PAPER

Linear-depth quantum circuits for loading Fourier approximations of arbitrary functions

To cite this article: Mudassir Moosa *et al* 2024 *Quantum Sci. Technol.* **9** 015002

View the article online for updates and enhancements.

Quantum Science and Technology

CrossMark

RECEIVED
28 March 2023

REVISED
10 September 2023

ACCEPTED FOR PUBLICATION
22 September 2023

PUBLISHED
11 October 2023

PAPER

Linear-depth quantum circuits for loading Fourier approximations of arbitrary functions

Mudassir Moosa[1,2,5,*], **Thomas W Watts**[3,5,*], **Yiyou Chen**[3,4], **Abhijat Sarma**[1] and **Peter L McMahon**[3,*]

[1] Department of Physics, Cornell University, Ithaca, NY 14853, United States of America
[2] Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, United States of America
[3] School of Applied and Engineering Physics, Cornell University, Ithaca, NY 14853, United States of America
[4] Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544, United States of America
[5] These authors contributed equally to this work.
[*] Authors to whom any correspondence should be addressed.

E-mail: mudassir@purdue.edu, tww55@cornell.edu and pmcmahon@cornell.edu

Keywords: quantum computing, trapped ion quantum computer, quantum state preparation, Gaussian state preparation, quantum algorithm, quantum Fourier transform

Abstract

The ability to efficiently load functions on quantum computers with high fidelity is essential for many quantum algorithms, including those for solving partial differential equations and Monte Carlo estimation. In this work, we introduce the Fourier series loader (FSL) method for preparing quantum states that exactly encode multi-dimensional Fourier series using linear-depth quantum circuits. Specifically, the FSL method prepares a (Dn)-qubit state encoding the 2^{Dn}-point uniform discretization of a D-dimensional function specified by a D-dimensional Fourier series. A free parameter, m, which must be less than n, determines the number of Fourier coefficients, $2^{D(m+1)}$, used to represent the function. The FSL method uses a quantum circuit of depth at most $2(n-2)+\lceil\log_2(n-m)\rceil+2^{D(m+1)+2}-2D(m+1)$, which is linear in the number of Fourier coefficients, and linear in the number of qubits (Dn) despite the fact that the loaded function's discretization is over exponentially many (2^{Dn}) points. The FSL circuit consists of at most $Dn+2^{D(m+1)+1}-1$ single-qubit and $Dn(n+1)/2+2^{D(m+1)+1}-3D(m+1)-2$ two-qubit gates; we present a classical compilation algorithm with runtime $O(2^{3D(m+1)})$ to determine the FSL circuit for a given Fourier series. The FSL method allows for the highly accurate loading of complex-valued functions that are well-approximated by a Fourier series with finitely many terms. We report results from noiseless quantum circuit simulations, illustrating the capability of the FSL method to load various continuous 1D functions, and a discontinuous 1D function, on 20 qubits with infidelities of less than 10^{-6} and 10^{-3}, respectively. We also demonstrate the practicality of the FSL method for near-term quantum computers by presenting experiments performed on the Quantinuum H1-1 and H1-2 trapped-ion quantum computers: we loaded a complex-valued function on 3 qubits with a fidelity of over 95%, as well as various 1D real-valued functions on up to 6 qubits with classical fidelities ≈99%, and a 2D function on 10 qubits with a classical fidelity ≈94%.

1. Introduction

Efficiently loading classical data on quantum computers is an important ingredient in many quantum algorithms. For example, quantum-linear-solver algorithms require an efficient preparation of a state $|b\rangle$ in order to compute the solution $|x\rangle \propto A^{-1}|b\rangle$ to a large linear system A [1–4]. In general, exactly loading 2^n dimensional data into a state of n qubits requires a quantum circuit of $O(2^n)$ quantum operations [5–8]. Exponential space and time complexities for the read-in component of a quantum algorithm will compromise any potential for an exponential speed-up over comparable classical algorithms [9]. Hence, finding an efficient method for accurately loading classical input data into quantum computers is a problem with vast applications.

In many practical applications, classical input data takes the form of values of a function or a distribution on a uniform discretized grid (or mesh) [10–19]. It is well-known that a large class of functions and distributions can be accurately approximated by a Fourier series of only a few terms. In this work, we exploit this fact in order to load a truncated Fourier-series approximation to a general multivariate target function $f:[0,1]^D\to\mathbb{C}$. More precisely, given a truncated Fourier series consisting of $2^{D(m+1)}$ Fourier modes, $f_{(m)}(x_1,\dots,x_D)$, that approximates the target function of D variables, $f(x_1,\dots,x_D)$, we present a method to exactly prepare a quantum state of Dn qubits of the form $|f_{(m)}\rangle=\sum_{k_1=0}^{2^n-1}\cdots\sum_{k_D=0}^{2^n-1}f_{(m)}$ $(k_1/2^n,\dots,k_D/2^n)|k_1\rangle\otimes\cdots\otimes|k_D\rangle$. For single variable functions (i.e. $D=1$), the target state becomes $|f_{(m)}\rangle=\sum_{k=0}^{2^n-1}f_{(m)}(k/2^n)|k\rangle$. This allows us to load the target function with practically arbitrarily high fidelity so long as a sufficient number of Fourier modes are used. We refer to our method as the Fourier series loader (FSL).

The FSL method consists of three main steps: First, we find the dominant 2^{m+1} Fourier coefficients for a given function. This can be done classically with a time complexity of $O(m2^m)$ using sparse Fourier transform algorithms [20–23]. The second step is to load the $2^{m+1}\ll 2^n$ Fourier coefficients into a sparse quantum state of n qubits. We achieve this using a quantum circuit of depth $O(\log(n)+2^m)$, which can be found in $O(8^m)$ classical runtime, and consists of $O(n+2^m)$ and $O(2^m)$ two-qubit and single-qubit gates, respectively. This is an improvement over a generic sparse state preparation algorithm that sports a classical runtime of $O(mn4^m)$ and produces a circuit of depth $O(n2^m)$ which consists of $O(n2^m)$ two-qubit gates and $O(m2^m+n)$ single-qubit gates [24]. Finally, the third step of the FSL method is to apply the inverse quantum Fourier transform (QFT) to generate a 'real-space' representation of the desired function from the loaded Fourier coefficients. The full quantum circuit implementing our method is shown in figure 1(a) and has a depth of $O(n+2^m)$ and consists of $O(n^2+2^m)$ two-qubit and $O(n+2^m)$ single-qubit gates. The only error introduced by the FSL method is due to the approximation of the given function by a truncated Fourier series. This *truncation error* is determined by the number of Fourier modes which dictates the requisite value of m. In particular, the infidelity between the target state and the state prepared by the FSL method decays exponentially with m.

It is worth mentioning that the FSL method is analogous to a state preparation method based on Fourier interpolation [25, 26]. The interpolation-based method loads the values of the target function at 2^{m+1} discretized points and then interpolates to 2^n points using the QFT. The FSL method, as discussed above, instead loads 2^{m+1} Fourier coefficients and then produces a Fourier series representation of the target function using the QFT. Despite their similarities, the FSL method has various advantages over the interpolation-based method including, but not limited to, superior accuracy and reduced gate count.

The FSL has several attractive features that make it appealing for loading classical functions on quantum computers. First, we have complete control over the accuracy of our results as it is determined by the number of Fourier coefficients used to represent the target function. Secondly, no time-consuming classical optimization of a parameterized quantum circuit or matrix product state (MPS) is needed unlike other comparable methods for function loading in the literature [25, 27–29]. In fact, all the angles of rotation in our quantum circuits can be determined from the Fourier coefficients of the function of interest as we explain in section 2. Thirdly, the FSL works equally well for loading real-valued and complex-valued functions. More remarkably, the FSL can be easily generalized to load a D-dimensional Fourier series into a quantum state of Dn qubits using a quantum circuit of $O(n)$ depth and $O(Dn^2)$ quantum gates. Finally, due to the low depth of the circuit in figure 1(a), the FSL method is capable of loading functions on near-term noisy quantum computers as we demonstrate in section 3.

The FSL method is particularly useful for quantum algorithms for solving differential equations that require the preparation of states encoding initial conditions [10–14]. Other contexts in which the FSL method is useful include quantum image processing [30], Monte Carlo methods based on quantum amplitude estimation algorithms [17, 18], and computing generalized inner products [19]. These last two contexts find applications in the domain of option pricing and risk analysis [17–19]. Though this is not the goal of the present work, we elaborate in the appendix F on how the FSL method can be utilized to load an image into a quantum state.

The rest of this paper is organized as follows: In section 2, we present the details of the FSL method and the accompanying quantum circuit implementations. Starting with the simplest use cases, we demonstrate how our FSL method can be used to load periodic functions of a single variable. Then we discuss how the FSL method can be easily modified to load non-periodic, piece-wise discontinuous, and multivariate functions. For each case, we demonstrate high fidelity loading of various functions using the FSL method by quantum simulations performed using Qiskit [31]. Next, in section 3, we discuss several experiments of

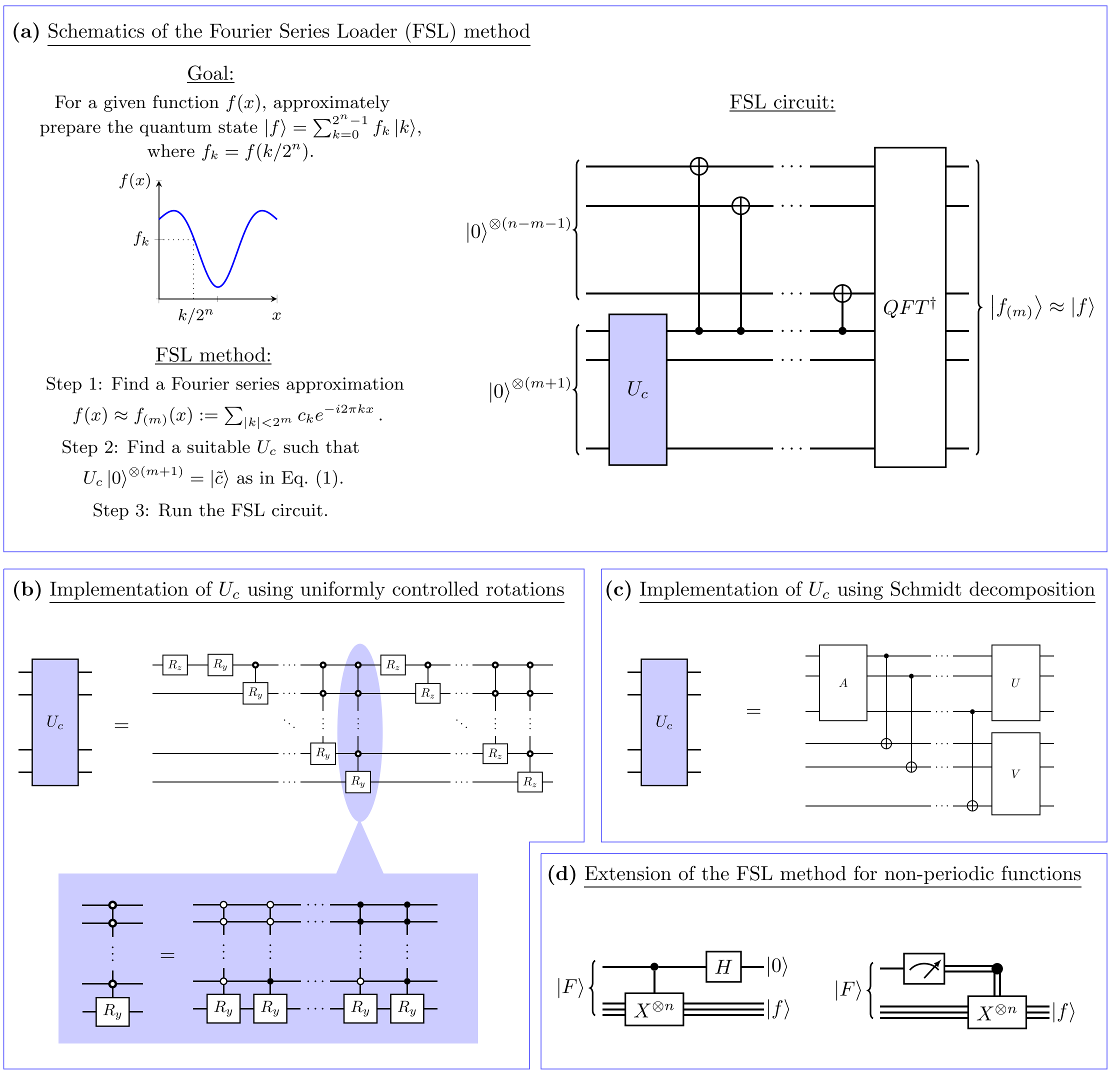
(a) Schematics of the Fourier Series Loader (FSL) method
Goal:
For a given function $f(x)$, approximately prepare the quantum state $|f\rangle = \sum_{k=0}^{2^n-1} f_k |k\rangle$, where $f_k = f(k/2^n)$.
$f(x)$
f_k
$k/2^n$
x
FSL method:
Step 1: Find a Fourier series approximation $f(x) \approx f_{(m)}(x) := \sum_{|k|<2^m} c_k e^{-i2\pi kx}$.
Step 2: Find a suitable U_c such that $U_c |0\rangle^{\otimes(m+1)} = |\tilde{c}\rangle$ as in Eq. (1).
Step 3: Run the FSL circuit.
FSL circuit:
$|0\rangle^{\otimes(n-m-1)}$
$|0\rangle^{\otimes(m+1)}$
U_c
$QFT^\dagger$
$|f_{(m)}\rangle \approx |f\rangle$
(b) Implementation of U_c using uniformly controlled rotations
U_c
=
R_z
R_y
(c) Implementation of U_c using Schmidt decomposition
U_c
=
A
U
V
(d) Extension of the FSL method for non-periodic functions
$|F\rangle$
H
$|0\rangle$
$X^{\otimes n}$
$|f\rangle$

Figure 1. (a) The Fourier series loader (FSL) method loads a Fourier series approximation of the target function into a quantum state of n qubits. Implementing the FSL method requires approximating the target function with a Fourier series of 2^{m+1} terms and finding a unitary circuit U_c to load the 2^{m+1} Fourier coefficients on $m+1$ qubits, both of which can be done on a classical computer. The integer m should be chosen such that the truncation error of the Fourier series, and hence, the infidelity of the prepared state is within a desired threshold. Once a suitable unitary circuit U_c is found, the Fourier series approximation of the target function is loaded on n qubits by running the FSL circuit on a quantum computer. The FSL circuit consists of three parts. First, the unitary U_c loads 2^{m+1} Fourier coefficients in $m+1$ qubits and prepares the state given in equation (1). Then, a cascade of CNOT gates entangles the remaining $n-m-1$ qubits with the $m+1$ qubits storing the Fourier coefficients which efficiently implements the technique of zero padding (see equation (2)). Finally, the inverse QFT at the end of the circuit transforms the loaded Fourier coefficients into a state encoding the Fourier series approximation of the target function. (b) One possible implementation of the unitary U_c is using a cascade of uniformly controlled rotations, which can be used to prepare an arbitrary quantum state. A uniformly k-controlled rotation operator, shown in the shaded region, decomposes as a series of controlled rotations conditioned on each computational basis state of k control qubits. (c) Another possible implementation of the unitary U_c is using a circuit that can prepare an arbitrary quantum state once the Schmidt decomposition of the target state is computed. The unitary A in the circuit loads the Schmidt coefficients whereas unitaries U and V rotate the computational basis into the Schmidt basis. (d) The FSL method loads a non-periodic function f by first loading a periodic extension F on n qubits and 1 ancilla qubit. Then the desired state is prepared either through disentangling the ancilla qubit (left) or through measurement-based operations (right).

loading functions of one and two variables on the Quantinuum H1-1 and H1-2 quantum computers [32]. The results of these experiments demonstrate that the FSL method can load functions on up to 10 qubits on a present-day noisy quantum computer. Finally, we conclude with a comparison of the FSL method with other state preparation methods for loading functions to quantum states and some possible future directions in section 4.

2. Methods

Our goal is to load the values of any arbitrary complex-valued function to the amplitudes of a quantum state of n qubits. For example, given a one-dimensional function $f\colon [0,1] \to \mathbb{C}$, we want to prepare an n-qubit state

$|f\rangle = \sum_{k=0}^{2^n-1} f_k |k\rangle$, where $|k\rangle$ is the kth computational basis state and $f_k = f(k/2^n)$ is the value of the function at kth grid point. Without loss of generality, we assume that the function f is such that the corresponding quantum state representation $|f\rangle$ is normalized.

We begin by considering the simplest case of a function f given by a finite Fourier series of the form $f(x) = \sum_{k=-M}^{M} c_k e^{-i2\pi kx}$, where $M := 2^m - 1 < 2^{n-1}$. We claim that the quantum state $|f\rangle$ corresponding to this Fourier series can be prepared exactly using the quantum circuit shown in figure 1(a) in which U_c denotes an arbitrary implementation of the unitary that maps $|0\rangle^{\otimes(m+1)}$ state to $|\tilde{c}\rangle$, where

$$|\tilde{c}\rangle := 2^{n/2} \sum_{k=0}^{M} c_k |k\rangle + 2^{n/2} \sum_{k=1}^{M} c_{-k} |2^{m+1} - k\rangle \,. \tag{1}$$

The cascade of $n-m-1$ CNOT gates that follow U_c entangle all n qubits, mapping $|0\rangle^{\otimes(n-m-1)} \otimes |\tilde{c}\rangle$ to an n-qubit state $|c\rangle$, where

$$|c\rangle := 2^{n/2} \sum_{k=0}^{M} c_k |k\rangle + 2^{n/2} \sum_{k=1}^{M} c_{-k} |2^{n} - k\rangle \,. \tag{2}$$

Finally, the inverse QFT maps $|c\rangle$ to the desired state $|f\rangle$. A more detailed derivation is provided in the appendix A.

The QFT and its inverse can be implemented with a quantum circuit of depth $O(n)$ and with $O(n^2)$ quantum gates. The cascade of CNOT gates in figure 1(a) can be implemented in $O(\log(n))$ depth. In the worst case, the implementation of $(m+1)$-qubit unitary U_c will require $O(2^m)$ quantum gates and a circuit of depth $O(2^m)$. However, note that m is entirely fixed by the number of Fourier modes in our function and does not scale with the total number of qubits, n. Hence, for a Fourier series with a fixed number of terms, the circuit in figure 1(a) has $O(n)$ depth and has $O(n^2)$ quantum gates.

The problem of preparing the n-qubit state $|f\rangle$ has now reduced to the problem of preparing a $(m+1)$-qubit state $|\tilde{c}\rangle$. When the target Fourier series consists of only a few terms, the construction of an efficient circuit that prepares $|\tilde{c}\rangle$ is usually straightforward. However, when loading more complex functions that require many Fourier coefficients, we propose two general methods for constructing the state $|\tilde{c}\rangle$.

2.1. Implementations of U_c

Our first proposal is to prepare the state $|\tilde{c}\rangle$ using a cascade of uniformly controlled rotations as shown in figure 1(b), which can be used to prepare any arbitrary quantum state as was shown by Möttönen *et al* [7]. Möttönen *et al* proposed this circuit as an extension of a circuit of Zalka, Grover, and Rudolph (ZGR) [5, 6] which only involves a cascade of uniformly controlled R_y gates, and hence, can only prepare states with positive, real-valued amplitudes. Moreover, Möttönen *et al* made two interesting observations that render this circuit a practical choice for implementing U_c. First, this circuit can be constructed using 2^{m+2} single-qubit rotations and 2^{m+2} CNOT gates. Secondly, there exists a set of formulae for calculating the angles of rotation for all single-qubit gates in the circuit when provided access to the amplitudes of the target state $|\tilde{c}\rangle$ [7]. These formulae can be evaluated on a classical computer in $O(8^m)$ time. (See appendix B for more details.)

Our second proposal is based on the Schmidt decomposition. We can treat $|\tilde{c}\rangle$ as the overall quantum state of a bipartite system by partitioning $m+1$ qubits into $(m+1)/2$ and $(m+1)/2$ qubits if m is odd or into $m/2$ and $m/2+1$ qubits if m is even. Then, we can classically compute the Schmidt decomposition of $|\tilde{c}\rangle$ and express it as $|\tilde{c}\rangle = U \otimes V \sum_{k=0}^{r} \alpha_k |k\rangle \otimes |k\rangle$, where α_k are the Schmidt coefficients (singular values), U and V are unitary operators, and $r = 2^{(m+1)/2}$ ($r = 2^{m/2}$) if m is odd (even). Given this decomposition, we can prepare the state $|\tilde{c}\rangle$ using the circuit shown in figure 1(c) [33]. The gate A in figure 1(c) is taken to be the uniformly controlled rotations that encode the Schmidt coefficients in the state $|\alpha\rangle = \sum_{k=0}^{r} \alpha_k |k\rangle$, and the gates U and V are generated using methods for unitary synthesis methods such as those described in [34]. The classical computational cost to implement this method grows exponentially in m, which is due to the cost of finding the Schmidt decomposition of the target state $|\tilde{c}\rangle$ and the cost of computing the gate decompositions of unitaries A, U, and V.

These two methods complement each other nicely. On the one hand, the first method has a lower classical pre-processing cost than the second method. On the other hand, the first method leads to a quantum circuit with a higher gate count than the second method for generic states. Since a lower gate count is more desirable when working with noisy quantum computers, we used the second method for most of the experiments we performed on real-world quantum computers as we discuss in section 3. However, we used the first method when performing quantum simulations which we discuss in the next subsection.

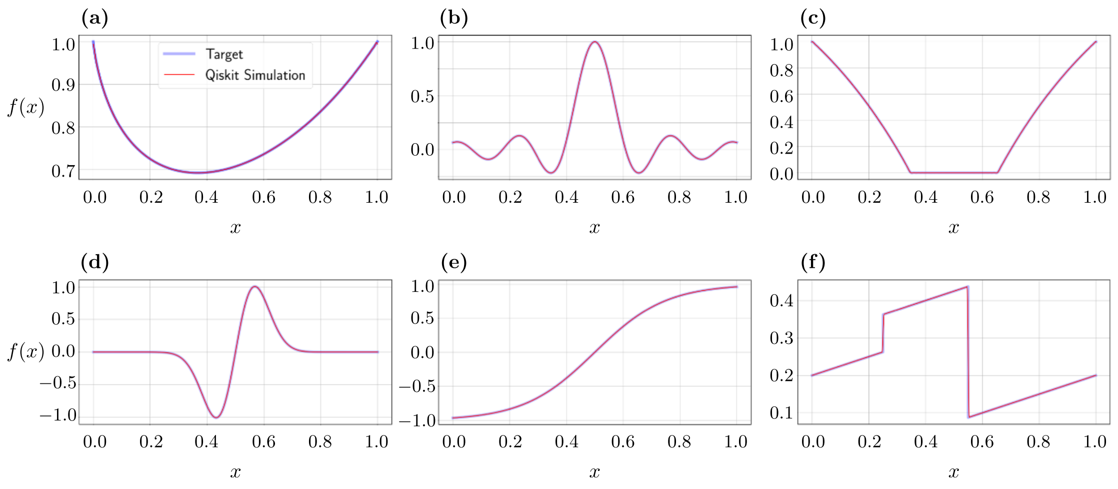

Figure 2. Quantum circuit simulations of loading each of the following 1D functions into a quantum state of $n = 20$ qubits using the FSL method: (a) x^x, (b) sinc, (c) reflected put option, (d) first excited state of a quantum oscillator, (e) hyperbolic tangent, and (f) a piece-wise discontinuous function. Continuous functions in (a)–(e) were approximated with 128 Fourier coefficients (i.e. $m = 6$) whereas the piece-wise discontinuous function in (f) was approximated with 512 Fourier coefficients (i.e. $m = 8$). These functions were loaded with infidelity of less than (a) 10^{-6}, (b) 10^{-7}, (c) 10^{-6}, (d) 10^{-12}, (e) 10^{-8}, and (f) 10^{-3}. All of the simulations are performed using the 'statevector_simulator' provided by Qiskit [31] and the Fourier coefficients were loaded using the uniformly controlled rotations shown in figure 1(b). The results are presented up to normalization.

Finally, even though the asymptotic scaling of the computational cost of finding U_c using the aforementioned two methods is exponential in m, we find that the computational time to find implement U_c is reasonable in practice. For example, for $m = 10$ (i.e. around 2000 Fourier coefficients), it takes less than 6 and 3 seconds to find U_c using the first and the second method respectively. More details about the benchmarking of the CPU wall clock time can be found in appendix C.

2.2. Function loading

Generally, only a few terms in the truncated Fourier series are needed to well-approximate any periodic function f. Now, given a Fourier series approximation $f_{(m)} = \sum_{k=-M}^{M} c_k e^{-i2\pi kx}$ of a periodic function f, we can use the FSL method to prepare the state $|f_{(m)}\rangle \approx |f\rangle$. The only source of error between the prepared state $|f_{(m)}\rangle$ and the target state $|f\rangle$ is the truncation error from the Fourier series approximation. The error as measured by infidelity, $\epsilon_{(m)} := 1 - |\langle f|f_{(m)}\rangle|^2$, decays exponentially with m in the limit of large n as we show in the appendix D. Moreover, the rate of exponential decay of infidelity depends on the smoothness of the function f. The bounds on infidelity derived in the appendix D can be used to determine the number of Fourier modes needed to prepare the state $|f\rangle$ with a specified infidelity.

In order to approximate a function by a Fourier series, we first need to find its Fourier coefficients, which leads to additional classical pre-processing costs. The fast Fourier transform (FFT) has a computational cost of $O(n2^n)$, which takes away any hope of exponential advantage. Fortunately, there exists various sparse Fourier transform algorithms that can be used to find $M \ll 2^n$ dominant Fourier coefficients efficiently [20]. In particular, algorithms developed in [21–23] compute the M non-zero coefficients in time $O(M\log M)$. However, for our implementations of the FSL method, we used the conventional FFT.

To demonstrate the usefulness of the FSL method, we performed the quantum simulation of loading various periodic functions into a 20 qubit state using the 'statevector_simulator' provided by Qiskit [31]. For concreteness, we chose $m = 6$ (i.e. 128 Fourier modes) and loaded the Fourier coefficients using the uniformly controlled rotations shown in figure 1(b). We were able to load non-trivial functions such as x^x, a sinc function, the reflected put option function of [35], and the wavefunction of the first excited state of a quantum harmonic oscillator with an infidelity of less than 10^{-6}, 10^{-7}, 10^{-6}, and 10^{-12}, respectively. The results of the simulation are shown in figures 2(a)–(d).

The FSL method can be modified to load non-periodic functions as well. Given a non-periodic function f on the domain $[0,1]$, we can define a function F on the extended domain $[0,2]$ such that $F(x) = f(x)/\sqrt{2}$ for $0 \leqslant x \leqslant 1$ and $F(x) = f(2-x)/\sqrt{2}$ for $1 \leqslant x \leqslant 2$. By construction, F is periodic on the domain $[0,2]$, and hence, can be loaded with high fidelity into a quantum state of $(n+1)$ qubits using the FSL method. We introduce an ancilla qubit so that there will be twice as many grid points on the extended domain which ensures that the grid spacing remains $1/2^n$. The state $|f\rangle$ of $n+1$ qubits is of the form

$$|f\rangle = \frac{1}{\sqrt{2}}|0\rangle \otimes \sum_{k=0}^{2^n-1} f_k|k\rangle + \frac{1}{\sqrt{2}}|1\rangle \otimes \sum_{k=0}^{2^n-1} f_{2^n-1-k}|k\rangle \,. \tag{3}$$

With the observation that $X^{\otimes n}|k\rangle = |2^n - 1 - k\rangle$, where X is the Pauli-X operator, we find that the state $|f\rangle$ can be written as

$$|f\rangle = \frac{1}{\sqrt{2}}|0\rangle \otimes |f\rangle + \frac{1}{\sqrt{2}}|1\rangle \otimes X^{\otimes n}|f\rangle \,. \tag{4}$$

Once we prepare $|f\rangle$ using the FSL, we measure the ancilla qubit in the computational basis. We do nothing to the n qubits if the measurement output is 0, but we apply X to each of the n qubits if the measurement output is 1. In either case, the state of the n qubits is $|f\rangle$ as desired. Alternatively, we can prepare the state $|f\rangle$ and apply CNOT gates on each of the n qubits and a Hadamard gate on the ancilla qubit. This disentangles the ancilla qubit from the other n qubits and maps $|f\rangle$ to $|0\rangle \otimes |f\rangle$. The circuits implementing both methods are provided in figure 1(d).

In order to demonstrate the FSL's ability to load non-periodic functions, we performed the simulation of loading a hyperbolic tangent function into a 20 qubit state. Once again, we chose $m = 6$ and used the uniformly controlled rotations to load the Fourier coefficients. We were able to load the hyperbolic tangent with infidelity of less than 10^{-8} and the result of this simulation is shown in figure 2(e).

Generalizing further, Fourier series approximations of piece-wise discontinuous functions typically exhibit large oscillations at the discontinuities resulting in overshooting; this is known as the Gibbs phenomenon. Depending on the application, these large oscillations may not be desirable. One way to suppress these oscillations is to apply Fourier filters, such as a Lanczos σ-factor, to the Fourier coefficients: $c_k \to (\mathrm{sinc}(\pi k/M))^a c_k$, where M is the order of the partial Fourier series and $a \in \mathbb{R}^+$ [36]. The Lanczos filter is just one of many Fourier filters available and is by no means the best choice for every discontinuous function of interest [37–39]. The FSL method can easily incorporate these Fourier filters with a small additional classical pre-processing cost. Instead of directly loading in the Fourier coefficients using the unitary U_c in figure 1(a), one can load the filtered Fourier coefficients. This ability of the FSL method to suppress the Gibbs phenomenon gives it an edge over other QFT-based state preparation methods [25, 29]. (See section 4 for more details.) As an example, we performed the quantum simulation of loading a piece-wise discontinuous shown in figure 2(f) into a 20 qubit state. We found that with $m = 8$, we were able to prepare the desired state with an infidelity of around 10^{-3}.

Remarkably, the FSL method can be generalized to functions of more than one variable. The approach is the same: we first find a multi-variable Fourier series approximation of the target function, prepare a sparse state of Fourier coefficients using either the uniformly controlled rotation or the Schmidt-decomposition circuit, and then apply inverse QFT operators. Further details, including the circuit diagrams for loading functions of two variables, are presented in the appendix A. Here, we simply demonstrate the practicality of the FSL method by presenting the simulation results of loading a two-dimensional sinc function into a 20 qubits state, 10 qubits for each dimension. We approximated this two-dimensional sinc function with 256 Fourier coefficients, which we loaded to 8 qubits using the cascade of uniformly controlled rotations method for implementing U_c. With only 256 Fourier modes, we were able to load the sinc function with infidelity of 5.0×10^{-4}. The results of the simulation performed using Qiskit's 'statevector_simulator' are presented in figure 3.

We now conclude this introduction to the mechanisms of the FSL method. In the next section, we present the experimental results from running the FSL method on the Quantinuum H1-1 and H1-2 quantum computers. Our results indicate that the FSL method is able to load functions with high fidelity even in the presence of noise in near-term quantum computers.

3. Experimental results

In the previous section, we demonstrated that the FSL method is capable of preparing function-encoding states with high fidelity in an ideal noiseless setting. However, present-day quantum computers are very noisy and exhibit complex errors when running highly structured quantum programs [40]. To test how the FSL method performs in the presence of noise, we performed experiments on the Quantinuum H1-1 and H1-2 quantum computers which we accessed remotely via an Open QASM-based API [32]. System Model H1 quantum computers have, on average, single-qubit gate infidelities of 4×10^{-5}, two-qubit gate infidelities of 3×10^{-3}, state preparation and measurement (SPAM) error of 3×10^{-3}, and a measurement cross-talk error of 2×10^{-5}. More impressively, System Model H1 quantum computers support full connectivity between qubits and allows for mid-circuit measurements, the latter of which we used extensively as we discuss below.

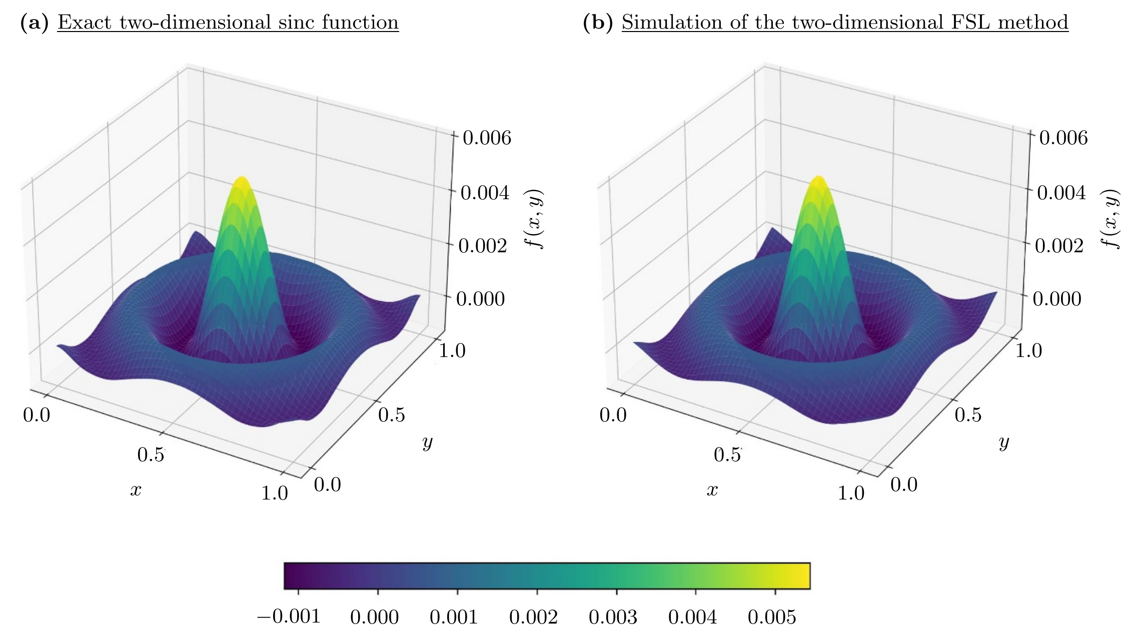

Figure 3. Quantum circuit simulation demonstrating the FSL method being used to load a 2D function. Specifically, a two-dimensional sinc function was loaded into a quantum state of 20 qubits (i.e. $n = 10$) with infidelity of 5.0×10^{-4}. In this simulation, 256 Fourier coefficients (i.e. $m = 3$) were loaded using the uniformly controlled rotations shown in figure 1(b). This simulation was performed using the 'statevector_simulator' provided by Qiskit [31].

More details about the specifications for the System Model H1 quantum computers, H1-1 and H1-2, are provided in the appendix E.

For our first experiment, we considered a complex-valued function $f(x) = (\cos(2\pi x) - 1.5i \cos(6\pi x))/\sqrt{13}$. We loaded this function into a quantum state of three qubits using the quantum circuit shown in figure 12 in the appendix E. To check how close the prepared state is to the target state, we performed the state tomography where we measured each qubit in the eigenbasis of Pauli-Z, Pauli-X, and Pauli-Y operators by applying $\{I, H, R_x(\pi/2)\}$ on each qubit before measuring it. From the results of 800 measurements for each of the 27 experiments, we estimated the probabilities of projection, P_{μ_1,μ_2,μ_3}, onto states $|\psi_{\mu_1,\mu_2,\mu_3}\rangle \equiv |\psi_{\mu_1}\rangle \otimes |\psi_{\mu_2}\rangle \otimes |\psi_{\mu_3}\rangle$, where $\mu_i \in \{0,1,2,3\}$ and $|\psi_0\rangle = |0\rangle$, $|\psi_1\rangle = |1\rangle$, $|\psi_2\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$, and $|\psi_3\rangle = (|0\rangle + i|1\rangle)/\sqrt{2}$. We assumed the density matrix to be of the form $\rho(T) = \left(T^\dagger T\right)/\mathrm{tr}\left(T^\dagger T\right)$, where T is a 8×8 lower triangular matrix with real-valued diagonal elements. The matrix $\rho(T)$ which best fits the measured probabilities of projection onto states $|\psi_{\mu_1,\mu_2,\mu_3}\rangle$ was determined by minimizing the cost function, [41, 42]

$$C(T) = \sum_{\mu_1=0}^{3}\sum_{\mu_2=0}^{3}\sum_{\mu_3=0}^{3} \frac{\left(\langle\psi_{\mu_1,\mu_2,\mu_3}|\rho(T)|\psi_{\mu_1,\mu_2,\mu_3}\rangle - P_{\mu_1,\mu_2,\mu_3}\right)^2}{\langle\psi_{\mu_1,\mu_2,\mu_3}|\rho(T)|\psi_{\mu_1,\mu_2,\mu_3}\rangle}, \tag{5}$$

with respect to T. The reconstructed state, ρ_{tomo} through this tomography method has a fidelity of 95.5% with the target state, $\rho_{\text{target}} = |f\rangle\langle f|$, and a purity of $\mathrm{tr}(\rho_{\text{tomo}}^2) = 0.918$. The matrix elements of the target density matrix, ρ_{target}, and the measured density matrix, ρ_{tomo}, are shown in figure 4(a). As evident from this figure, the matrix elements of ρ_{target} are in good agreement with those of ρ_{tomo}. Moreover, the comparison of the target function with the 'measured' function is shown in figure 4(b). Since the measured density matrix ρ_{tomo} is not a pure state, the notion of the measured function is not uniquely defined. For concreteness, we defined the measured function such that its values at 8 discretized points are the elements of the state $|f_{\text{tomo}}\rangle$ in the computational basis, where $|f_{\text{tomo}}\rangle$ is the eigenvector of ρ_{tomo} corresponding to the dominant eigenvalue [43].

The above approach, based on the maximum likelihood estimation, is especially useful as it restricts the resulting density matrix, $\rho(T)$, to being Hermitian, positive, and normalized. In the appendix E, we discuss an alternate 'direct reconstruction' approach where we measured the density matrix without imposing any constraints. When using the 'direct reconstruction' of the density matrix, we calculated a fidelity of around 94.0% between the measured and the target state. However, as is usually the case with such unconstrained approaches [41], the reconstructed matrix has negative eigenvalues, and hence, is not a valid density matrix.

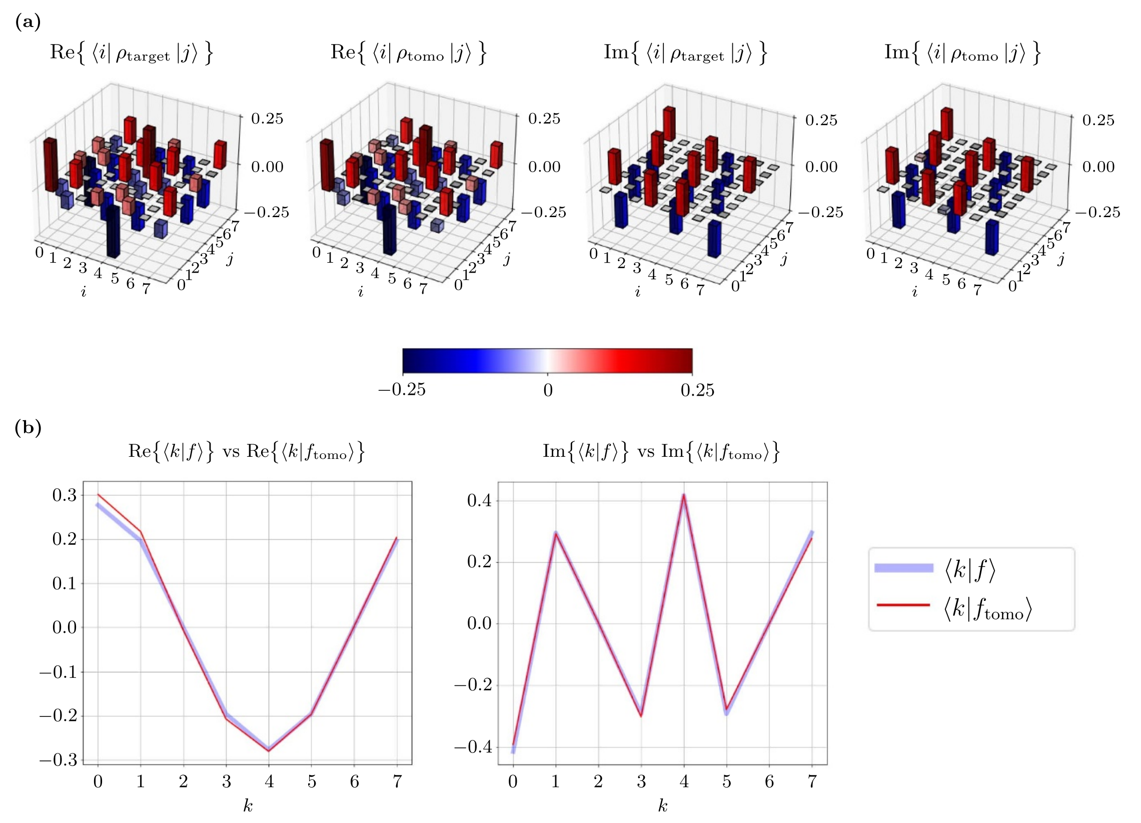

Figure 4. Experimental results of loading a complex-valued function, $f(x) = (\cos(2\pi x) - 1.5i\cos(6\pi x))/\sqrt{13}$, into a three qubit state on the Quantinuum H1-1 quantum computer using the quantum circuit shown in figure 12. (a) The loaded quantum state was reconstructed through quantum state tomography based on maximum likelihood estimation, and has a fidelity of 95.5% with the target state and a purity of 0.918. (b) Comparison of the elements of the target function state with the eigenvector of ρ_{tomo} corresponding to the dominant eigenvalue. As evident from these plots, the reconstructed state, ρ_{tomo}, is in good agreement with the target state, $\rho_{\text{target}} = |f\rangle\langle f|$. Hence, the FSL method can accurately load both the phases and the amplitudes of a complex-valued target state.

We now discuss experiments we performed to test the FSL method's potential for loading functions into a quantum state of five or more qubits. Note that due to the high cost of running a quantum computer, performing similar quantum state tomography experiments to verify the accuracy of the FSL method is not a viable option for more than three qubits. Even though there are economical algorithms to perform state tomography using randomized measurements [44, 45] or machine learning techniques [46–49], they are beyond the scope of this work.

To circumvent the issue of the verification of the prepared quantum state, we simplified our task in the rest of the experiments and only verified the amplitudes of the prepared quantum state by performing measurements in the computational basis. In particular, we first loaded $\sqrt{f(x)}$ for a given function $f(x)$ into a quantum state $|\sqrt{f}\rangle = \sum_{k=0}^{2^n-1}\sqrt{f_k}|k\rangle$ using the FSL method and then estimated the measurement probabilities, $|\langle k|\sqrt{f}\rangle|^2$, by measuring all the qubits in the computational basis sufficiently many times. This allowed us to compare the measurement probabilities with the value of the target function $f(x)$ at $x = k/2^n$. It is worth mentioning that even though we are unable to verify the accuracy of the local phases in the quantum state prepared using the FSL method, we still expect the experiments described above to test the performance of the FSL method on a present-day noisy quantum computer. Note that we already have enough evidence that the FSL method is, in principle, capable of loading correct amplitudes and local phases based on the simulation results in figures 2 and 3 and the proof of the FSL method provided in the appendix A. Therefore, the only thing that needs to be checked is if the quantum state prepared using the FSL circuit can survive the hardware noise present in noisy quantum computers. For example, if the depth of the FSL quantum circuit, despite being $O(n)$, is not small enough, the resultant noisy state will be far from the target state, in which case even the measured amplitudes will be far from the actual amplitudes. Thus, measuring amplitudes of the prepared state and comparing them with those of the target state provides a nice alternate to more expensive state tomography to check the practicality of the FSL method for present-day noisy quantum computers.

Before we proceed, let us also point out that since we are only comparing the amplitudes of the prepared state to the target function, it seems more reasonable to quantify the accuracy of the FSL method in terms of

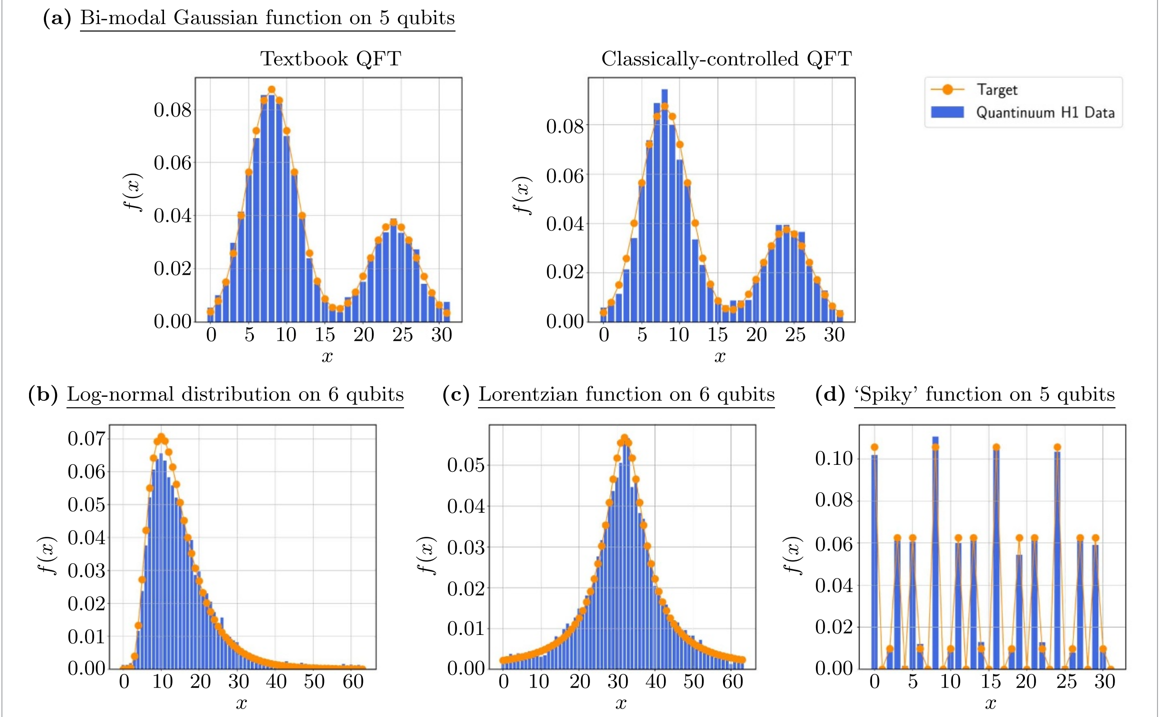

Figure 5. Experimental results of loading functions on the Quantinuum H1-1 and H1-2 quantum computers using the FSL method. The histograms (blue bars) represent the measurement probabilities (in the computational basis) determined from 5,000 measurements for (a) and (d) and 10,000 measurements for (b) and (c), whereas the orange dots represent the values of target functions. The classical fidelities between the empirical measurement probabilities and the target function are (a) more than 99.7%, (b) 98.8%, (c) 99.7%, and (d) 98.9%. The Schmidt-decomposition circuit shown in figure 1(c) was used to load the Fourier coefficients in experiments (a)–(c), whereas the quantum circuit in figure 13 was used to load the 'spiky' function in experiment (d).

the classical fidelity between the measurement probabilities (in the computational basis) and the target function. Recall that the classical fidelity between probability distributions p and q is defined as $F(p,q) = \left(\sum_i \sqrt{p_i}\sqrt{q_i}\right)^2$.

For our first experiment involving five or more qubits, we considered a bi-modal Gaussian function given by a superposition of two Gaussian functions; see equation (E1) in the appendix E. We loaded this function into a 5-qubit state and ran 5000 shots of measurements. The measurement probabilities are in good agreement with the target function as shown in figure 5(a). In fact, we obtained a classical fidelity of 99.8% between the measured probabilities and the target function. In this experiment, we used the Schmidt-decomposition circuit shown in figure 1(c) to load the Fourier coefficients. Further details about this circuit can be found in the appendix E.

When all the qubits are measured right after the application of the inverse QFT, as is the case in the above experiment, it is possible to replace all the controlled phase gates in the inverse QFT with classically-controlled single-qubit rotation gates [50]. Utilizing this classically-controlled implementation of the inverse QFT, we repeated the above bi-modal Gaussian function experiment and obtained a fidelity of 99.7% between the measured probabilities and the target function. The measurement probabilities for this experiment are also shown in figure 5(a).

As evident from the measured fidelities and the experimental results in figure 5(a), the FSL method works equally well with the 'textbook' implementation and the classically-controlled implementation of the inverse QFT. In other words, the effect of hardware noise is similar for both implementations of the inverse QFT. However, the implementation of the classically-controlled QFT was more economical as the experiment using the classically-controlled QFT required almost three times fewer compute credits than the experiment with the textbook QFT. For this reason, we used the classically-controlled QFT to perform the rest of the function-loading experiments that we discuss.

Next, we loaded a log-normal distribution and a Lorentzian function given in equations (E2) and (E3), respectively, to six qubits. From the results of 10,000 measurements, we obtained a classical fidelity of 98.8% for the log-normal distribution and a classical fidelity of 99.7% for the Lorentzian function. The measured probabilities for these experiments are shown in figures 5(b) and (c) respectively. For these experiments, we

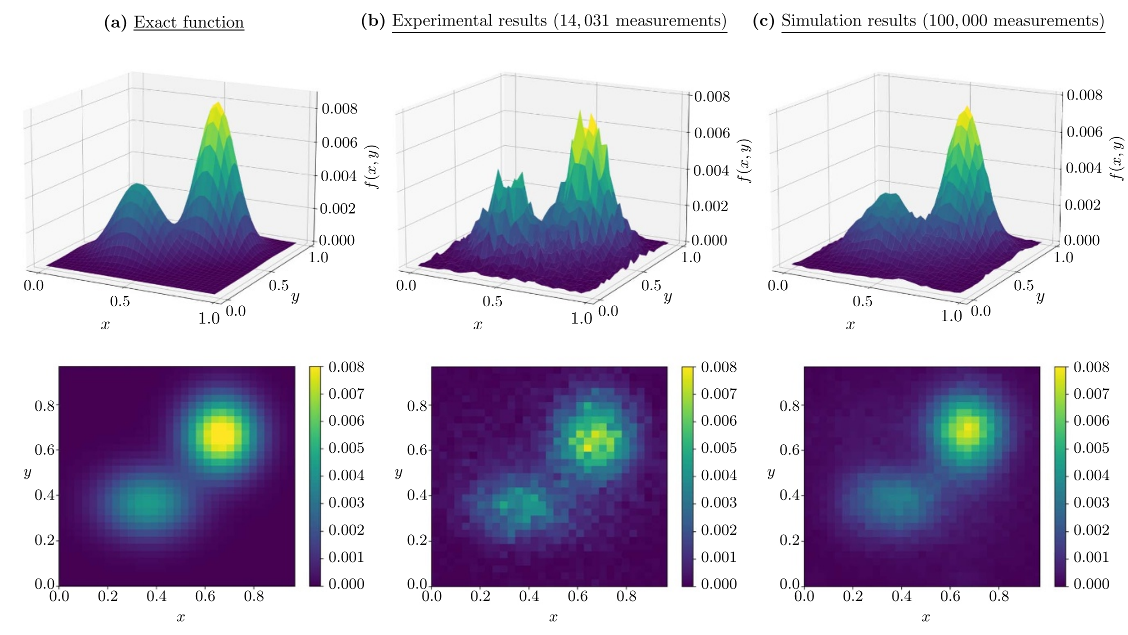

Figure 6. (a) The function $f(x,y) = \exp\left(-(x-\mu_1)^2/\sigma_{11}^2 - (y-\mu_1)^2/\sigma_{12}^2\right) + \lambda \exp\left(-(x-\mu_2)^2/\sigma_{21}^2 - (y-\mu_2)^2/\sigma_{22}^2\right)$ with $\mu_1 = 0.65$, $\mu_2 = 0.35$, $\sigma_{11} = \sigma_{22} = \sqrt{1/50}$, $\sigma_{12} = \sqrt{1/40}$, $\sigma_{21} = \sqrt{1/30}$, and $\lambda = 0.5$ we use to test the performance of the FSL method to load functions of two variables on a present-day noisy quantum computer. (b) Measurement probabilities are determined by first loading $\sqrt{f(x,y)}$ into a quantum state of 10 qubits of the Quantinuum H1-1 and H1-2 quantum computers and then measuring all the qubits in the computational basis. The Schmidt-decomposition circuit shown in figure 1(c) was used to load the Fourier coefficients of this function. Even with 14, 031 measurements, we observe that the general shape of the target function has emerged and we obtain a fidelity of 94.0% between the measured probabilities and the target function. (c) To see how the experimental results in (b) would improve with the number of measurements, we perform the simulation of 100 000 measurements on the Quantinuum H1-1 and H1-2 emulators whose noise models are calibrated to simulate the noise present in the Quantinuum H1-1 and H1-2 quantum computers. These results demonstrate that the FSL method can load multi-variable functions on up to 10 qubits on a near-term noisy quantum computer.

used the Schmidt-decomposition circuit shown in figure 1(c) to load the Fourier coefficients. Further details about these circuits are provided in the appendix E.

After testing the FSL method's ability to load smooth functions in the presence of noise, we next considered a 'spiky' function given by a superposition of low-frequency ($\omega = 4\pi$) and high-frequency ($\omega = 20\pi$) cosine waves; see equation (E4). We loaded this function into a 5-qubit state using the quantum circuit shown in figure 13 and performed 5000 measurements. From these measurements, we determined the measurement probabilities which are shown in figure 5(d) and obtained a classical fidelity of 98.9% between the measurement probabilities and the target function.

Encouraged by the success of the FSL method to load functions of a single variable on five and six qubits states, we wanted to verify how the FSL method performs when loading a function of two variables on 10 qubits of a noisy quantum computer. Specifically, we considered a function $f(x,y) = \exp(-(x-\mu_1)^2/\sigma_{11}^2 - (y-\mu_1)^2/\sigma_{12}^2) + \lambda \exp\left(-(x-\mu_2)^2/\sigma_{21}^2 - (y-\mu_2)^2/\sigma_{22}^2\right)$ with $\mu_1 = 0.65$, $\mu_2 = 0.35$, $\sigma_{11} = \sigma_{22} = \sqrt{1/50}$, $\sigma_{12} = \sqrt{1/40}$, $\sigma_{21} = \sqrt{1/30}$, and $\lambda = 0.5$. We then loaded $\sqrt{f(x,y)}$ into a 10-qubit state (five qubits for each variable). A graph of the function $f(x,y)$ is shown in figure 6(a). For concreteness, we used the Schmidt-decomposition circuit shown in figure 1(c) to load the Fourier coefficients of this function. Performing this experiment is expensive due to the large number of measurements required to convincingly compare the amplitudes of the prepared state with the target function, $f(x,y)$.

Due to limited compute resources, we only managed to perform 14, 031 measurements on the Quantinuum H1-1 and H1-2 quantum computers. Even with this small number of measurements, we observe the correct overall shape of the target function as shown in figure 6(b). Furthermore, we obtained a fidelity of 94.0% between the measured probabilities and the target function. We expect these results to improve further with the number of measurements. To justify this expectation, we performed a simulation of the experiment on the Quantinuum H1-1 and H1-2 emulators whose noise models and parameters match those of the Quantinuum H1-1 and H1-2 quantum computers [51]. The resulting probabilities determined from 100 000 simulated measurements are shown in figure 6(c) and have a fidelity of 94.1% with the target function.

The results presented in this section clearly demonstrate that the FSL method can load functions into a quantum state of up to 10 qubits on present-day noisy quantum computers. We now conclude with a comparison of the FSL method with previously known state preparation methods and some possible future directions.

4. Discussion

4.1. Comparison with previous work

Many function-loading algorithms have been proposed in the literature. In this section, we discuss a variety of algorithms that are comparable with the FSL method. A comparative summary of the algorithms we consider is provided in table 1.

We start by discussing other function-loading methods that also use the QFT and have a close resemblance with the FSL method. One such method is based on Fourier interpolation or trigonometric interpolation, which uses a Fourier transform to approximate the values of a function everywhere given the values of the function on a subset of points in its domain. A quantum circuit implementing this interpolation technique was proposed in [25, 26], and was generalized for loading images on a quantum computer in [52]. This circuit is similar to the circuit that we have proposed in figure 1(a) but with U_c replaced with $\mathrm{QFT}_{m'}U_f$, where U_f is any unitary that loads the values of the function f sampled at $2^{m'}$ equidistant grid points to an m'-qubit state, and $\mathrm{QFT}_{m'}$ is a QFT operator acting on m' qubits. Hence, the circuit implementing Fourier interpolation also has $O(n^2)$ gate count and $O(n)$ depth. However, despite their similarity, the FSL method has several advantages over the interpolation method. Firstly, depending on how the unitary U_f is implemented, the circuit implementing the Fourier interpolation may have a higher gate count than the FSL circuit in figure 1(a) due to an additional QFT operator. Secondly, it is known that the error associated with Fourier interpolation is always greater than the truncation error due to aliasing [53]. Therefore, the FSL method is capable of higher accuracy than an interpolation method. Thirdly, as we demonstrated in section 2, it is easy to incorporate Fourier filters when using the FSL method, which can suppress the Gibbs phenomenon present in the Fourier-series approximation of piece-wise discontinuous functions. It is not clear how to suppress the Gibbs phenomenon using the interpolation method. Finally, we do not expect the interpolation method to perform well with functions having sharp peaks as these peaks may not be captured by the values of the functions at some subset of points. Consider the 5-qubit spiky function state in figure 4(d) as an example. Since this spiky function has a Fourier mode with frequency $\omega = 10\pi$, the Nyquist theorem dictates that we need the value of the function at $2^{m'} \geqslant 20$ points to get an accurate result using the interpolation method. This implies that the interpolation method with $m' < n = 5$ cannot accurately load this spiky function into a 5-qubit state. The FSL method, on the other hand, can accurately load this function into a 5-qubit state as we have demonstrated.

The Fourier transform has also been used in [29] to load probability distributions as the amplitudes of a quantum state. In fact, the circuit proposed in [29] to load a distribution consists of a unitary operator acting on a small subset of qubits followed by a QFT operator acting on all of the qubits, and hence, is similar to the circuit that we have proposed in figure 1(a). However, there are several key differences. First, the unitary operator that precedes the QFT operator is taken to be a function of variational parameters which needs to be determined through classical optimization. Secondly, and more importantly, the subset of qubits on which the variational unitary acted on were not entangled with the rest of the qubits before the application of the QFT operator. In other words, there are no CNOT gates similar to those in figure 1(a). This has interesting consequences as it can be shown that this circuit can only load functions of the form $f(x) = \sum_{k=0}^{2^{m}-1} c_k e^{-i2\pi kx}$, where c_k depend on the variational parameters. This is only half of the Fourier series as it is missing the complex conjugate terms $c_{-k} = c_k^*$. Therefore, even though this circuit can be trained to sample a target probability distribution, it can not be used to load distributions as a subroutine of another quantum algorithm.

We now discuss other state preparation methods that are not based on Fourier transforms and compare them with the FSL method. Consider the variational algorithm based on the generative adversarial network (GAN) which involves a variational quantum circuit (generator) and a classical neural network (discriminator) [27]. The generator and the discriminator are simultaneously optimized until the discriminator cannot distinguish the samples from the generator with the given samples from the target distribution. The variational circuit that was used in [27] consists of L consecutive layers, each of depth $O(n)$ and containing $O(n)$ quantum gates. However, the computational cost of optimizing a variational quantum circuit scales at least linearly in the number of parameters p, which in this case is $O(nL)$. Despite the usefulness of a quantum GAN for state preparations, there are limitations. For example, by construction, a quantum GAN can only learn to load distributions and not arbitrary complex-valued functions into a quantum state. Moreover, the performance of a quantum GAN-based method highly depends on the choice of the initial state on which the variational circuit acts [27]. In general, it can take a significant amount of time to train a quantum GAN, further hindering the overall time complexity of this state preparation

Table 1. A comparison between various function-loading algorithms. The classical cost of the Fourier interpolation method is determined by the maximum frequency, denoted by $\omega_{\max}$, in the spectrum of the target function as per the Shannon–Nyquist theorem. In variational algorithms, such as Variational QFT and Quantum GAN, the number of parameters and the number of layers in a variational circuit are denoted by p and L respectively. The degree of the polynomial used to approximate the target function and the number of rounds of amplitude amplification needed in the QSVT method are denoted by d_{poly} and R respectively. In fully quantum algorithms, such as Hamiltonian simulation and the Kitaev–Webb method, the function/Hamiltonian/angles of rotation are stored using quantum registers. The number of qubits used to store this information is denoted by n'. Furthermore, T_{oracle} denotes the circuit depth of the oracle used in the implementation of the corresponding method. Finally, the 'NISQ-y' category classifies those methods whose circuits are relatively shallow [e.g. circuit depth of less than 1000 for states of less than 100 qubits] thereby allowing for the preparation of states with high fidelity (e.g. >90%) on noisy, intermediate-scale quantum (NISQ) devices.

Methods	Type of state	Requires classical optimization	Classical time complexity	Classical space complexity	Gate count	Circuit depth	Ancilla qubits	NISQ-y?
Fourier Interpolation [25]	Smooth functions	No	$O(\log\omega_{\max})$	$O(\log\omega_{\max})$	$O(n^2)$	$O(n)$	0 to 1	Yes
QFT Sampler [29]	Periodic distributions	Yes	$\Omega(p)$	$O(p)$	$O(n^2)$	$O(n)$	0	Yes
Quantum GAN [27]	Arbitrary distributions	Yes	$\Omega(p)$	$O(p)$	$O(nL)$	$O(nL)$	0	Yes
Matrix Product State [28]	Smooth functions	Yes	$O(n)$	$O(n)$	$O(n)$	$O(n)$	0	Yes
Variational ZGR [54]	Smooth functions	Yes	$O(2^n)$	$O(2^n)$	$\Omega(n^2)$	$\Omega(n)$	0	Yes
Hamiltonian Simulation [55]	Arbitrary functions	No	$O(1)$	$O(1)$	$O(T_{\text{oracle}})$	$O(T_{\text{oracle}})$	$O(n+n')$	No
QSVT [56]	Arbitrary functions	No	$O(d^3_{\text{poly}})$	$O(d_{\text{poly}})$	$O(nd_{\text{poly}}R)$	$O(nd_{\text{poly}}R)$	3 to 4	No
Kitaev–Webb [57]	Gaussian distribution	No	$O(1)$	$O(1)$	$O(\text{poly}(n))$	$O(\text{poly}(n))$	$\Omega(n')$	No
Discrete Random Walks [58]	Gaussian distribution	No	$O(1)$	$O(1)$	$O(n\log n)$	$O(n\log n)$	$O(n)$	Yes
Powers of Cosine Approximation [59]	Gaussian distribution	No	$O(1)$	$O(1)$	$O(n^2)$	$O(n)$	0	Yes
FSL (This work)	Arbitrary functions	No	$O(8^m)$	$O(2^m)$	$O(n^2)$	$O(n)$	0 to 1	Yes

method. Therefore, we expect that the FSL method, which does not suffer from these limitations, offers a good alternative for loading functions and distributions into a quantum state.

Another interesting state preparation method is based on MPSs [12, 28]. The main insight that was used in [28] was that smooth, differentiable, real-valued (SDR) functions can be well-approximated by piece-wise polynomials. This led to an approximation of SDR function states using MPS of bond dimension $K(p+1)$, where K is the number of subdomains and p is the order of the polynomials [60]. This MPS was further approximated by a MPS of bond dimension 2 using the MPS compression algorithm which has a time complexity of $O(n)$ [61]. The resulting compressed MPS was converted into a quantum circuit of $O(n)$ depth and gate count [62]. The linear depth circuit and gate count make this approach exceedingly efficient for loading SDR functions into a quantum state. However, the approximation of the SDR functions by piece-wise polynomials and the compression of the MPS lead to a build-up of error. Indeed, the fidelities reported in [28] are lower than the fidelities that we obtained in section 2 from the FSL method for which the only source of error is the truncation error in the Fourier series. Furthermore, the FSL method can load a larger class of functions than the MPS method. Specifically, the FSL method can load complex-valued and non-smooth functions.

In in section 2, we proposed a method for encoding the target Fourier coefficients via a unitary U_c where U_c is taken to be the circuit of ZGR which consists of a cascade of uniformly controlled rotations. Recently, another state preparation algorithm also based on the ZGR circuit was proposed in [54]. In fact, it was observed in [54] that for a class of positive integrable functions for which $|\partial_x^2 \log f^2(x)| \leqslant 8\pi$ on the domain $[0,1]$, most of the angles appearing in the ZGR circuits are close to each other. Hence, it is possible to replace uniformly k-controlled rotation gates with single rotation gates for all $k > k_0$ for some $k_0 > 0$. This approximation results in a reduction of the gate count and the depth of the circuit from $O(2^n)$ to $O(2^{k_0})$. However, for functions that are non-positive or piece-wise defined, not all of the k-controlled rotation gates for $k > k_0$ can be replaced with single qubit gates. Furthermore, the angles of these gates have to be determined through classical optimization. Hence, the total gate count for this variational algorithm is $\Omega(n^2)$ whereas the classical time and space complexity is exponential in n [54].

The methods that we have discussed until now have all required some sort of classical pre-processing. A state preparation method that leverages adiabatic time evolution was recently proposed in [55] which does not require classical pre-processing at the cost of significantly deeper circuits. The idea behind this method is to encode the target state as a rank-1 projector Hamiltonian $\hat{H} = |f\rangle\langle f|$, then perform the adiabatic evolution using low-rank, 1-sparse Hamiltonian simulation techniques [63–66]. The time evolution is simulated using Trotterization-like time steps that suppress adiabatic and simulation errors. Each time step requires $O(1)$ call to an oracle which implements the Hamiltonian using the arithmetic operations. Interestingly, it was proven in [55] that the query complexity of this algorithm approaches a constant asymptotically in the number of qubits. However, note that the simulation results reported in [55] show that on the order of 10^5 Trotterization steps are required to achieve an error of 10^{-5} for $n > 5$ qubits. Given that the implementation of the oracle requires quadratic gate count and circuit depth [67], the total resources needed to run this fully-quantum state preparation algorithm are orders of magnitude higher than the present-day technology. This is not surprising as the authors intended the method would be executed on a fault-tolerant quantum computer. It is worth appreciating that this method is also capable of preparing arbitrary quantum states (i.e. loading arbitrary discrete functions), however, the associated complexity can differ greatly from the function loading use case.

Recently, a function-loading algorithm based on the quantum singular value transformation (QSVT), was proposed [56]. The basic idea of the QSVT is to find a block encoding of $h(A)$ given a block-encoding U_A of a Hermitian operator $A = \sum_k a_k|k\rangle\langle k|$, where $h(x)$ is a polynomial of degree d_{poly}. By cleverly choosing $A = \sum_k \sin(k/N)|k\rangle\langle k|$ and choosing $h(x)$ to be a polynomial approximation of $f(\arcsin(x))$, the authors were able to prepare a block-encoding of $\sum_k f(k/N)|k\rangle\langle k|$. The desired state could then be prepared with high probability by applying this block-encoding to $|+\rangle^{\otimes n} \otimes |0\rangle_{\text{ancilla}}$. In many cases, several rounds of amplitude amplification are required to boost the probability of success closer to 1. The number of rounds of amplitude estimation, R, needed is inversely related to the 'L_2-norm filling ratio' of the target function which is approximately the ratio $\|f\|_2/\|f\|_\infty$. Given that quantum circuit that implements the QSVT requires $d_{\text{poly}}/2$ applications of U_A and $U_A^\dagger$ [56], and U_A for $A = \sum_k \sin(k/N)|k\rangle\langle k|$ can be implemented with a circuit of depth n, the full quantum circuit needed to prepare the desired state has a depth of $O(nd_{\text{poly}}R)$. Moreover, as discussed in [56], the full quantum circuit consists of $O(nd_{\text{poly}}R)$ quantum gates and requires at most 4 ancilla qubits. Just like the Hamiltonian simulation method discussed above, this method is only suitable for fault-tolerant quantum computers. However, this method replaces the requirement for an oracle needed in the Hamiltonian simulation method with a classical pre-processing step in which all the angles of rotations in the quantum circuit can be determined using algorithms such as those described in [68–70].

There are many other state preparation algorithms that are tailored to a specific distribution. The most notable example is the algorithm of Kitaev and Webb that loads a Gaussian distribution using a recursive algorithm [57]. At each recursive step, the algorithm calculates the angles of rotations directly on an ancilla quantum register and rotates the non-ancillary qubits by that angle. Even though this algorithm asymptotically only requires polynomial resources, a recent study argued that this algorithm is more costly than a generic exponentially scaling algorithm for $n \leqslant 15$ [71]. This is due to the large complexity of the quantum arithmetic circuits [67]. The method of Kitaev and Webb can also be generalized to load D-dimensional correlated Gaussian distributions. The algorithm first loads D uncorrelated one-dimensional Gaussian distributions and then applies a 'shearing' transformation which implements the change of basis required to introduce the correlations between different dimensions. The implementation of the shearing transformation requires a circuit involving $(2n+1) + 2\log_2(D+1)$ ancillae qubits and $O(n^2D^2)$ CNOT gates [71]. In contrast, the FSL method does not require the shearing transformation as it can load the correlated distribution directly and has a gate count of $O(n^2D)$.

There are a few other proposals to load a Gaussian distribution that are, unlike the Kitaev–Webb algorithm, suitable for near-term quantum devices For example, Rattew *et al* have proposed a Gaussian distribution loading method that is provably robust to bit-flip and phase-flip errors [58]. This method is based on a discrete random walk inspired by Galton machines whose walk operator maps a computational basis state $|j\rangle$ to a superposition state $|j\rangle + |j+1\rangle$ up to a normalization factor. Rattew *et al* propose first loading a low-resolution Gaussian distribution on a small number of qubits using a method such as the cascade of controlled rotations depicted in figure 1(b), then iteratively qubits are added in the $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ state followed by a small number of applications of a quantum walk operator until the target number of qubit is reached. The computational complexity of this approach depends upon the specific implementation of the walk operator which requires a choice of quantum adder circuit. Hence, if the logarithmic depth adder presented by Draper *et al* [72] is used, then the overall quantum gate complexity of this Gaussian loading method is $O(n\log n)$ and $O(n)$ ancilla qubits are required. Alternatively, a QFT-based adder can be used which leads to an overall quantum gate complexity of $O(n^2)$ and only a single ancilla is required.

Another method to approximately load Gaussian distributions into a quantum state was introduced by Markov *et al* [59]. This method is based on approximations of Gaussian distributions by powers of trigonometric functions. In particular, these approximations can be derived from the limit $\lim_{r\to\infty}\cos^r(x/\sqrt{r}) = e^{-x^2/2}$. To illustrate their method, consider the task of loading the 2nd power approximation $p_2(x) = \sigma^{-1}\cos^2(\pi(x-\mu)/2\sigma)$ for $\mu-\sigma \leqslant x \leqslant \mu+\sigma$. The main insight of [59] is that a quantum state $|\sqrt{p_2}\rangle$ can be prepared by applying the inverse QFT operator applied to the state $(|0\rangle - |2^n-1\rangle)/\sqrt{2}$. This method is nearly a special case of the FSL method introduced in this work with the exception that their method neglects to control for complex phases introduced by the QFT operator.

4.2. Future directions

The FSL method inherits the majority of its circuit complexity from the QFT. Therefore, a natural question to ask is how well the FSL method performs when using an approximate QFT (AQFT) sporting which would lead to reduced circuit complexity. Specifically, it has been shown that an AQFT with error bounded by ε can be implemented by a circuit with depth bounded above by $O(\log n + \log\log(1/\varepsilon))$ and bounded below by $\Omega(\log n)$ [73]. Therefore, using an AQFT to implement an approximate FSL would yield a logarithmic-depth function loading circuit. In order to justify the usage of an AQFT for practical applications, a rigorous study of the performance of an approximate FSL is needed.

In this work, we have only focused on amplitude embedding where the values of the functions are loaded as the amplitudes of the quantum state. There exists another type of embedding known as the key-value embedding, in which the function $f : \{0,\ldots,2^n-1\} \to [0, 2^m-1]$ is encoded as an $(n+m)$-qubit state, $\frac{1}{\sqrt{2^n}}\sum_k |k\rangle_n \otimes |f_k\rangle_m$, where $|k\rangle_n$ is the n-qubit 'key' register and $|f_k\rangle$ is the m-qubit 'value' register. The difficulty in preparing such a key-value pair state is that the circuit preparing the state on the value register has to be controlled by the key register, which leads to a large gate count. Various efficient algorithms have been proposed for the key-value embedding of the functions/distributions [74–76]. An interesting future direction is to see if the FSL method introduced in this work can be generalized to efficiently generate key-value embeddings.

It is also worth emphasizing that the Fourier series is just one possible orthonormal series representation of functions of interest. There are many other basis functions—such as wavelets, Chebyshev polynomials, etc – that could be more suitable than a Fourier series expansion for some classes of functions. In this work, we only considered the Fourier-series representation given that the implementation of the Fourier transform on a quantum computer requires fewer gates than the implementation of the wavelet transform [77, 78] and discrete cosine transform [79]. Nevertheless, it could be worthwhile to investigate the performance of the truncated wavelet series, and other possible representations, for function loading on quantum computers.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://doi.org/10.5281/zenodo.7613143.

A user-friendly implementation of the FSL method, along with various examples, is available at https://github.com/mcmahon-lab/Fourier-Series-Loader.

Acknowledgments

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. We thank Travis Humble (Oak Ridge National Laboratory) for his support of our research. We also thank Ryan Landfield (ORNL/OLCF User Support) and Brian Neyenhuis (Quantinuum) for technical assistance and for providing the hardware specifications included in this work. It is also a pleasure to thank Javier Gonzalez-Conde, Keisuke Fujii, Constantin Gonciulea, Vanio Markov, Juan José Garcia-Rípoll, Arthur Rattew, Mikel Sanz, Charlee Stefanski, Naoki Yamamoto, and Christa Zoufal for useful feedback on a draft of this manuscript. The work of M M was supported by the US Department of Energy under Grant Numbers DE-SC0014123 and DE-SC0007884, and the QuantiSED Fermilab consortium. P L M gratefully acknowledges support from a Sloan Foundation Fellowship and a David and Lucile Packard Foundation Fellowship, and acknowledges membership in the CIFAR Quantum Information Science Program as an Azrieli Global Scholar.

Software used

The circuit diagrams in this paper and the appendices were prepared using quantikz package [80]. All the plots were prepared using Matplotlib [81]. To perform the maximum likelihood estimation for the tomography experiment, the cost function was minimized using the generic minimize function in SciPy [82].

Appendix A. Derivation of the FSL method

In section 2, we claimed that the FSL method can load a Fourier series of one or more variables into a quantum state. We demonstrated this capability by loading a function of one and two variables on the Quantinuum H1-1 and H1-2 quantum computers. In this appendix, we provide a detailed derivation of the FSL method.

A.1. FSL method for a Fourier series of single variable

Consider a truncated Fourier series of the form

$$f(x) = \sum_{p=-M}^{M} c_p\, e^{-i2\pi px}, \tag{A1}$$

where $M = 2^m - 1$ denotes the number of Fourier coefficients. Our goal is then to prepare the following state of n qubits:

$$|f\rangle_n = \sum_{k=0}^{2^n-1} f_k\, |k\rangle_n, \tag{A2}$$

where $f_k = f(x = k/2^n) = \sum_{p=-M}^{M} c_p\, e^{-i2\pi pk/2^n}$. Here, the subscript n in $|\cdot\rangle_n$ specifies the number of qubits.

Let us assume the unitary U_c in the quantum circuit in figure 1(a) acts on $(m+1)$ qubits and maps $|0\rangle^{\otimes(m+1)}$ state to $|\tilde{c}\rangle$, where

$$|\tilde{c}\rangle_{m+1} := 2^{n/2}\sum_{p=0}^{M} c_p|p\rangle_{m+1} + 2^{n/2}\sum_{p=1}^{M} c_{-p}|2^{m+1}-p\rangle_{m+1}. \tag{A3}$$

Then we claim that the circuit in figure 1(a) maps initial $|0\rangle^{\otimes n}$ state to the target $|f\rangle_n$ state in equation (A2). To verify this claim, we traverse through the circuit in figure 1(a) and analyze the state after each step. After the action on U_c on the lower $(m+1)$ qubits, the initial $|0\rangle^{\otimes n}$ state becomes

$$2^{n/2}\sum_{p=0}^{M} c_p|0\rangle^{\otimes(n-m-1)}\otimes|p\rangle_{m+1} + 2^{n/2}\sum_{p=1}^{M} c_{-p}|0\rangle^{\otimes(n-m-1)}\otimes|2^{m+1}-p\rangle_{m+1}, \tag{A4}$$

which can equivalently be written as

$$2^{n/2}\sum_{p=0}^{M} c_p|0\rangle^{\otimes(n-m-1)}\otimes|0\rangle\otimes|p\rangle_m + 2^{n/2}\sum_{p=1}^{M} c_{-p}|0\rangle^{\otimes(n-m-1)}\otimes|1\rangle\otimes|2^m-p\rangle_m. \tag{A5}$$

The next step following the action of U_c is the action of the cascade of $n-m-1$ CNOT gates as shown in figure 1(a). After the action of these CNOT gates, the state of n qubits becomes

$$2^{n/2}\sum_{p=0}^{M} c_p|0\rangle^{\otimes(n-m)}\otimes|p\rangle_m + 2^{n/2}\sum_{p=1}^{M} c_{-p}|1\rangle^{\otimes(n-m)}\otimes|2^m-p\rangle_m, \tag{A6}$$

which can equivalently be written as

$$2^{n/2}\sum_{p=0}^{M} c_p|p\rangle_n + 2^{n/2}\sum_{p=1}^{M} c_{-p}|2^n-p\rangle_n. \tag{A7}$$

The last step in the FSL circuit shown in figure 1(a) is the action of the inverse QFT. Recall that the inverse QFT acts on a computational basis state as

$$(\mathrm{QFT})^\dagger\,|p\rangle_n = \frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1} e^{-i2\pi pk/2^n}\,|k\rangle_n. \tag{A8}$$

This implies that the state of n qubits after the action of the inverse QFT becomes

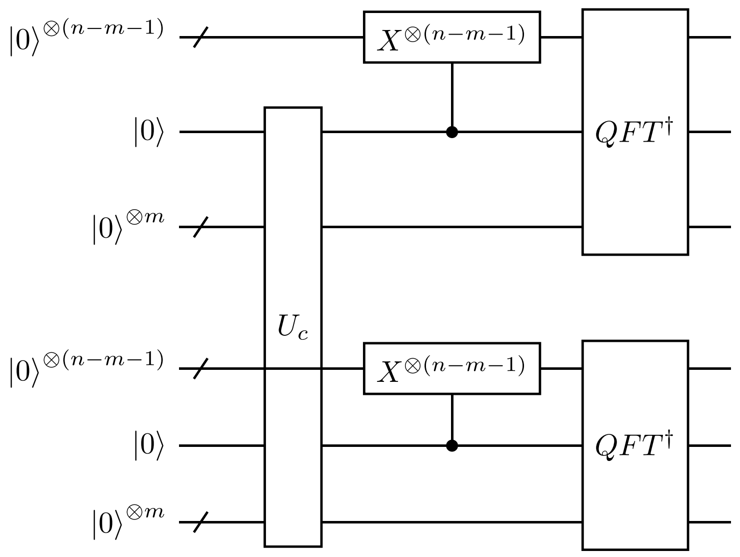

Figure 7. The quantum circuit implementing the two-dimensional FSL method. Similarly to the one-dimensional FSL circuit in figure 1, this circuit consists of an unitary U_c which loads the Fourier coefficients, a cascade of CNOT gates, and inverse QFTs.

$$\begin{aligned}
&2^{n/2}\sum_{p=0}^{M} c_p\,(\mathrm{QFT})^\dagger\,|p\rangle_n + 2^{n/2}\sum_{p=1}^{M} c_{-p}\,(\mathrm{QFT})^\dagger\,|2^n-p\rangle_n\,,\\
&=\sum_{p=0}^{M} c_p \sum_{k=0}^{2^n} e^{-i2\pi\,kp/2^n}\,|k\rangle_n + \sum_{p=1}^{M} c_{-p}\sum_{k=0}^{2^n} e^{-i2\pi\,k(2^n-p)/2^n}\,|k\rangle_n\,,\\
&=\sum_{k=0}^{2^n}\sum_{p=-M}^{M} c_p e^{-i2\pi\,kp/2^n}\,|k\rangle_n\,,
\end{aligned}\tag{A9}$$

which is precisely the target state in equation (A2). This finishes our derivation of the FSL method for single-variable functions.

A.2. FSL method for a multi-variate Fourier series

Here, we provide the details for extending the FSL method to loading higher-dimensional functions and provide the corresponding quantum circuit. For simplicity, we consider the case of a Fourier series of two variables. Although, our construction can easily be generalized to Fourier series of more than two variables.

Any arbitrary truncated Fourier Series of two variables can be written as

$$f(x,y) = \sum_{p=-M}^{M}\sum_{q=-M}^{M} c_{p,q}\,e^{-i2\pi\,px}\,e^{-i2\pi\,qy}\,,\tag{A10}$$

where $M = 2^m - 1$. Our goal is then to prepare the following state of $2n$ qubits (n qubits for each variable):

$$|f\rangle_{2n} = \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} f_{k,j}\,|k\rangle_n\otimes|j\rangle_n\,,\tag{A11}$$

where $f_{k,j} = f(k/2^n, j/2^n) = \sum_{p=-M}^{M}\sum_{q=-M}^{M} c_{p,q}\,e^{-i2\pi\,pk/2^n}\,e^{-i2\pi\,qj/2^n}$. Here, the subscript n in $|\cdot\rangle_n$ specifies the number of qubits.

We claim that the state in equation (A11) can be prepared using the circuit shown in figure 7. In this circuit, the unitary operator U_c is any unitary operator that maps $|0\rangle^{\otimes 2(m+1)}$ to state $|\tilde{c}\rangle_{2(m+1)}$, where

$$|\tilde{c}\rangle_{2(m+1)} = 2^n \sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\,|p\rangle_{m+1}\otimes|q\rangle_{m+1} + 2^n \sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\,|p\rangle_{m+1}\otimes|2^{m+1}-q\rangle_{m+1}$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\,|2^{m+1}-p\rangle_{m+1}\otimes|q\rangle_{m+1} + 2^n \sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\,|2^{m+1}-p\rangle_{m+1}\otimes|2^{m+1}-q\rangle_{m+1}\,. \tag{A12}$$

Note that this state is analogous to the state $|\tilde{c}\rangle$ in equation (A3).

To verify our claim, we traverse through the circuit in figure 7 and analyze the state after each step: the initial state of the $2n$ qubits is $|0\rangle^{\otimes 2n}$. Then, after applying the U_c operator, the state of $2n$ qubits is

$$2^n \sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\,|0\rangle^{\otimes(n-m-1)}\otimes|p\rangle_{m+1}\otimes|0\rangle^{\otimes(n-m-1)}\otimes|q\rangle_{m+1}$$
$$+ 2^n \sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\,|0\rangle^{\otimes(n-m-1)}\otimes|p\rangle_{m+1}\otimes|0\rangle^{\otimes(n-m-1)}\otimes|2^{m+1}-q\rangle_{m+1}$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\,|0\rangle^{\otimes(n-m-1)}\otimes|2^{m+1}-p\rangle_{m+1}\otimes|0\rangle^{\otimes(n-m-1)}\otimes|q\rangle_{m+1}$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\,|0\rangle^{\otimes(n-m-1)}\otimes|2^{m+1}-p\rangle_{m+1}\otimes|0\rangle^{\otimes(n-m-1)}\otimes|2^{m+1}-q\rangle_{m+1}\,. \tag{A13}$$

The above state can equivalently be written as

$$2^n \sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\,|0\rangle^{\otimes(n-m-1)}\otimes|0\rangle\otimes|p\rangle_m\otimes|0\rangle^{\otimes(n-m-1)}\otimes|0\rangle\otimes|q\rangle_m$$
$$+ 2^n \sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\,|0\rangle^{\otimes(n-m-1)}\otimes|0\rangle\otimes|p\rangle_m\otimes|0\rangle^{\otimes(n-m-1)}\otimes|1\rangle\otimes|2^m-q\rangle_m$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\,|0\rangle^{\otimes(n-m-1)}\otimes|1\rangle\otimes|2^m-p\rangle_m\otimes|0\rangle^{\otimes(n-m-1)}\otimes|0\rangle\otimes|q\rangle_m$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\,|0\rangle^{\otimes(n-m-1)}\otimes|1\rangle\otimes|2^m-p\rangle_m\otimes|0\rangle^{\otimes(n-m-1)}\otimes|1\rangle\otimes|2^m-q\rangle_m. \tag{A14}$$

Following the application of the controlled $X^{\otimes(n-m-1)}$ gates in figure 7, the above state is mapped to

$$2^n \sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\,|0\rangle^{\otimes(n-m)}\otimes|p\rangle_m\otimes|0\rangle^{\otimes(n-m)}\otimes|q\rangle_m$$
$$+ 2^n \sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\,|0\rangle^{\otimes(n-m)}\otimes|p\rangle_m\otimes|1\rangle^{\otimes(n-m)}\otimes|2^m-q\rangle_m$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\,|1\rangle^{\otimes(n-m)}\otimes|2^m-p\rangle_m\otimes|0\rangle^{\otimes(n-m)}\otimes|q\rangle_m$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\,|1\rangle^{\otimes(n-m)}\otimes|2^m-p\rangle_m\otimes|1\rangle^{\otimes(n-m)}\otimes|2^m-q\rangle_m\,. \tag{A15}$$

We now denote this state by $|c\rangle_{2n}$ and write it equivalently as

$$|c\rangle_{2n} = 2^n \sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\,|p\rangle_n\otimes|q\rangle_n + 2^n \sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\,|p\rangle_n\otimes|2^n-q\rangle_n$$
$$+ 2^n \sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\,|2^n-p\rangle_n\otimes|q\rangle_n + 2^n \sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\,|2^n-p\rangle_n\otimes|2^n-q\rangle_n\,. \tag{A16}$$

Finally, we consider the action of the inverse QFTs. The state of $2n$ qubits following the application of the inverse QFTs is given by

$$
\begin{aligned}
(\mathrm{QFT})^{\dagger} \otimes (\mathrm{QFT})^{\dagger}\, |c\rangle_{2n} = & \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1}\sum_{p=0}^{M}\sum_{q=0}^{M} c_{p,q}\, e^{-i2\pi pk/2^n} e^{-i2\pi qj/2^n}\, |k\rangle_n \otimes |j\rangle_n \\
& + \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1}\sum_{p=0}^{M}\sum_{q=1}^{M} c_{p,-q}\, e^{-i2\pi pk/2^n} e^{i2\pi qj/2^n}\, |k\rangle_n \otimes |j\rangle_n \\
& + \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1}\sum_{p=1}^{M}\sum_{q=0}^{M} c_{-p,q}\, e^{i2\pi pk/2^n} e^{-i2\pi qj/2^n}\, |k\rangle_n \otimes |j\rangle_n \\
& + \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1}\sum_{p=1}^{M}\sum_{q=1}^{M} c_{-p,-q}\, e^{i2\pi pk/2^n} e^{i2\pi qj/2^n}\, |k\rangle_n \otimes |j\rangle_n \,. \qquad \text{(A17)}
\end{aligned}
$$

By combining terms, we can write this state as

$$
(\mathrm{QFT})^{\dagger} \otimes (\mathrm{QFT})^{\dagger}\, |c\rangle_{2n} = \sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} \left(\sum_{p=-M}^{M}\sum_{q=-M}^{M} c_{p,q}\, e^{-i2\pi pk/2^n} e^{-i2\pi qj/2^n} \right) |k\rangle_n \otimes |j\rangle_n \,, \qquad \text{(A18)}
$$

which is exactly the desired state $|f\rangle_{2n}$ in equation (A11). This verifies our claim that the circuit in figure 7 can be used for the exact loading of a Fourier series of two or more variables.

Appendix B. Uniformly controlled rotations

In the main text, we proposed a cascade of uniformly controlled rotations shown in figure 1(b) as one possible method to load the Fourier coefficients to the amplitudes of a quantum state $|\tilde{c}\rangle = U_c|0\rangle^{\otimes m}$. This method has many desirable features. For example, an efficient implementation of this circuit U_c in terms of CNOT gates and rotation gates is known [7]. Moreover, all the angles of rotation can be classically determined given the target state [7]. Since this method plays a significant role in the FSL method, we briefly review the results of [7] in this appendix.

As discussed in section 2, the m-qubit circuit shown in figure 1(b) can be used to prepare an arbitrary m-qubit state $|\psi\rangle$. The circuit in figure 1(b) has three distinct parts, (i) a R_z gate, (ii) uniformly controlled R_y gates, and (iii) uniformly controlled R_z gates. Hence, the action of this circuit can be succinctly written as

$$
|\psi\rangle = \prod_{j=0}^{m-1} F_j^{(z)}\left[\boldsymbol{\alpha}_{m-1-j}^{(z)}\right] \cdot \prod_{j=0}^{m-1} F_j^{(y)}\left[\boldsymbol{\alpha}_{m-1-j}^{(y)}\right] \cdot R_z(-\phi_\psi)\, |0\rangle^{\otimes n}. \qquad \text{(B1)}
$$

In this equation, $\boldsymbol{\alpha}_j^{(y/z)} = (\alpha_{j,0}^{(y/z)}, \alpha_{j,1}^{(y/z)}, \dots, \alpha_{j,2^{m-1-j}-1}^{(y/z)})$ is a vector of size 2^{m-1-j}, and $F_j^{(y/z)}[\boldsymbol{\alpha}_{m-1-j}^{(y/z)}]$ is a uniformly j-controlled rotation operator which acts on the $(j+1)$th qubit as $R_{y/z}\big(\alpha_{m-1-j,k}^{(y/z)}\big)$ if the state of the first j qubits is $|k\rangle$ for $k = \{0,1,\dots,2^j-1\}$ [5–7]. A pictorial representation of $F_j^{(y)}$ for $j=3$ is given in figure 8.

Given the m-qubit target state $|\psi\rangle = \sum_{i=0}^{2^m-1} |\psi_i|\, e^{i\omega_i}\, |i\rangle$, the angles $\boldsymbol{\alpha}_j^{(y)}$ for $j = \{0,1,\dots,m-1\}$ in equation (B1) can be determined using [5–7]

$$
\alpha_{j,k}^{(y)} = 2\arcsin\left[\frac{\sum_{\ell=0}^{2^j-1} |\psi_{(2k+1)2^j+\ell}|^2}{\sum_{\ell=0}^{2^{j+1}-1} |\psi_{k2^{j+1}+\ell}|^2} \right]. \qquad \text{(B2)}
$$

Moreover, the angles $\boldsymbol{\alpha}_j^{(z)}$ for $j = \{0,1,\dots,m-1\}$ and ϕ_ψ can be determined using [7]

$$
\alpha_{j,k}^{(z)} = 2^{-j} \sum_{\ell=0}^{2^j} \left(\omega_{(2k+1)2^j+\ell} - \omega_{k2^{j+1}+\ell} \right), \qquad \text{(B3)}
$$

and

$$
\phi_\psi = 2^{1-m} \sum_{j=0}^{2^m-1} \omega_j\,, \qquad \text{(B4)}
$$

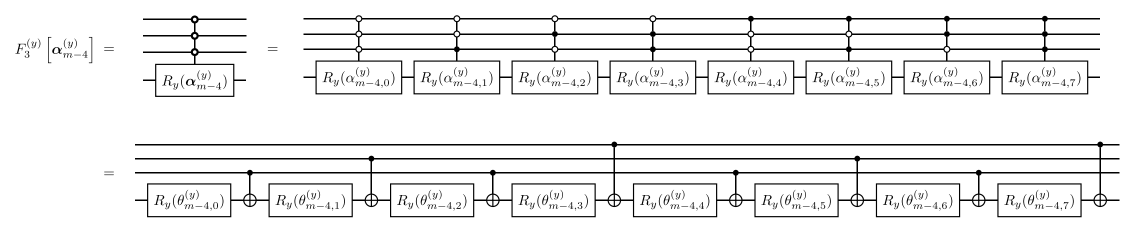
$F_3^{(y)}\left[\boldsymbol{\alpha}_{m-4}^{(y)}\right]$ =
$R_y(\boldsymbol{\alpha}_{m-4}^{(y)})$
=
$R_y(\alpha_{m-4,0}^{(y)})$
$R_y(\alpha_{m-4,1}^{(y)})$
$R_y(\alpha_{m-4,2}^{(y)})$
$R_y(\alpha_{m-4,3}^{(y)})$
$R_y(\alpha_{m-4,4}^{(y)})$
$R_y(\alpha_{m-4,5}^{(y)})$
$R_y(\alpha_{m-4,6}^{(y)})$
$R_y(\alpha_{m-4,7}^{(y)})$
=
$R_y(\theta_{m-4,0}^{(y)})$
$R_y(\theta_{m-4,1}^{(y)})$
$R_y(\theta_{m-4,2}^{(y)})$
$R_y(\theta_{m-4,3}^{(y)})$
$R_y(\theta_{m-4,4}^{(y)})$
$R_y(\theta_{m-4,5}^{(y)})$
$R_y(\theta_{m-4,6}^{(y)})$
$R_y(\theta_{m-4,7}^{(y)})$

Figure 8. Equivalent implementations of uniformly 3-controlled rotations. The implementation in terms of rotation gates and CNOT gates was derived in [7].

respectively. Hence, given the target state $|\psi\rangle$, all the angles in equation (B1) can be evaluated on a classical computer.

The next step after finding the angles of uniformly controlled rotations, is to efficiently implement the uniformly controlled gates in terms of simple gates such as rotations and CNOT gates. It was shown in [7] that $F_j^{(y/z)}$ for $j > 0$ can be implemented using 2^j $R_{y/z}$ gates acting on the $(j+1)$th qubit. The angle of the kth rotation for $k = \{0, 1, \ldots, 2^j - 1\}$ is given by [7]

$$\theta_{m-1-j,k}^{(y/z)} = \sum_{\ell=0}^{2^j-1} M_{k\ell}\, \alpha_{m-1-j,\ell}^{(y/z)}; \qquad M_{k\ell} = 2^{-j}\,(-1)^{b_\ell \cdot g_k}, \tag{B5}$$

where b_k and g_k denote the binary code and binary Gray code of the integer k respectively. In addition to the 2^j rotation gates, the efficient implementation of $F_j^{(y/z)}$ also has 2^j CNOT gates acting on the $(j+1)$th qubit as shown in figure 8. The last CNOT gate is controlled by the first qubit. Whereas, the control qubit for the kth CNOT, excluding the last CNOT, has to be determined based on which bit of the binary Gray code of integer k differs from that of $k-1$. For example, the control qubit for the first CNOT in figure 8 is the 3rd qubit because the third bit in the Gray code of 1 (001) and 0 (000) are different. Whereas, the control qubit for the second CNOT is the 2$^{\text{nd}}$ qubit because the second bit in the Gray code of 2 (011) and 1 (001) are different.

The Python code used to implement the cascade of uniformly controlled rotations and to perform the pre-processing step discussed above is provided in the code repository accompanying this paper. This finishes our review of [7].

Appendix C. Benchmarking classical pre-processing time

An attractive feature of the FSL method is that the classical pre-processing cost needed to implement the FSL circuit is constant with respect to the total number of qubits, n. However, an important step during the classical pre-processing involves compiling a circuit implementation of a unitary U_c that loads the $M = 2^{(m+1)}$ Fourier coefficients on $m+1$ qubits. This cost scales polynomially with the number of Fourier coefficients. In fact, the asymptotic cost of implementing U_c either using the cascade of uniformly controlled rotations as in figure 1(b) or using the Schmidt decomposition method as in figure 1(c) scales as $O(M^3)$. If this classical pre-processing cost were too high, then our method would not be suitable for functions for which a large number of Fourier coefficients are needed to approximate the target function with the desired error. For this reason, we benchmark the classical pre-processing time needed to implement the unitary U_c. More precisely, we recorded the CPU wall clock time, T_{cl}, required to find a circuit implementation of the unitary U_c for randomly generated 2^{m+1} Fourier coefficients for $6 \leqslant m \leqslant 14$, and the results are presented in figure 9. We found that the classical pre-processing time, T_{cl}, for $m = 12$ is less than two minutes using the uniformly controlled rotations and is less than a minute using the Schmidt decomposition. The benchmark was performed on a 2020 13-inch Macbook Pro with 3.2 GHz M1 CPU and 8 GB main memory. This benchmark demonstrates that the classical pre-processing time needed to implement the FSL method is computationally tractable even for sufficiently large values of m.

Furthermore, we recorded the total compute time needed to perform a classical simulation of the FSL method's quantum circuit loading the discontinuous function we presented in figure 2(f) with $m = 17$ (i.e. approximately a quarter million Fourier coefficients) and $n = 20$. We found that it took a total of around 15 minutes (13 minutes for finding U_c using the Schmidt decomposition method and 2 minutes for executing the circuit in Qiskit). The results of the simulation are presented in figure 10.

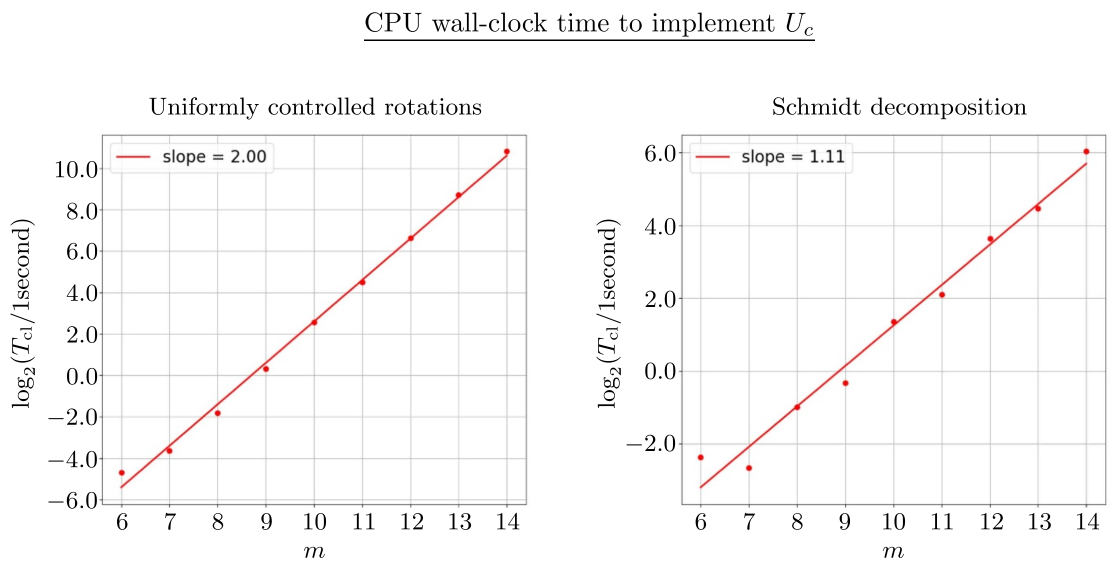

Figure 9. Plots for the CPU wall clock time, T_{cl}, passed during the classical pre-processing step of finding a circuit implementation of the unitary U_c using the uniformly controlled rotations (left) and using the Schmidt decomposition method (right). We find that even though T_{cl} scales polynomially with the number of Fourier coefficients $M = 2^{m+1}$, the values of T_{cl} are reasonable for large enough values of M. For example, $T_{cl} < 6$ seconds for uniformly controlled rotations and $T_{cl} < 3$ seconds for the Schmidt decomposition method for $M \sim 2000$ ($m = 10$).

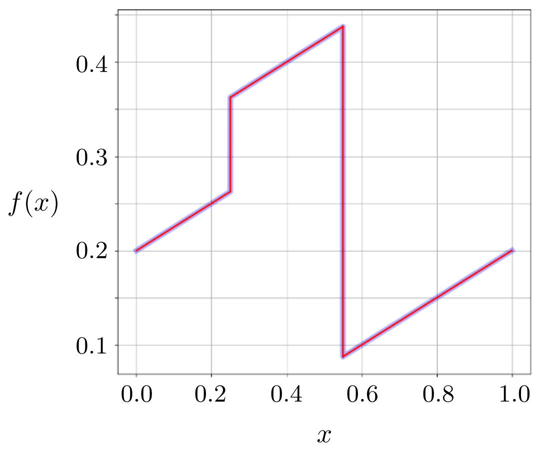

Figure 10. Quantum circuit simulation of loading a piece-wise discontinuous function into a quantum state of $n = 20$ qubits using the FSL method with a quarter million Fourier coefficients. The total CPU wall clock time for this simulation was around 15 minutes which included the classical pre-processing time needed to find a circuit implementation of loading Fourier coefficients and the time needed to simulate the execution of the circuit.

Appendix D. Error analysis

In the main text, we stated that the infidelity of the FSL method decays exponentially with m at a rate that depends on the smoothness of the target function. In figure 11, we explicitly verify this exponential decay of infidelity for all of the functions considered in section 2. In particular, we show that the infidelity decays as $O(2^{-m})$ for the discontinuous function in figure 2(f), whereas it decays as $O(2^{-3m})$ for continuous functions such as x^x in figure 2(a), sinc function in figure 2(b), reflected put option function in figure 2(c), hyperbolic tangent function in figure 2(e), and the two-dimensional sinc function considered in figure 3. Moreover, for an infinitely differentiable function such as the wavefunction of the first excited state of the quantum harmonic oscillator considered in figure 2(d), the infidelity decays doubly exponential in m.

In the rest of this appendix, we argue that the exponential decay of infidelity is true in general. We do this by deriving a bound on infidelity as a function of m. For simplicity, we only consider functions of a single variable in our analysis, but the generalization to multivariate functions is straightforward. Our derivation is similar to the analysis in [83] in which the bounds on the truncation error, as a function of the number of terms in the Fourier series, were obtained.

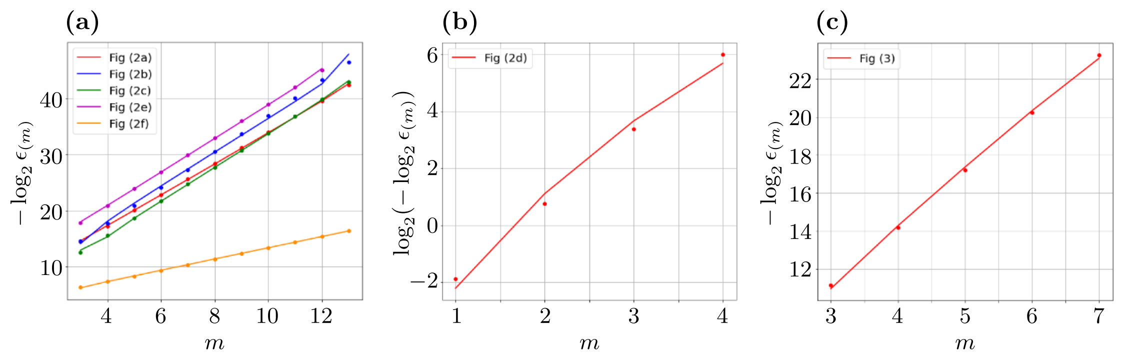

Figure 11. Plots for infidelity, $\epsilon_{(m)}$, as a function of m for all of the functions considered in section 2. The colored lines are used to plot the values of the infidelities, whereas the colored dots denote the linear function determined by linear fitting the values of the infidelities. The slope of the linear fit function describing the infidelity for the function in figure 2(f) is approximately 1, whereas the slope is approximately 3 for the functions in figures 2(a)–(c), (e), and 3. Plot (b) illustrates the doubly exponential decay of infidelity for the function in figure 2(d).

Note that any function encoding state $|f\rangle = \sum_{k=0}^{2^n-1} f_k|k\rangle$ can exactly be written as $|f\rangle = (\mathrm{QFT})^\dagger |c\rangle$, where

$$|c\rangle = \sum_{k=0}^{2^{n-1}} c_k|k\rangle + \sum_{k=1}^{2^{n-1}-1} c_{-k}|2^n - k\rangle\,, \tag{D1}$$

and c_k are the *discrete* Fourier coefficients of $f(x)$:

$$c_k = \frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} f_\ell\, e^{i2\pi k\ell/2^n}\,. \tag{D2}$$

The normalization of the state $|f\rangle$ implies that the state $|c\rangle$ is normalized. That is,

$$\sum_{k=-2^{n-1}+1}^{2^{n-1}} |c_k|^2 = 1\,. \tag{D3}$$

The FSL method approximates the state $|f\rangle$ by the state $|f_{(m)}\rangle = (\mathrm{QFT})^\dagger |c_{(m)}\rangle$, where

$$|c_{(m)}\rangle = \mathcal{N}^{-1/2} \sum_{k=0}^{2^m-1} c_k|k\rangle + \mathcal{N}^{-1/2} \sum_{k=1}^{2^m-1} c_{-k}|2^n - k\rangle\,, \tag{D4}$$

and $\mathcal{N}$ is normalization constant:

$$\mathcal{N} = \sum_{k=-2^m+1}^{2^m-1} |c_k|^2\,. \tag{D5}$$

The infidelity between the state $|f\rangle$ and $|f_{(m)}\rangle$ can be calculated in terms of the Fourier coefficients as follows:

$$\begin{aligned}
\epsilon_{(m)} &= 1 - \left|\langle f|f_{(m)}\rangle\right|^2\,,\\
&= 1 - \left|\langle c|c_{(m)}\rangle\right|^2\,,\\
&= 1 - \mathcal{N}^{-1}\left(\sum_{k=-2^m+1}^{2^m-1} |c_k|^2\right)^2\,,\\
&= 1 - \sum_{k=-2^m+1}^{2^m-1} |c_k|^2\,,\\
&= \sum_{k=2^m}^{2^{n-1}} |c_k|^2 + \sum_{k=2^m}^{2^{n-1}-1} |c_{-k}|^2\,,
\end{aligned} \tag{D6}$$

where we have used equation (D3) in the last step and equation (D5) in the step before that.

In order to derive a bound on the infidelity, we first need to derive a bound on the Fourier coefficients. To do this, we define a *difference* operator Δ such that

$$\Delta f_\ell =: f_{\ell+1} - f_\ell . \tag{D7}$$

From this definition, we deduce that the action of this difference operator on a product of functions is given by

$$\Delta\left(f_\ell g_\ell\right) = \left(\Delta f_\ell\right)\cdot g_{\ell+1} + f_\ell \cdot \left(\Delta g_\ell\right). \tag{D8}$$

This is a discrete analog of the action of the derivative operator on a product of functions. Now, if we take $g_\ell = e^{i2\pi k\ell/2^n}$ in the above expression and rearrange the terms, we get

$$f_\ell e^{i2\pi k\ell/2^n} = \frac{1}{e^{i2\pi k/2^n}-1}\Delta\left(f_\ell e^{i2\pi k\ell/2^n}\right) - \frac{1}{1-e^{-i2\pi k/2^n}}\left(\Delta f_\ell\right)\cdot e^{i2\pi k\ell/2^n}. \tag{D9}$$

Note we have applied the identity $\Delta e^{i2\pi k\ell/2^n} = \left(e^{i2\pi k/2^n}-1\right)\cdot e^{i2\pi k\ell/2^n}$ to obtain the above expression. Now using equation (D2), we find that the Fourier coefficient c_k can be written as

$$\begin{aligned} c_k &= \frac{1}{2^{n/2}}\frac{1}{e^{i2\pi k/2^n}-1}\sum_{\ell=0}^{2^n-1}\Delta\left(f_\ell e^{i2\pi k\ell/2^n}\right) - \frac{1}{2^{n/2}}\frac{1}{1-e^{-i2\pi k/2^n}}\sum_{\ell=0}^{2^n-1}\left(\Delta f_\ell\right)\cdot e^{i2\pi k\ell/2^n}, \\ &= \frac{1}{2^{n/2}}\frac{1}{e^{i2\pi k/2^n}-1}\left(f_{2^n}-f_0\right) - \frac{1}{2^{n/2}}\frac{1}{1-e^{-i2\pi k/2^n}}\sum_{\ell=0}^{2^n-1}\left(\Delta f_\ell\right)\cdot e^{i2\pi k\ell/2^n}, \\ &= -\frac{1}{2^{n/2}}\frac{1}{1-e^{-i2\pi k/2^n}}\sum_{\ell=0}^{2^n-1}\left(\Delta f_\ell\right)\cdot e^{i2\pi k\ell/2^n}, \end{aligned} \tag{D10}$$

where we have used the periodicity of f to cancel the first term. Now, using the triangle inequality, we get

$$\begin{aligned} |c_k| &= \frac{1}{2^{n/2}}\frac{1}{2\sin\left(\pi k/2^n\right)}\left|\sum_{\ell=0}^{2^n-1}\left(\Delta f_\ell\right)\cdot e^{i2\pi k\ell/2^n}\right|, \\ &\leqslant \frac{1}{2^{n/2}}\frac{1}{2\sin\left(\pi k/2^n\right)}\sum_{\ell=0}^{2^n-1}\left|\left(\Delta f_\ell\right)\right|, \\ &= \frac{1}{2^{n/2}}\frac{1}{2\sin\left(\pi k/2^n\right)}\left\||\Delta f\rangle\right\|_1, \end{aligned} \tag{D11}$$

where $\left\||\Delta f\rangle\right\|_1$ is the 1-norm of the state $|\Delta f\rangle = \sum_{\ell=0}^{2^n-1}\Delta f_\ell|\ell\rangle$.

Due to this bound on the Fourier coefficients, the infidelity in equation (D6) is bounded by

$$\epsilon_{(m)} \leqslant \frac{\left\||\Delta f\rangle\right\|_1^2}{2^{n+1}}\sum_{k=2^m}^{2^{n-1}}\frac{1}{\left(\sin\left(\pi k/2^n\right)\right)^2}. \tag{D12}$$

Bounding the sum by an integral provides the bound

$$\begin{aligned} \epsilon_{(m)} &\leqslant \frac{\left\||\Delta f\rangle\right\|_1^2}{2^{n+1}}\int_{k=2^m}^{2^{n-1}}\mathrm{d}k\,\frac{1}{\left(\sin\left(\pi k/2^n\right)\right)^2}, \\ &= \frac{1}{2\pi}\left\||\Delta f\rangle\right\|_1^2\cot\left(\pi 2^m/2^n\right). \end{aligned} \tag{D13}$$

The bound derived above holds for an arbitrary number of qubits, n. To find the asymptotic limit of this bound, we first need to determine how the 1-norm of the state $|\Delta f\rangle$ scales with large values of n. Note that the 1-norm of $|\Delta f\rangle$ is determined by the values of f_ℓ at local maximas, local minimas, and the end-points of the domain. That is,

$$\begin{aligned}\left\| |\Delta f\rangle \right\|_1 &= \sum_{\ell=0}^{2^n-1} \left| (\Delta f_\ell) \right| , \\ &= -\left(\mathrm{sgn}\left(\Delta f_0\right) - \mathrm{sgn}\left(\Delta f_{2^n-1}\right)\right) f_0 + 2\sum_{\ell\in\ell_{\max}} f_\ell - 2\sum_{\ell\in\ell_{\min}} f_\ell , \end{aligned} \tag{D14}$$

where $\ell_{\max}$ is the set of points when f_ℓ is locally maximum (i.e. $f_\ell > f_{\ell\pm1}$) and $\ell_{\min}$ is the set of points when f_ℓ is locally minimum (i.e. $f_\ell < f_{\ell\pm1}$). The normalization condition for the state $|f\rangle$ implies that $f_\ell \sim 1/\sqrt{2^n}$ for large values of n. This means that the 1-norm also scales as

$$\left\| |\Delta f\rangle \right\|_1 \sim \frac{C}{\sqrt{2^n}} , \tag{D15}$$

where $C > 0$ is a constant. With this scaling result, we find that the bound on $\epsilon_{(m)}$ approaches

$$\epsilon_{(m)} \leqslant \frac{C^2}{\pi^2 2^{m+1}} \tag{D16}$$

in the large limit as $n\to\infty$.

The bound in equation (D16) is valid for any periodic target function, f. We can obtain stronger bounds if we make certain assumptions about the *continuity* of $\Delta^p f_k$, where

$$\Delta^p f_\ell =: \Delta^{p-1} f_{\ell+1} - \Delta^{p-1} f_\ell , \tag{D17}$$

where p is a positive integer, and f_ℓ is periodically extended beyond $\ell\in\{0,1,\ldots,2^n-1\}$ according to $f_{2^n+\ell} = f_\ell$. Let us assume that the $\Delta^q f_\ell$ for all $q\leqslant p$ is *continuous* by which we mean that $\Delta^q f_\ell \sim (1/2^n)^{q+1/2}$ for $q\leqslant p$ and that $\Delta^q f_\ell \sim (1/2^n)^{p+1/2}$ for $q \geqslant p$. This implies that

$$\left\| |\Delta^{p+1} f\rangle \right\|_1 \sim \frac{C_p}{(2^n)^{p+1/2}} . \tag{D18}$$

To see why this leads to a stronger bound, we repeat the analysis in equation (D10) p times to get

$$c_k = \frac{1}{2^{n/2}} \left(\frac{-1}{1-e^{-i2\pi k/2^n}}\right)^{p+1} \sum_{\ell=0}^{2^n-1} \left(\Delta^{p+1} f_\ell\right)\cdot e^{i2\pi k\ell/2^n} . \tag{D19}$$

This leads to a stronger bound of the Fourier coefficients,

$$|c_k| \leqslant \frac{1}{2^{n/2}} \frac{1}{\left(2\sin\left(\pi k/2^n\right)\right)^{p+1}} \left\| |\Delta^{p+1} f\rangle \right\|_1 , \tag{D20}$$

which leads to a stronger bound on infidelity,

$$\begin{aligned}\epsilon_{(m)} &\leqslant \frac{\left\| |\Delta^{p+1} f\rangle \right\|_1^2}{2^{n+1+2p}} \sum_{k=2^m}^{2^{n-1}} \frac{1}{\left(\sin\left(\pi k/2^n\right)\right)^{2p+2}} , \\ &\leqslant \frac{\left\| |\Delta^{p+1} f\rangle \right\|_1^2}{2^{n+1+2p}} \int_{k=2^m}^{2^{n-1}} \mathrm{d}k \frac{1}{\left(\sin\left(\pi k/2^n\right)\right)^{2p+2}} , \\ &= \frac{\left\| |\Delta^{p+1} f\rangle \right\|_1^2}{2^{2p+2}\pi} \left[B\left(1; -\frac{p+1}{2}, \frac{1}{2}\right) - B\left(\sin^2\left(\frac{\pi 2^m}{2^n}\right); -\frac{p+1}{2}, \frac{1}{2}\right)\right] , \end{aligned} \tag{D21}$$

where $B(x;a,b)$ is the incomplete Beta function. Asymptotically, this bound approaches

$$\epsilon_{(m)} \leqslant \frac{C_p^2}{(2p+1)\,\pi^{2p+2} 2^{(m+1)(2p+1)}} . \tag{D22}$$

This concludes our discussion of the upper bound on the infidelity.

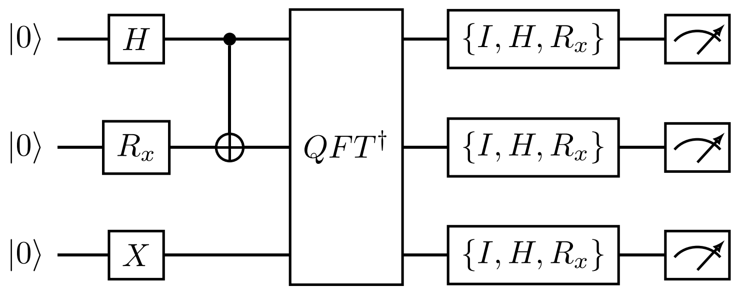

Figure 12. The FSL circuit used to load the complex-valued function $f(x) = (\cos(2\pi x) - 1.5i\cos(6\pi x))/\sqrt{13}$ on 3-qubit state. In order to verify the prepared state, we apply single qubit gates from $\{I, H, R_x(\pi/2)\}$ to each qubit before measuring it in the computational basis. We repeat this process for all operations in the set $\{I, H, R_x(\pi/2)\}^{\otimes 3}$ as depicted in the above figure. The results of these 27 experiments are processed to reconstruct the density matrix through maximum likelihood estimation.

Appendix E. Details of experiments on the quantiuum system model H1 quantum computers

In section 3, we presented the results of the experiments we performed on the Quantinuum H1-1 and H1-2 quantum computers. Here, we provide further details about those experiments and their implementation.

The first experiment we performed involved loading a complex-valued function $f(x) = (\cos(2\pi x) - 1.5i\cos(6\pi x))/\sqrt{13}$. Since this function is already in the form of a Fourier series, the pre-processing step of approximating a function by a truncated Fourier series is not needed. Moreover, since there is no truncation error in this case, the FSL method can load this function exactly. The FSL circuit used to load this function into a 3-qubit state is shown in figure 12. It is worthwhile to point out that this circuit is a special case of the circuit shown in figure 1(a) as the unitary U_c is acting on all of the $n = 3$ qubits. This, of course, is only possible for functions that are sufficiently sparse in Fourier space so that all the non-zero Fourier coefficients can be loaded with a shallow circuit.

To verify the state prepared by the circuit in figure 12, we performed quantum state tomography where we applied single-qubit gates from the set $\{I, H, R_x(\pi/2)\}$ on each of the qubits before measuring them. The reason for using H rather than $R_y(\pi/2)$ for rotating the measurement basis was the observation that the inverse QFT, used in the FSL circuit, ends with a H gate. Therefore, in some of the 27 experiments, we were able to ignore two successive H gates due to their cancellation.

As we discussed in section 3, we performed maximum likelihood estimation to reconstruct the density matrix from the measurement results. Here, we discuss the results of an alternate reconstruction of the state. Recall that a 3-qubit density matrix can be written as: $\rho = \frac{1}{8}\sum_{\mu_1=0}^{3}\sum_{\mu_2=0}^{3}\sum_{\mu_3=0}^{3} s_{\mu_1\mu_2\mu_3}\sigma_{\mu_1}\otimes\sigma_{\mu_2}\otimes\sigma_{\mu_3}$, where $\{\sigma_0, \sigma_1, \sigma_2, \sigma_3\} = \{I, \sigma_x, \sigma_y, \sigma_z\}$ are the standard Pauli matrices. Thus, the reconstruction of the state is reduced to the reconstruction of the coefficients $s_{\mu_1\mu_2\mu_3}$. Since $s_{\mu_1\mu_2\mu_3} = \mathrm{tr}\left(\rho\cdot\sigma_{\mu_1}\otimes\sigma_{\mu_2}\otimes\sigma_{\mu_3}\right)$ are just the expectation values of tensor products of Pauli operators, they can be estimated from the results of the measurements in 27 different bases. We found that the matrix reconstructed in this way has a fidelity of 94.0% which is almost the same as the fidelity we obtained through maximum likelihood estimation. However, this matrix is not a valid density matrix as it possesses some negative eigenvalues. Nevertheless, we find it encouraging that the fidelities obtained using two different methods are almost in agreement with each other.

In the rest of the experiments we performed, we used the FSL method to load $\sqrt{f(x)}$ into a quantum state for a given function $f(x)$ and then compared the amplitudes of the prepared state to the value of the target function. The precise expressions for the functions we considered are:

$$\text{Bi-modal Gaussian:} \quad f(x) \sim (1-\lambda)\,e^{-(x-1/4)^2/2\sigma^2} + \lambda e^{-(x-3/4)^2/2\sigma^2} \quad \text{with } (\lambda,\sigma) = (0.3, 0.1)\,, \tag{E1}$$

$$\text{Log-normal:} \quad f(x) \sim \frac{1}{x} e^{-(\log(x/q))^2/2\sigma^2} \quad \text{with } (q,\sigma) = (0.2, 0.5)\,, \tag{E2}$$

$$\text{Lorentzian:} \quad f(x) \sim \frac{1}{1 + (x-0.5)^2/\sigma^2} \quad \text{with } \sigma = 0.1\,, \tag{E3}$$

$$\text{‘Spiky’:} \quad f(x) \sim \left(\cos(4\pi x) + \lambda\cos(20\pi x)\right)^2 \quad \text{with } \lambda = 2.5\,, \tag{E4}$$

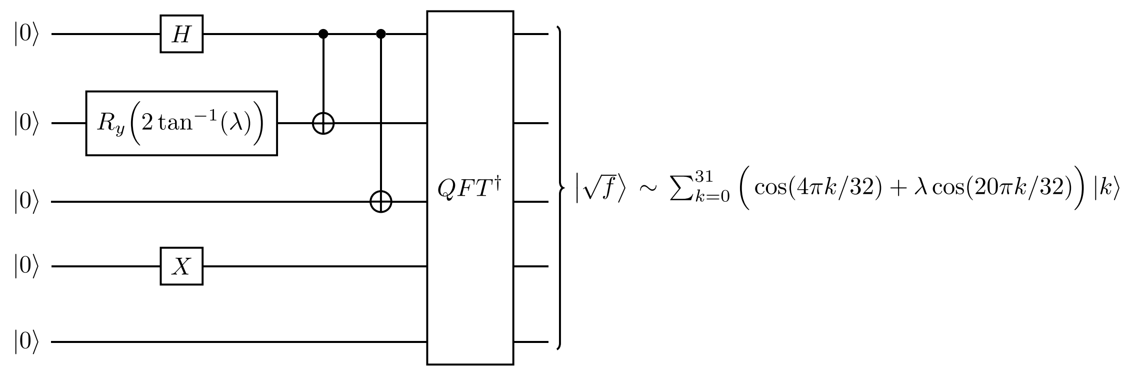
$|0\rangle$
H
$R_y\left(2\tan^{-1}(\lambda)\right)$
X
$QFT^\dagger$
$|\sqrt{f}\rangle \sim \sum_{k=0}^{31}\left(\cos(4\pi k/32)+\lambda\cos(20\pi k/32)\right)|k\rangle$

Figure 13. The quantum circuit to load the 'spiky' function in equation (E4) into a quantum state of 5-qubit. Since there were only two Fourier modes present in this case, we were able to construct an efficient to load the Fourier coefficients.

where $\sim$ denotes equality up to normalization.sssssssss In addition to these functions of a single variable, we also considered the following function of two variables:

$$f(x,y) \sim \exp\left(-(x-\mu_1)^2/\sigma_{11}^2-(y-\mu_1)^2/\sigma_{12}^2\right)+\lambda\exp\left(-(x-\mu_2)^2/\sigma_{21}^2-(y-\mu_2)^2/\sigma_{22}^2\right), \quad \text{(E5)}$$

with $\mu_1=0.65$, $\mu_2=0.35$, $\sigma_{11}=\sigma_{22}=\sqrt{1/50}$, $\sigma_{12}=\sqrt{1/40}$, $\sigma_{21}=\sqrt{1/30}$, and $\lambda=0.5$.

To load these functions on a quantum computer using the FSL method, we need to find the first few dominant Fourier coefficients of the square root of these functions. For bi-modal Gaussian function, log-normal distribution, and Lorentzian function, we used the first 8, 16, and 16 Fourier coefficients, respectively. For the two-variable function, we used the first 64 Fourier coefficients. Once the Fourier coefficients are determined, we need to find a suitable unitary U_c which can load these Fourier coefficients. For all the functions except the 'spiky' function in equation (E4), we implemented U_c using the Schmidt-decomposition circuit shown in figure 1(c). We first determined the unitary operators U and V in figure 1(c) by performing the Schmidt decomposition of the state $|\tilde{c}\rangle$ from equation (1). Then, we decomposed the unitary operators U and V into single and two-qubit gates using the built-in transpiler available in Qiskit [31].

On the other hand, when loading the 'spiky' function, we do not need to perform any such classical pre-processing given that only two Fourier modes are active. This is another instance in which the target state is suitably sparse in Fourier space such that we are able to construct a shallow quantum circuit to load all the non-zero Fourier coefficients. The full FSL circuit that we used to load the 'spiky' function is shown in figure 13.

Before we executed the FSL circuit on the Quantinuum H1-1 and H1-2 quantum computers, we reduced the gate count of the circuits as much as possible in order to reduce the effect of hardware noise. One obvious but crucial simplification that we exploited was to ignore all the SWAP gates that appear in the implementation of the inverse QFT. Instead of applying these SWAP gates, we manually changed the ordering of the qubits prior to applying the inverse QFT without SWAPs.

The experiments we performed on the Quantinuum H1-1 and H1-2 quantum computers were subject to various sources of noise. A list of sources of error present in System Model H1 quantum computers is available in the product data sheet [84]. For instance, the typical SPAM error is around 3×10^{-3}, whereas the typical infidelities for single qubit and two qubit gates are 4×10^{-5} and 3×10^{-3}, respectively.

Since we are making use of multiple mid-circuit measurements in our experiments, another source of error is the decoherence of the unmeasured qubits. This measurement crosstalk error is around 2×10^{-5} on average. Additionally, the error due to dephasing is 4×10^{-4} on average per depth-1 circuit execution time (the time required to apply a single round of single-qubit and two-qubit gates on all pairs of the qubits) [84].

The hardware performance specifications for the Quantinuum System Model H1 quantum computers provided in [84] are correct as of 3rd October 2022. However, we performed our experiments over the course of several months, and the hardware specifications fluctuated during that time. Therefore, we provide in table 2 the gate infidelities, measurement crosstalk errors, and SPAM errors as measured by Quantinuum for each of the dates we have performed the experiments [85].

Table 2. A record of hardware performance specifications for the Quantinuum System Model H1 quantum computers used on the days the experiments were performed. These specifications were measured by Quantinuum.

Dates	SPAM error (Average)	Measurement crosstalk error (Worst)	Single-qubit gate infidelity (Average)	Two-qubit gate infidelity (Average)	Experiments performed
4 December 2021	3.7×10^{-3}	5.0×10^{-4}	4.4×10^{-5}	3.0×10^{-3}	figure 5(b)
7 December 2021	3.4×10^{-3}	4.0×10^{-4}	3.9×10^{-5}	2.9×10^{-3}	figure 5(c)
15 December 2021	4.0×10^{-3}	2.0×10^{-4}	7.1×10^{-5}	2.1×10^{-3}	figures 5(b) and (c)
12 January 2022	3.3×10^{-3}	3.0×10^{-4}	5.9×10^{-5}	3.2×10^{-3}	figure 5(d)
17 February 2022	4.0×10^{-3}	6.0×10^{-4}	5.7×10^{-5}	2.9×10^{-3}	figure 6
14 April 2022	3.0×10^{-3}	4.0×10^{-4}	5.3×10^{-5}	2.9×10^{-3}	figure 6
29 April 2022	3.0×10^{-3}	3.0×10^{-4}	3.5×10^{-5}	2.5×10^{-3}	figure 5(a)
17 May 2022	2.9×10^{-3}	2.0×10^{-4}	5.3×10^{-5}	2.8×10^{-3}	figure 5(a)
18 May 2022	3.3×10^{-3}	2.0×10^{-4}	6.0×10^{-5}	2.5×10^{-3}	figure 6
21 June 2022	2.7×10^{-3}	2.0×10^{-4}	4.6×10^{-5}	2.3×10^{-3}	figure 4
22 June 2022	2.4×10^{-3}	4.0×10^{-4}	3.4×10^{-5}	2.4×10^{-3}	figure 4
23 June 2022	2.6×10^{-3}	1.0×10^{-4}	2.9×10^{-5}	2.1×10^{-3}	figure 4
12 July 2022	2.6×10^{-3}	2.0×10^{-4}	5.6×10^{-5}	2.4×10^{-3}	figure 6
22 July 2022	2.7×10^{-3}	2.0×10^{-4}	4.1×10^{-5}	2.1×10^{-3}	figure 4
9 August 2022	3.4×10^{-3}	2.0×10^{-4}	4.8×10^{-5}	2.0×10^{-3}	figure 4
23 August 2022	2.8×10^{-3}	1.0×10^{-4}	5.5×10^{-5}	2.0×10^{-3}	figure 6
23 September 2022	3.2×10^{-3}	1.0×10^{-4}	7.1×10^{-5}	3.4×10^{-3}	figure 6

Appendix F. Image loading using FSL method

Digital image processing is useful in many fields such as artificial intelligence, computer vision, and medical imagining to name a few. Quantum computers are expected to speed up various image processing tasks [86, 87] provided there exists an efficient way to load images on a quantum computer. In this appendix, we argue that the FSL method can be modified to load images on a quantum computer. This is not surprising as commonly used classical image compression algorithms, such as JHEP, are based on spectral transforms.

We can view a grayscale image $\mathcal{I}$ as a two-dimensional matrix whose elements $\mathcal{I}_{jk}$ store the brightness of the corresponding pixel. The matrix elements can be normalized such that $\mathcal{I}_{jk} = 0$ for black-colored pixels and $\mathcal{I}_{jk} = 1$ for white-colored pixels. For simplicity, we will assume that the image $\mathcal{I}$ is composed of $2^n \times 2^n$ pixels. One way to encode the image $\mathcal{I}$ on a quantum computer is the Flexible Representation of Quantum Images (FRQIs). FRQI stores image data as the following quantum state of $(2n+1)$-qubits [30]:

$$|\mathcal{I}\rangle = \frac{1}{2^n}\sum_{j=0}^{2^n-1}\sum_{k=0}^{2^n-1} |\mathcal{I}_{jk}\rangle \otimes |j\rangle \otimes |k\rangle\,, \tag{F1}$$

where

$$|\mathcal{I}_{jk}\rangle = \cos\left(\frac{\pi \mathcal{I}_{jk}}{2}\right)|0\rangle + \sin\left(\frac{\pi \mathcal{I}_{jk}}{2}\right)|1\rangle\,. \tag{F2}$$

It follows that, $|\mathcal{I}_{jk}\rangle = |0\rangle$ represents a purely black pixel, whereas $|\mathcal{I}_{jk}\rangle = |1\rangle$ represents a purely white pixel.

While the state $|\mathcal{I}\rangle$ could be prepared using uniformly controlled rotations, that would require $O(2^n)$ quantum gates [30]. We can do better by observing that the problem of preparing $|\mathcal{I}\rangle$ can be reduced to the problem of loading functions on a quantum computer. This becomes apparent if we rewrite $|\mathcal{I}\rangle$ in equation (F1) as

$$|\mathcal{I}\rangle = \frac{1}{2^n}|0\rangle \otimes \left[\sum_{j=0}^{2^n-1}\sum_{k=0}^{2^n-1} \cos\left(\frac{\pi \mathcal{I}_{jk}}{2}\right)|j\rangle\otimes|k\rangle\right] + \frac{1}{2^n}|1\rangle \otimes \left[\sum_{j=0}^{2^n-1}\sum_{k=0}^{2^n-1} \sin\left(\frac{\pi \mathcal{I}_{jk}}{2}\right)|j\rangle\otimes|k\rangle\right]. \tag{F3}$$

It follows that we may prepare $|\mathcal{I}\rangle$ by initializing the first qubit in a (non-uniform) superposition, and then loading $\cos\left(\frac{\pi\mathcal{I}}{2}\right)$ and $\sin\left(\frac{\pi\mathcal{I}}{2}\right)$ on the other $2n$ qubits when the first qubit is in $|0\rangle$ and $|1\rangle$ state, respectively. While controlled versions of any of the methods discussed in section 4 can be used to prepare the state $|\mathcal{I}\rangle$ as written in equation (F3), the FSL method is more suitable and efficient. This is because only a shallow depth portion of the quantum circuit used by the FSL method depends on the target function i.e. U_c in figures 1(a)

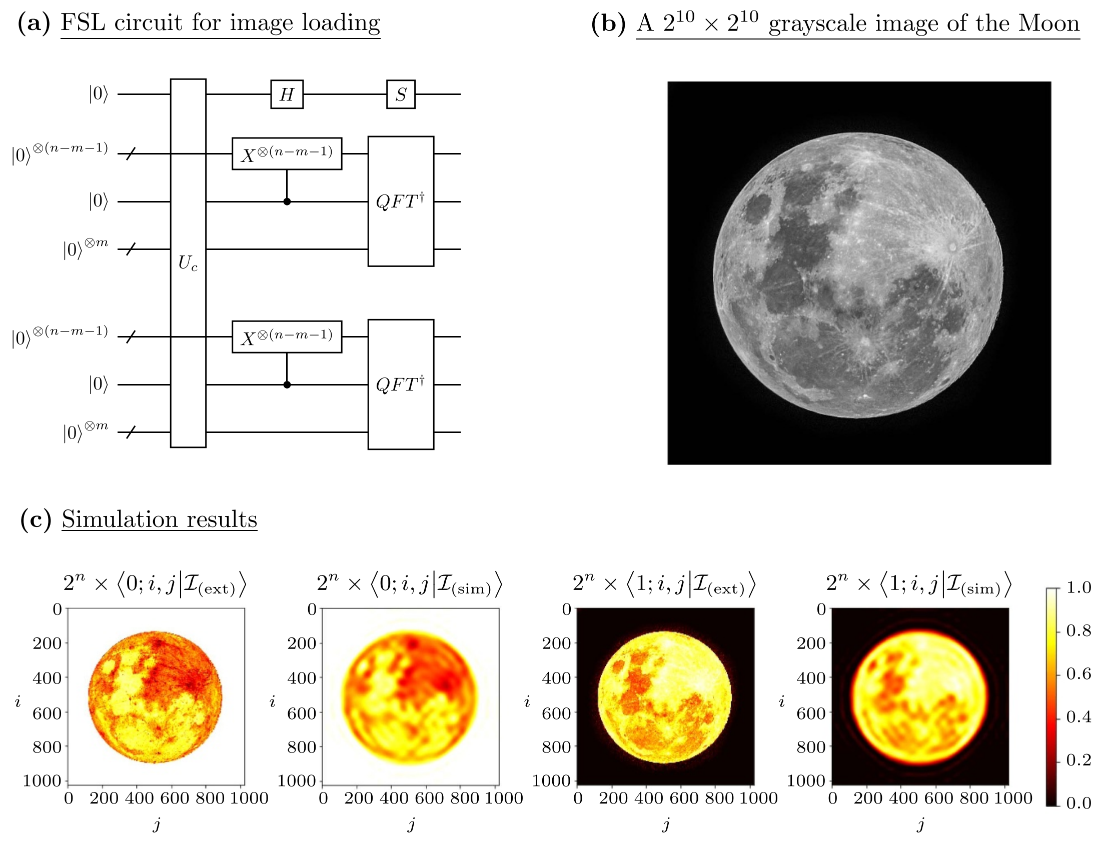
(a) FSL circuit for image loading
(b) A $2^{10} \times 2^{10}$ grayscale image of the Moon
(c) Simulation results
$|0\rangle$
$|0\rangle^{\otimes(n-m-1)}$
$|0\rangle$
$|0\rangle^{\otimes m}$
$|0\rangle^{\otimes(n-m-1)}$
$|0\rangle$
$|0\rangle^{\otimes m}$
U_c
H
S
$X^{\otimes(n-m-1)}$
$QFT^\dagger$
$2^n \times \langle 0; i, j | \mathcal{I}_{(\mathrm{ext})} \rangle$
$2^n \times \langle 0; i, j | \mathcal{I}_{(\mathrm{sim})} \rangle$
$2^n \times \langle 1; i, j | \mathcal{I}_{(\mathrm{ext})} \rangle$
$2^n \times \langle 1; i, j | \mathcal{I}_{(\mathrm{sim})} \rangle$
i
j
0
200
400
600
800
1000
0.0
0.2
0.4
0.6
0.8
1.0

Figure 14. (a) Quantum circuit to prepare the FRQI representation of an image using the FSL method. The first qubit stores the brightness of a pixel, the next n qubits store the horizontal coordinates of a pixel, and the last n qubits store the vertical coordinates of a pixel. (b) A grayscale image of the Moon [88] that we used as a test case for the FSL method. Reproduced from [88]. CC BY 4.0. (c) Simulation results of approximately preparing the FRQI representation of the image of the Moon using the FSL method with $m = 4$. As can be seen from these results, the state prepared by the FSL method $|\mathcal{I}_{(\mathrm{sim})}\rangle$ is a good approximation of the exact state $|\mathcal{I}_{(\mathrm{ext})}\rangle$ which can be further improved by taking higher values of m.

and 7. The rest of the FSL circuit is independent of the target function. Therefore, unlike other methods, we need only vary U_c instead of the entire circuit in order to prepare $|\mathcal{I}\rangle$. This renders the FSL method very efficient for loading images on quantum computers.

In order to construct the modified quantum circuit implementing the FSL method for image loading, notice that we can further rewrite the state $|\mathcal{I}\rangle$ as

$$|\mathcal{I}\rangle = \frac{1}{\sqrt{2}}|+i\rangle \otimes |g^+\rangle + \frac{1}{\sqrt{2}}|-i\rangle \otimes |g^-\rangle, \tag{F4}$$

where $|\pm i\rangle = \frac{1}{\sqrt{2}}|0\rangle \pm \frac{i}{\sqrt{2}}|1\rangle$, and

$$|g^\pm\rangle = \frac{1}{2^n}\sum_{j=0}^{2^n-1}\sum_{k=0}^{2^n-1} \exp\left(\mp\frac{i\pi\mathcal{I}_{jk}}{2}\right)|j\rangle \otimes |k\rangle. \tag{F5}$$

The advantage of writing the state $|\mathcal{I}\rangle$ in this way is that the Fourier coefficients of the state $|g^+\rangle$ and $|g^-\rangle$ are no longer independent. Hence, we only need to find the discrete Fourier coefficients of $|g^+\rangle$ which reduces the classical pre-processing time. If we now assume that $|g^\pm\rangle$ can be well-approximated by $4^{(m+1)}$ Fourier modes, then the state $|\mathcal{I}\rangle$ can be approximately prepared using the circuit shown in figure 14(a). In this circuit, the unitary U_c acts on $2(m+1)+1$ qubits as

$$U_c\left(|0\rangle \otimes |0\rangle^{\otimes 2(m+1)}\right) = |0\rangle \otimes |\tilde{c}^+\rangle + |1\rangle \otimes |\tilde{c}^-\rangle, \tag{F6}$$

where $|\tilde{c}^\pm\rangle$ are $2(m+1)$-qubit states encoding the Fourier coefficients of $|g^\pm\rangle$ and are analogous to the state $|\tilde{c}\rangle$ in equation (A12).

In order to assess the accuracy of the FSL method, we chose a $2^{10} \times 2^{10}$ pixel grayscale image of the Moon provided in figure 14(b). We performed a classical simulation of the version of the quantum circuit shown in figure 14(a) for loading the image of the Moon into a state of 21 qubits using the 'statevector_simulator' provided by Qiskit [31]. The resulting image-encoding state $|\mathcal{I}\rangle$ was loaded with the infidelity of around 10^{-2} for $m = 4$, and the results of the simulation are shown in figure 14(c).

ORCID iDs

Mudassir Moosa https://orcid.org/0000-0003-4510-4527
Thomas W Watts https://orcid.org/0000-0002-4588-997X
Peter L McMahon https://orcid.org/0000-0002-1177-9887

References

[1] Harrow A W, Hassidim A and Lloyd S 2009 Quantum algorithm for linear systems of equations *Phys. Rev. Lett.* **103** 150502
[2] Childs A M, Kothari R and Somma R D 2017 Quantum algorithm for systems of linear equations with exponentially improved dependence on precision *SIAM J. Comput.* **46** 1920
[3] Tong Y, An D, Wiebe N and Lin L 2021 Fast inversion, preconditioned quantum linear system solvers, fast Green's-function computation and fast evaluation of matrix functions *Phys. Rev.* A **104** 032422
[4] Clader B D, Jacobs B C and Sprouse C R 2013 Preconditioned quantum linear system algorithm *Phys. Rev. Lett.* **110** 250504
[5] Zalka C 1998 Simulating quantum systems on a quantum computer *Proc. R. Soc.* A **454** 313
[6] Grover L and Rudolph T 2002 Creating superpositions that correspond to efficiently integrable probability distributions (arXiv:quant-ph/0208112)
[7] Möttönen M, Vartiainen J, Bergholm V and Salomaa M 2005 Transformation of quantum states using uniformly controlled rotations *Quantum Inf. Comput.* **5** 467
[8] Plesch M and Brukner Č 2011 Quantum-state preparation with universal gate decompositions *Phys. Rev.* A **83** 032302
[9] Aaronson S 2015 Read the fine print *Nat. Phys.* **11** 291
[10] Benenti G and Strini G 2008 Quantum simulation of the single-particle Schrödinger equation *Am. J. Phys.* **76** 657
[11] Berry D W, Childs A M, Ostrander A and Wang G 2017 Quantum algorithm for linear differential equations with exponentially improved dependence on precision *Commun. Math. Phys.* **356** 1057
[12] Lubasch M, Joo J, Moinier P, Kiffner M and Jaksch D 2020 Variational quantum algorithms for nonlinear problems *Phys. Rev.* A **101** 010301
[13] Childs A M, Liu J-P and Ostrander A 2021 High-precision quantum algorithms for partial differential equations *Quantum* **5** 574
[14] Liu J-P, Kolden H Ø, Krovi H K, Loureiro N F, Trivisa K and Childs A M 2021 Efficient quantum algorithm for dissipative nonlinear differential equations *Proc. Natl Acad. Sci.* **118** e2026805118
[15] Lemieux J, Heim B, Poulin D, Svore K and Troyer M 2020 Efficient quantum walk circuits for Metropolis-Hastings algorithm *Quantum* **4** 287
[16] Wiebe N, Braun D and Lloyd S 2012 Quantum algorithm for data fitting *Phys. Rev. Lett.* **109** 050505
[17] Stamatopoulos N, Egger D J, Sun Y, Zoufal C, Iten R, Shen N and Woerner S 2020 Option pricing using quantum computers *Quantum* **4** 291
[18] Woerner S and Egger D J 2019 Quantum risk analysis *npj Quantum Inf.* **5** 15
[19] Markov V, Stefanski C, Rao A and Gonciulea C 2022 A generalized quantum inner product and applications to financial engineering (arXiv:2201.09845)
[20] Gilbert A C, Indyk P, Iwen M and Schmidt L 2014 Recent developments in the sparse Fourier transform: a compressed Fourier transform for big data *IEEE Signal Process. Mag.* **31** 91
[21] Lawlor D, Wang Y and Christlieb A 2013 Adaptive sub-linear time Fourier algorithms *Adv. Adapt. Data Anal.* **05** 1350003
[22] Ghazi B, Hassanieh H, Indyk P, Katabi D, Price E and Shi L 2013 Sample-optimal average-case sparse Fourier transform in two dimensions *2013 51st Annual Allerton Conf. on Communication, Control and Computing (Allerton)* p 1258
[23] Pawar S A and Ramchandran K 2013 Computing a k-sparse n-length discrete Fourier transform using at most 4k samples and O(k log k) complexity *2013 IEEE Int. Symp. on Information Theory* p 464
[24] Gleinig N and Hoefler T 2021 An efficient algorithm for sparse quantum state preparation *2021 58th ACM/IEEE Design Automation Conf. (DAC)* pp 433–8
[25] García-Ripoll J J 2021 Quantum-inspired algorithms for multivariate analysis: from interpolation to partial differential equations *Quantum* **5** 431
[26] García-Molina P, Rodríguez-Mediavilla J and García-Ripoll J J 2022 Quantum Fourier analysis for multivariate functions and applications to a class of Schrödinger-type partial differential equations *Phys. Rev.* A **105** 012433
[27] Zoufal C, Lucchi A and Woerner S 2019 Quantum Generative Adversarial Networks for learning and loading random distributions *npj Quantum Inf.* **5** 103
[28] Holmes A and Matsuura A Y 2020 Efficient quantum circuits for accurate state preparation of smooth, differentiable functions (arXiv:2005.04351)
[29] Endo K, Nakamura T, Fujii K and Yamamoto N 2020 Quantum self-learning Monte Carlo and quantum-inspired Fourier transform sampler *Phys. Rev. Res.* **2** 043442
[30] Le P Q, Dong F and Hirota K 2011 A flexible representation of quantum images for polynomial preparation, image compression and processing operations *Quantum Inf. Process.* **10** 63
[31] Anis M S *et al* 2021 Qiskit: an open-source framework for quantum computing (https://doi.org/10.5281/zenodo.2573505)
[32] Quantinuum 2021 (available at: www.quantinuum.com)
[33] Abhijith J *et al* 2018 Quantum algorithm implementations for beginners (arXiv:1804.03719)
[34] Krol A M, Sarkar A, Ashraf I, Al-Ars Z and Bertels K 2021 Efficient decomposition of unitary matrices in quantum circuit compilers (arXiv:2101.02993)
[35] Gonzalez-Conde J, Rodríguez-Rozas A, Solano E and Sanz M 2021 Simulating option price dynamics with exponential quantum speedup (arXiv:2101.04023)
[36] Hamming R W 1986 *Numerical Methods for Scientists and Engineers* 2nd edn (Dover) pp 534–6 (available at: https://store.doverpublications.com/0486652416.html)

[37] Gottlieb D, Shu C-W, Solomonoff A and Vandeven H 1992 On the gibbs phenomenon I: recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function *J. Comput. Appl. Math.* **43** 81

[38] Cai W, Gottlieb D and Shu C-W 1992 On one-sided filters for spectral Fourier approximations of discontinuous functions *SIAM J. Numer. Anal.* **29** 905

[39] Kopriva D A 1987 A practical assessment of spectral accuracy for hyperbolic problems with discontinuities *J. Sci. Comput.* **2** 249

[40] Proctor T, Seritan S, Rudinger K, Nielsen E, Blume-Kohout R and Young K 2021 Scalable randomized benchmarking of quantum computers using mirror circuits (arXiv:2112.09853)

[41] James D F V, Kwiat P G, Munro W J and White A G 2001 Measurement of qubits *Phys. Rev.* A **64** 052312

[42] Takeda K, Noiri A, Nakajima T, Yoneda J, Kobayashi T and Tarucha S 2021 Quantum tomography of an entangled three-qubit state in silicon *Nat. Nanotechnol.* **16** 965

[43] Of course, $|f_{\text{tomo}}\rangle$ is only defined up to a global phase. We fix this phase by demanding that $\langle f|f_{\text{tomo}}\rangle$ is real and positive

[44] Huang H-Y, Kueng R and Preskill J 2020 Predicting many properties of a quantum system from very few measurements *Nat. Phys.* **16** 1050

[45] Elben A, Flammia S T, Huang H-Y, Kueng R, Preskill J, Vermersch B and Zoller P 2022 The randomized measurement toolbox (arXiv:2203.11374)

[46] Torlai G, Mazzola G, Carrasquilla J, Troyer M, Melko R and Carleo G 2018 Neural-network quantum state tomography *Nat. Phys.* **14** 447

[47] Neugebauer M, Fischer L, Jäger A, Czischek S, Jochim S, Weidemüller M and Gärttner M 2020 Neural-network quantum state tomography in a two-qubit experiment *Phys. Rev.* A **102** 042604

[48] Cha P, Ginsparg P H, Wu F, Carrasquilla J F, McMahon P L and Kim E-A 2021 Attention-based quantum tomography *Mach. Learn.: Sci. Technol.* **3** 01LT01

[49] Zuo Y, Cao C, Cao N, Lai X, Zeng B and Du S 2021 All-optical neural network quantum state tomography (arXiv:2103.06457)

[50] Griffiths R B and Niu C-S 1996 Semiclassical Fourier transform for quantum computation *Phys. Rev. Lett.* **76** 3228

[51] Quantinuum 2022 Quantinuum system model H1 emulator (available at: https://assets.website-files.com/62b9d45fb3f64842a96c9686/6398c899bb181e5138578789_Quantinuum%20H1%20Emulator%20Product%20Data%20Sheet%20v6%2001DEC22.pdf)

[52] Ramos-Calderer S 2022 Efficient quantum interpolation of natural data (arXiv:2203.06196)

[53] Kompenhans M 2016 Adaptation strategies for discontinuous galerkin spectral element methods by means of truncation error estimations *Doctoral* Universidad Politécnica de Madrid (https://doi.org/10.20868/upm.thesis.40549)

[54] Marin-Sanchez G, Gonzalez-Conde J and Sanz M 2021 Quantum algorithms for approximate function loading (arXiv:2111.07933)

[55] Rattew A G and Koczor B 2022 Preparing arbitrary continuous functions in quantum registers with logarithmic complexity (arXiv:2205.00519)

[56] McArdle S, Gilyén A and Berta M 2022 Quantum state preparation without coherent arithmetic (arXiv:2210.14892)

[57] Kitaev A and Webb W A 2008 Wavefunction preparation and resampling using a quantum computer (arXiv:0801.0342)

[58] Rattew A G, Sun Y, Minssen P and Pistoia M 2021 The efficient preparation of normal distributions in quantum registers *Quantum* **5** 609

[59] Markov V, Stefanski C, Rao A and Gonciulea C 2022 A generalized quantum inner product and applications to financial engineering (arXiv:2201.09845)

[60] Oseledets I 2013 Constructive representation of functions in low-rank tensor formats *Constr. Approx.* **37** 1

[61] Schollwoeck U 2011 The density-matrix renormalization group in the age of matrix product states *Ann. Phys., NY* **326** 96

[62] Ran S-J 2020 Encoding of matrix product states into quantum circuits of one- and two-qubit gates *Phys. Rev.* A **101** 032310

[63] Rebentrost P, Steffens A, Marvian I and Lloyd S 2018 Quantum singular-value decomposition of nonsparse low-rank matrices *Phys. Rev.* A **97** 012327

[64] Aharonov D and Ta-Shma A 2003 Adiabatic quantum state generation and statistical zero knowledge (arXiv:quant-ph/0301023)

[65] Childs A M, Cleve R, Deotto E, Farhi E, Gutmann S and Spielman D A 2002 Exponential algorithmic speedup by quantum walk (arXiv:quant-ph/0209131)

[66] Berry D W, Ahokas G, Cleve R and Sanders B C 2007 Efficient quantum algorithms for simulating sparse Hamiltonians *Commun. Math. Phys.* **270** 359

[67] Häner T, Roetteler M and Svore K M 2018 Optimizing quantum circuits for arithmetic (arXiv:1805.12445)

[68] Haah J 2019 Product decomposition of periodic functions in quantum signal processing *Quantum* **3** 190

[69] Dong Y, Meng X, Whaley K B and Lin L 2021 Efficient phase-factor evaluation in quantum signal processing *Phys. Rev.* A **103** 042419

[70] Chao R, Ding D, Gilyen A, Huang C and Szegedy M 2020 Finding angles for quantum signal processing with machine precision (arXiv:2003.02831)

[71] Bauer C W, Deliyannis P, Freytsis M and Nachman B 2021 Practical considerations for the preparation of multivariate gaussian states on quantum computers (arXiv:2109.10918)

[72] Draper T G, Kutin S, Rains E M and Svore K M 2006 A logarithmic-depth quantum carry-lookahead adder *Quantum Inf. Comput.* **6** 351

[73] Cleve R and Watrous J 2000 Fast parallel circuits for the quantum Fourier transform (arXiv:quant-ph/0006004)

[74] Araujo I F, Park D K, Petruccione F and da Silva A J 2021 A divide-and-conquer algorithm for quantum state preparation *Sci. Rep.* **11** 6329

[75] Zhang X-M, Li T and Yuan X 2022 Quantum state preparation with optimal circuit depth: implementations and applications (arXiv:2201.11495)

[76] Stefanski C, Markov V and Gonciulea C 2022 Quantum amplitude interpolation (arXiv:2203.08758)

[77] Fijany A and Williams C P 1998 Quantum wavelet transforms: fast algorithms and complete circuits (arXiv:quant-ph/9809004)

[78] Li H, Fan P, Xia H, Song S and He X 2018 The multi-level and multi-dimensional quantum wavelet packet transforms *Sci. Rep.* **8** 13884

[79] Klappenecker A and Roetteler M 2001 Discrete cosine transforms on quantum computers (arXiv:quant-ph/0111038)

[80] Kay A 2018 Tutorial on the quantikz package (arXiv:1809.03842)

[81] Hunter J D 2007 Matplotlib: a 2D graphics environment *Comput. Sci. Eng.* **9** 90

[82] Virtanen P *et al* (SciPy 1.0 Contributors) 2020 SciPy 1.0: fundamental algorithms for scientific computing in Python *Nat. Methods* **17** 261

[83] Giardina C and Chirlian P 1972 Bounds on the truncation error of periodic signals *IEEE Trans. Circuit Theory* **19** 206

[84] Quantinuum 2022 Quantinuum system model h1 data sheet (available at: https://assets.website-files.com/62b9d45fb3f64842a96c9686/6398c899bb181e5138578789_Quantinuum%20H1%20Emulator%20Product%20Data%20Sheet%20v6%2001DEC22.pdf)
[85] These specifications were provided by Quantinuum upon request
[86] Zhang Y, Lu K and Gao Y 2014 QSobel: a novel quantum image edge extraction algorithm *Science China Inf. Sci.* **58** 1
[87] Yao X *et al* 2017 Quantum image processing and its application to edge detection: theory and experiment (arXiv:1801.01465)
[88] Owen M 2020 Photograph of the moon (available at: https://unsplash.com/photos/G9bDsVeHM7I?utm_source=unsplash&utm_medium=referral&utm_content=creditShareLink)