Quantum state preparation for multivariate functions

Matthias Rosenkranz[1], Eric Brunner[1], Gabriel Marin-Sanchez[1], Nathan Fitzpatrick[2], Silas Dilkes[2], Yao Tang[2], Yuta Kikuchi[3,4], and Marcello Benedetti[1]

[1]Quantinuum, Partnership House, Carlisle Place, London SW1P 1BX, United Kingdom
[2]Quantinuum, Terrington House, 13-15 Hills Road, Cambridge CB2 1NL, United Kingdom
[3]Quantinuum K.K., Otemachi Financial City Grand Cube 3F, 1-9-2 Otemachi, Chiyoda-ku, Tokyo, Japan
[4]Interdisciplinary Theoretical and Mathematical Sciences Program (iTHEMS), RIKEN, Wako, Saitama 351-0198, Japan

A fundamental step of any quantum algorithm is the preparation of qubit registers in a suitable initial state. Often qubit registers represent a discretization of continuous variables and the initial state is defined by a multivariate function. We develop protocols for preparing quantum states whose amplitudes encode multivariate functions by linearly combining block-encodings of Fourier and Chebyshev basis functions. Without relying on arithmetic circuits, quantum Fourier transforms, or multivariate quantum signal processing, our algorithms are simpler and more effective than previous proposals. We analyze requirements both asymptotically and pragmatically in terms of near/medium-term resources. Numerically, we prepare bivariate Student's t-distributions, 2D Ricker wavelets and electron wavefunctions in a 3D Coulomb potential, which are initial states with potential applications in finance, physics and chemistry simulations. Finally, we prepare bivariate Gaussian distributions on the Quantinuum H2-1 trapped-ion quantum processor using 24 qubits and up to 237 two-qubit gates.

1 Introduction

State preparation is a computational problem that appears as a sub-routine in many quantum algorithms. In Hamiltonian simulation, we approximate the real-time evolution $e^{-itH}|\psi\rangle$ of an initially prepared state $|\psi\rangle$. Quantum algorithms for linear systems of equations $A|x\rangle = |b\rangle$ require the preparation of $|b\rangle$ in the first place. Other examples are found in the literature of quantum algorithms for partial differential equations [1–3], financial instrument pricing [4–6], quantum chemistry simulations [7–12], high-energy or nuclear physics [13–15], and other applications [16].

The computational complexity of state preparation is well understood [17–22] and it is connected to that of synthesizing circuits for arbitrary unitary matrices. Algorithms for preparing generic n-qubit states incur an unavoidable exponential scaling, either in terms of circuit depth [23] or number of ancilla qubits [24]. Under certain assumptions, however, the scaling can be improved. For the special case of k-sparse states, i.e. states with k non-zero amplitudes, there exists an efficient preparation using $\log(nk)$ circuit depth and $\mathcal{O}(nk \log k)$ ancilla qubits [24]. For more structured states, such as when the amplitudes are defined by a function, there exist a number of efficient algorithms. For example, [25–30] consider certain classes of probability density functions; [31–33] consider real-valued functions approximated by polynomials; [34] considers complex-valued continuous functions. Quantum arithmetic circuits are among the most expensive components in state preparation algorithms. Recent work has aimed at limiting their use, potentially leading to a reduction of the number of qubits and gates. When the amplitudes are unknown and provided by a black-box, the use of arithmetic circuits can be limited to simple comparisons against some initial reference state [35–37], or replaced by linear combination of unitaries circuits [38]. When amplitudes are given by a known function instead, quantum signal processing (QSP) [39–43] can be used to avoid arithmetic circuits altogether [44].

In this work, we contribute to the state preparation problem by considering the case where the amplitudes are defined by multivariate functions. This is an essential step for many quantum computing applications of practical interest. For example, quantum algorithms for simulating wave propagation in D-dimensional space requires preparation of a D-dimensional function in the amplitudes of a quantum state as initial condition [3]. Another example is the simulation of quantum chemistry in first quantization [12]. Here, a typical initial state is an antisymmetrized many-body wave function represented in a D-dimensional discrete Fourier basis, which can be understood as a multivariate function. Despite its practical importance, the multivariate state preparation problem is rarely discussed in the literature and presents its unique challenges. For example, multivariate versions of QSP appear to be much more complicated and highly constrained [45–47]. We attack

Matthias Rosenkranz: matthias.rosenkranz@quantinuum.com
Marcello Benedetti: marcello.benedetti@quantinuum.com

the multivariate state preparation problem using a simple set of primitives and without using arithmetic circuits. We make use of (i) function approximation by truncated Fourier or Chebyshev series, (ii) block-encoding of the corresponding basis functions, and (iii) linear combination of unitaries. The resources required by our algorithms depend on three hyperparameters: the number of dimensions D, the number of bits n used for the discretization of each dimension, and the degree d used for function approximation in each dimension. We obtain a number of two-qubit gates that scales as $\mathcal{O}(d^D + Dn\log d)$ for Fourier series and as $\mathcal{O}\left(d^D + Ddn\log n\right)$ for Chebyshev series. The total number of qubits scales as $\mathcal{O}\left(Dn + D\log d\right)$ for both methods.

Some of the primitives we use were originally introduced by Childs et al. [48] in the context of quantum linear system solvers. In the context of state preparation the closest work to ours is the Fourier series loader (FSL) [49]. This method loads the Fourier coefficients in the main qubit register, hence avoiding the use of ancilla qubits. It then employs the quantum Fourier transform (QFT) to prepare the target state using additional $\mathcal{O}(n^2)$ gates. Related ideas of using the QFT for Fourier series interpolation of a target function from a coarse grid to a finer grid were earlier explored in [50]. In comparison, our method uses ancilla qubits, but it is more general, and, in the limit of large n, cuts down the number of two-qubit gates by avoiding QFT. A related method called the Walsh series loader (WSL) [51] approximates the target function by a Walsh series, which is loaded on the quantum computer using techniques from [52]. While WSL is shallower than FSL, it introduces a further approximation error which also affects the probability of success. Our method does not have this additional error. We provide a thorough comparison of multivariate state preparation techniques in Appendix D. The code implementing the presented protocols and the data to reproduce all results are available at GitHub and Zenodo [53].

2 Methods

We wish to prepare a quantum state with amplitudes proportional to a multivariate target function. We consider complex target functions, $f\colon [-1,1]^D \to \mathbb{C}$. We assume that f admits a Fourier or Chebyshev series representation. A Lipshitz condition on f is sufficient for uniform convergence of this series (in the Fourier case we also assume periodicity on $[-1,1]^D$) [54]. For simplicity of exposition, let us first discuss the one-dimensional case, $D = 1$. The Fourier and Chebyshev series of f are $f(x) = \sum_{k=-\infty}^{\infty} \hat{c}_k \mathrm{e}^{\mathrm{i}\pi kx}$ and $f(x) = \sum_{k=0}^{\infty} \hat{c}_k T_k(x)$, respectively, with $T_k(x)$ the k-th Chebyshev polynomial of the first kind. For the Fourier case, this requires the function to be periodic with period 2 on the whole interval $[-1,1]$. However, we also work with non-periodic target functions as Fourier series by focusing on the interval $[0,1]$ and using the interval $[-1,0)$ for its periodic extension (see Appendix B).

Next, we approximate the target function with a finite Fourier or Chebyshev series of the form

$$f_d^F(x) = \sum_{k=-d}^{d} c_k \mathrm{e}^{\mathrm{i}\pi kx} \quad \text{(Fourier)} \tag{1}$$

and

$$f_d^C(x) = \sum_{k=0}^{d} c_k T_k(x) \quad \text{(Chebyshev)}, \tag{2}$$

respectively. The largest index in the sum, d, is the *degree* of the polynomial (for which c_d or c_{-d} does not vanish). We use f_d to indicate either f_d^F or f_d^C, which should be clear from context. Two common methods for finding the coefficients c_k are *truncation* and *interpolation*. If the coefficients of the infinite series are known, truncation sets $c_k = \hat{c}_k$ for all $|k| \le d$. Interpolation matches $f_d(x_j)$ with $f(x_j)$ in $d+1$ (Chebyshev) or $2d+1$ (Fourier) interpolation points x_j. Computing the exact coefficients for this interpolant can be done with the fast Fourier transform (FFT) with classical preprocessing cost $\mathcal{O}(d\log d)$ (see Appendix A–B). Interpolation is often advantageous because the series coefficients $\hat{c}_k$ may not be known analytically for target functions of practical interest and numerically integrating their definition, Eq. (43), could introduce higher preprocessing cost or an additional error. The uniform error of interpolation is at most a factor of 2 larger than the uniform error of truncation [55]. A rule-of-thumb is that the smoother the target function the faster its finite series approximations converges with increasing d [55–57].

For the multivariate case, we consider approximations with products of the corresponding basis functions. For example, in two dimensions

$$f_{d_x,d_y}^F(x,y) = \sum_{k=-d_x}^{d_x} \sum_{l=-d_y}^{d_y} c_{k,l} \mathrm{e}^{\mathrm{i}\pi kx} \mathrm{e}^{\mathrm{i}\pi ly} \quad \text{(Fourier)} \tag{3}$$

and

$$f_{d_x,d_y}^C(x,y) = \sum_{k=0}^{d_x} \sum_{l=0}^{d_y} c_{k,l} T_k(x) T_l(y) \quad \text{(Chebyshev)}. \tag{4}$$

This generalizes to D dimensions with degrees $\boldsymbol{d} = (d_1, d_2, \ldots, d_D)$. The largest value in $\boldsymbol{d}$ is the *maximal degree* of the polynomial. The following protocols prepare a quantum state $\sum_{\boldsymbol{x}} g_{\boldsymbol{d}}(\boldsymbol{x}) |\boldsymbol{x}\rangle$ with $g_{\boldsymbol{d}}(\boldsymbol{x}) = f_{\boldsymbol{d}}(\boldsymbol{x}) / \sqrt{\sum_{\boldsymbol{x}} |f_{\boldsymbol{d}}(\boldsymbol{x})|^2}$ for all $\boldsymbol{x}$ on a grid of size $2^{n_1} \times 2^{n_2} \times \cdots \times 2^{n_D}$. A generic way to compute coefficients of $f_{\boldsymbol{d}}$ is via the multi-dimensional FFT

with preprocessing cost $\mathcal{O}[\bar{d}\log(\bar{d})]$ requiring at most $\bar{d}$ function evaluations, where $\bar{d} = \Pi_{j=1}^{D} d_j$. For more structured multivariate functions, other ways of finding coefficients may be more efficient in practice. For example, one could attempt to compute a low-rank approximation of f, e.g. via iterative Gaussian elimination, adaptive cross approximation etc. (see e.g. Ref. [58] for their usage in classical numerical computation). Our protocols are agnostic to this preprocessing and only require the coefficients of a finite Fourier or Chebyshev series as inputs.

2.1 Discretization

In this work, each variable is assigned to its own 'main' register. We use the notation x for the variable and n_x for the number of qubits in the register. The variable is therefore discretized on a grid of 2^{n_x} points. In the univariate setting, we simplify the notation to $n = n_x$.

Recall that when working with Fourier series approximations, we focus on the non-negative part of the domain, $x \in [0, 1]$. In this case the uniform grid of points is represented by the n-qubit diagonal matrix

$$H^{\mathrm{F}} = \frac{1}{2^n - 1} \begin{pmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 2^n - 1 \end{pmatrix}. \tag{5}$$

For example, with $n = 2$ we have grid points $0, \frac{1}{3}, \frac{2}{3}, 1$.

When working with Chebyshev series approximations we don't need periodic extensions. Thus we use the negative part of the domain as well, $x \in [-1, 1]$. The uniform grid of point is then represented by the n-qubit diagonal matrix

$$H^{\mathrm{C}} = 2H^{\mathrm{F}} - I^{\otimes n}, \tag{6}$$

where $I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ is the identity matrix. For example, with $n = 2$ we have grid points $-1, -\frac{1}{3}, \frac{1}{3}, 1$. It is important to note that this grid is uniform and should not be confused with the grid of Chebyshev nodes commonly used in polynomial interpolation.

In the following we often use H to indicate either H^{F} or H^{C}, which should be clear from the context. Note that both matrices (5) and (6) have the operator norm $\|H\| = 1$. Grids for multivariate functions are built by taking tensor products $H \otimes \cdots \otimes H$, and this does not change the operator norm.

2.2 Block-encoding

To construct the Fourier and Chebyshev approximations we shall block-encode the corresponding basis functions. We begin with the standard definition of block-encoding. An $(n + m)$-qubit unitary W_L is an (λ, m, ϵ)-block-encoding of $L \in \mathbb{C}^{2^n \times 2^n}$ if

$$\left\| L - \lambda \left(\langle 0|^{\otimes m} \otimes I^{\otimes n} \right) W_L \left(|0\rangle^{\otimes m} \otimes I^{\otimes n} \right) \right\| \leq \epsilon, \tag{7}$$

where m is the number of ancilla qubits, λ is a normalization constant such that $\lambda \geq \|L\| - \epsilon$ and ϵ is the error.

We make use of the linear combination of unitaries (LCU) [59] to construct block-encoding circuits. Assume the input matrix is given as a weighted sum of K unitary operators $\{U_k\}_k$, i.e. $L = \sum_{k=0}^{K-1} \lambda_k U_k$. To deal with negative and complex coefficients, we write them in polar form, $\lambda_k = \mathrm{e}^{\mathrm{i}\gamma_k} |\lambda_k|$, and compute the normalization $\lambda = \sum_{k=0}^{K-1} |\lambda_k|$. LCU uses $m = \lceil \log_2 K \rceil$ ancilla qubits and the following unitary operators

$$A = \sum_{k=0}^{K-1} \sqrt{\frac{|\lambda_k|}{\lambda}} \, |k\rangle\langle 0|^{\otimes m} \otimes I^{\otimes n} + \text{ u.c. }, \tag{8}$$

$$B = \sum_{k=0}^{K-1} |k\rangle\langle k| \otimes U_k, \tag{9}$$

$$C = \sum_{k=0}^{K-1} \mathrm{e}^{\mathrm{i}\gamma_k} \, |k\rangle\langle k| \otimes I^{\otimes n}, \tag{10}$$

where u.c. stands for unitary completion. With these definitions, the operator $A^\dagger CBA$ yields

$$\left(\langle 0|^{\otimes m} \otimes I^{\otimes n} \right) \, A^\dagger CBA \, \left(|0\rangle^{\otimes m} \otimes I^{\otimes n} \right) = \frac{1}{\lambda} \sum_{k=0}^{K-1} \lambda_k U_k, \tag{11}$$

which is a $(\lambda, m, 0)$-block-encoding of L.

Let us discuss the important special case where $U_k = U^k$ are powers of some unitary U. When $k \geq 0$ is encoded to the state as a binary string $|k\rangle = |k_{m-1}\rangle \otimes \cdots \otimes |k_0\rangle$, corresponding to the binary representation $k = 2^0 k_0 + 2^1 k_1 + \cdots + 2^{m-1} k_{m-1}$, the B operator in Eq. (9) takes the form

$$\sum_{k=0}^{K-1} |k\rangle\langle k| \otimes U^k = \prod_{i=0}^{m-1} \left[|0\rangle\langle 0|_i \otimes I^{\otimes n} + |1\rangle\langle 1|_i \otimes U^{2^i} \right] \tag{12}$$

where we highlight the relevant qubits with a subscript and assume identity gates on all other qubits. The identity in Eq. (12) is also discussed in Lemma 8 of [48], and is implicitly used in many other quantum algorithms, e.g. it is a key component of quantum phase estimation. This simple identity is illustrated in Fig. 1.

Note that the approximate functions we wish to encode, Eqs. (1) and (2), are weighted sums of basis functions. In the following sections we implement the k-th basis functions as the operator U_k and then use LCU to block-encode the whole weighted sum. The circuit construction for the operators A and C

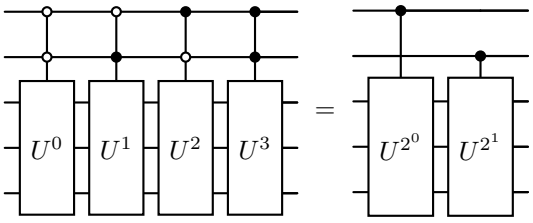

Figure 1: Circuit identity Eq. (12) for $\sum_k |k\rangle\langle k| \otimes U^k$ with $K = 4$ and $m = 2$.

does not change depending on whether the Fourier or Chebyshev basis is used, while the circuit construction for the operator B does.

2.3 Fourier basis functions

We seek a circuit construction for the Fourier basis B operator $B^F = \sum_{k=0}^{K-1} |k\rangle\langle k| \otimes U_k$ with U_k a block-encoding of the k-th Fourier basis function. As the univariate Fourier basis in Eq. (1) is defined as $\{e^{i\pi k' x}\}_{k'=-d}^{d}$, the k'-th Fourier basis function applied to H^F is the real-time evolution operator $U_{k'} = U^{k'} = e^{i\pi k' H^F}$. By the definition in Eq. (7), a unitary operator is a trivial $(1, 0, 0)$-block-encoding of itself. However, we need to show that there exists an efficient circuit to implement the operator for any value of k'. It can be verified by inspection that

$$e^{i\pi k' H^F} = \bigotimes_{j=0}^{n-1} P_j\left(\frac{\pi k' 2^j}{2^n - 1}\right), \tag{13}$$

where

$$P(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}, \tag{14}$$

is the phase shift gate and subscript j indicates application of the gate to the j-th qubit. This is a block-encoding circuit for any Fourier basis function and uses only n single-qubit phase shift gates.

To map this block-encoding to the special case discussed at the end of Sec. 2.2, we first write $U_k = U^{-d}U^k$ with $k = 0, 1, \ldots, 2d$. Next we implement the operator $\sum_{k=0}^{K-1} |k\rangle\langle k| \otimes e^{i\pi k H^F}$ using Eq. (12) with $U = e^{i\pi H^F}$ and an ancilla register of size $m = a = \log_2(2d + 1)$, and synthesize the circuit efficiently as

$$\sum_{k=0}^{K-1} |k\rangle\langle k| \otimes e^{i\pi k H^F} = \prod_{i=0}^{a-1} \prod_{j=0}^{n-1} \left[|0\rangle\langle 0|_i \otimes I_j + |1\rangle\langle 1|_i \otimes P_j\left(\frac{\pi 2^{j+i}}{2^n - 1}\right)\right]. \tag{15}$$

This can be verified using the factorization $e^{i\pi k H^F} = \prod_{i=0}^{a-1} e^{i\pi k_i 2^i H^F}$, followed by Eq. (13), and using $P_j(0) = I_j$. An example circuit diagram is provided in Fig. 2. Finally, we construct

$$B^F = U^{-d} \sum_{k=0}^{K-1} |k\rangle\langle k| \otimes e^{i\pi k H^F} \tag{16}$$

by applying the circuit Eq. (15) followed by the circuit for U^{-d} using Eq. (13) one more time. We have thus obtained a controlled $(1, 0, 0)$-block-encoding circuit for the Fourier basis functions. It requires only $a \times n$ controlled-phase gates and additional n single-qubit gates from the circuit for U^{-d}. A controlled-phase gate can be implemented with two controlled-not (CX) gates plus single-qubit gates.

2.4 Chebyshev basis functions

We seek a circuit construction for the Chebyshev basis B operator $B^C = \sum_{k=0}^{K-1} |k\rangle\langle k| \otimes U_k$ with U_k a block-encoding of the k-th Chebyshev polynomial. As the univariate Chebyshev basis in Eq. (2) is defined as $\{T_k(x) = \cos\big(k \cos^{-1}(x)\big)\}_{k=0}^{d}$, the k-th Chebyshev basis function applied to H^C is not a unitary operator. We proceed by first block-encoding H^C, and then using the algebra of block-encoding circuits to obtain $T_k(H^C)$.

Using Eqs. (5) and (6), we obtain

$$H^C = \sum_{j=0}^{n-1} \left(-\frac{2^j}{2^n - 1}\right) Z_j = \sum_{j=0}^{n-1} \left(\frac{2^j}{2^n - 1}\right) X_j Z_j X_j. \tag{17}$$

Here Z_j and X_j indicate the Pauli operators $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ and $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ applied to the j-th qubit, respectively. The result is a linear combination of n unitaries with which we can define a $(1, b, 0)$-block-encoding of H^C, $V = A_V^\dagger C_V B_V A_V$, where $C_V = I$ and

$$A_V = \sum_{j=0}^{n-1} \sqrt{\frac{2^j}{2^n - 1}}\, |j\rangle\langle 0|^{\otimes m} \otimes I^{\otimes n} + \text{ u.c. }, \tag{18}$$

$$B_V = \sum_{j=0}^{n-1} |j\rangle\langle j| \otimes X_j Z_j X_j. \tag{19}$$

Such a block-encoding requires $m = b = \lceil \log_2 n \rceil$ ancilla qubits.

Next, Lemma 8 in Ref. [60] shows that a block-encoding V can be used to derive another block-encoding U_V that satisfies the qubitization condition

$$U_V = \bigoplus_{\xi} \begin{pmatrix} \xi & -\sqrt{1 - \xi^2} \\ \sqrt{1 - \xi^2} & \xi \end{pmatrix}_{\xi}, \tag{20}$$

where ξ's are the eigenvalues of the block-encoded marix and each term is represented in the basis

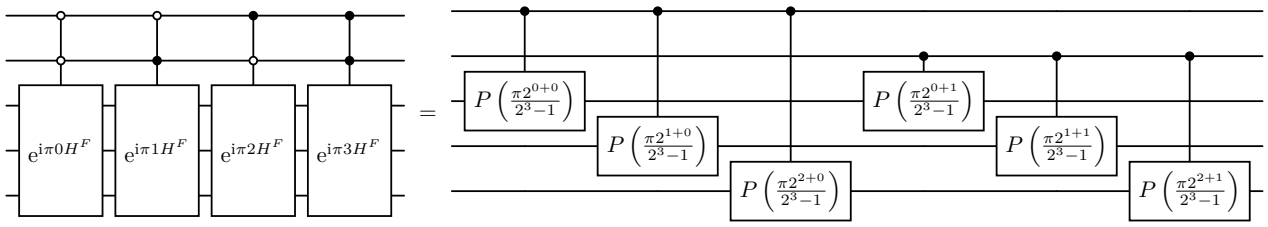

Figure 2: Example of circuit synthesis for the Fourier basis with Hamiltonian H^F, $a = 2$ ancilla and $n = 3$ main qubits. The special structure of this Hamiltonian leads to an efficient circuit made of $a \times n$ controlled-phase gates.

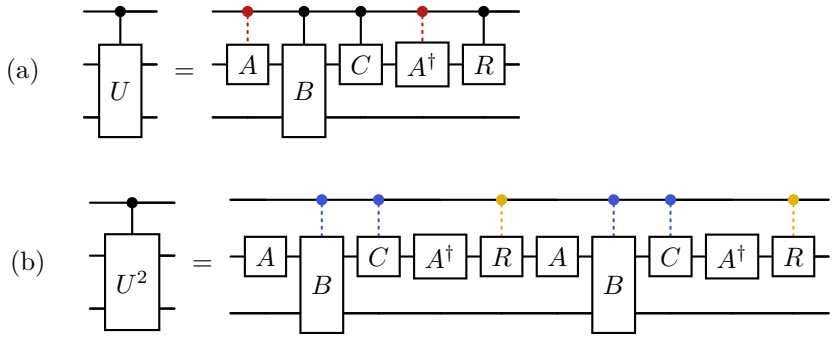

Figure 3: Simplifications used in the block-encoding circuit of Chebyshev basis functions. (a) For controlled-U, where U is any LCU circuit, one can remove the controls depicted in red. (b) For the specific case where $B^2 = I$ and $C^2 = I$ it is possible to simplify controlled-U^2 as well. One can remove either the blue or the yellow colored controls, depending on which option is more convenient. Our method uses controlled-U^{2^i} and thus benefits significantly from these techniques.

$\left\{|G_\xi\rangle, |G_\xi^\perp\rangle\right\}$ defined in Ref. [60]. Moreover, Corollary 9 in Ref. [60] shows that in the special case where $V^2 = I$ (which holds true here given that $B_V^2 = I$ and $C_V = I$) the qubitized block-encoding takes a simple form

$$U_V = \left(\left(2\,|0\rangle\langle 0|^{\otimes b} - I^{\otimes b}\right) \otimes I^{\otimes n}\right) V. \tag{21}$$

We now combine two results from the literature. The first result is Lemma 16 in Ref. [48]. It states that for positive integer k

$$\begin{pmatrix} \xi & -\sqrt{1-\xi^2} \\ \sqrt{1-\xi^2} & \xi \end{pmatrix}^k = \begin{pmatrix} T_k(\xi) & -\sqrt{1-\xi^2}\, u_{k-1}(\xi) \\ \sqrt{1-\xi^2}\, u_{k-1}(\xi) & T_k(\xi) \end{pmatrix}, \tag{22}$$

where u_k are Chebyshev polynomials of the second kind. For $k = 0$ we define the above expression as the identity matrix. The second result we use is Lemma 54 in Ref. [40] which deals with products of block-encodings. In the case where the encoded matrices are unitaries and their normalization constant is $\lambda = 1$, the product of $(1, b, 0)$-block-encodings yields the $(1, b, 0)$-block-encoding of the product. Moreover, since U_V is the direct sum in Eq. (20), powers of U_V preserve the qubitization condition. It follows that

$$U_V^k = \bigoplus_\xi \begin{pmatrix} T_k(\xi) & -\sqrt{1-\xi^2}\, u_{k-1}(\xi) \\ \sqrt{1-\xi^2}\, u_{k-1}(\xi) & T_k(\xi) \end{pmatrix}_\xi, \tag{23}$$

and $U_V^0 = I$. This is a $(1, b, 0)$-block-encoding of the k-th Chebyshev polynomial applied to H^C, i.e. $T_k(H^C) = (\langle 0|^{\otimes b} \otimes I^{\otimes n}) U_V^k (|0\rangle^{\otimes b} \otimes I^{\otimes n})$. Note that the algebra of qubitized block-encodings is also used in Ref. [61] to implement Krylov basis vectors and Lanczos' method on a quantum computer. We emphasize that our method works because $\lambda = \|H^C\| = 1$. For $\lambda \neq 1$ we would not obtain the correct result after rescaling the eigenvalues of H since $T_k(H/\lambda) \neq T_k(H)/\lambda$ (but see Section 5.1 of Ref. [62] for approximate solutions when this issue arises).

To construct the finite Chebyshev series Eq. (2) from the block-encoding circuit U_V^k we insert U_V^k into Eq. (9). To implement U_V^k controlled on the value of k we again use the identity in Eq. (12) with $m = a$ and obtain

$$\begin{aligned} B^C &= \sum_{k=0}^{K-1} |k\rangle\langle k| \otimes U_V^k \\ &= \prod_{i=0}^{a-1} \left[|0\rangle\langle 0|_i \otimes I^{\otimes n+b} + |1\rangle\langle 1|_i \otimes U_V^{2^i} \right]. \end{aligned} \tag{24}$$

We can save some resources when controlling U_V. First, we can remove the control on A_V and $A_V^\dagger$ since

they cancel when no operation is performed in between. Second, when controlling U_V^2, we can either remove the control on B_V or on $R = 2\,|0\rangle\langle 0|^{\otimes b} - I^{\otimes b}$. These improvements are shown in the circuit diagrams in Fig. 3 and for our purposes we remove the control on B_V.

To provide a gate count for B^C in Eq. (24), we start with a gate count for $|1\rangle\langle 1|_i \otimes U_V^2$ recalling $U_V = RA_V^\dagger C_V B_V A_V$. The operators A_V and $A_V^\dagger$ in U_V^2 can be implemented with a b-qubit state preparation circuit with real coefficients, requiring 2^b CX gates each [19]. B_V costs n b-qubit controlled Z gates, which we denote as C$^b Z$. Controlled R costs one C$^b Z$ gate. In total $|1\rangle\langle 1|_i \otimes U_V^2$ costs $2(n+1)$ C$^b Z$ gates and $2 \cdot 2^b$ CX gates. For Eq. (24) we need to implement 2^{i-1} copies of $|1\rangle\langle 1|_i \otimes U_V^2$. By summing up the gate counts for each copy we get the following totals for B^C: $\sum_{i=0}^{a-1} 2^i(n+1) = (2^a-1)(n+1)$ C$^b Z$ gates and $\sum_{i=0}^{a-1} 2^i \cdot 2^b = (2^a-1)2^b$ CX gates.

We wish to reduce gate counts to two-qubit gates only. Each C$^b Z$ gate can be implemented with $\mathcal{O}(b)$ gates [63], giving a total for B^C of $\mathcal{O}\left[(2^a-1)\left((n+1)b+2^b\right)\right] = \mathcal{O}(2^a nb)$ CX gates. Here we kept only dominant terms in d and n.

The best choice decompositions of C$^b Z$ will depend on b and the resources available, with alternative C$^b Z$ decompositions, such as in [64], or strategies for synthesising series of C$^b Z$ with varying control conditions, such as with the unary iteration circuits proposed in [65], potentially providing reduced CX count.

2.5 State preparation in 1 and 2 dimensions

We now utilize our building-blocks for the construction of quantum states defined by truncated series expansions. In this section we drop the superscripts F for Fourier and C for Chebyshev, since the protocols are almost identical. Then, the n-qubit discretization of the domain (2^n points) is denoted as H, and the block-encoding of the k-th basis function is denoted as U_k.

We begin with one-dimensional functions f approximated by either Eq. (1) or Eq. (2), here both denoted by f_d. Without further assumptions, the number of terms is $K = 2d+1$ for Fourier series and $K = d+1$ for Chebyshev series. Each basis function is associated to a complex coefficient c_k which we rewrite as $c_k = \mathrm{e}^{\mathrm{i}\phi_k}|c_k|$. Then

$$f_d(H) = \sum_{k=0}^{K-1} c_k U^k = \mathcal{N} \sum_k \mathrm{e}^{\mathrm{i}\phi_k} \frac{|c_k|}{\mathcal{N}} U^k, \tag{25}$$

where we normalize the coefficients with $\mathcal{N} = \sum_k |c_k|$. We encode the coefficients using operators A and C (Eqs. (8) and (10)), and an ancilla register of size $a = \lceil \log_2 K \rceil$. We then use the operator B defined in Eq. (16) for Fourier and Eq. (24) for Chebyshev noting, however, that the Chebyshev method requires additional b ancilla qubits. We thus obtain a $(\mathcal{N}, a+b, 0)$-block-encoding $A^\dagger CBA$ with

$$\left(\langle 0|^{\otimes(a+b)} \otimes I^{\otimes n}\right) A^\dagger CBA \left(|0\rangle^{\otimes(a+b)} \otimes I^{\otimes n}\right) = \frac{1}{\mathcal{N}} f_d(H). \tag{26}$$

We apply $A^\dagger CBA$ to $|0\rangle^{\otimes(a+b)} \otimes |+\rangle^{\otimes n}$ with the uniform superposition $|+\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle$ on the main register. Upon measuring the zero state for all the ancilla qubits, the protocol generates the unnormalized state $\frac{1}{\mathcal{N}\sqrt{2^n}} \sum_x f_d(x)\,|x\rangle$. The probability of success is given by $p_{\text{success}} = \frac{\sum_x |f_d(x)|^2}{\mathcal{N}^2 2^n}$. Renormalizing the state (dividing by $\sqrt{p_{\text{success}}}$) we obtain the desired result $\sum_x g_d(x)\,|x\rangle := \frac{1}{\sqrt{\sum_x |f_d(x)|^2}} \sum_x f_d(x)\,|x\rangle$. The full circuit is sketched in Fig. 4.

With the one-dimensional warm up, we have all the elements to construct states defined by bivariate functions $f(x,y)$. Again we start by discretizing the variables using n_x and n_y qubits in the main registers, which gives us a two-dimensional grid of $2^{n_x} \times 2^{n_y}$ points. We then approximate f as a finite series of degrees d_x and d_y. The coefficients are rewritten as $c_{k,l} = \mathrm{e}^{\mathrm{i}\phi_{k,l}}|c_{k,l}|$ where $k = 0, \ldots, K_x - 1$ and $l = 0, \ldots, K_y - 1$, and K_x, K_y depend on the series type and degree as before. The normalization constant becomes $\mathcal{N} = \sum_k \sum_l |c_{k,l}|$. The LCU operators are now defined as:

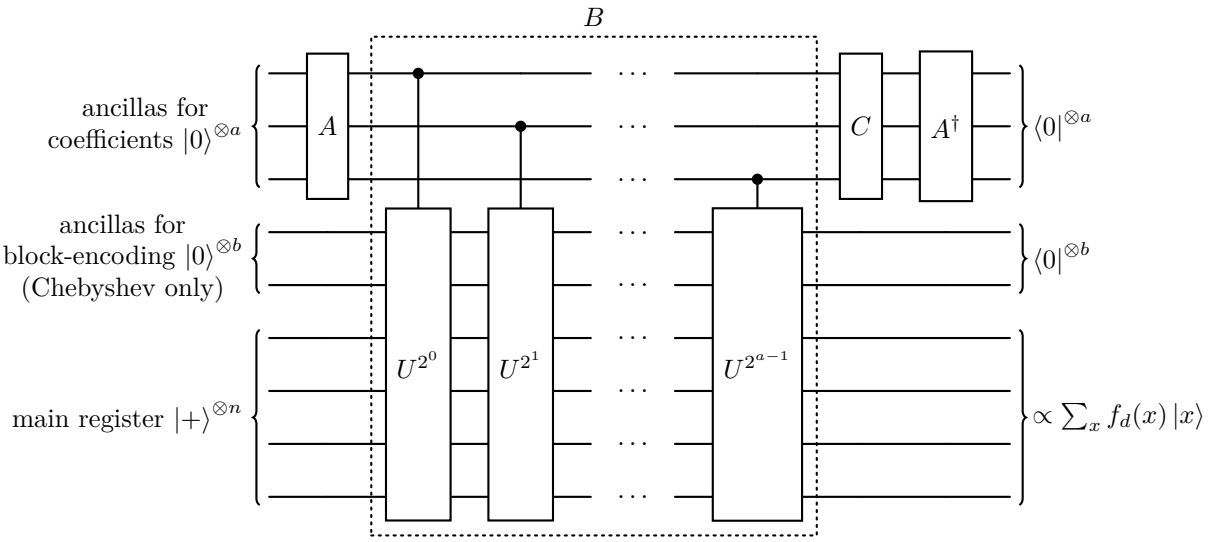

Figure 4: Quantum circuit for state preparation in one dimension. The amplitudes are defined by a finite series, such as Fourier or Chebyshev. Circuit U is the block-encoding of the corresponding basis functions and may require b additional ancilla qubits. Each bit in the coefficient register drives the application of powers of the block-encoding circuit. For example, in the subspace where the coefficient ancillas are in state $|k\rangle$, we apply U^k to the other two registers. The algorithm succeeds if we measure the zero state on all $a+b$ ancilla qubits.

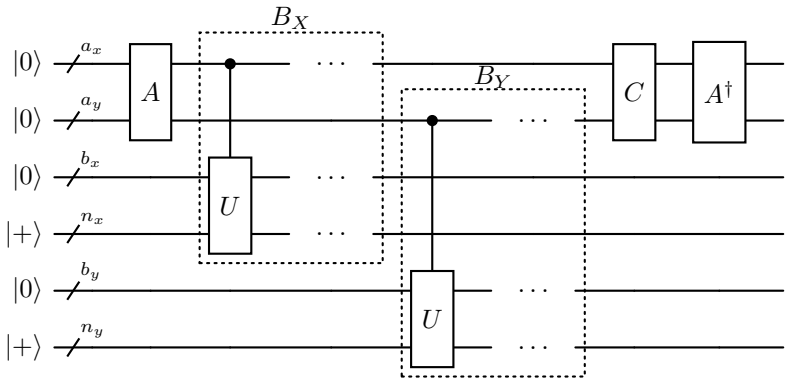

Figure 5: Quantum circuit for state preparation in two dimensions.

$$A = \sum_{k=0}^{K_x-1} \sum_{l=0}^{K_y-1} \sqrt{\frac{|c_{k,l}|}{\mathcal{N}}}\, |k\rangle\langle 0|^{\otimes a_x} \otimes |l\rangle\langle 0|^{\otimes a_y} \otimes I^{\otimes(b_x+b_y+n_x+n_y)} + \text{ u.c. }, \tag{27}$$

$$B_X = \sum_{k=0}^{K_x-1} |k\rangle\langle k| \otimes I^{\otimes a_y} \otimes U^k \otimes I^{\otimes(b_y+n_y)}, \tag{28}$$

$$B_Y = \sum_{l=0}^{K_y-1} I^{\otimes a_x} \otimes |l\rangle\langle l| \otimes I^{\otimes(b_x+n_x)} \otimes U^l, \tag{29}$$

$$C = \sum_{k=0}^{K_x-1} \sum_{l=0}^{K_y-1} \mathrm{e}^{\mathrm{i}\phi_{k,l}}\, |k\rangle\langle k| \otimes |l\rangle\langle l| \otimes I^{\otimes(b_x+b_y+n_x+n_y)}. \tag{30}$$

Applying the circuit to the uniform superposition of both main registers and post-selecting on the zero state of all the ancillas yields the desired state

$$\left(\langle 0|^{\otimes(a_x+a_y+b_x+b_y)} \otimes I^{\otimes(n_x+n_y)}\right) A^\dagger C B_Y B_X A \left(|0\rangle^{\otimes(a_x+a_y+b_x+b_y)} \otimes |+\rangle^{\otimes(n_x+n_y)}\right) = \frac{1}{\mathcal{N}\sqrt{2^{n_x+n_y}}} \sum_x \sum_y f_d(x,y)\,|x\rangle \otimes |y\rangle\,. \tag{31}$$

The protocol succeeds with probability

$$p_{\text{success}} = \frac{\sum_x \sum_y |f_d(x,y)|^2}{\mathcal{N}^2 2^{n_x+n_y}}. \tag{32}$$

To obtain the desired result we renormalize Eq. (31) by dividing by $\sqrt{p_{\text{success}}}$. The full circuit for bivariate state preparation is sketched in Fig. 5. The generalization of this protocol to arbitrary number of variables is straightforward.

2.6 Resource scaling for state preparation in arbitrary dimension

We can extrapolate the resource costs given in the previous sections for the one-dimensional case to estimate the scaling of state preparation in arbitrary dimensions. We focus on the number of qubits and CX gate count as a proxy for the two-qubit gate count, which is the main resource cost for our algorithm on near-term hardware. We neglect single-qubit gates.

Let us recall the resource requirements for state preparation with one variable ($D = 1$). The operator B costs $\mathcal{O}(an)$ CX gates for the Fourier approach (Sec. 2.3) and $\mathcal{O}\left(2^a nb\right)$ CX gates for the Chebyshev approach (Sec. 2.4). The size of the ancilla register a is determined by the number of coefficients in the approximating series f_d, namely, $a = \lceil \log_2(2d+1) \rceil$ for Fourier series and $a = \lceil \log_2(d+1) \rceil$ for Chebyshev series. The block-encodings of the Chebyshev basis functions require an additional ancilla register of size $b = \lceil \log_2 n \rceil$. A, $A^\dagger$, Eq. (8), and C, Eq. (10), can be implemented with a-qubit circuits requiring $\mathcal{O}\left(2^a\right)$ CX gates each.

We now provide the resource requirements for state preparation in arbitrary dimension. First, we assume that there are D variables, each discretized with n qubits, i.e. a total of Dn qubits in the main registers. Second, we assume the same degree d for each variable. The coefficients are encoded in ancilla registers $a_1, a_2, \ldots, a_D$ each with size a. Then the total number of ancilla qubits is $Da = D\lceil \log_2(2d+1) \rceil$ and $D(a+b) = D(\lceil \log_2(d+1) \rceil + \lceil \log_2 n \rceil)$ for the Fourier and Chebyshev approaches, respectively, where we inserted the respective expressions from the 1D case for a and b. The qubit counts are summarized in Table 1.

In general, the coefficients in a multivariate approximation f_d may take arbitrary values. Then the operators A, $A^\dagger$ and C can be implemented with Da-qubit circuits requiring $\mathcal{O}\left(2^{Da}\right)$ CX gates each. We have D operators $B_1, B_2, \ldots, B_D$. Each B_i is controlled only on the corresponding coefficient ancilla register a_i and acts only on the corresponding main register n_i. For the Chebyshev case each B_i also acts on the corresponding block-encoding ancilla register b_i, cf. Fig. 5. Hence, implementing all B_i is D times the 1D implementation cost of B: $\mathcal{O}(Dan)$ CX gates for the Fourier approach and $\mathcal{O}\left(D2^a nb\right)$ CX gates for the Chebyshev approach. Summing the costs for A, $A^\dagger$, C and all B_i and inserting the values for a and b yields $\mathcal{O}\left(2^{D\lceil \log_2(2d+1) \rceil} + Dn\lceil \log_2(2d+1) \rceil\right)$ CX gates for the Fourier approach and $\mathcal{O}\left(2^{D\lceil \log_2(d+1) \rceil} + Dn\lceil \log_2 n \rceil 2^{\lceil \log_2(d+1) \rceil}\right)$ CX gates for the Chebyshev approach. The simplified expressions $\mathcal{O}\left(d^D + Dn \log d\right)$ and $\mathcal{O}\left(d^D + Ddn \log n\right)$, respectively, are provided as the two-qubit gate count in Table 1. Appendix D and Table 4 compare our resource requirements to those of other state preparation methods for multivariate functions.

Some remarks are in order. First, if we wanted to boost the probability of success to unity, we could use amplitude estimation. That would increase the number of gates by a multiplicative factor of $\mathcal{O}\left(p_{\text{success}}^{-1/2}\right)$. Second, the degree d for each dimension must be chosen depending on the desired accuracy ε. The scaling of $d(\varepsilon)$ is problem-dependent and is not analyzed in this work. However, the protocol is efficient only if $d \in \mathcal{O}(\text{poly}(n))$, which limits the accuracy that can be achieved in general. Third, any method that implements a discretized version of the target function must rely on a grid of points. In our case, each qubit added to a main register increases the grid resolution exponentially, while increasing the number of two-qubit gates only mildly.

3 Results

With our first numerical simulations we visually inspect the block-encodings of Fourier and Chebyshev basis functions. In Fig. 6(a) we plot the diagonal of Eq. (13) for $k = 0, \ldots, 4$. In Fig. 6(b) we show the diagonal of $(\langle 0|^{\otimes b} \otimes I^{\otimes n}) U^k (|0\rangle^{\otimes b} \otimes I^{\otimes n})$ where U^k is given in Eq. (23). In both cases we use $n = 5$, leading to a x-axis discretization with 32 points. As expected the data points closely track $\mathrm{e}^{\mathrm{i}k\pi H}$ in Fig. 6(a), and $T_k(H)$ in Fig. 6(b).

3.1 2D Ricker wavelet via Chebyshev series

To illustrate the Chebyshev approach, we consider a 2D Ricker wavelet centered at $(0, 0)$,

$$f(x,y) = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{33}$$

Wavelets of this form are used, e.g. in image or seismic data applications [66–68].

We interpolate the function on the domain $[-1, 1]^2$ with the Chebyshev series Eq. (4) and compute its coefficients via the fast Fourier transform (see Appendix A). We choose the same degree and same grid size in each variable, i.e. $d_x = d_y = d$ and $n_x = n_y = n$. Figure 7 shows results for approximating the 2D Ricker wavelet with $\sigma = 0.5$ by a Chebyshev series of varying degree d and for different grid sizes $2^n \times 2^n$. Figure 7(a) illustrates the discretized target function and its absolute difference to

	Fourier	Chebyshev
Target function	$[0,1]^D \to \mathbb{C}$	$[-1,1]^D \to \mathbb{C}$
Classical preprocessing	Cost of computing coefficients, e.g. $\mathcal{O}(Dd^D \log d)$ for FFT	
Main qubits	Dn	Dn
Ancilla qubits	$D\lceil \log_2(2d+1) \rceil$	$D\lceil \log_2(d+1) \rceil + D\lceil \log_2 n \rceil$
Two-qubit gates	$\mathcal{O}(d^D + Dn \log d)$	$\mathcal{O}(d^D + Ddn \log n)$
Success probability	$\mathcal{N}^{-2} 2^{-Dn} \sum_{\boldsymbol{x}} \lvert f_d(\boldsymbol{x}) \rvert^2$	$\mathcal{N}^{-2} 2^{-Dn} \sum_{\boldsymbol{x}} \lvert f_d(\boldsymbol{x}) \rvert^2$

Table 1: Resources of our algorithm for preparing a quantum state with amplitudes proportional to a D-dimensional function represented on an equidistant grid of total size 2^{Dn}. The method uses a Fourier or Chebyshev polynomial approximation f_d with maximal degree d in each dimension. $\mathcal{N} = \sum_{\boldsymbol{k}} |c_{\boldsymbol{k}}|$ is a normalization dependent on the expansion coefficients $c_{\boldsymbol{k}}$.

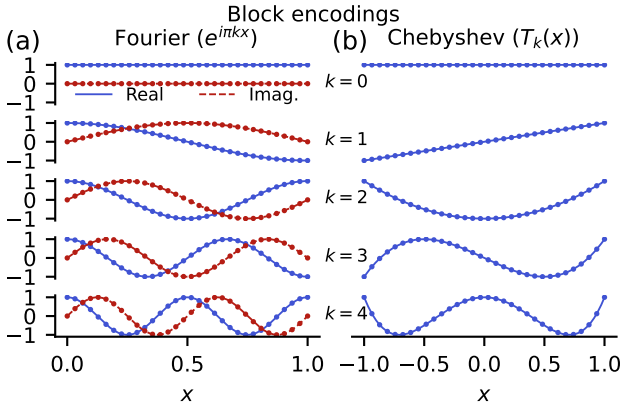

Figure 6: Block-encodings of the Fourier and Chebyshev basis functions. Markers are the diagonal of the Fourier block encoding, Eq. (13) in (a), and $(\langle 0|^{\otimes b} \otimes I^{\otimes n}) U^k (|0\rangle^{\otimes b} \otimes I^{\otimes n})$ with Chebyshev block encoding U^k, Eq. (23) in (b), for $n = 5$ qubits (32 grid points). Lines are the corresponding continuous basis functions $e^{i\pi kx}$ and $T_k(x)$.

the amplitudes of the prepared quantum state for degree $d = 7$ and grid size $2^4 \times 2^4$. For this comparison we rescale the amplitudes from a noiseless simulation of the algorithm appropriately,

$$\tilde{g}_d(x,y) = g_d(x,y) \sqrt{\sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} |f(x_i,y_i)|^2}, \qquad (34)$$

where x_i, y_i are the evaluated grid points. Figure 7(b) shows the maximum error $\max_{(x,y)} |f(x,y) - \tilde{g}_d(x,y)|$ where the maximum is over grid points. Increasing the degree leads to rapid convergence of the amplitudes to the target function. We observe that the error between the rescaled amplitudes and the Chebyshev approximation f_d is at machine precision (not shown). Hence, the convergence of our method to the target function is determined by the convergence of the Chebyshev approximation. The grey line shows the maximum error of the Chebyshev approximation, $\max_{(x,y)\in[-1,1]^2} |f(x,y) - f_d(x,y)|$, evaluated numerically over the whole domain (not just the grid) via global optimization. The line is consistent with supergeometric convergence to the target function. The numerics are floored at machine precision from around degree 30. The data from the algorithm's simulation is indeed upper bounded by this maximum error demonstrating that the accuracy tracks the Chebyshev approximation error. The dashed line illustrates a supergeometric bound $\mathcal{O}\left(e^{-d \log d}\right)$. This indicates that the accuracy ε is bounded by $\mathcal{O}\left(e^{-d \log d}\right)$. Hence, we can choose degree $d = \mathcal{O}\left(e^{W_0[\log(1/\varepsilon)]}\right) = \mathcal{O}\left[\log(1/\varepsilon)\right]$ for a fixed target error $0 < \varepsilon < 1$ with W_0 the principal branch of the Lambert W function. As a rule-of-thumb we expect at least geometric convergence to hold for analytic functions.

Figure 7(c) shows the number of two-qubit gates for varying degrees and different grid sizes. Circuits are compiled with `pytket` for the default native gate set of the H2-1 quantum computer at optimization level 2 [69]. The default native two-qubit gate is $ZZ\text{Phase}(\theta) = e^{-i\frac{\theta}{2} Z\otimes Z}$ with arbitrary angle θ. We perform a least-square fit of the data to the scaling of two-qubit gates derived in Sec. 2.6, $\alpha_0 + \alpha_1 D 2^{\lceil \log_2(d+1) \rceil} n \lceil \log_2 n \rceil + \alpha_2 2^{D\lceil \log_2(d+1) \rceil}$, where $\alpha_0, \alpha_1, \alpha_2$ are the fitted parameters and we added a constant offset. The fit in Fig. 7(c) shows good agreement. For large n and d this scaling law is upper bounded by the simplified expression in Table 1. The best parameters from the fit (see caption of Fig. 7) are in line with expectations from a worst-case analysis. For example, the term $\alpha_2 2^{D\lceil \log_2(d+1) \rceil}$ comes from the implementation of the operators A, C, $A^\dagger$ in Fig. 5 and the value $\alpha_2 = 3$ indicates that each of the operators is implemented with the expected worst-case cost $2^{D\lceil \log_2(d+1) \rceil}$. We note some residual variance in the coefficients α_0 and α_1 depending on n, which we attribute to suppressed, non-dominant terms and compiler optimizations. The total number of qubits can be computed from Table 1. For example, the largest circuit in Fig. 7(c) at $d = 63$, $n = 6$ requires 30 qubits (12 for the main register and 18 for

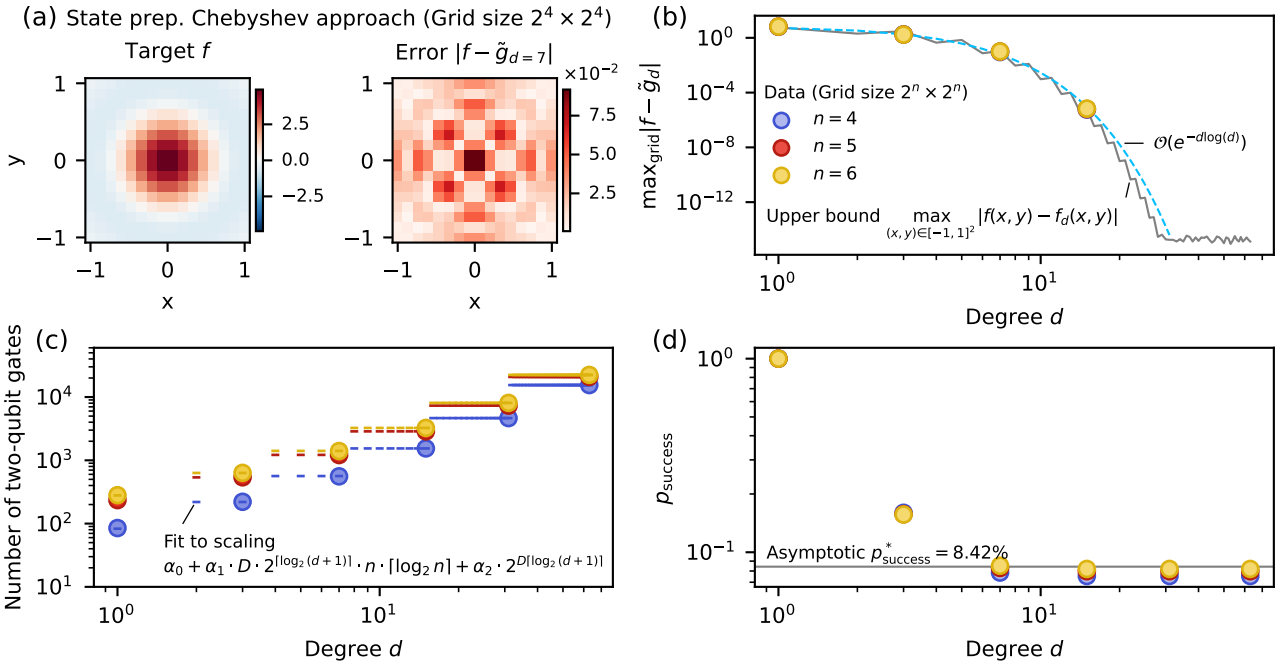

Figure 7: Resources for preparing an approximation to the 2D Ricker wavelet with products of Chebyshev polynomials of degree d in each dimension. (a) Discretized target function and the absolute difference to the rescaled amplitudes from a noiseless simulation of the algorithm for $d = 7$. (b) Maximum absolute error of the prepared state evaluated at the grid points for different grid sizes. The difference between the prepared state and the Chebyshev approximation is at machine precision (not shown). Hence, the maximum error of the noiseless simulation against the target function is upper bounded by the maximum error of the degree-d Chebyshev interpolant over the (non-discretized) domain (grey line – floored at machine precision). The dashed line illustrates a super-exponential asymptotic error bound $\mathcal{O}\left(\mathrm{e}^{-d \log d}\right)$. (c) Two-qubit gate count for different grid sizes compiled with `pytket` optimization level 2 for the H2-1 default gate set. The line markers are a least-square fit of the data (round markers) to the worst-case scaling with $\alpha_0 = -29.3, -47.3, -49.3$, $\alpha_1 = 3.14, 4.47, 4.39$, $\alpha_2 = 3.00, 3.00, 3.00$ for $n = 4, 5, 6$ and $D = 2$. (d) Empirical (markers) and asymptotic success probability for $n \to \infty$ and $d \to \infty$ (line).

ancilla registers). The success probability shown in Fig. 7(d) saturates at around 8%, which is close to the asymptotic success probability for $n \to \infty$, $d \to \infty$. We compute the asymptotic success probability as $p^*_{\text{success}} = \int_{[-1,1]^2} \mathrm{d}x\mathrm{d}y |f(x,y)|^2/4 \left(\sum_{k,l=0}^{30} |c_{k,l}| \right)^2$, which is the continuous version of Eq. (32). The sum in the numerator runs to $k, l = 30$ because higher-order coefficients do not contribute noticeably and fall off super-exponentially. The factor $1/4$ is from the size of the integration domain.

3.2 Bivariate Student's t-distribution via Fourier series

To illustrate the Fourier approach, we consider the bivariate non-standardized Student's t-distribution with one degree of freedom ($\nu = 1$, this is also known as the bivariate Cauchy distribution)

$$f(\boldsymbol{x}) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \left[1 + (\boldsymbol{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right]^{-\frac{3}{2}}, \tag{35}$$

where $\boldsymbol{x} = (x, y)^T$ and $\boldsymbol{\mu}$ and Σ are location and scale parameters, respectively. Student's t-distribution is characterized by heavy tails with applications in finance to model heavy-tailed return distributions [70] among others.

For the Fourier approach we focus on the domain $\boldsymbol{x} \in [0,1]^2$ and choose $\boldsymbol{\mu} = (0.5, 0.5)^T$ and, $\Sigma = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.05 \end{pmatrix}$. Note that the bivariate Student's t-distribution does not factorize into two univariate Student's t-distributions. First, we periodically extend the function from $[0,1]^2$ to $[-1,1]^2$, interpolate it with the Fourier series in Eq. (3) and compute coefficients with the fast Fourier transform following Appendix B. We choose the same degree and same grid size in each variable, i.e. $d_x = d_y = d$ and $n_x = n_y = n$. Figure 8 shows the results of approximating the periodically extended Student's t-distribution by a Fourier series of varying degree d and for different grid sizes $2^n \times 2^n$. The maximum error in Fig. 8(b) scales asymptotically as $\mathcal{O}(1/d)$. This is caused by the periodic extension introducing a discontinuity in the first derivative of the extended function at the boundaries $x = 0, 1$ and $y = 0, 1$. In the univariate case a rule-of-thumb is that for $k - 1$ times continuously differentiable functions and a k-th derivative with bounded variation, the asymptotic in-

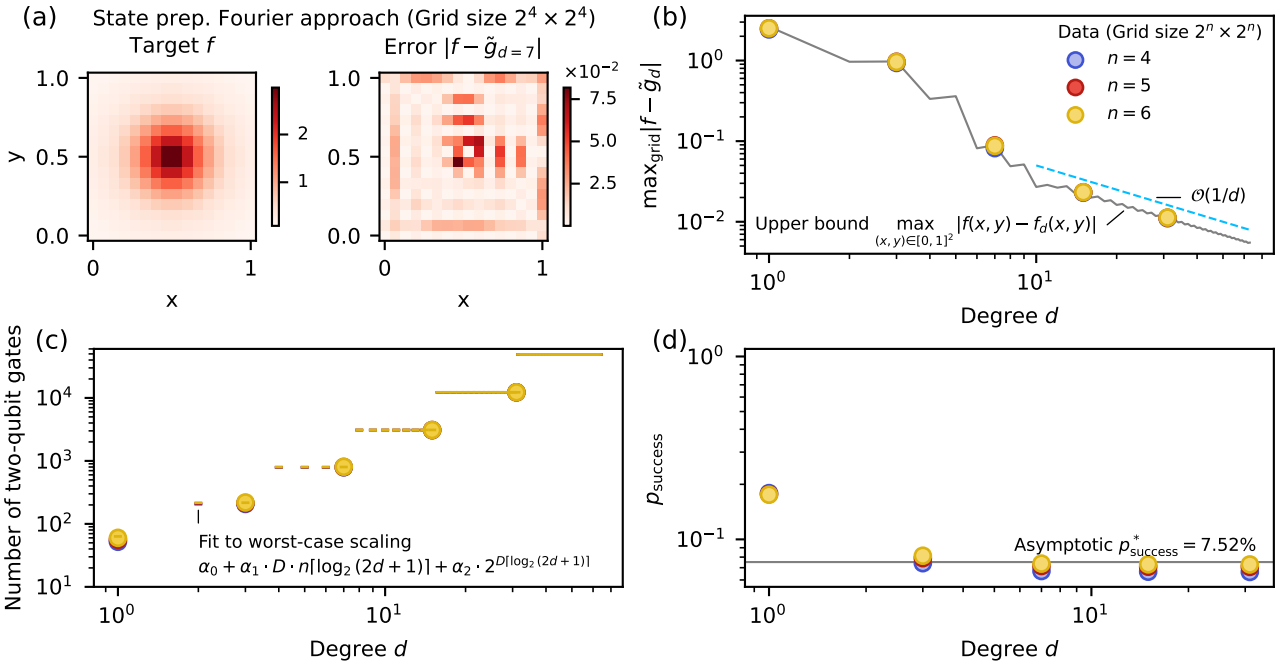

Figure 8: Resources for preparing an approximation to the bivariate Student's t-distribution with a 2D Fourier series approximation of degree d in each dimension. (a) Discretized target function and the difference to the rescaled amplitudes from a noiseless simulation of the algorithm for $d = 7$. (b) Maximum absolute error of the prepared state evaluated at the grid points for different grid sizes. The difference between the prepared state and the Fourier approximation is at machine precision (not shown). Hence, the maximum error of the noiseless simulation against the target function is upper bounded by the maximum error of the maximal degree-d Fourier interpolant over the (non-discretized) domain (grey line – floored at machine precision). The dashed line illustrates the asymptotic error bound $\mathcal{O}(1/d)$. (c) Two-qubit gate count of the algorithm for different grid sizes compiled with `pytket` optimization level 2 for the H2-1 gate set. The line markers are a least-square fit of the data (round markers) to the worst-case scaling with $\alpha_0 = -4.91, -4.91, -4.91$, $\alpha_1 = 0.78, 0.82, 0.85$, $\alpha_2 = 2.99, 2.99, 2.99$ for $n = 4, 5, 6$ and $D = 2$. (d) Empirical (markers) and asymptotic success probability for $n \to \infty$ and $d \to \infty$ (line).

terpolation error is $\mathcal{O}(1/d^k)$ [56, see Theorem 7.2 for an equivalent statement for Chebyshev interpolants]. Our numerics indicate this also holds for the bivariate case of the considered function. To reach higher convergence rate we could e.g. adjust the periodic extension method to smoothen the sharp corners at the boundaries.

Figure 8(c) shows the number of two-qubit gates after compilation to the Quantinuum native gate set with optimization level 2. A least-square fit of the data to the scaling of two-qubits gates derived in Sec. 2.6, $\alpha_0 + \alpha_1 Dn\lceil \log_2(2d+1) \rceil + \alpha_2 2^{D\lceil \log_2(2d+1) \rceil}$, shows excellent agreement. The small coefficients $\alpha_1 < 1$ (see caption of Fig. 8) indicate savings from the compilation step. As in the Chebyshev case, $\alpha_2 \approx 3$ indicates worst-case synthesis of the operators A, C, $A^\dagger$ in the LCU on the coefficient registers. The largest circuit in Fig. 8(c) at $d = 63$, $n = 6$ requires 26 qubits – 12 for the main register and 14 for the ancilla register, cf. Table 1. Similar to the Chebyshev case the success probability in Fig. 8(d) saturates close to its asymptotic value of $p^*_{\text{success}} = 7.52\%$. This is computed as $p^*_{\text{success}} = \int_{[0,1]^2} \mathrm{d}x\mathrm{d}y |f(x,y)|^2 / \left(\sum_{k,l=-63}^{63} |c_{k,l}| \right)^2$. The sum runs to degree 63 because the coefficients beyond this degree did not contribute noticeably (tested up to $k, l = 128$) and fall off with increasing degree.

3.3 Single electron in periodic 3D Coulomb potential

Our protocols are applicable even when the exact target function is unknown, but we have access to the coefficients of an approximation by other means, e.g. through the solution of a differential equation. To demonstrate this, in this section we use our Fourier approach for the preparation of wavefunctions of a single electron in a 3D translationally invariant Coulomb potential.

The prepared quantum state can serve as an initial state for the quantum simulation of materials in a unit cell of atoms with nuclear electronic Coulomb interactions. The unit cell wavefunction with cell volume equal to 1, is expressed as a linear combination of N^3 plane wave basis functions via

$$\psi(\boldsymbol{r}) = \sum_{\boldsymbol{k} \in G} c_{\boldsymbol{k}} \mathrm{e}^{\mathrm{i}\pi \boldsymbol{k}^T \boldsymbol{r}}. \tag{36}$$

Here, $\boldsymbol{k} = (k_x, k_y, k_z)^T$ runs over $G = \left[-\frac{N}{2}, \frac{N-1}{2} \right]^3 \subset \mathbb{Z}^3$, $\boldsymbol{r} = (x, y, z)^T \in [0,1]^3$

is the real-space position of the electron in the unit cell. Equation (36) is a trigonometric polynomial in 3 variables with maximal degree $N/2$ in each dimension, to which we apply our Fourier approach.

In order to obtain the coefficients $c_{\boldsymbol{k}}$, we insert the expansion (36) into the time-independent Schrödinger equation and solve the resulting set of linear equations. In the plane wave basis the kinetic energy matrix is diagonal with entries given by

$$T_{\boldsymbol{k},\boldsymbol{k}} = \frac{(\hbar\pi\|\boldsymbol{k}\|)^2}{2m} |\boldsymbol{k}\rangle\langle\boldsymbol{k}| \,. \tag{37}$$

The electron-nuclear attraction matrix elements have the following convenient analytical form in the basis of plane waves

$$U_{\boldsymbol{k},\boldsymbol{k}'} = \frac{4}{\pi} \sum_{\ell=1}^{L} w_\ell \frac{e^{i\pi(\boldsymbol{k}'-\boldsymbol{k})^T \mathbf{R}_\ell}}{\|\boldsymbol{k}-\boldsymbol{k}'\|^2} |\boldsymbol{k}\rangle\langle\boldsymbol{k}'| \,. \tag{38}$$

$\mathbf{R}_\ell$ is the position of nucleus ℓ in real space and w_ℓ is the corresponding nuclear weight and L is the number of nuclei. The π factors arise from the definition of the state in Eq. (36).

The electronic Hamiltonian matrix elements are then given by $H^{(\mathrm{el})}_{\boldsymbol{k},\boldsymbol{k}'} = -T_{\boldsymbol{k},\boldsymbol{k}}\delta_{\boldsymbol{k},\boldsymbol{k}'} - U_{\boldsymbol{k},\boldsymbol{k}'}$ and the time-independent Schrödinger equation becomes $H^{(\mathrm{el})}\boldsymbol{c} = E\boldsymbol{c}$ with $\boldsymbol{c}$ the vector of coefficients in Eq. (36). Solutions $\boldsymbol{c}$ for the first and second lowest eigenenergies result in the upper bounds to the ground and excited state wavefunction of the non-interacting single electronic system. We use those coefficients in our multivariate Fourier approach to prepare the quantum states for the ground and first excited state wavefunctions.

We choose the simplest 3D periodic unit cell for demonstration, which is a single Coulomb potential of nuclear weight and charge 1 at the center of a (1×1×1) unit cell at position $(0.5, 0.5, 0.5)$. Other more advanced unit cells can be used, such as face-centered cubic (FCC). In the left panels of Fig. 9 we present the probability density $|\psi(\boldsymbol{r})|^2$ of the ground (top) and first excited state wavefunctions (bottom) computed directly with Eq. (36) and coefficients from the solution of the time-independent Schrödinger equation above. We then use the same set of coefficients to construct the circuits of our Fourier-based protocol for preparing those wavefunctions. In the right panels of Fig. 9 we present the probability density of the resulting wavefunctions obtained from a statevector simulation of those circuits. The simulated circuits replicate to numerical precision the results from the direct numerical computation.

Table 2 shows the resource requirements from compilation of the circuits of our method for preparing the wavefunctions with different numbers of grid points and coefficients. `pytket` [69] is used to construct the circuits, using the gateset $\{CX, R_Z, R_X\}$. This confirms that the dominant cost of the algorithm comes

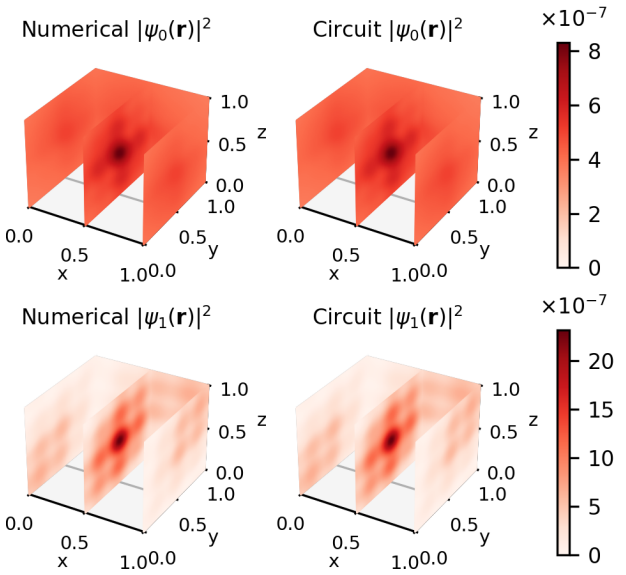

Figure 9: Probability density of the ground state (top panels) and first excited state (bottom) electronic wavefunctions for a unit cell with a single nucleus at the center at $(0.5, 0.5, 0.5)$. Left panels: direct evaluation of Eq. (36) with a total of $N^3 = 512$ coefficients from the numerical solution of the time-independent Schrödinger equation. Right panels: equivalent solutions from the Fourier approach of our method evaluated with a statevector simulator using $n = 7$ qubits per spatial dimension and $a = 3$ ancilla qubits per dimension for a total of 30 qubits.

from the addition of the ancilla qubits that contain the Fourier coefficients, which is expected from Table 1. Increasing the spatial resolution (given by 2^{3n}, where n is the number of qubits per spatial dimension), leads to little increase in gate count due to the efficient control structure presented in Fig. 2. Using a total of 9 and 12 ancilla qubits (i.e. registers of size $a = 3, 4$ per dimension) allows one to encode 512 and 4096 Fourier coefficients, respectively. Thus, the large increase in the number of gates is due to the 9 vs. 12 qubit circuits for A, C and $A^\dagger$ (cf. the term $\mathcal{O}(d^D)$ for two-qubit gates in Table 1). For the 30 qubit case marked with *, the success probability of generating the ground (ψ_0) and first excited (ψ_1) states are 0.7047 and 0.2392, respectively.

Although these are only single electron wavefunctions, they can be used as a starting point for many-electron wavefunctions by anti-symmetrizing the spatial registers of multiple single electron states following the procedure proposed in Ref. [12], or as the initial state for real or imaginary time evolution under an interacting electron Hamiltonian in the first quantized picture.

3.4 Bivariate Gaussian on quantum hardware

This section demonstrates our state preparation protocol on the H2-1 trapped-ion quantum computer and illustrates the effect of noise. We choose to use the

n	a	D	Grid points (2^{Dn})	Coefficients (2^{Da})	Total qubits ($Dn + Da$)	Two-qubit gates
4	3	3	4096	512	21	1602
5	3	3	32 768	512	24	1620
6	3	3	262 144	512	27	1638
* 7	3	3	2 097 152	512	30	1656
7	4	3	2 097 152	4096	33	12 450

Table 2: Circuit compilation results for the solutions of the Schrödinger equation in a 3D periodic Coulomb potential with varying numbers of grid points and Fourier coefficients. With n the number of qubits per spatial dimension and $a = \lceil \log_2 N \rceil$ the number of ancillas per dimension the total number of qubits is $3n + 3a$. The number of two-qubit gates is determined by compiling the circuits to the gate set given in the main text. The row marked with * corresponds to the setting of Fig. 9.

		$\mu_{d,x}, \hat{\mu}_{d,x}$	$\mu_{d,y}, \hat{\mu}_{d,y}$	$\sigma^2_{d,x}, \hat{\sigma}^2_{d,x}$	$\sigma^2_{d,y}, \hat{\sigma}^2_{d,y}$	$\rho_d, \hat{\rho}_d$	F_x	F_y	F
Uncorrelated Gaussian	$\mathfrak{f}_d(x,y)$	0.5	0.5	0.0413	0.0299	0.0	0.998	0.984	0.986
	$\hat{\mathfrak{f}}_d(x,y)$	0.494	0.495	0.0468	0.0396	−0.015			
Correlated Gaussian	$\mathfrak{f}_d(x,y)$	0.5	0.5	0.0415	0.0321	0.353	0.991	0.963	0.918
	$\hat{\mathfrak{f}}_d(x,y)$	0.493	0.488	0.0519	0.0510	0.088			

Table 3: Experimental results for preparing a 2D Fourier series approximation of the bivariate Gaussian distribution, see Fig. 10. We consider two settings, the preparation of an uncorrelated and a correlated Gaussian. The table collects, for both settings, mean values, variances and correlation coefficients [cf. Eq. (40)] of the target densities $\mathfrak{f}_d(x,y)$ and their estimates $\hat{\mathfrak{f}}_d(x,y)$, based on kernel density estimation. Moreover, we show the fidelities F_x, F_y [cf. Eq. (41)] between marginal estimates, $\hat{\mathfrak{f}}_{d,x}, \hat{\mathfrak{f}}_{d,y}$, and target functions' marginals, $\mathfrak{f}_{d,x}, \mathfrak{f}_{d,y}$, and the full fidelity F between $\hat{\mathfrak{f}}_d$ and $\mathfrak{f}_d$ [cf. Eq. (41)].

Fourier basis functions as they lead to circuits with fewer two-qubit gates. We focus on analyzing the output signal without any error mitigation. Error mitigation protocols, if needed, would likely be performed in combination with a downstream quantum algorithm, not after state preparation, which is only the first step of the computation.

We choose a two-dimensional Gaussian distribution as a target function. The low overheads of our method allow us to prepare the Gaussian on a fine grid with 24 qubits using up to 237 two-qubit gates with meaningful output signal. The Gaussian distribution is given by

$$f(\boldsymbol{x}) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \exp\left[-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right], \tag{39}$$

where $\boldsymbol{x} = (x,y)^T$, $\boldsymbol{\mu}$ is the mean and Σ is the positive semi-definite covariance matrix. Let $Z = X \times Y$ be a 2D random variable distributed according to $f(x,y)$. The covariance matrix can be parametrized by the marginal variances $\sigma_x^2 = \mathrm{Var}(X)$ and $\sigma_y^2 = \mathrm{Var}(Y)$ (diagonal entries of Σ), and the (Pearson) correlation coefficient $\rho = \mathrm{Cov}(X,Y)/\sigma_x\sigma_y$ (setting off-diagonal entries $\rho\sigma_x\sigma_y$). For this demonstration, we use $\boldsymbol{\mu} = (\mu_x, \mu_y)^T = (0.5, 0.5)^T$, $\sigma_x = 0.22, \sigma_y = 0.18$, and the two choices $\rho = 0.0$ and $\rho = 0.4$. The first setting defines an uncorrelated Gaussian, i.e. the bivariate density $f(x,y)$ is the product of two univariate densities. The second setting defines a correlated Gaussian, which does not factorize.

For both settings we use $n_x = n_y = 9$ qubits for spatial discretization of the domain $[0,1]^2$ (cf. Fig. 5) into $512 \times 512 = 262\,144$ grid points (for comparison, Ref. [49] prepared a bivariate function on a grid of size 32×32). We use a Fourier series approximation $f_d(x,y)$ of degrees $d = d_x = d_y = 3$, which leads to a total of $(2d+1)^2 = 49$ coefficients. Note that we use a different method to compute the Fourier coefficients than the one used in Sec. 3.2 because the approximation at the chosen degree is better for the specific target, see Appendix C.2. To encode the Fourier coefficients we require ancilla registers with $a_x = a_y = 3$ qubits each, cf. Fig. 5. Overall, this results in a circuit with 24 qubits. We test our approach on the H2-1 trapped-ion quantum computer [71, 72]. At the time of this demonstration in April and May 2024, it provides 32 fully-connected qubits with average single- and two-qubit gate infidelities of 3×10^{-5} and 2×10^{-3}, respectively, and average state preparation and measurement error of 2×10^{-3}. Detailed specifications are given in Appendix C.1.

The Fourier approximation $f_{d=3}(x,y)$ of the (positive) Gaussian density in Eq. (39) can have noticeable negative values in $[0,1]^2$. This is visible in the small kinks of the red dashed line (showing a marginal over $|f_d(x,y)|$) in Fig. 10(c) close to the boundaries of the domain $[0,1]$. To avoid expensive state-tomographic methods, we only read out the magnitudes $|f_d(x,y)|$ of the Fourier series via measurements in the compu-

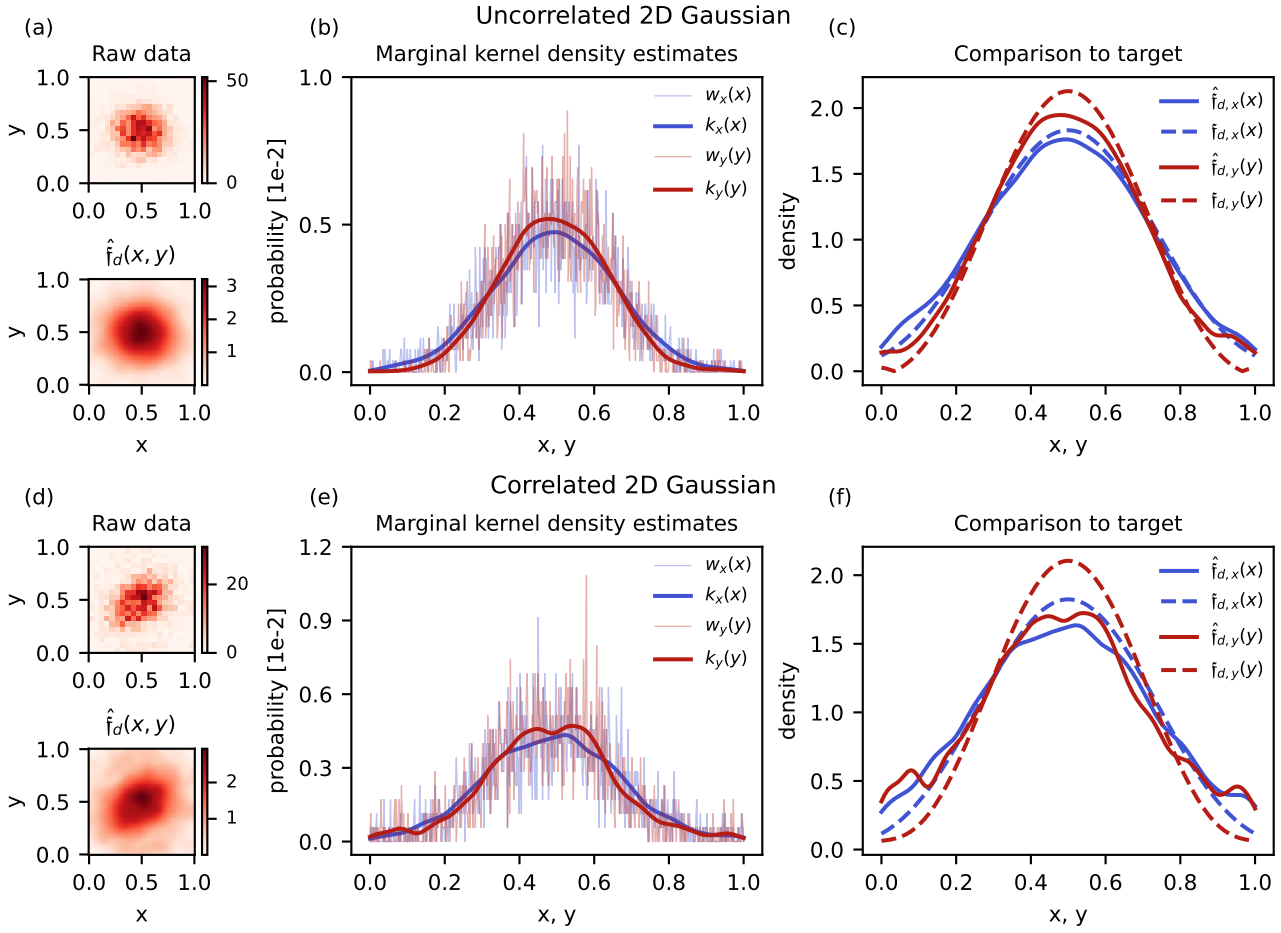

Figure 10: Experimental results for preparing a 2D Fourier series approximation of the bivariate Gaussian distribution. We show two setting, an uncorrelated Gaussian (upper panel) and a correlated Gaussian (lower panel); parameters are given in the main text. (a, d) Histograms of the raw two-dimensional measurement counts with 50 bins along each axis (upper plots), and the resulting estimates $\hat{\mathfrak{f}}_d(x,y)$ (lower plots, obtained from kernel density estimates) of the target functions $\mathfrak{f}_d(x,y)$. (b, e) Marginal probabilities $w_x(x), w_y(y)$ of the measurement data (thin solid lines) and the corresponding kernel density estimates $k_x(x), k_y(y)$ (bold solid lines). (c, f) Direct comparison between marginal estimates $\hat{\mathfrak{f}}_{d,x}(x), \hat{\mathfrak{f}}_{d,y}(y)$ (solid lines) and the target functions' marginals $\mathfrak{f}_{d,x}(x), \mathfrak{f}_{d,y}(y)$ (dashed lines).

tational basis and compare our experimental data to the positive target function $\mathfrak{f}_d(x,y) = |f_d(x,y)|$. We run each experiment with 20 000 shots. The success probability in our setting is around 10% (for details see Appendix C.2). Let $w(x_i, y_j)$ be the probability to measure grid point $(x_i, y_j) \in [0,1]^2, i,j = 1,\ldots,512$, after post-selection on the zero state in the ancilla register (cf. Fig. 4), and let $w_x(x_i) = \sum_{j=1}^{512} w(x_i, y_j)$, $w_y(y_j) = \sum_{i=1}^{512} w(x_i, y_j)$ be its marginals. The upper panels of Fig. 10(a) and (d) show histograms of the joint measurement distribution $w(x,y)$ for the uncorrelated and correlated settings, respectively. Figures 10(b) and (e) show the corresponding x- and y-marginals as thin lines. Notably, the measured signals suffer from substantial fluctuations due to shot noise.

To compare the empirical probabilities to the target $\mathfrak{f}_d(x,y)$ in a meaningful way, we smoothen the experimental data with kernel density estimation (KDE) [73–75]. The KDE yields a continuous approximation of the density underlying a finite data sample. This is obtained by replacing the raw counts with, in our case, Gaussian kernels of a given bandwidth h (see Appendix C.3). The KDE becomes smoother with increasing bandwidth h. On the other hand, for too large h the KDE loses information about the underlying data. In the limit $h \to \infty$, the KDE approaches a perfectly flat distribution. The "optimal" bandwidth can be computed via minimization of an appropriate cost function. A naive choice would be, for example, to consider a distance measure or statistical divergence (such as the relative entropy) between KDE and the target function $\mathfrak{f}_d(x,y)$. However, this choice introduces an obvious bias. Instead we choose to optimize h via cross validation from the measurement data alone (see Appendix C).

Let $k(x,y)$, $k_x(x)$ and $k_y(y)$ be the KDE [cf. Eq. (57)] corresponding to the empirical probabilities $w(x_i,y_j)$, $w_x(x_i)$ and $w_y(y_j)$ $(i,j = 1,\ldots,512)$, rescaled such that $\sum_{i,j=1}^{512} k(x_i,y_j) = \sum_{i=1}^{512} k_x(x_i) = \sum_{j=1}^{512} k_y(y_j) = 1$.

The thick solid lines in Figs. 10(b) and (e) are the marginal KDE $k_x(x_i)$ and $k_y(y_j)$. To compare to $\mathfrak{f}_d(x,y)$, we need to take the square root, which defines our experimental estimates $\hat{\mathfrak{f}}_d(x,y) = \sqrt{k(x,y)}$, $\hat{\mathfrak{f}}_{d,x}(x) = \sqrt{k_x(x)}$ and $\hat{\mathfrak{f}}_{d,y}(y) = \sqrt{k_y(y)}$. The lower panels of Fig. 10(a) and (d) display $\hat{\mathfrak{f}}_d(x,y)$. In Fig. 10(c) and (f), we compare the estimated marginals (solid lines) to the target marginals $\mathfrak{f}_{d,x}, \mathfrak{f}_{d,y}$ of $\mathfrak{f}_d$ (dashed lines). For direct comparison, we rescale all functions $\hat{\mathfrak{f}}_d, \hat{\mathfrak{f}}_{d,x}, \hat{\mathfrak{f}}_{d,y}, \mathfrak{f}_{d,x}, \mathfrak{f}_{d,y}$ such that they integrate to one on their corresponding domains, i.e. that they represent proper probability densities. Visually, we observe a good agreement. We clearly recognize an elongated shape in the lower plot of Fig. 10(d), as expected from a correlated 2D Gaussian.

For a quantitative comparison we determine the mean values and the covariance matrix of the target function $\mathfrak{f}_d(x,y)$ and its estimate $\hat{\mathfrak{f}}_d(x,y)$ (obtained from the experimental data):

$$\begin{aligned}
\mu_{d,x} &= \int_{[0,1]^2} \mathrm{d}x\mathrm{d}y\, x\mathfrak{f}_d(x,y),\\
\mu_{d,y} &= \int_{[0,1]^2} \mathrm{d}x\mathrm{d}y\, y\mathfrak{f}_d(x,y),\\
\sigma^2_{d,x} &= \int_{[0,1]^2} \mathrm{d}x\mathrm{d}y\, (x-\mu_{d,x})^2\mathfrak{f}_d(x,y),\\
\sigma^2_{d,y} &= \int_{[0,1]^2} \mathrm{d}x\mathrm{d}y\, (y-\mu_{d,y})^2\mathfrak{f}_d(x,y),\\
\rho_d &= \frac{\int_{[0,1]^2} \mathrm{d}x\mathrm{d}y\, (x-\mu_{d,x})(y-\mu_{d,y})\mathfrak{f}_d(x,y)}{\sigma_{d,x}\sigma_{d,y}},
\end{aligned} \tag{40}$$

and $\hat{\mu}_{d,x}, \hat{\mu}_{d,y}, \hat{\sigma}^2_{d,x}, \hat{\sigma}^2_{d,x}, \hat{\rho}_d$ computed equivalently by replacing all quantities with their "hat" versions (e.g. replace $\mathfrak{f}_d(x,y)$ with $\hat{\mathfrak{f}}_d(x,y)$). Furthermore, we compute the classical fidelity between the target and measurements

$$F = \frac{\left(\sum_{i,j=1}^{512} \sqrt{\mathfrak{f}_d(x_i,y_j)\hat{\mathfrak{f}}_d(x_i,y_j)}\right)^2}{\sum_{i,j=1}^{512} \mathfrak{f}_d(x_i,y_j) \sum_{i,j=1}^{512} \hat{\mathfrak{f}}_d(x_i,y_j)}, \tag{41}$$

where x_i, y_i runs over the grid points. Table 3 compares the results for both settings. Overall, we observe a good agreement between the targets and the experimentally prepared functions, both in terms of statistical quantifiers and fidelities. For the uncorrelated 2D Gaussian we obtain an overall fidelity of $F = 0.986$, for the correlated one we obtain $F = 0.918$.

We observe a larger deviation between target and prepared function for the correlated Gaussian. For example, the observed variances $\hat{\sigma}^2_{d,x}, \hat{\sigma}^2_{d,x}$ overestimate the target variances $\sigma^2_{d,x}, \sigma^2_{d,x}$ in this setting. Although we clearly identify a non-zero correlation coefficient in the setting of the correlated Gaussian, the estimated $\hat{\rho}_d = 0.088$ is significantly smaller, roughly by a factor of four, than the target value $\rho_d = 0.353$. The pronounced discrepancy between $\mathfrak{f}_d$ and $\hat{\mathfrak{f}}_d$ in the correlated setting, in comparison to the uncorrelated Gaussian, is expected. This is because, in the latter scenario, we are able to exploit the factorization of $\mathfrak{f}_d(x,y)$ into two univariate functions also at the level of circuit synthesis. Indeed we obtain much shallower circuits for the uncorrelated setting, which are much less prone to noise, as discussed in more detail in Appendix C.2.

4 Conclusions

This work introduces methods for the preparation of quantum states that encode multivariate functions, and analyzes their resource requirements. As a proof-of-concept, we demonstrate the successful preparation of representative initial states (with potential applications e.g. in quantum chemistry, physics and finance simulations) on comparatively high-resolution grids. We execute a 24-qubit experiment on Quantinuum's H2-1 trapped-ion quantum computer and analyse the impact of noise on the prepared states. The code implementing the presented protocols and the data to reproduce all results are available at GitHub and Zenodo [53].

Many functions of interest cannot be naturally encoded in a quantum state (e.g. $1/x$ due to a singularity), and function approximation becomes a necessary step to obtain practical protocols. Our methods are based on multivariate versions of Fourier and Chebyshev series approximations. We note that power series can be mapped to Chebyshev series [76, 77] where our method applies as-is. Walsh series used e.g. in [51, 52] can be implemented as a special case of our Fourier method. It is also possible to use different basis functions for each dimension in a hybrid Fourier-Chebyshev method. Moreover, the building blocks of our method can be applied recursively to create more expressive basis functions, e.g. wavelets and kernels, which are then linearly combined to form more complex functions. Ultimately, the approximation method and error budget will depend on the use case (e.g. whether the target function is periodic) and specifics of the available quantum hardware (e.g. two-qubit gate fidelity).

QSP [39–43] has attracted significant interest as a framework for developing quantum algorithms via polynomial transformations of block-encoded matrices. Multivariate versions of QSP have been proposed and analyzed, although questions about expressive power remain open [45–47]. Our work shows that linear combination of unitaries [59], with its simplicity and effectiveness, is a valid alternative for the task of preparing quantum states that encode multivariate functions in their amplitudes. An interesting research direction consists of using (univariate) QSP to further process the matrices produced by our method. For example, suppose we wanted to prepare

a state that encodes a hypersurface – a set of dimension $(D-1)$ defined by a polynomial equation of the form $f(x_1,\ldots,x_D)=0$. After discretizing each dimension via $H^{\otimes D}$ and using our method to construct the diagonal operator $f(H^{\otimes D})$, we use QSP to filter its eigenvalues. In particular, we want a polynomial p such that $p(0)=1$ and $|p(y)|$ is small for $y\in[-1,-\epsilon]\cup[\epsilon,1]$. An optimal filter is given by Lin and Tong in [78]. Applying the resulting operator to the uniform superposition we obtain a state proportional to $\sum_{x_1}\cdots\sum_{x_D}p(f(x_1,\ldots,x_D))\,|x_1,\ldots,x_D\rangle$, approximately encoding the hypersurface. Remarkably, this proposal uses univariate QSP for a fundamentally multivariate problem.

Acknowledgements

We thank Hitomi Mori for useful discussions, and Alexandre Krajenbrink and Enrico Rinaldi for helpful feedback on the manuscript.

References

[1] Sarah K. Leyton and Tobias J. Osborne. "A quantum algorithm to solve nonlinear differential equations" (2008). arXiv:0812.4423.

[2] Andrew M. Childs, Jin-Peng Liu, and Aaron Ostrander. "High-precision quantum algorithms for partial differential equations". Quantum **5**, 574 (2021).

[3] Pedro C. S. Costa, Stephen Jordan, and Aaron Ostrander. "Quantum Algorithm for Simulating the Wave Equation". Physical Review A **99**, 012323 (2019). arXiv:1711.05394.

[4] Dylan Herman, Cody Googin, Xiaoyuan Liu, Alexey Galda, Ilya Safro, Yue Sun, Marco Pistoia, and Yuri Alexeev. "A survey of quantum computing for finance" (2022). arXiv:2201.02773.

[5] Nikitas Stamatopoulos and William J. Zeng. "Derivative Pricing using Quantum Signal Processing". Quantum **8**, 1322 (2024).

[6] Ismail Yunus Akhalwaya, Adam Connolly, Roland Guichard, Steven Herbert, Cahit Kargi, Alexandre Krajenbrink, Michael Lubasch, Conor Mc Keever, Julien Sorci, Michael Spranger, and Ifan Williams. "A Modular Engine for Quantum Monte Carlo Integration" (2023). arXiv:2308.06081.

[7] Alán Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon. "Simulated quantum computation of molecular energies". Science **309**, 1704–1707 (2005).

[8] Ivan Kassal, Stephen P. Jordan, Peter J. Love, Masoud Mohseni, and Alán Aspuru-Guzik. "Polynomial-time quantum algorithm for the simulation of chemical dynamics". Proceedings of the National Academy of Sciences **105**, 18681–18686 (2008).

[9] Yudong Cao, Jonathan Romero, Jonathan P Olson, Matthias Degroote, Peter D Johnson, Mária Kieferová, Ian D Kivlichan, Tim Menke, Borja Peropadre, Nicolas P D Sawaya, Sukin Sim, Libor Veis, and Alán Aspuru-Guzik. "Quantum chemistry in the age of quantum computing". Chem. Rev. **119**, 10856–10915 (2019).

[10] Sam McArdle, Suguru Endo, Alán Aspuru-Guzik, Simon C Benjamin, and Xiao Yuan. "Quantum computational chemistry". Rev. Mod. Phys. **92**, 015003 (2020).

[11] Jie Liu, Yi Fan, Zhenyu Li, and Jinlong Yang. "Quantum algorithms for electronic structures: basis sets and boundary conditions". Chem. Soc. Rev. **51**, 3263–3279 (2022).

[12] Hans Hon Sang Chan, Richard Meister, Tyson Jones, David P. Tew, and Simon C. Benjamin. "Grid-based methods for chemistry simulations on a quantum computer". Science Advances **9**, eabo7484 (2023).

[13] Stephen P. Jordan, Keith S. M. Lee, and John Preskill. "Quantum algorithms for quantum field theories". Science **336**, 1130–1133 (2012).

[14] Mari Carmen Bañuls, Rainer Blatt, Jacopo Catani, Alessio Celi, Juan Ignacio Cirac, Marcello Dalmonte, Leonardo Fallani, Karl Jansen, Maciej Lewenstein, Simone Montangero, Christine A. Muschik, Benni Reznik, Enrique Rico, Luca Tagliacozzo, Karel Van Acoleyen, Frank Verstraete, Uwe-Jens Wiese, Matthew Wingate, Jakub Zakrzewski, and Peter Zoller. "Simulating lattice gauge theories within quantum technologies". The European Physical Journal D **74**, 165 (2020).

[15] Christian W. Bauer, Zohreh Davoudi, A. Baha Balantekin, Tanmoy Bhattacharya, Marcela Carena, Wibe A. de Jong, Patrick Draper, Aida El-Khadra, Nate Gemelke, Masanori Hanada, Dmitri Kharzeev, Henry Lamm, Ying-Ying Li, Junyu Liu, Mikhail Lukin, Yannick Meurice, Christopher Monroe, Benjamin Nachman, Guido Pagano, John Preskill, Enrico Rinaldi, Alessandro Roggero, David I. Santiago, Martin J. Savage, Irfan Siddiqi, George Siopsis, David Van Zanten, Nathan Wiebe, Yukari Yamauchi, Kübra Yeter-Aydeniz, and Silvia Zorzetti. "Quantum simulation for high-energy physics". PRX Quantum **4**, 027001 (2023).

[16] Alexander M. Dalzell, Sam McArdle, Mario Berta, Przemyslaw Bienias, Chi-Fang Chen, András Gilyén, Connor T. Hann, Michael J. Kastoryano, Emil T. Khabiboulline, Aleksander Kubica, Grant Salton, Samson Wang, and Fernando G. S. L. Brandão. "Quantum algorithms: A survey of applications and end-to-end complexities" (2023). arXiv:2310.03011.

[17] Juha J. Vartiainen, Mikko Möttönen, and Martti M. Salomaa. "Efficient decomposition of quantum gates". Phys. Rev. Lett. **92**, 177902 (2004).

[18] Mikko Möttönen, Juha J. Vartiainen, Ville Bergholm, and Martti M. Salomaa. "Transformation of quantum states using uniformly controlled rotations". Quant. Inf. Comp. **5**, 467 (2005). arXiv:quant-ph/0407010.

[19] Ville Bergholm, Juha J. Vartiainen, Mikko Möttönen, and Martti M. Salomaa. "Quantum circuits with uniformly controlled one-qubit gates". Phys. Rev. A **71**, 052330 (2005).

[20] Vivek V. Shende, Stephen S. Bullock, and Igor L. Markov. "Synthesis of quantum-logic circuits". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**, 1000–1010 (2006). arXiv:quant-ph/0406176.

[21] Martin Plesch and Časlav Brukner. "Quantum-state preparation with universal gate decompositions". Phys. Rev. A **83**, 032302 (2011).

[22] Guang Hao Low, Vadym Kliuchnikov, and Luke Schaeffer. "Trading T-gates for dirty qubits in state preparation and unitary synthesis". Quantum **8**, 1375 (2024). arXiv:1812.00954.

[23] Xiaoming Sun, Guojing Tian, Shuai Yang, Pei Yuan, and Shengyu Zhang. "Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **42**, 3301–3314 (2023).

[24] Xiao-Ming Zhang, Tongyang Li, and Xiao Yuan. "Quantum state preparation with optimal circuit depth: Implementations and applications". Phys. Rev. Lett. **129**, 230504 (2022).

[25] Christof Zalka. "Simulating quantum systems on a quantum computer". Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **454**, 313–322 (1998). arXiv:quant-ph/9603026.

[26] Lov Grover and Terry Rudolph. "Creating superpositions that correspond to efficiently integrable probability distributions" (2002). arXiv:quant-ph/0208112.

[27] Alexei Kitaev and William A. Webb. "Wavefunction preparation and resampling using a quantum computer" (2009). arXiv:0801.0342.

[28] Christian W. Bauer, Plato Deliyannis, Marat Freytsis, and Benjamin Nachman. "Practical considerations for the preparation of multivariate gaussian states on quantum computers" (2021). arXiv:2109.10918.

[29] Arthur G. Rattew, Yue Sun, Pierre Minssen, and Marco Pistoia. "The Efficient Preparation of Normal Distributions in Quantum Registers". Quantum **5**, 609 (2021).

[30] Gabriel Marin-Sanchez, Javier Gonzalez-Conde, and Mikel Sanz. "Quantum algorithms for approximate function loading". Phys. Rev. Res. **5**, 033114 (2023).

[31] Adam Holmes and A. Y. Matsuura. "Efficient quantum circuits for accurate state preparation of smooth, differentiable functions". 2020 IEEE International Conference on Quantum Computing and Engineering (QCE) (2020). arXiv:2005.04351.

[32] Javier Gonzalez-Conde, Thomas W. Watts, Pablo Rodriguez-Grasa, and Mikel Sanz. "Efficient quantum amplitude encoding of polynomial functions". Quantum **8**, 1297 (2024).

[33] Jason Iaconis, Sonika Johri, and Elton Yechao Zhu. "Quantum state preparation of normal distributions using matrix product states". npj Quantum Information **10**, 15 (2024).

[34] Arthur G. Rattew and Bálint Koczor. "Preparing arbitrary continuous functions in quantum registers with logarithmic complexity" (2022). arXiv:2205.00519.

[35] Yuval R. Sanders, Guang Hao Low, Artur Scherer, and Dominic W. Berry. "Black-box quantum state preparation without arithmetic". Phys. Rev. Lett. **122**, 020502 (2019).

[36] Johannes Bausch. "Fast Black-Box Quantum State Preparation". Quantum **6**, 773 (2022).

[37] Jessica Lemieux, Matteo Lostaglio, Sam Pallister, William Pol, Karthik Seetharam, Sukin Sim, and Burak Şahinoğlu. "Quantum sampling algorithms for quantum state preparation and matrix block-encoding" (2024). arXiv:2405.11436.

[38] Shengbin Wang, Zhimin Wang, Guolong Cui, Shangshang Shi, Ruimin Shang, Lixin Fan, Wendong Li, Zhiqiang Wei, and Yongjian Gu. "Fast black-box quantum state preparation based on linear combination of unitaries". Quantum Information Processing **20**, 270 (2021).

[39] Guang Hao Low and Isaac L. Chuang. "Optimal Hamiltonian Simulation by Quantum Signal Processing". Phys. Rev. Lett. **118**, 010501 (2017).

[40] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. "Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics". Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (2019). arXiv:1806.01838.

[41] Thais L. Silva, Márcio M. Taddei, Stefano Carrazza, and Leandro Aolita. "Fragmented imaginary-time evolution for early-stage quantum signal processors". Scientific Reports **13**, 18258 (2023).

[42] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang. "Grand unification of quantum algorithms". PRX Quantum **2**, 040203 (2021).

[43] Yuta Kikuchi, Conor Mc Keever, Luuk Coopmans, Michael Lubasch, and Marcello Benedetti.

"Realization of quantum signal processing on a noisy quantum computer". npj Quantum Information **9**, 93 (2023).

[44] Sam McArdle, András Gilyén, and Mario Berta. "Quantum state preparation without coherent arithmetic" (2022). arXiv:2210.14892.

[45] Zane M. Rossi and Isaac L. Chuang. "Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle". Quantum **6**, 811 (2022). arXiv:2205.06261.

[46] Hitomi Mori, Keisuke Fujii, and Kaoru Mizuta. "Comment on "Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle"". Quantum **8**, 1512 (2024). arXiv:2310.00918.

[47] Balázs Németh, Blanka Kövér, Boglárka Kulcsár, Roland Botond Miklósi, and András Gilyén. "On variants of multivariate quantum signal processing and their characterizations" (2023). arXiv:2312.09072.

[48] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision". SIAM Journal on Computing **46**, 1920–1950 (2017).

[49] Mudassir Moosa, Thomas W Watts, Yiyou Chen, Abhijat Sarma, and Peter L McMahon. "Linear-depth quantum circuits for loading fourier approximations of arbitrary functions". Quantum Science and Technology **9**, 015002 (2023).

[50] Juan José García-Ripoll. "Quantum-inspired algorithms for multivariate analysis: From interpolation to partial differential equations". Quantum **5**, 431 (2021).

[51] Julien Zylberman and Fabrice Debbasch. "Efficient Quantum State Preparation with Walsh Series" (2023). arXiv:2307.08384.

[52] Jonathan Welch, Daniel Greenbaum, Sarah Mostame, and Alan Aspuru-Guzik. "Efficient quantum circuits for diagonal unitaries without ancillas". New Journal of Physics **16**, 033040 (2014).

[53] Matthias Rosenkranz, Eric Brunner, Gabriel Marin-Sanchez, Nathan Fitzpatrick, Silas Dilkes, Yao Tang, Yuta Kikuchi, and Marcello Benedetti. "Quantum state preparation for multivariate functions - data and implementation (v1.0.0)". Zenodo `https://doi.org/10.5281/zenodo.14621127` (2025). `https://github.com/CQCL/mvsp`. Accessed 2025-04-03.

[54] J. C. Mason. "Near-best multivariate approximation by Fourier series, Chebyshev series and Chebyshev interpolation". Journal of Approximation Theory **28**, 349–358 (1980).

[55] John P. Boyd. "Chebyshev and Fourier Spectral Methods". Dover Publications. Mineola, N.Y. (2001). 2nd revised edition.

[56] Lloyd N. Trefethen. "Approximation Theory and Approximation Practice, Extended Edition". Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics. Philadelphia, PA (2019).

[57] Lloyd N. Trefethen. "Multivariate polynomial approximation in the hypercube". Proceedings of the American Mathematical Society **145**, 4837–4844 (2017).

[58] Alex Townsend and Lloyd N. Trefethen. "An Extension of Chebfun to Two Dimensions". SIAM Journal on Scientific Computing **35**, C495–C518 (2013).

[59] Andrew M. Childs and Nathan Wiebe. "Hamiltonian simulation using linear combinations of unitary operations". Quantum Info. Comput. **12**, 901–924 (2012). arXiv:1202.5822.

[60] Guang Hao Low and Isaac L. Chuang. "Hamiltonian simulation by qubitization". Quantum **3**, 163 (2019).

[61] William Kirby, Mario Motta, and Antonio Mezzacapo. "Exact and efficient Lanczos method on a quantum computer". Quantum **7**, 1018 (2023).

[62] Daan Camps, Lin Lin, Roel Van Beeumen, and Chao Yang. "Explicit quantum circuits for block encodings of certain sparse matrices". SIAM Journal on Matrix Analysis and Applications **45**, 801–827 (2024). arXiv:2203.10236.

[63] Xiaoming Sun Junhong Nie, Wei Zi. "Quantum circuit for multi-qubit toffoli gate with optimal resource" (2024). arXiv:2402.05053.

[64] Adenilton J. da Silva and Daniel K. Park. "Linear-depth quantum circuits for multiqubit controlled gates". Phys. Rev. A **106**, 042602 (2022).

[65] Ryan Babbush, Craig Gidney, Dominic W. Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven. "Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity". Phys. Rev. X **8**, 041015 (2018).

[66] Yanghua Wang. "Generalized seismic wavelets". Geophysical Journal International **203**, 1172–1178 (2015).

[67] J. P. Antoine. "Wavelet analysis of signals and images, a grand tour". Ciencias Matemáticas **18**, 113–144 (2000).

[68] Lewis Wright, Conor Mc Keever, Jeremy T. First, Rory Johnston, Jeremy Tillay, Skylar Chaney, Matthias Rosenkranz, and Michael Lubasch. "Noisy intermediate-scale quantum simulation of the one-dimensional wave equation". Physical Review Research **6**, 043169 (2024). arXiv:2402.19247.

[69] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. "T|ket⟩: A retargetable compiler for NISQ

devices". Quantum Science and Technology **6**, 014003 (2020). arxiv:2003.10611.

[70] Ching-Wai (Jeremy) Chiu, Haroon Mumtaz, and Gábor Pintér. "Forecasting with VAR models: Fat tails and stochastic volatility". International Journal of Forecasting **33**, 1124–1143 (2017).

[71] S. A. Moses, C. H. Baldwin, M. S. Allman, R. Ancona, L. Ascarrunz, C. Barnes, J. Bartolotta, B. Bjork, P. Blanchard, M. Bohn, J. G. Bohnet, N. C. Brown, N. Q. Burdick, W. C. Burton, S. L. Campbell, J. P. Campora, C. Carron, J. Chambers, J. W. Chan, Y. H. Chen, A. Chernoguzov, E. Chertkov, J. Colina, J. P. Curtis, R. Daniel, M. DeCross, D. Deen, C. Delaney, J. M. Dreiling, C. T. Ertsgaard, J. Esposito, B. Estey, M. Fabrikant, C. Figgatt, C. Foltz, M. Foss-Feig, D. Francois, J. P. Gaebler, T. M. Gatterman, C. N. Gilbreth, J. Giles, E. Glynn, A. Hall, A. M. Hankin, A. Hansen, D. Hayes, B. Higashi, I. M. Hoffman, B. Horning, J. J. Hout, R. Jacobs, J. Johansen, L. Jones, J. Karcz, T. Klein, P. Lauria, P. Lee, D. Liefer, S. T. Lu, D. Lucchetti, C. Lytle, A. Malm, M. Matheny, B. Mathewson, K. Mayer, D. B. Miller, M. Mills, B. Neyenhuis, L. Nugent, S. Olson, J. Parks, G. N. Price, Z. Price, M. Pugh, A. Ransford, A. P. Reed, C. Roman, M. Rowe, C. Ryan-Anderson, S. Sanders, J. Sedlacek, P. Shevchuk, P. Siegfried, T. Skripka, B. Spaun, R. T. Sprenkle, R. P. Stutz, M. Swallows, R. I. Tobey, A. Tran, T. Tran, E. Vogt, C. Volin, J. Walker, A. M. Zolot, and J. M. Pino. "A race-track trapped-ion quantum processor". Phys. Rev. X **13**, 041052 (2023).

[72] "Quantinuum H2-1". url: https://www.quantinuum.com/. (accessed: 2025-04-03).

[73] Duin. "On the Choice of Smoothing Parameters for Parzen Estimators of Probability Density Functions". IEEE Transactions on Computers **C-25**, 1175–1179 (1976).

[74] Mats Rudemo. "Empirical choice of histograms and kernel density estimators". Scandinavian Journal of Statistics **9**, 65–78 (1982). url: http://www.jstor.org/stable/4615859.

[75] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. "The Elements of Statistical Learning". Springer Series in Statistics. Springer New York. New York, NY (2009).

[76] Henry C. Thacher. "Conversion of a power to a series of Chebyshev polynomials". Commun. ACM **7**, 181–182 (1964).

[77] Hippolyte Nyengeri, Rénovat Nizigiyimana, Jean-Pierre Mutankana, Henry Bayaga, and Ferdinand Bayubahe. "Power and Chebyshev Series Transformation Formulas with Applications to Solving Ordinary Differential Equations via the Fröbenius and Taylor's Methods". Open Access Library Journal **8**, 1–19 (2021).

[78] Lin Lin and Yu Tong. "Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems". Quantum **4**, 361 (2020).

[79] D. Elliott, D.F. Paget, G.M. Phillips, and P.J. Taylor. "Error of truncated chebyshev series and other near minimax polynomial approximations". Journal of Approximation Theory **50**, 49–57 (1987).

[80] Charles Fefferman. "On the convergence of multiple Fourier series". Bulletin of the American Mathematical Society **77**, 744–745 (1971).

[81] Charles Fefferman. "On the divergence of multiple Fourier series". Bulletin of the American Mathematical Society **77**, 191–195 (1971).

[82] Adrian W. Bowman. "An Alternative Method of Cross-Validation for the Smoothing of Density Estimates". Biometrika **71**, 353–360 (1984).

[83] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine learning in Python". Journal of Machine Learning Research **12**, 2825–2830 (2011). url: http://jmlr.org/papers/v12/pedregosa11a.html.

[84] Lov K. Grover. "Synthesis of Quantum Superpositions by Quantum Computation". Physical Review Letters **85**, 1334–1337 (2000).

[85] Steven Herbert. "No quantum speedup with Grover-Rudolph state preparation for quantum Monte Carlo integration". Physical Review E **103**, 063302 (2021). arXiv:2101.02240.

[86] C. Schön, E. Solano, F. Verstraete, J. I. Cirac, and M. M. Wolf. "Sequential Generation of Entangled Multiqubit States". Physical Review Letters **95**, 110503 (2005). arXiv:quant-ph/0501096.

[87] Raban Iten, Roger Colbeck, Ivan Kukuljan, Jonathan Home, and Matthias Christandl. "Quantum circuits for isometries". Physical Review A **93**, 032318 (2016).

[88] Hitomi Mori, Kosuke Mitarai, and Keisuke Fujii. "Efficient state preparation for multivariate Monte Carlo simulation" (2024). arXiv:2409.07336.

[89] Thomas Häner, Martin Roetteler, and Krysta M. Svore. "Optimizing Quantum Circuits for Arithmetic" (2018). arXiv:1805.12445.

[90] Román Orús. "A practical introduction to tensor networks: Matrix product states and projected entangled pair states". Annals of Physics **349**, 117–158 (2014). arXiv:1306.2164.

[91] Norbert Schuch, Michael M. Wolf, Frank Verstraete, and J. Ignacio Cirac. "Entropy Scaling and Simulability by Matrix Product States". Physical Review Letters **100**, 030504 (2008). arXiv:0705.0292.

[92] Lars Grasedyck. "Polynomial approximation in hierarchical Tucker format by vector-tensorization". Report 308. Inst. für Geometrie und Praktische Mathematik, RWTH Aachen (2010). url: https://www.igpm.rwth-aachen.de/Download/reports/pdf/IGPM308_k.pdf.

[93] I. V. Oseledets. "Constructive Representation of Functions in Low-Rank Tensor Formats". Constructive Approximation **37**, 1–18 (2013).

[94] Michael Lubasch, Pierre Moinier, and Dieter Jaksch. "Multigrid Renormalization". Journal of Computational Physics **372**, 587–602 (2018). arXiv:1802.07259.

[95] Juan José Rodríguez-Aldavero, Paula García-Molina, Luca Tagliacozzo, and Juan José García-Ripoll. "Chebyshev approximation and composition of functions in matrix product states for quantum-inspired numerical analysis" (2024). arXiv:2407.09609.

[96] Shi-Ju Ran. "Encoding of matrix product states into quantum circuits of one- and two-qubit gates". Physical Review A**101** (2020).

[97] Manuel S. Rudolph, Jing Chen, Jacob Miller, Atithi Acharya, and Alejandro Perdomo-Ortiz. "Decomposition of Matrix Product States into Shallow Quantum Circuits" (2022). arXiv:2209.00595.

A Classical preprocessing for the Chebyshev method

A.1 One dimension

A Lipshitz continuous function f defined on $[-1,1]$ has a unique expansion as a Chebyshev series,

$$f(x) = \sum_{k=0}^{\infty} \hat{c}_k T_k(x) \tag{42}$$

with coefficients

$$\hat{c}_k = \frac{2-\delta_{k,0}}{\pi} \int_{-1}^{1} \frac{\mathrm{d}x}{\sqrt{1-x^2}} f(x) T_k(x), \tag{43}$$

where $\delta_{k,j}$ is the Kronecker delta (see e.g. [56, Theorem 3.1]). The Chebyshev polynomials $T_k(x)$ satisfy the orthogonality condition

$$\int_{-1}^{1} \frac{\mathrm{d}x}{\sqrt{1-x^2}} T_k(x) T_j(x) = \begin{cases} 0 & \text{if } k \neq j, \\ \frac{\pi}{2} & \text{if } k = j \neq 0, \\ \pi & \text{if } k = j = 0 \end{cases} \tag{44}$$

as well as the discrete orthogonality condition

$$\sum_{m=0}^{N-1} T_k\left(x_m^{(N)}\right) T_j\left(x_m^{(N)}\right) = \begin{cases} 0 & \text{if } k \neq j, \\ \frac{N}{2} & \text{if } k = j \neq 0, \\ N & \text{if } k = j = 0, \end{cases} \tag{45}$$

where $x_m^{(N)} = \cos\left(\frac{\pi(2m+1)}{2N}\right)$ for $m = 0, 1, \ldots, N-1$ are the roots of the Chebyshev polynomial $T_N(x)$.

The analysis can be extended to complex functions. The Chebyshev series remains convergent within an ellipse in the complex plane with foci ± 1 that does not contain any poles, branch points or other singularities of f [55, Theorem 7].

For the numerical results in this paper we construct the maximal degree-d polynomial approximation Eq. (2) via interpolation. The interpolant f_d is uniquely determined by finding coefficients c_k such that $f_d(x_m) = f(x_m)$ on a set of $d+1$ points $\{x_m\}_{m=0}^{d}$. We choose the roots of the degree-$(d+1)$ Chebyshev polynomial as interpolation points, $\left\{x_m^{(d+1)}\right\}_{m=0}^{d}$, as this entails favorable convergence properties to the target function.

First, let us consider the interpolation error [79]. If the Chebyshev series of a function f exists, the error is bounded by $\max_{x\in[-1,1]} |f(x) - f_d(x)| \leq 2\sum_{k=d+1}^{\infty} |\hat{c}_k|$ [55, Theorem 21]. Typically, the magnitude of coefficients decreases rapidly in k with a rate depending on the smoothness of f. Given a target function $f \in C^{(d+1)}[-1,1]$ (i.e. f is a complex function whose $(d+1)$-st derivative exists and is continuous on the interval $[-1,1]$), the degree-d Chebyshev interpolation incurs an error

$$\varepsilon_d := \max_{x\in[-1,1]} |f(x) - f_d(x)| = \frac{|f^{(d+1)}(\xi)|}{2^d (d+1)!} \tag{46}$$

for some $\xi \in (-1,1)$. The same formula holds for truncation instead of interpolation, possibly for some other ξ [79]. If M is an upper bound for $f^{(d+1)}$ on $[-1,1]$ (this exists because $f^{(d+1)}$ is continuous on the bounded interval $[-1,1]$), then the truncation and interpolant error is upper-bounded by

$$\varepsilon_d \leq \frac{M}{2^d(d+1)!}. \tag{47}$$

The take-away message is that the Chebyshev interpolant rapidly converges to the target function if it is sufficiently smooth. The function does not need to be periodic as is the case for the Fourier expansion.

The coefficients c_k of the interpolant can be computed with the fast Fourier transform. The interpolant f_d is a Chebyshev series for a polynomial of degree at most d. Its coefficients are given exactly by applying the $(d+1)$-point trapezoidal rule to the integral in Eq. (43) evaluated at the $d+1$ Chebyshev roots [55, Theorem 21]

$$c'_k = \frac{2}{d+1}\sum_{m=0}^{d} f_d\left(x_m^{(d+1)}\right) T_k\left(x_m^{(d+1)}\right) = \frac{2}{d+1}\sum_{m=0}^{d} f_d\left(x_m^{(d+1)}\right)\cos\left(k\pi\frac{2m+1}{2(d+1)}\right). \tag{48}$$

This is the discrete cosine transform, which we evaluate via the fast Fourier transform with complexity $\mathcal{O}(d\log d)$. The notation c'_k means we need to divide the $k=0$ term by 2, $c_0 = c'_0/2$ while $c_k = c'_k$ for $k>0$. The coefficients of the interpolant and infinite Chebyshev series are related via a simple formula [55, Theorem 21].

A.2 Two and higher dimensions

To compute the 2D interpolant $f_{d_x,d_y}(x,y)$, Eq. (4), we match the target function at the grid points $\left\{\left(x_m^{(d_x+1)}, y_n^{(d_y+1)}\right)\right\}_{m=0,n=0}^{d_x,d_y}$ given by the Chebyshev roots of degrees (d_x+1) and (d_y+1). In analogy to the 1D case, the coefficients $c_{k,l}$ of this interpolant are given by the 2D cosine transform on this grid

$$c'_{k,l} = \frac{4}{(d_x+1)(d_y+1)}\sum_{m=0}^{d_x}\sum_{n=0}^{d_y} f(x_m^{(d_x+1)}, y_n^{(d_y+1)})\cos\left(k\pi\frac{2m+1}{2(d_x+1)}\right)\cos\left(l\pi\frac{2n+1}{2(d_y+1)}\right). \tag{49}$$

This can be evaluated with the fast Fourier transform with cost $\mathcal{O}(d_x d_y \log(d_x d_y))$. The notation $c'_{k,l}$ means we need to divide the $k=0, l=0$ terms appropriately, $c_{k,l} = c'_{k,l}/(1+\delta_{k,0})(1+\delta_{l,0})$.

The above procedure readily generalizes to higher dimensions. However, for high degrees or high dimensions it may become computationally more efficient to compute a low-rank approximation of f and approximate the univariate, constituent functions as Chebyshev polynomials [58].

Error bounds are more difficult to establish rigorously for the multivariate case. Ref. [57] shows that multivariate functions with a certain property display similar behavior of the maximum error as the univariate case, namely exponential convergence to the target function with increasing (total or maximal) degree.

B Classical preprocessing for the Fourier method

Here we discuss two preprocessing steps that may be required for implementing the Fourier approach. We used those methods in Sec. 3.2. First, we describe one way to periodically extend functions which are not periodic with period 2. Second, we show how to compute coefficients in a degree-d Fourier series approximation via the fast Fourier transform.

Without loss of generality consider a *bivariate* target function $f : [0,1]^2 \to \mathbb{C}$, which is not periodic with period 2. To obtain a converging Fourier series, we turn the function into a periodic one. We begin by extending the domain to $[-1,1]^2$ by considering the function $\tilde{f}(x,y) = f(|x|,|y|)$. This step guarantees that the function is mirrored with respect to both coordinate axes and for both the real and imaginary part. There are no jump discontinuities in the resulting function, although the first derivative may be discontinuous. Next, we extend the domain to $\mathbb{R}^2$ by considering the function

$$\tilde{f}(x+2n, y+2m) = \tilde{f}(x,y) \qquad \forall n,m \in \mathbb{N}, \tag{50}$$

which is a periodic function of period 2 in both variables. Then we use this extended target function for finding Fourier series coefficients.

Next we discuss a method for computing the coefficients of a degree-d trigonometric polynomial approximation to a function via interpolation and the fast Fourier transform. For simplicity let us consider the one-dimensional

case, i.e. a periodic function $f(x)$ with period 2 on the interval $[-1,1]$ (after applying a periodic extension if necessary). We assume f admits a Fourier series $f(x) = \sum_{k=-\infty}^{\infty} \hat{c}_k e^{i\pi kx}$. Its coefficients are given by

$$\hat{c}_k = \frac{1}{2}\int_{-1}^{1} \mathrm{d}x f(x) e^{-i\pi kx}. \tag{51}$$

If those coefficients are known, a valid approximation is truncation of the infinite Fourier series at degree d. If they are unknown, it is often numerically favourable to seek an interpolant f_d of the form Eq. (1) such that $f_d(x_m) = f(x_m)$ on the set of points $x_m = 2m/(2d+1)$ for $m = 0, \ldots, 2d$. The coefficients of f_d are given by

$$c_k = \frac{1}{2d+1}\sum_{m=0}^{2d} f(x_m) e^{-i\frac{2\pi km}{2d+1}}. \tag{52}$$

This follows from multiplying $f_d(x_m)$ by $e^{-i\pi kx_m}$, summing over m and using discrete orthogonality of the Fourier basis functions:

$$\sum_{m=0}^{2d} f_d(x_m) e^{-i\pi kx_m} = \sum_{m=0}^{2d}\sum_{l=-d}^{d} c_l e^{i\pi x_m(l-k)} = \sum_{l=-d}^{d} c_l(2d+1)\delta_{l,k} = (2d+1)c_k. \tag{53}$$

Inserting x_m yields Eq. (52).

Note that Eq. (52) is the discrete Fourier transform of $\{f(x_m)\}_{m=0}^{2d}$, which can be evaluated with cost $\mathcal{O}(d\log d)$ via the fast Fourier transform. This method generalizes to D dimensions and use of the D-dimensional fast Fourier transform with cost $\mathcal{O}[d_1 d_2 \cdots d_D \log(d_1 d_2 \cdots d_D)]$.

We do not discuss the convergence of multivariate Fourier series in general. We shall mention however that the examples presented in this paper do enjoy this property. For bivariate functions in L^p, with $p > 1$, and with a careful definition of the partial sums, the Fourier series converges as shown by Fefferman [80, 81].

C Hardware experiment

C.1 Hardware specifications

For the experimental demonstration of our state preparation algorithm, we use the H2-1 trapped-ion quantum computer. Here we summarize the relevant specifications of H2-1 at the time of the experiments performed in April and May 2024. For details, see Ref. [72]. H2-1 operates with 32 qubits implemented with $S_{1/2}$ hyperfine clock states of $^{171}\mathrm{Yb}^+$ ions. The native gate set consists of single-qubit rotation gates $\mathrm{PhasedX}(\alpha,\beta) = R_Z(\beta)R_X(\alpha)R_Z(-\beta)$ and two-qubit gates $ZZ\mathrm{Phase}(\theta) = e^{-i\frac{\theta}{2}Z\otimes Z}$, parametrized by real angles α, β and θ. All the gate operations as well as initializations and measurements are performed in one of four gate zones, each operating in parallel. The native two-qubit gates can be applied to an arbitrary pair of qubits by shuttling ions to one of the gate zones, enabling all-to-all connectivity. Average single- and two-qubit gate infidelities are typically 0.003% and 0.2%, respectively. State preparation and measurement error is typically 0.2% on average.

C.2 Implementation details

As discussed in Sec. 3.4, we consider the preparation of an uncorrelated and a correlated 2D Gaussian distribution $f(x,y)$, Eq. (39), with the same mean value $\boldsymbol{\mu} = (\mu_x, \mu_y)^T$, covariance matrix Σ parameterized by the same marginal variances σ_x^2, σ_y^2 and two values of the correlation coefficient ρ. We prepare both functions on a 2D grid with 512×512 grid points, requirering $n_x = n_y = 9$ qubits for the spatial discretization of both axes.

For both settings, we compute the Fourier series coefficients in $f_d(x,y)$ with maximal degree $d = d_x = d_y = 3$ using the following method. First, we note that both Gaussians are close to 0 at the boundary of the domain $[0,1]^2$. We, thus, extend $f(x,y)$ to $[-1,1]^2$ by setting the function to 0 in the extended domain. Next, assuming this function admits a Fourier series, its coefficients can be approximated via

$$\hat{c}_{k,\ell} = \frac{1}{4}\int_{[-1,1]^2} \mathrm{d}x\mathrm{d}y\, f(x,y) e^{-i\pi(xk+y\ell)} \approx \frac{1}{4}\int \mathrm{d}x\mathrm{d}y\, f(x,y) e^{i(x,y)^T\boldsymbol{\omega}} = \frac{1}{4} e^{i\boldsymbol{\mu}^T\boldsymbol{\omega} - \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\Sigma}\boldsymbol{\omega}}. \tag{54}$$

The first equality is a generalisation of Eq. (51) to the bivariate case. The second approximate equality follows from extending the integration domain to the full space $\mathbb{R}^2$, and setting $\boldsymbol{\omega} = -\pi(k,\ell)^T$. This integral is the

characterisic function of the bivariate Gaussian, for which there exists an analytic formula, given by the last expression in Eq. (54). The coefficients go to 0 exponentially in $|k|$ and $|l|$, allowing us to truncate the series at relatively small degrees d_x and d_y. We numerically observe that this method leads to a faster convergence for our specific target function compared to the method used in Sec. 3.2. This allows us to use a lower degree at a similar approximation quality. A lower degree reduces circuit depth which is crucial for the experimental implementation.

Note that in the setting of the uncorrelated Gaussian, the density $f(x, y)$ [cf. Eq. (39)] and its Fourier series approximation $f_d(x, y)$ factorize into a product of univariate Gaussian densities. This factorization can be exploited in the circuit construction as well. More precisely, we prepare the two univariate functions with two separate circuits as given in Fig. 4 (each with $a = 3, n = 9$ qubits for the ancilla and main registers, respectively) and concatenate those. This considerably simplifies the circuit and reduces the gate complexity, in comparison to the full 2D state preparation circuit, cf. Fig. 5. For the uncorrelated Gaussian we obtain 123 single-qubit gates (Phased$X$), 80 two-qubit gates (ZZPhase) and a depth of 42 for the full state preparation circuit. For the correlated Gaussian we obtain 266 single-qubit gates, 237 two-qubit gates and circuit depth 352.

The success probability of the algorithm is predominantly controlled by the weight $\sum_x \sum_y |f_d(x, y)|^2$ of the prepared function in the domain $[0, 1]^2$, see Eq. (32). Based on a noiseless simulation of the circuit, the ideal success probabilities are given by $p_{\text{success}} = 0.134$ and $p_{\text{success}} = 0.139$ for the case of the correlated and uncorrelated Gaussian, respectively. The interplay between noise and circuit depth also affects the success probability. Deeper circuits increase overall noise when executed on noisy hardware, which, in general, reduces the overlap of the prepared (mixed) state with the target state $|0\rangle^{\otimes(a_x+a_y)}$ in the ancilla registers, cf. Fig. 5. Generally, this reduces the success probability. Experimentally, we obtain a success probability $p_{\text{success}} = 0.1299$ for the uncorrelated setting and a success probability $p_{\text{success}} = 0.0877$ for the correlated setting.

C.3 Kernel density estimation

Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\text{data}}}\}$, $\boldsymbol{x}_i \in \mathbb{R}^D$ be the observed data points in a given D-dimensional domain. Assume the data is independently drawn from a density $f(\boldsymbol{x})$. The kernel density estimate (KDE) of f based on dataset $\boldsymbol{X}$ is given by

$$k_h^{\boldsymbol{X}}(\boldsymbol{x}) = \frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right), \tag{55}$$

where K is the kernel, i.e. a smooth normalized function (which integrates to one over the given domain), which is often chosen to be Gaussian. The parameter h is the bandwidth. The KDE yields a smooth approximation of the density f underlying the observations $\boldsymbol{x}_i$. The bandwidth h needs to be chosen based on the competition between the smoothness of the estimate and the information it contains about the dataset. For a small bandwidth h the KDE tends to overfit the data, and in the limit of $h \to 0$ the KDE just represents the original dataset without smoothing. For $h \to \infty$, on the other hand, the KDE approaches a flat distribution, which does not contain any information about the dataset $\boldsymbol{X}$.

We optimize h from the data, i.e. we do not assume access to the underlying density f, using cross validation. For this let $\boldsymbol{X}_i$ be the dataset obtained from $\boldsymbol{X}$ by excluding the point $\boldsymbol{x}_i$, and $k_h^{\boldsymbol{X}_i}$ the corresponding KDE. The probability to observe the excluded point is given by $k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)$. We want to maximize this probability, averaged over all choices of i. In practice one often works with log-probabilities. Note that, since the logarithm is a monotonically increasing function, maximizing $k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)$ is equivalent to maximizing $\log k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)$. This defines our final optimizer, the mean of log-probabilities taken over i:

$$q(h) = \frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} \log k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)\,. \tag{56}$$

The optimal $\hat{h}$ is obtained by maximizing q over h. The KDE estimate of the target density f is then given by

$$k^{\boldsymbol{X}} = k_{\hat{h}}^{\boldsymbol{X}}\,. \tag{57}$$

For a given h, the log-probability $\log k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)$ is an unbiased estimator (under the distribution of $\boldsymbol{X}$) of the Kullback-Leibler divergence (up to a sign and a constant additive factor depending on f) between f and $k_h^{\boldsymbol{X}}$. What this means is that $\mathbb{E}[\log k_h^{\boldsymbol{X}_i}(\boldsymbol{x}_i)]$ asymptotically approaches $\mathbb{E}[\int \mathrm{d}\boldsymbol{x} f(\boldsymbol{x}) \log k_h^{\boldsymbol{X}}(\boldsymbol{x})]$ for $N_{\text{data}} \to \infty$, where the expectation $\mathbb{E}[\cdot]$ is taken with respect to the distribution of $\boldsymbol{X}$, see e.g. [73, 74]. There exist alternatives to the log-probabilities which are, for example, more closely related to the mean integrated squared error (MISE) between the target density f and the estimator [74], see also [82] for a comparison of both.

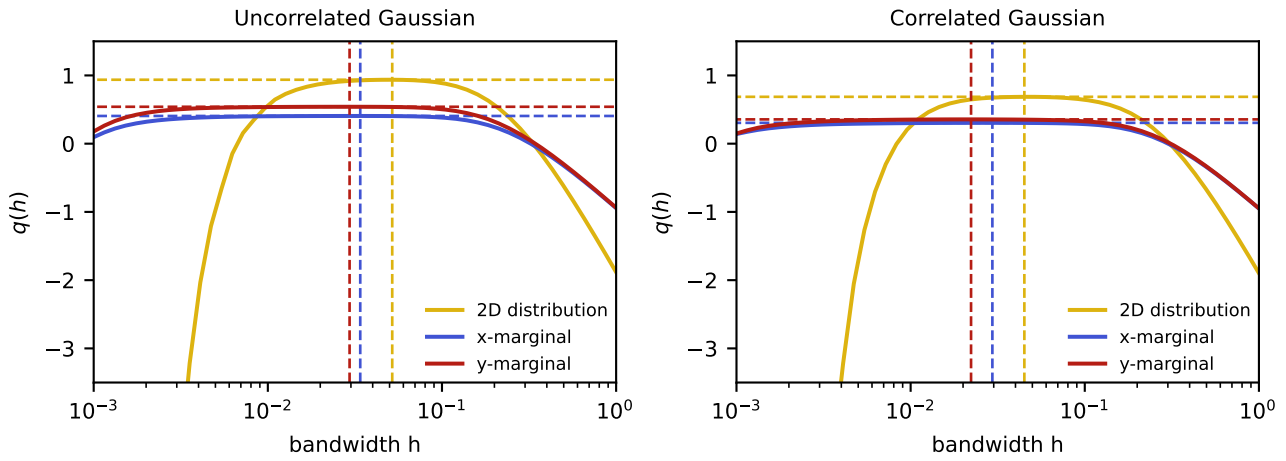

Figure 11: Cross validation of the kernel density estimates for $\mathfrak{f}_d(x,y) = |f_d(x,y)|$ and its marginals. The results are shown for the uncorrelated and the correlated Gaussian in the left and right plots, respectively. The maximal value of the mean of log-probabilities $q(h)$, Eq. (56), and the corresponding bandwidth h are highlighted via vertical and horizontal dashed lines. For the uncorrelated Gaussian, the maximum of $q(h)$ is achieved for $\hat{h} \approx 0.052$ for the full 2D distribution, and for $\hat{h}_x \approx 0.034$ and $\hat{h}_y \approx 0.029$ for its marginals. For the correlated Gaussian, we obtain optimal bandwidths $\hat{h} \approx 0.045$, $\hat{h}_x \approx 0.029$ and $\hat{h}_y \approx 0.022$.

In Fig. 11 we apply KDE to the data obtained from our state preparation hardware experiments, cf. Sec. C.2. To optimize $q(h)$ [cf. Eq. (56)] we use the `scikit-learn` [83] implementations of KDE and cross validation. We first consider the datasets of the full 2D state preparation circuit. The dataset $\boldsymbol{X}$ is given by the two-dimensional points $\boldsymbol{x}_i = (x_i, y_i)$ (as represented via bitstrings, see Sec. 2.1) measured at the output of our state preparation algorithm, see Appendix C.1 and C.2. In case of the uncorrelated Gaussian, we have $N_{\text{data}} = 2598$ data points, for the correlated one we have $N_{\text{data}} = 1754$. The yellow solid curves in both plots of Fig. 11 show q as a function of the bandwidth h, for both settings. We obtain optimal bandwidths (vertical yellow dashed lines) of $\hat{h} \approx 0.052$ (uncorrelated Gaussian) and $\hat{h} \approx 0.045$ (correlated Gaussian). We repeat the same procedure for the marginals. The blue and red solid curves in Fig. 11 show $q(h)$ when we use $\boldsymbol{X} = \{x_1, \ldots, x_{N_{\text{data}}}\}$ (the x-marginal) and $\boldsymbol{X} = \{y_1, \ldots, y_{N_{\text{data}}}\}$ (the y-marginal), respectively. Optimal marginal KDE are obtained for bandwidths $\hat{h}_x \approx 0.034, \hat{h}_y \approx 0.029$ for the setting of the uncorrelated Gaussian, and $\hat{h}_x \approx 0.029, \hat{h}_y \approx 0.022$ for the correlated Gaussian, as indicated by the vertical dashed blue and red lines in the plots, respectively.

D Comparison to prior work

Here we review prior works that (i) are not primarily heuristic and (ii) focus on or can be extended in a straightforward way to multivariate functions. Table 4 summarizes their resource estimates discussed in the following subsections.

We consider a multivariate function f defined on $[-1,1]^D$. We discretize the domain with 2^n points per variable. We denote the ℓ^p norm of the function evaluated on 2^{Dn} grid points $\boldsymbol{x}$ with

$$\|\boldsymbol{f}\|_p = \left(\sum_{\boldsymbol{x}} |f(\boldsymbol{x})|^p\right)^{1/p}, \tag{58}$$

and the ℓ^p norm filling fraction with

$$\mathcal{F}_p = \frac{\|\boldsymbol{f}\|_p}{2^{Dn/p}|f|_{\max}}, \tag{59}$$

where

$$|f|_{\max} = \max_{\boldsymbol{x}\in[-1,1]^D} |f(\boldsymbol{x})|. \tag{60}$$

Computing $|f|_{\max}$ is hard in general, especially in higher dimensions. Several prior works assume that this quantity is known or that a good upper bound $h \geq |f|_{\max}$ can be found. Moreover, the cost of such preprocessing step is rarely discussed. An advantage of our protocol is that it does not require additional preprocessing steps. Let us assume that we want to implement a maximal degree-d Fourier or Chebyshev approximation f_d to the

	Two-qubit gates	Success probability	Ancilla qubits	Function type
Black-box [35, 84]	$\mathcal{O}(d^D m^2)$	$\mathcal{F}_2^2$	$\mathcal{O}(d^D m)$	any
Grover-Rudolph [25, 26, 85]	$\mathcal{O}(Dnd^D m^2)$	1	$\mathcal{O}(d^D m)$	efficiently integrable distribution
Adiabatic [34]	$\mathcal{O}(d^D m^2/\epsilon^2 \mathcal{F}_1^4)$	$1-\mathcal{O}(\epsilon^2)$	$\mathcal{O}(d^D m + Dn)$	any
MPS [86, 87]	$\mathcal{O}(\mathrm{poly}(\chi)Dn)$	1	$\log \chi$	low entanglement
FSL [49]	$\mathcal{O}(d^D + Dn^2)$	1	0	Fourier series
M-QSP [88]	$\mathcal{O}(Dnd)$	$\mathcal{F}_2^2$	1	limited & unclear
LCU Fourier (this work)	$\mathcal{O}(d^D + Dn \log d)$	$\mathcal{F}_{2,\|\boldsymbol{c}\|_1}^2$	$D\lceil \log_2(2d+1)\rceil$	Fourier series
LCU Chebyshev (this work)	$\mathcal{O}(d^D + Ddn \log n)$	$\mathcal{F}_{2,\|\boldsymbol{c}\|_1}^2$	$D\lceil \log_2(d+1)\rceil + D\lceil \log_2 n\rceil$	Chebyshev series

Table 4: Summary of computational resources required to prepare a Dn-qubit quantum state representing a D-dimensional target function. For the oracles in the black-box, Grover-Rudolph and adiabatic approaches we assume a maximal degree-d polynomial approximation to the target stored in a register with m-bit precision using the construction in Ref. [89] (Eq. (66)). χ is the bond order of the MPS representation of the target function. The success probability p_{success} can be amplified to 1 at the cost of multiplicative overhead $\mathcal{O}(p_{\text{success}}^{-1/2})$ in gate count. $\mathcal{F}_p$ is the ℓ_p-norm filling fraction (Eq. (59)) and $\mathcal{F}_{2,\|\boldsymbol{c}\|_1} \leq \mathcal{F}_2$. In practice, often only a bound on $\mathcal{F}_p$ is known, which can lead to additional error. Note that the dependence of resources on state preparation errors is mostly hidden in d, m and χ. For the adiabatic algorithm, ϵ denotes the spectral distance to an exact state preparation unitary.

target and we have computed coefficients $\boldsymbol{c}$. Then we can use the following bound

$$h = \|\boldsymbol{c}\|_1 = \sum_{\boldsymbol{k}} |c_{\boldsymbol{k}}| \geq \max_{\boldsymbol{x}} |f_d(\boldsymbol{x})| = |f_d|_{\max} \tag{61}$$

which only costs addition of $\mathcal{O}(d^D)$ numbers.

Inserting any bound h in Eq. (59) leads to a lower bound on the filling fraction

$$\mathcal{F}_{p,h} = \frac{\|\boldsymbol{f}\|_p}{2^{Dn/p} h} \leq \mathcal{F}_p. \tag{62}$$

Another useful quantity is obtained by replacing f with the approximating function, such as a maximal degree-d polynomial f_d or a finite precision representation depending on the context. We denote with $\tilde{\mathcal{F}}_p$ an approximate ℓ^p norm filling fraction.

D.1 Black-box algorithm

In the *black-box approach* we are given two oracles O_{amp} and O_{rot}. Oracle O_{amp} writes a set of real numbers $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{2^n-1})$ with $0 \leq \alpha_i \leq 1$ for all $i = 0, \ldots, 2^n - 1$ into an m-qubit register according to

$$O_{\text{amp}} |i\rangle |z\rangle = |i\rangle \left| z \oplus \alpha_i^{(m)} \right\rangle. \tag{63}$$

The first register has size n, z is an m-bit integer and $\alpha_i^{(m)} = \lfloor 2^m \alpha_i \rfloor$. Oracle O_{rot} implements

$$O_{\text{rot}} \left| \alpha_i^{(m)} \right\rangle |0\rangle = \left| \alpha_i^{(m)} \right\rangle (\sin\theta_i |0\rangle + \cos\theta_i |1\rangle) \tag{64}$$

with $\theta_i = \arcsin\left(\alpha_i^{(m)}/2^m\right)$ so that $\sin\theta_i = \alpha_i^{(m)}/2^m \approx \alpha_i$. The goal is to prepare the state $\frac{1}{\|\boldsymbol{\alpha}\|_2} \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$. The original black-box algorithm by Grover [84] first performs

$$(I^{\otimes n} \otimes O_{\text{rot}})(O_{\text{amp}} \otimes I)(H^{\otimes n} \otimes I^{\otimes(m+1)}) |0\rangle^{\otimes n} |0\rangle^{\otimes m} |0\rangle = \frac{1}{\sqrt{2^n}} \sum_i |i\rangle \left| \alpha_i^{(m)} \right\rangle (\sin\theta_i |0\rangle + \cos\theta_i |1\rangle) \tag{65}$$

and then uncomputes the second register by applying O_{amp}. Post-selecting on the last register being 0 outputs the approximate target state with success probability $p_{\text{success}} = \frac{\|\boldsymbol{\alpha}^{(m)}/2^m\|_2^2}{2^n} \approx \frac{\|\boldsymbol{\alpha}\|_2^2}{2^n}$. It only approximates the target state because of the m-bit precision of $\alpha_i^{(m)}$ and the finite precision of computing the arcsine inside the oracle (we assume the same precision m for asymptotic scaling statements). The original protocol prescribes

amplitude amplification to boost the success probability close to 1. Instead here we state the success probability of post-selection without amplitude amplification to align with the presentation of our algorithm. As noted in the main text, our algorithm – or any probabilistic algorithm discussed here – could also use amplitude amplification at the cost of increasing the number of gates by a factor of $\mathcal{O}(p_{\text{success}}^{-1/2})$. The black-box protocol can be generalized to complex amplitudes by considering two amplitude oracles, one for the magnitude $|\alpha_i|$ and one for its phase [35].

To connect with our work, consider $\alpha_i' = f_d(\boldsymbol{x}_i)$ with grid points $\boldsymbol{x}_i \in [-1,1]^D$ for $i = 0, \ldots, 2^{Dn}-1$ and f_d a multivariate polynomial approximation of f with, without loss of generality, degree d for each variable. We use the rescaled amplitudes $\alpha_i = \alpha_i'/|f_d|_{\max}$ in the oracle O_{amp} of the black-box approach. Then the second register holds $f_d(\boldsymbol{x})/|f_d|_{\max}$ with m-bit precision. We assume that the $\mathcal{O}(d^D)$ coefficients of f_d are computed in a classical preprocessing step similar to ours. Then we get the following resources for this oracle from Ref. [89][1]

$$G_{\text{oracle}} = \mathcal{O}(d^D m^2) \text{ two-qubit gates} \qquad \text{and} \qquad A_{\text{oracle}} = \mathcal{O}(d^D m) \text{ ancillary qubits.} \tag{66}$$

The same asymptotic complexity holds for computing the arcsine in O_{rot} with d replaced by the degree for a sufficiently accurate polynomial approximation to the arcsine (and $D = 1$). The oracle stores the computed value in another register to desired precision (for simplicity assumed to be m). Then its value is transduced to a phase by a series of controlled rotations, which do not change the asymptotic complexity. Hence, the overall asymptotic resources for both oracles are the same. The success probability is

$$p_{\text{success}} = \frac{\|\boldsymbol{f}_d^{(m)}\|_2^2}{2^{Dn}|f_d|_{\max}^2} = \tilde{\mathcal{F}}_2^2. \tag{67}$$

For sufficiently high precision m and degree d this success probability is approximated by $\mathcal{F}_2^2$.

The preprocessing cost of the above implementation of the black-box approach is the same as ours because both require computing coefficients for a maximal degree-d approximating polynomial. The black-box approach also requires precomputing a bound on the maximum of f_d. This is an additional classical preprocessing cost and the accuracy of this bound influences the required precision m. As discussed at the beginning of Sec. D we could use the bound $\|\boldsymbol{c}\|_1$ using the coefficients of f_d. Then the success probability of the black-box approach and this work (see Table 4) are essentially the same (up to e.g. differences due to m-bit precision in the black-box approach).

The two-qubit gate count for the black-box approach is likely higher than ours. This can be seen from the prefactors of the leading term $\mathcal{O}(d^D)$ (m^2 vs 3 in our algorithm from the unitaries A, $A^\dagger$, C). The number of Toffoli gates for several univariate functions in Ref. [89, Table II] suggests an order of magnitude of ten thousands for preparing univariate functions. Gate counts for multivariate functions will be higher due to the factor $\mathcal{O}(d^D)$. This should be compared to our explicit gate counts, e.g. in Figs. 7-8. Note, however, that this comparison is only indicative because we report optimized ZZPhase gate counts whereas Ref. [89] reports Toffolis. We also expect the black-box approach to require considerably more ancillas than our approach based on the explicit numbers in Ref. [89, Table II] for univariate functions and the favorable asymptotic scaling of our method in Table 4.

D.2 Grover-Rudolph algorithm

The algorithm by Zalka [25] applies to target functions $f(x) = \sqrt{p(x)}$, where $p(x)$ is an efficiently integrable probability density function. The algorithm was independently rediscovered by Grover and Rudolph [26] and is commonly called Grover-Rudolph algorithm. A construction for efficiently integrable multivariate distributions is given in Ref. [85]. At each step $l = 1, \ldots, Dn$ of the algorithm an oracle encodes angles $\theta_i = \arccos\left(\alpha_{l,i}^{(m)}\right)$ for $i = 0, \ldots, 2^l$ into an ancilla register, performs controlled rotations on a fresh qubit added to the main register, and uncomputes the ancilla register. The angles are stored in an ancilla register with sufficient precision (assumed to be m). The value of $\alpha_i^{(m)}$ is determined by the integral over a part of the target distribution into an ancilla register. The oracle is essentially O_{amp} in the black-box approach.

To compare resource requirements to our method we assume the same implementation of O_{amp} as in App. D.1. This does not take into account resources for the coherent integration required for computing the angles. The total number of two-qubit gates is higher by a factor Dn than in the black-box because of the iterative procedure of building up the main register over Dn steps. Hence, in general, we expect much higher gate counts than in our algorithm (cf. the leading term in two-qubit gates in Table 4). The algorithm also requires complicated

[1] For simplicity, we only consider one polynomial over the entire domain instead of a piecewise polynomial approximation. This does not affect asymptotic complexity. Furthermore, it is conceivable that the quantum algorithm for evaluating polynomials in the monomial basis in Ref. [89] can be generalized to evaluating polynomials in the Fourier and Chebyshev polynomials at similar cost by using, e.g., the Fourier/Chebyshev polynomial recurrence relation or the Clenshaw algorithm instead of a Horner scheme.

arithmetic circuits to implement, while ours does not. We expect high numbers of ancillas similar to the black-box approach, i.e. considerably higher than ours. The success probability of Grover-Rudolph is 1, whereas ours is smaller in general. The algorithm itself does not require classical preprocessing as the arithmetics are absorbed in the oracle. However, an explicit oracle implementation may require preprocessing. For example, the implementation above requires finding polynomial coefficients; likely with similar or worse preprocessing cost as ours. Variants of Grover-Rudolph have been proposed for the special case of multivariate Gaussian distributions [27]; however, explicit circuit constructions [28] showed that they require more resources than generic state preparation methods ($\mathcal{O}(2^n)$ gates) in practice for moderate n.

D.3 Adiabatic algorithm

The adiabatic algorithm of Ref. [34] works for any complex function f assuming efficient pointwise evaluation.[2] It assumes access to the oracle O_{amp} defined in the black-box approach, App. D.1, to evaluate and store intermediate function evaluations in an ancilla register with m-bit precision.

The algorithm performs a discretized adiabatic evolution under the rank-1 Hamiltonian $H(s) \propto |\psi_{[s]}\rangle\langle\psi_{[s]}|$ with $|\psi_{[s]}\rangle \propto \sum_{\boldsymbol{x_i}} f_{[s]}^{(m)}(\boldsymbol{x}_i)\,|\boldsymbol{x}_i\rangle$ and the parameterized function $f_{[s]} = (1-s)f_0 + sf$, where $f_0 \propto 1$ has an easy state preparation circuit, f is target function of interest and $0 \geq s \geq 1$. Each step of the discretized evolution is implemented with a sparse Hamiltonian simulation algorithm. The authors show that the total evolution time is lower-bounded by a constant independent of n. The number of steps for error ϵ (in spectral distance to the exact unitary) is $\mathcal{O}(1/\epsilon^2 \tilde{\mathcal{F}}_1^4)$. The authors state that knowledge of the filling fraction is not required to run the algorithm.

Resources are dominated by the oracle evaluating the function. For the comparison in Table 4 we assume the implementation discussed in App. D.1. At intermediate steps O_{amp} computes and stores matrix elements $A_{ij}(s) = (f_{[s]}^{(m)}(\boldsymbol{x}_i))^* f_{[s]}^{(m)}(\boldsymbol{x}_j)$ in an ancilla register with m-bit precision. In addition, $\mathcal{O}(Dn)$ ancillas are required to store the location of the nonzero entries of a 1-sparse matrix required for the sparse Hamiltonian simulation. Compared to our algorithm, adiabatic state preparation is nearly deterministic. Assuming the above oracle implementation, the classical pre-processing cost is similar to ours. However, we expect that an implementation of this algorithm requires considerably more ancillas than our algorithm (see discussion in App. D.1). We also expect more two-qubit gates based on the larger prefactor of the term $\mathcal{O}(d^D)$ (m^2 vs. 3 in our case).

D.4 Matrix product states

Matrix product states (MPS) algorithms are often heuristic in nature making a precise comparison to non-heuristic quantum state preparation methods such as ours difficult. We seek to represent a D-dimensional function discretized on 2^{Dn} grid points, which requires an MPS with Dn tensors. Apart from the number of tensors, the complexity of an MPS is determined by its bond dimension χ, and typical operations on the MPS will cost $\mathcal{O}(\text{poly}(\chi)Dn)$ [90]. In the context of state preparation we need to consider (i) the effectiveness and classical cost of representing a multivariate target function with an MPS, and (ii) the resources required to map the MPS to a quantum circuit.

We first discuss point (i). States representing smooth univariate functions with bounded derivative have small von Neumann entropy [50]. While this suggests an MPS representation with small bond dimension, the equivalence is not rigorous [91]. On the other hand, univariate degree-d polynomials can be efficiently represented by an MPS with $\chi = d+1$ [92, 93]. A way to extend those results to D-dimensional functions is to prescribe the additional variables into the ordering of the tensors of an MPS of size Dn. The ordering can have significant influence on the bond dimension and it is unclear a-priori whether a given multivariate function is well represented by an MPS. For example, Ref. [50] observed a increased, but relatively small, von Neumann entropy for 2D Gaussians. In contrast, Ref. [94] observed a significant increase of the von Neumann entropy with the dimension for the solutions of the nonlinear Schrödinger equation, suggesting that the MPS is not well suited for those functions. Ref. [95] studied low-rank approximations to a restricted class of multivariate functions that avoids the exponential-in-D cost $\mathcal{O}(d^D)$ of computing coefficients such that they can be represented by MPS Chebyshev interpolants with low bond order. We note that similar low-rank approximations could also be used in our quantum algorithm as a preprocessing step.

With regards to our point (ii) above, Ref. [86] constructs an exact MPS with χ-level qudits, which can be represented with qubits and a quantum circuit using $\log \chi$ ancillas and depth $\mathcal{O}(\text{poly}(\chi)Dn)$ [87]. Refs. [96, 97]

[2] The authors also discuss an alternative state preparation for efficiently integrable functions that could be beneficial in practice for discontinuous functions. Here we focus on state preparation with pointwise evaluation.

propose heuristic mappings from MPS to quantum circuits using T layers of gates, for a total number of two-qubit gates scaling as $\mathcal{O}(TDn)$, and without using ancilla qubits.

There are very few end-to-end studies of quantum state preparation using MPS. Ref. [31] considers smooth, real, univariate functions. They perform a piecewise polynomial approximation of the target function, translate each degree-d polynomial into an MPS, and add them up. The result is mapped to a quantum circuit using Ref. [96]. They assume $\chi = 2$ approximates the target function well enough on a 2^n grid which yields a two-qubit gate count of $\mathcal{O}(n)$. The assumption $\chi = 2$ is, however, in general not satisfied. Ref. [33] considers univariate normal distributions and obtains comparable results.

The discussion above suggests that, generally, the bond order will increase in higher dimensions. Hence, in Table 4 we state the more general two-qubit count $\mathcal{O}(\text{poly}(\chi)Dn)$. Overall, we expect MPS to lead to shallower circuits than ours for smooth univariate functions. For higher dimensions the comparison is less clear and requires a dedicated study due to the heuristic nature of MPS algorithms. We emphasize that our multivariate state preparation method is not heuristic and it has a simple circuit construction.

D.5 Fourier series loader

The Fourier series loader (FSL) [49] applies to complex multivariate functions that can be written as a Fourier series. This is essentially the same class of functions representable with our Fourier approach. FSL prepares a truncated Fourier series of the form of Eq. 3 (in arbitrary dimensions D) on 2^{Dn} equidistant grid points on $[0,1]^D$. A minor difference in presentation is that Ref. [49] also allows discontinuous functions, whereas we chose to exclude them via our Lipshitz assumption on f. The Fourier series of a discontinuous function exhibits large errors (Gibbs phenomenon). Ref. [49] discusses heuristic modifications of the Fourier coefficients to suppress those errors. We note that our algorithm would also work if we loaded such modified coefficients at the expense of the loss of fast uniform convergence guarantees and a small preprocessing cost to compute them.

FSL first loads the $(2d+1)^D$ Fourier coefficients into a $(a = \lceil D \log_2(2d+1) \rceil)$-qubit register. This is essentially a generic state preparation circuit for arbitrary complex amplitudes similar in purpose to the unitaries A, C, and $A^\dagger$ in our method. The authors suggest two suitable implementations: uniformly controlled rotations [18], and using the Schmidt decomposition of the coefficient state. Next, their algorithm prescribes a ladder of CX on the remaining $Dn - a$ qubits controlled on the top qubit of the coefficient register. The application of one inverse quantum Fourier transform (QFT) per variable register of size n yields the desired state on the full register of size Dn. The D inverse QFTs require $\mathcal{O}(Dn^2)$ two-qubit gate, which is their leading gate count scaling in n. We expect that their approach uses fewer two-qubits in regimes with small n and ours uses fewer in the high n regime. Their algorithm is deterministic and uses no ancillas. On the other hand, our protocols can also represent Chebyshev series.

D.6 Multivariate quantum signal processing

The algorithm in Ref. [44] applies to univariate functions $f\colon [a,b] \to \mathbb{C}$. Given a polynomial approximation of $f((b-a)\arcsin(y)+a)/|f|_{\max}$ their method implements this polynomial with quantum signal processing (QSP). Similar to our work, their algorithm works with continuous polynomial approximations rather than storing a discretization in an auxiliary register, such as in the black-box or Grover-Rudolph approach. Without amplitude amplification the resource requirements are $\mathcal{O}(nd)$ two-qubit gates, 2 (definite parity polynomials) or 3 ancillas (mixed parity) and $\tilde{\mathcal{F}}_2^2$ success probability. This success probability is approximated by $\mathcal{F}_2^2$.

Ref. [88] proposed a multivariate extension based on a multivariate QSP (M-QSP) ansatz [45–47]. The implemented function has the form $f(2\arccos(y_1) - 1, 2\arccos(y_2) - 1, \ldots, 2\arccos(y_D) - 1))$. The authors construct commuting block-encodings U_j of $\sum_{x_j} \cos(x_j)\,|x_j\rangle\langle x_j|$ for each variable $j = 1, \ldots, D$, where the sum runs over 2^n grid points of x_j. This can be done with one ancilla and a total of Dn controlled rotations. The algorithm prescribes a length-d sequence alternating between one of the block-encodings selected per step followed by a single-qubit rotation on the ancilla plus one final single-qubit rotation. The angles are determined by a maximal degree-d polynomial approximation of the target function. Post-selecting on the ancilla being in state 0 gives the desired quantum state in the main register of size Dn. The success probability is again $\tilde{\mathcal{F}}_2^2$. Similar to the univariate QSP protocol in Ref. [44] the multivariate protocol requires pre-computation of $|f|_{\max}$ with the caveats discussed above.

While this proposal uses asymptotically fewer gates and ancillas than ours, the class of functions representable by M-QSP is limited and largely not characterized (except for the homogeneous, commuting bivariate case [47]). This is clear from a counting argument. A D-dimensional maximal degree-d polynomial has at most $\mathcal{O}(d^D)$ coefficients, whereas M-QSP only has $\mathcal{O}(d)$ free phases. Moreover, the classical cost and numerical stability of computing rotation angles for general M-QSP is currently not well understood. In contrast, our algorithm works for a wide class of multivariate functions and does not require this additional classical preprocessing.