Implementation and Evaluation of Exact and Approximate Quantum Fourier Transforms Using NumPy

Anastasios Karakoulakis
KTH Royal Institute of Technology
Stockholm, Sweden
karako@kth.se

I. Introduction

Quantum computing has been evolving from a theoretical concept into a promising technological frontier, where steady progress is being done in hardware and algorithms. Major technology companies (Google, IBM) report that they have created devices with over 100 physical qubits [1]. At the same time an expanding ecosystem of startups and labs is advancing software tooling, error correction, and algorithms. This growing demand for quantum computing arises from its potential to solve classically hard problems in areas such as cryptography, optimization, chemistry, and machine learning. Yet practical fault tolerance remains out of reach due to limited coherence, gate infidelity, and broader noise. As a result, much of research on quantum computing emphasizes on how to simulate, optimize, and approximate circuits under tight resource budgets.

Modern frameworks like Qiskit [2] and Cirq [3] provide high-performance simulators that make quantum circuit prototyping accessible and scalable. However, these platforms often abstract away the low-level dynamics of quantum operations. In contrast, lightweight simulators built in `NumPy` [4] expose intermediate amplitudes and phases, offering clearer insight into gate effects and state evolution. Such transparency makes them valuable for education, debugging, and analyzing fundamental quantum algorithms.

In this work, we present `QSimLite`, a lightweight statevector simulator built in `NumPy` for implementing and analyzing the Quantum Fourier Transform (QFT). The simulator is validated by comparing its gates and QFT results with Qiskit's reference implementation, confirming identical statevectors. We then study the runtime scaling of the QFT and show that the approximate version reduces computational cost by omitting small controlled rotations with minimal loss of fidelity. This design provides an interpretable, efficient framework for exploring and teaching core quantum algorithms.

*All work in this report was completed independently by Anastasio Karakoulakis. Generative AI was used solely for grammar correction, phrasing refinement, and formatting assistance. The implementation, simulations, and analysis were carried out entirely by the author. The expected final course grade is B, reflecting completion of all bonus exercises and adherence to the assignment and formatting guidelines.

II. Background

We consider an n-qubit quantum system represented by a complex statevector $|\Psi\rangle \in \mathbb{C}^{2^n}$, defined as

$$|\Psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle , \tag{1}$$

where $\alpha_x \in \mathbb{C}$ denotes the complex amplitude associated with the computational basis state $|x\rangle$ labeled by binary digits $b_{n-1} \cdots b_0$. The normalization condition $\sum_x |\alpha_x|^2 = 1$ ensures valid quantum probabilities.

A single-qubit unitary gate $G_t \in \mathbb{C}^{2\times 2}$ acting on qubit t is embedded in the n-qubit Hilbert space through the Kronecker product

$$U = I_2^{\otimes(n-t-1)} \otimes G_t \otimes I_2^{\otimes t}, \qquad |\Psi'\rangle = U |\Psi\rangle , \tag{2}$$

where I_2 denotes the 2×2 identity matrix. This operation transforms the global statevector by locally applying G_t to the selected qubit, while leaving the others unchanged.

A. Fundamental Gates

Quantum circuit operations are built from a universal set of single and multi qubit gates, each represented by a unitary matrix acting on the global statevector. These gates define the fundamental transformations of quantum information and form the basis for constructing higher-level algorithms.

a) Single-qubit gates

The elementary Pauli and Hadamard operations are given by

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad H = \tfrac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{3}$$

which implement bit-flip (X), phase-flip (Z), and superposition (H) operations.

b) Two-qubit gates

Entangling operations are realized through controlled gates such as the Controlled-NOT (CNOT), Controlled-Phase

(CPHASE), and SWAP:

$$U_{\mathrm{CNOT}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

$$\mathrm{U}_{\mathrm{CPHASE}}(\phi) = \mathrm{diag}(1,\, 1,\, 1,\, e^{i\phi}),$$

$$U_{\mathrm{SWAP}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

c) Three-qubit gate

The Toffoli (controlled-controlled-NOT) gate extends the control NOT concept to three qubits, flipping the target only when both control qubits are in the $|1\rangle$ state:

$$U_{\mathrm{TOFF}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

Each gate acts unitarily on the composite system and is embedded in the n-qubit Hilbert space through tensor products, forming the core primitives of our statevector simulator.

B. Quantum Fourier Transform

The Quantum Fourier Transform (QFT) is a unitary operation that performs the discrete Fourier transform on the amplitudes of a quantum state. For an n-qubit register of dimension $N = 2^n$, the transformation is defined as

$$\mathrm{QFT}_N \,|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i x y/N} \,|y\rangle \,, \tag{4}$$

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

The QFT is a unitary transformation that preserves normalization and satisfies $\mathrm{QFT}_N^{-1} = \mathrm{QFT}_N^{\dagger}$. In circuit form, it is implemented as a sequence of Hadamard and controlled-phase gates, followed by a bit-reversal operation to restore qubit order. As a reference we used, the QFT circuit of Qiskit, where the case for the four-qubit system is shown in Fig. 1.

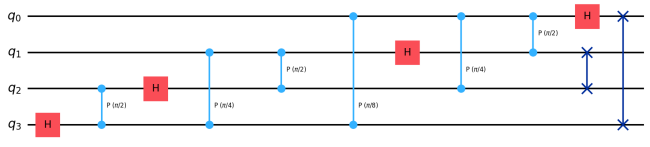

Fig. 1. Quantum Fourier Transform (QFT) circuit for four qubits generated with Qiskit.

C. Approximate QFT (AQFT)

The Approximate QFT [5] omits small rotations with angle $2\pi/2^k$ for $k > m$, keeping only terms where $j + k \geq n - m$. This reduces the number of two-qubit gates from $O(n^2)$ to $O(nm)$, while the phase error remains bounded by $O(2^{-m})$.

D. Fidelity

To quantify how closely two quantum states match, a good metric is the fidelity:

$$F(\Psi_{\mathrm{ideal}}, \Psi_{\mathrm{actual}}) = |\,\langle \Psi_{\mathrm{ideal}} \rangle\, \Psi_{\mathrm{actual}}|^2,$$

A fidelity of $F = 1$ indicates identical states up to a global phase.

III. Methodology

We follow a simulation-based analysis of the Quantum Fourier Transform (QFT) and its approximate variant (AQFT) to evaluate their performance and accuracy. The quality of the approximation is quantified using the fidelity metric, which measures the similarity between the AQFT output state and the corresponding full QFT state.

A. Overview of Approach

A quantum circuit is represented as a product of unitary operations applied sequentially to an initial state,

$$|\Psi'\rangle = U_k U_{k-1} \cdots U_1 \,|\Psi_0\rangle \,. \tag{5}$$

Using the fundamental gates defined earlier, the QFT and its truncated form (AQFT) are implemented through combinations of Hadamard and controlled-phase operations. The truncation level specifies the number of phase rotations retained: lower values produce approximations with fewer gates, whereas higher values approach the exact QFT, defining the trade-off between computational cost and accuracy.

B. Experimental Procedure

Two main experiments are conducted to assess performance and accuracy:

1) **Runtime scaling of the QFT**
   This experiment investigates how the simulation time scales with the number of qubits n.
   a) Prepare initial quantum states for multiple system sizes.
   b) Apply the full QFT to each state.
   c) Record the total runtime for every case.
   d) Analyze the dependence of runtime on n to identify computational scaling behavior.
2) **Fidelity of the AQFT vs. truncation level**
   This experiment examines how phase truncation affects the accuracy of the Approximate QFT (AQFT).
   a) Select representative input states such as square-wave, Gaussian, and alternating patterns.
   b) Apply both the full QFT and AQFT to each state.
   c) Vary the truncation level m, which controls the omission of small-angle phase rotations.

d) Compute the fidelity between QFT and AQFT outputs,

$$F = |\langle \Psi_{\text{QFT}} \rangle \Psi_{\text{AQFT}}|^2,$$

to quantify accuracy loss.

e) Examine how fidelity changes with m and system size n.

Performance is assessed using two metrics, runtime which reflects the computational cost with increasing qubit count n, and fidelity, indicating how closely the approximate state matches the exact one under different truncation levels.

IV. Implementation

The simulator was implemented incrementally, beginning with the definition of all fundamental quantum gates as their corresponding unitary matrices. Each gate operation was validated through direct matrix vector application on multi-qubit states to ensure consistency with the theoretical formalism described earlier. Functional correctness was verified by comparing all gate and circuit transformations against the *Qiskit* Statevector simulator, used as a reference model. Tests were performed on both randomly generated and structured input states across multiple qubit sizes, confirming the numerical accuracy and stability of the implementation.

A. QFT and Approximate QFT Algorithms

The Quantum Fourier Transform (QFT) is implemented using Hadamard and controlled-phase gates followed by SWAP operations for bit reversal. The Approximate QFT (AQFT) follows the same structure but omits controlled rotations below a specified *truncation level*, reducing computational cost while retaining high fidelity.

Algorithm 1 Quantum Fourier Transform (QFT)

Require: State vector ψ
  $n \leftarrow \log_2(\text{length of } \psi)$
  $\psi' \leftarrow \psi$
  for $j = n-1$ **downto** 0 **do**
    $\psi' \leftarrow$ apply_h(ψ', j)
    for $k = j-1$ **downto** 0 **do**
      $\theta \leftarrow \pi/2^{(j-k)}$
      $\psi' \leftarrow$ apply_cphase(ψ', k, j, θ)
    end for
  end for
  return ψ' =0

Algorithm 2 Approximate Quantum Fourier Transform (Approx-QFT)

Require: State vector ψ, truncation level t
  $n \leftarrow \log_2(\text{length of } \psi)$
  $\psi' \leftarrow \psi$
  Initialize counters: $H \leftarrow 0$, $CPHASE \leftarrow 0$
  for $j = n-1$ **downto** 0 **do**
    $\psi' \leftarrow$ apply_h(ψ', j)
    $H \leftarrow H + 1$
    for $k = j-1$ **downto** 0 **do**
      if $t > 0$ **and** $(j-k) > t$ **then**
        continue
      end if
      $\theta \leftarrow \pi/2^{(j-k)}$
      $\psi' \leftarrow$ apply_cphase(ψ', k, j, θ)
      $CPHASE \leftarrow CPHASE + 1$
    end for
  end for
  return $(\psi', \{H, CPHASE\})$ =0

B. Experimental Setup

All simulations were carried out in Python using `NumPy` for matrix operations and `Matplotlib` for visualization. Two experiments were conducted: the first evaluated QFT runtime as a function of the number of qubits n, using identical initial states for consistency, while the second analyzed the fidelity between QFT and AQFT outputs across truncation levels m. For the fidelity study, input states included Gaussian and square-wave patterns (Figs. 2 and 3).

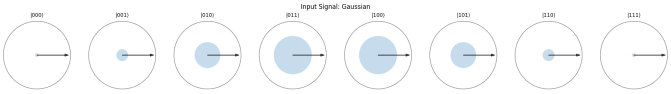

Fig. 2. Example of a Gaussian initial state for $n = 3$ qubits.

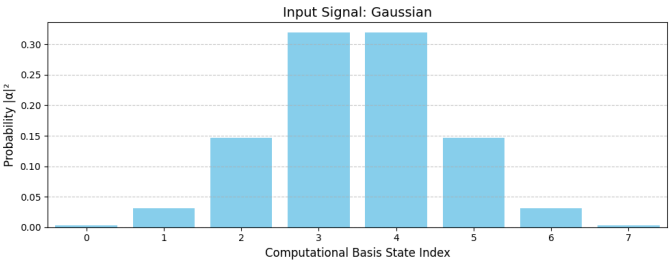

Fig. 3. Amplitude distribution of the Gaussian initial state.

V. Results

In this section we present the performance and accuracy results obtained from the QFT and AQFT simulations.

Figure 4 shows the QFT runtime as a function of qubit count (n). The results follow the expected $\mathcal{O}(n2^n)$ scaling of state-vector simulation, with manageable runtimes for $n \leq 10$ and rapid growth for larger systems due to the exponential Hilbert space size. Each point represents the mean of 50 runs to reduce processor noise and ensure reproducible measurements.

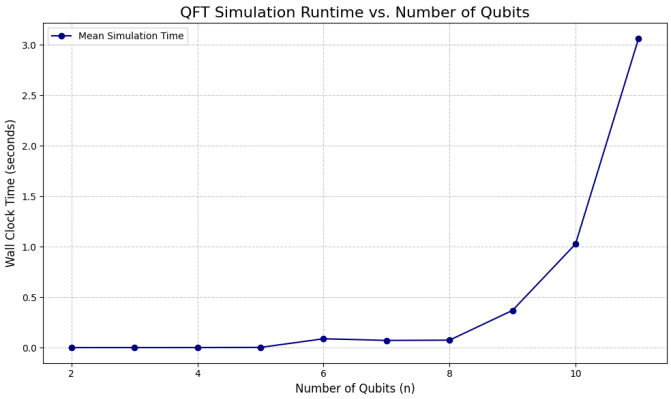

Fig. 4. Mean QFT simulation runtime as a function of the number of qubits n.

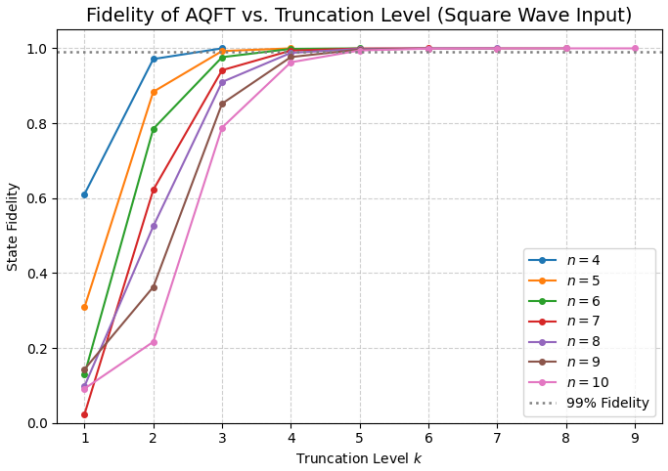

Fig. 5. Fidelity of AQFT vs Truncation Level for square wave input

Figure 5 shows the fidelity of the Approximate Quantum Fourier Transform (AQFT) as a function of the truncation level (k) for input states derived from a square-wave signal in the computational basis. As observed, the state fidelity increases monotonically with k and saturates near unity for $k \geq 3$ across all tested system sizes. For systems with a larger number of qubits, the fidelity decreases more rapidly at low truncation levels, reflecting their higher sensitivity to omitted phase rotations. Overall, the results demonstrate that many small-angle rotations can be discarded without significant loss of information, allowing for substantial computational savings with minimal impact on accuracy.

Figure 6 presents the fidelity of the Approximate Quantum Fourier Transform (AQFT) as a function of the truncation level (k) for input states corresponding to a Gaussian-shaped amplitude distribution. Compared to the square-wave case, the AQFT performs slightly better under this smoother input, achieving high fidelity even at lower truncation levels. This behavior can be attributed to the fact that, in Gaussian states, information is more localized across the qubits, qubits farther from the most significant positions carry less amplitude weight

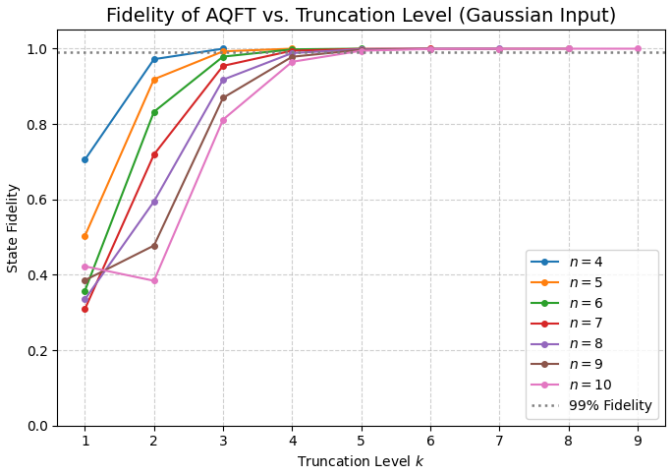

Fig. 6. Fidelity of AQFT vs Truncation Level for gaussian wave input

and therefore contribute less to the overall phase structure. Consequently, omitting small angle rotations associated with these qubits has a smaller impact on the final state fidelity. Although this effect depends on the exact initialization of the state amplitudes, the results indicate that the AQFT maintains strong robustness across different input profiles, demonstrating its practicality as a resource-efficient alternative to the full QFT.

VI. Conclusion

In this work, we developed a NumPy-based simulator for the Quantum Fourier Transform (QFT) and its approximate variant (AQFT), using Qiskit implementations as a reference for correctness. The simulator enables controlled experimentation with circuit size and truncation level, providing both performance and fidelity insights. The results confirm the expected exponential runtime scaling of the QFT and demonstrate that the AQFT can achieve near-exact fidelity while discarding small-angle rotations, significantly reducing computational cost. Overall, the developed simulator offers a lightweight, transparent, and extensible framework for studying quantum algorithms and their classical approximations.

References

[1] H. Neven. (2024) Meet willow: Our state-of-the-art quantum chip. [Online]. Available: https://blog.google/technology/research/google-willow-quantum-chip/

[2] I. Quantum. (2025) Ibm quantum documentation. [Online]. Available: https://quantum.cloud.ibm.com/docs/en

[3] G. Q. AI. (2025) Cirq: A python framework for quantum computing. [Online]. Available: https://quantumai.google/cirq

[4] T. N. Developers. (2025) Numpy documentation. [Online]. Available: https://numpy.org/doc/

[5] D. Coppersmith, "An approximate fourier transform useful in quantum factoring," 1994.