10.1109/TCSI.2005.851999
31. Karam, M., Khazaal, H., Aglan, H., Cole, C.: Noise removal in speech processing using spectral subtraction. J Signal Inf Process (2014). https://doi.org/10.4236/jsip.2014.52006
32. Karim, S.A.A., Kamarudin, M.H., Karim, B.A., Hasan, M.K., Sulaiman, J.: Wavelet transform and fast Fourier transform for signal compression: a comparative study. In: 2011 International Conference on Electronic Devices, Systems and Applications (ICEDSA), pp. 280–285 (2011). https://doi.org/10.1109/ICEDSA.2011.5959031
33. Kawano, Y., Sekigawa, H.: Quantum Fourier transform over symmetric groups—improved result. J. Symb. Comput. **75**, 219–243 (2016). https://doi.org/10.1016/j.jsc.2015.11.016. (**Special issue on the conference ISSAC 2014: Symbolic computation and computer algebra**)
34. Liang, Z., Zhao, Y.: Number theoretic transform and its applications in lattice-based cryptosystems: a survey (2022). arxiv:2211.13546
35. Liu, B., Ortiz, M., Cirak, F.: Towards quantum computational mechanics. Comput. Methods Appl. Mech. Eng. **432**, 117403 (2024). https://doi.org/10.1016/j.cma.2024.117403
36. Liu, X., Jing, L., Wang, L., Wang, M.: Quantum Fourier transform simulation on Sunway Taihulight. In: 2020 15th International Conference on Computer Science & Education (ICCSE), pp. 833–837 (2020). https://doi.org/10.1109/ICCSE49874.2020.9201643
37. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum principal component analysis. Nat. Phys. **10**, 631–633 (2014). https://doi.org/10.1038/nphys3029
38. Mahmud, N., Haase-Divine, B., Kuhnke, A., Rai, A., MacGillivray, A., El-Araby, E.: Efficient computation techniques and hardware architectures for unitary transformations in support of quantum algorithm emulation. J. Signal Process. Syst. **92**(9), 1017–1037 (2020). https://doi.org/10.1007/s11265-020-01569-4
39. Markidis, S.: What is quantum parallelism, anyhow? ISC HPC 2024 conference (2024) https://doi.org/10.48550/arXiv.2405.07222
40. Marquezino, F.L., Portugal, R., Sasse, F.D.: J. Comput. Appl. Math. **235**(1), 74–81 (2010). https://doi.org/10.1016/j.cam.2010.05.012
41. Martin, A., Lamata, L., Solano, E., Sanz, M.: Digital-analog quantum algorithm for the quantum Fourier transform. Phys. Rev. Res (2020). https://doi.org/10.1103/PhysRevResearch.2.013012
42. Massey, P., Clark, J.A., Stepney, S.: Human-competitive evolution of quantum computing artefacts by genetic programming. Evol. Comput. (2006). https://doi.org/10.1162/evco.2006.14.1.21
43. Mistry, R., Ortega, J.: Experiments in quantum frequency detection using quantum Fourier transform, pp. 275–295 (2022). https://doi.org/10.1007/978-3-031-13909-3_11
44. Mohammadbagherpoor, H., Oh, Y.-H., Dreher, P., Singh, A., Yu, X., Rindos, A.J.: An improved implementation approach for quantum phase estimation on quantum computers. In: 2019 IEEE International Conference on Rebooting Computing (ICRC), pp. 1–9 (2019). https://doi.org/10.1109/ICRC.2019.8914702
45. Moore, C., Russell, A., Schulman, L.J.: The symmetric group defies strong Fourier sampling. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05) (2007). https://doi.org/10.1109/SFCS.2005.73

46. Mukhamedieva, D.T., Sobirov, R.A., Turgunova, N.M., Samijonov, B.N.: Quantum Fourier transform in image processing. In: Information Technologies and Intelligent Decision Making Systems (2024). https://doi.org/10.1007/978-3-031-60318-1_12
47. Nagy, M., Akl, S.G.: Coping with decoherence: Parallelizing the quantum Fourier transform. In: Proceedings of the ISCA 19th International Conference on Parallel and Distributed Computing Systems, September 20-11, 2006, pp. 108–113. ISCA, San Francisco, California, USA (2006). https://doi.org/10.1142/s012962641000017x
48. Nam, Y., Su, Y., Maslov, D.: Approximate quantum Fourier transform with $o(n \log(n))$ t gates. NPJ Quantum Inf **6**(1), 1–6 (2020). https://doi.org/10.1038/s41534-020-0257-5
49. Neumann, N.M.P., Houte, R., Attema, T.: Imperfect distributed quantum phase estimation. Computat. Sci. ICCS **2020**(12142), 605–615 (2020). https://doi.org/10.1007/978-3-030-50433-5_46
50. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, Cambridge (2010) anniversary edition edn., Cambridge University Press, Cambridge (2010)
51. NTi Audio: Fast Fourier Transform (FFT). https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft. Accessed: 30 Sep 2024 (2024)
52. Ouedrhiri, O., Banouar, O., Hadaj, S.E., Raghay, S.: Quantum phase estimation based algorithms for machine learning. In: 2021 2nd International Informatics and Software Engineering Conference (IISEC), pp. 1–6 (2021). https://doi.org/10.1109/IISEC54230.2021.9672406
53. Pang, C., Zhou, R., Hu, B., Hu, W., El-Rafei, A.: Signal and image compression using quantum discrete cosine transform. Inf. Sci. **473**, 121–141 (2019)
54. Paula Neto, F.M., Ludermir, T.B., Oliveira, W.R., Silva, A.J.: Implementing any nonlinear quantum neuron. IEEE Trans. Neural Netw. Learn. Syst. **31**(9), 3741–3746 (2020). https://doi.org/10.1109/TNNLS.2019.2938899
55. Pereira, F.R.F., Guedes, E.B., Assis, F.M.d.: Simulating the quantum Fourier transform. In: 2013 2nd Workshop-School on Theoretical Computer Science, pp. 40–44 (2013). https://doi.org/10.1109/WEIT.2013.10
56. Rai, V.K., Mohanty, A.R.: Bearing fault diagnosis using FFT of intrinsic mode functions in Hilbert–Huang transform. Mech. Syst. Signal Process. **21**(6), 2607–2615 (2007). https://doi.org/10.1016/j.ymssp.2006.12.004
57. Rocca, P., Tosi, L., Anselmi, N., Massa, A.: On the use of quantum Fourier transform for array antenna analysis. In: 2022 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI), pp. 699–700 (2022). https://doi.org/10.1109/AP-S/USNC-URSI47032.2022.9886119
58. Rodriguez, S.: Quantum fourier transform (qft) over galois fields. In: Donkor, E., Pirich, A.R., Brandt, H.E. (eds.) Quantum Information and Computation XI, p. 87490. SPIE, Baltimore, Maryland, United States. International Society for Optics and Photonics (2013). https://doi.org/10.1117/12.2015194
59. Savran, I., Demirci, M., Yilmaz, A.H.: Accelerating Shor's factorization algorithm on gpus1. Can. J. Phys. **96**(7), 759–761 (2018). https://doi.org/10.1139/cjp-2017-0768
60. Schönhage, A., Strassen, V.: Schnelle multiplikation großer zahlen. Computing **7**, 281–292 (1971)
61. Sharma, A., Uehara, G., Narayanaswamy, V., Miller, L., Spanias, A.: Signal analysis-synthesis using the quantum fourier transform. In: ICASSP 2023—2023 International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2023). https://doi.org/10.1109/ICASSP49357.2023.10096524
62. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997). https://doi.org/10.1137/S0097539795293172
63. Solar, M., Cisterna, F., Villacura, J.P., Dombrovskaia, L.: A review on quantum machine learning and quantum cryptography. Memoria Investigaciones en Ingeniería **27**, 180–199 (2024). https://doi.org/10.36561/ing.27.12
64. Solar, M., Villacura, J.P., Cisterna, F., Dombrovskaia, L.: Detecting a Spy with quantum cryptography. Memoria Investigaciones en Ingeniería **27**, 200–219 (2024). https://doi.org/10.36561/ing.27.13
65. Somsuk, K.: The improved algorithm to limit scope for recovering private key in multi-prime RSA by utilizing quantum Fourier transform. Cogent Eng. **11**(1), 2301149 (2024). https://doi.org/10.1080/23311916.2023.2301149
66. Song, P.H., Shepelyansky, D.L.: Quantum computing of quantum chaos and imperfection effects. Phys. Rev. Lett. **86**, 2162–2165 (2001). https://doi.org/10.1103/PhysRevLett.86.2162
67. Steijl, R., Barakos, G.N.: Parallel evaluation of quantum algorithms for computational fluid dynamics. Comput Fluids **173**, 22–28 (2018). https://doi.org/10.1016/j.compfluid.2018.03.080

68. Sun, D., Zhang, N., Franchetti, F.: Optimization and performance analysis of Shor's algorithm in Qiskit. In: 2023 IEEE High Performance Extreme Computing Conference (HPEC) (2024). https://doi.org/10.1109/HPEC58863.2023.10363522
69. Tomita, A., Nakamura, K.: Measured quantum Fourier transform of 1024 qubits on fiber optics. Int. J. Quantum Inf. **2**, 119–131 (2004). https://doi.org/10.1142/S0219749904000122
70. Volya, D., Mishra, P.: Special session: Impact of noise on quantum algorithms in noisy intermediate-scale quantum systems. In: 2020 IEEE 38th International Conference on Computer Design (ICCD), pp. 1–4 (2020). https://doi.org/10.1109/ICCD50377.2020.00013
71. Vorobyov, V., Zaiser, S., Abt, N., Meinel, J., Dasari, D., Neumann, P., Wrachtrup, J.: Quantum Fourier transform for nanoscale quantum sensing. NPJ Quantum Inf. (2021) https://doi.org/10.1038/s41534-021-00463-6
72. Weinstein, Y.S., Pravia, M.A., Fortunato, E.M., Lloyd, S., Cory, D.G.: Implementation of the quantum Fourier transform. Phys. Rev. Lett. **86**, 1889–1891 (2001). https://doi.org/10.1103/PhysRevLett.86.1889
73. Yimsiriwattana, A.: Distributed quantum computing: A distributed Shor algorithm. Proc. SPIE Int. Soc. Opt. Eng. (2004). https://doi.org/10.1117/12.546504
74. Yung, M.-H., Benjamin, S.C., Bose, S.: Processor core model for quantum computing. Phys. Rev. Lett. **96**, 220501 (2006). https://doi.org/10.1103/PhysRevLett.96.220501
75. Zilic, Z., Radecka, K.: Scaling and better approximating quantum Fourier transform by higher radices. IEEE Trans. Comput. **56**, 202–207 (2007). https://doi.org/10.1109/TC.2007.35