

ARTICLE OPEN



Long-time simulations for fixed input states on quantum hardware

Joe Gibbs¹✉, Kaitlin Gili^{1,2}, Zoë Holmes³✉, Benjamin Commeau^{3,4}, Andrew Arrasmith¹, Lukasz Cincio¹, Patrick J. Coles¹✉ and Andrew Sornborger³

Publicly accessible quantum computers open the exciting possibility of experimental dynamical quantum simulations. While rapidly improving, current devices have short coherence times, restricting the viable circuit depth. Despite these limitations, we demonstrate long-time, high fidelity simulations on current hardware. Specifically, we simulate an XY-model spin chain on Rigetti and IBM quantum computers, maintaining a fidelity over 0.9 for 150 times longer than is possible using the iterated Trotter method. Our simulations use an algorithm we call fixed state Variational Fast Forwarding (fsVFF). Recent work has shown an approximate diagonalization of a short time evolution unitary allows a fixed-depth simulation. fsVFF substantially reduces the required resources by only diagonalizing the energy subspace spanned by the initial state, rather than over the total Hilbert space. We further demonstrate the viability of fsVFF through large numerical simulations, and provide an analysis of the noise resilience and scaling of simulation errors.

npj Quantum Information (2022)8:135; <https://doi.org/10.1038/s41534-022-00625-0>

INTRODUCTION

The simulation of physical systems is valuable for basic science and has technological applications across a diverse range of industries, from materials design to pharmaceutical development. Relative to classical computers, quantum computers have the potential to provide an exponentially more efficient means of simulating quantum mechanical systems. Quantum hardware has progressed substantially in recent years^{1,2}. However, despite continual progress, we remain in the ‘noisy intermediate-scale quantum’ (NISQ) era in which the available hardware is limited to relatively small numbers of qubits and prone to errors. Simulation algorithms designed for fault-tolerant quantum computers, such as Trotterization methods^{3,4}, qubitization methods⁵, and Taylor series methods⁶, require deeper circuits than viable given the short coherence times of current hardware. Thus alternative approaches are needed to successfully implement useful simulations on NISQ hardware.

Variational quantum algorithms^{7–22}, where a classical computer optimizes a cost function measured on a quantum computer, show promise for NISQ quantum simulations. An early approach introduced an iterative method, where the state is variationally learned on a step-by-step basis using action principles^{18,19,23,24}. Subsequently, a generalization of the variational quantum eigensolver¹⁰ was developed for simulations in low lying energy subspaces²⁰. Very recently, quantum-assisted methods have been proposed that perform all necessary quantum measurements at the start of the algorithm instead of employing a classical-quantum feedback loop^{25–27}.

In this work, we improve upon a recently proposed variational quantum algorithm known as Variational Fast Forwarding (VFF)²¹. VFF allows long time simulations to be performed using a fixed depth circuit, thus enabling a quantum simulation to be ‘fast forwarded’ beyond the coherence time of noisy hardware. The VFF algorithm requires finding a full diagonalization of the short time

evolution operator U of the system of interest. Once found, the diagonalization enables any initial state of that system to be fast forwarded. However, for practical purposes, one is often interested in studying the evolution of a particular fixed initial state of interest. In that case a full diagonalization of U is overkill. Instead, it suffices to find a diagonal compilation of U that captures its action on the given initial state. Here, we show that focusing on this commonly encountered but less exacting task can substantially reduce the resources required for the simulation.

Specifically, we introduce the fixed state VFF algorithm (fsVFF) for fast forwarding a fixed initial state beyond the coherence time of a quantum computer. This approach is tailored to making dynamical simulation more suitable for NISQ hardware in two key ways. First, the cost function requires half as many qubits as VFF. This not only allows larger scale simulations to be performed on current resource-limited hardware, but also has the potential to enable higher fidelity simulations since larger devices tend to be noisier. Second, fsVFF can utilize simpler ansätze than VFF both in terms of the depth of the ansatz and the number of parameters that need to be learnt. Thus, fsVFF can reduce the width, depth, and total number of circuits required to fast forward quantum simulations, hence increasing the viability of performing simulations on near-term hardware.

We demonstrate these advantages by implementing long-time high fidelity quantum simulations of the 2-qubit XY spin chain on Rigetti’s and IBM’s quantum computers. Specifically, while the iterated Trotter approach has a fidelity of less than 0.9 after 4 time steps and has completely thermalized by 25 time steps, with fsVFF we achieve a simulation fidelity greater than 0.9 for over 600 time steps. We further support the effectiveness of this approach for NISQ simulations, with 4 qubit noisy and 8 qubit noiseless numerical simulations of the XY model and Fermi–Hubbard model respectively.

¹Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM, USA. ²Department of Physics, University of Oxford, Clarendon Laboratory, Oxford, UK. ³Information Sciences, Los Alamos National Laboratory, Los Alamos, NM, USA. ⁴Department of Physics, University of Connecticut, Storrs, CT, USA. ✉email: joegibbs@gmail.com; zholmes@lanl.gov

The fsVFF algorithm marks a conceptual advance on other variational simulation algorithms in that it draws on ideas from quantum machine learning by using training data consisting of the evolution of the system at short times to formulate our cost. That is, in effect, we are using training data composed of the short-time evolution of our initial state to predict its long-time evolution. Moreover, to ground this approach and prove the faithfulness of our cost function, we utilize the recently developed Quantum No Free Lunch theorems^{28,29}, which bound the amount of training data required to learn an unknown unitary. In our analytical results, we further provide a proof of the noise resilience of the fsVFF cost function, specifically the optimal parameter resilience³⁰. Finally, we perform an analysis of simulation errors under fast-forwarding.

The diagonalizations obtained using fsVFF may further be useful for determining the eigenstates and eigenvalues of the Hamiltonian on the subspace spanned by the initial state and its evolved states. This can be done using a time series analysis, by using fsVFF to reduce the depth of the quantum phase estimation (QPE) algorithm, or using a simple sampling method. We demonstrate on IBM's quantum computer that, while standard QPE fails on real hardware, fsVFF can be used to obtain accurate estimates of the spectrum.

RESULTS

Fixed state variational fast forwarding algorithm

Before presenting our fsVFF algorithm, let us first review the original VFF algorithm from ref. ²¹. Consider a Hamiltonian H on a $d = 2^n$ dimensional Hilbert space (i.e., on n qubits) evolved for a short time Δt with the simulation unitary $e^{-iH\Delta t}$, and let T (larger than Δt) denote the desired simulation time. Then the VFF algorithm consists of the following steps:

1. Approximate $e^{-iH\Delta t}$ with a single-timestep Trotterized unitary denoted $U = U(\Delta t)$.
2. Variationally search for an approximate diagonalization of U by compiling it to a unitary with a structure of the form

$$V(\mathbf{a}, \Delta t) := W(\boldsymbol{\theta})D(\boldsymbol{\gamma}, \Delta t)W(\boldsymbol{\theta})^\dagger, \quad (1)$$

where $\mathbf{a} = (\boldsymbol{\theta}, \boldsymbol{\gamma})$ is a vector of parameters. Here, $D(\boldsymbol{\gamma}, \Delta t)$ is a parameterized unitary that will (after training) encode the eigenvalues of $U(\Delta t)$, while $W(\boldsymbol{\theta})$ is a parameterized unitary matrix that will consist of the corresponding eigenvectors²¹. The compilation is performed using the local Hilbert–Schmidt test¹³ to find the parameters $\boldsymbol{\theta}_{\text{opt}}$ and $\boldsymbol{\gamma}_{\text{opt}}$ that minimize the local Hilbert–Schmidt cost.

3. Use the compiled form to simulate for time $T = N\Delta t$ using the circuit

$$W(\boldsymbol{\theta}_{\text{opt}})D(\boldsymbol{\gamma}_{\text{opt}}, N\Delta t)W(\boldsymbol{\theta}_{\text{opt}})^\dagger. \quad (2)$$

VFF has proven effective for providing a fixed quantum circuit structure with which to fast-forward beyond the coherence time of current noisy quantum devices. However, the algorithm requires a full diagonalization of U over the entire Hilbert space. The local Hilbert–Schmidt test used to find this diagonalization requires $2n$ qubits. Additionally, the ansatz must be sufficiently expressible to diagonalize the full unitary U to a high degree of approximation^{31–33}. This typically requires a large number of parameters and a reasonably deep circuit. These overheads limit VFF's utility on current hardware.

In what follows, we introduce a more NISQ-friendly refinement to VFF that reduces these overheads when one is interested in fast-forwarding a *fixed* initial state $|\psi_0\rangle$, rather than any possible initial state. The fixed state VFF algorithm (fsVFF) is summarized in Fig. 1.

We note that VFF, like the standard iterated Trotter approach to quantum simulation, necessarily incurs a Trotter error by approximating $e^{-iH\Delta t}$ with $U = U(\Delta t)$. This Trotter error may be removed using the Variational Hamiltonian Diagonalization algorithm (VHD), which directly diagonalizes the Hamiltonian H ²². However, VHD is yet more resource intensive than VFF on current hardware, so we focus here on refining VFF.

Cost function

In fsVFF, instead of searching for a full diagonalization of U over the entire Hilbert space, we search for a diagonal compilation of U that captures the action of U on the initial state $|\psi_0\rangle$ and its future evolution, $e^{-iHt}|\psi_0\rangle$. Here, we introduce a cost function tailored to this task.

To make precise what is required of the cost for fsVFF, let us first note that as the state $|\psi_0\rangle$ evolves, it remains within a fixed energy subspace. This can be seen by expanding the initial state in terms of the energy eigenbasis $\{|E_k\rangle\}_{k=1}^{2^n}$ (the eigenbasis of H) as

$$|\psi_0\rangle = \sum_{k=1}^{n_{\text{eig}}} a_k |E_k\rangle, \quad (3)$$

where $a_k = \langle E_k | \psi_0 \rangle$, and noting that

$$e^{-iHt}|\psi_0\rangle = \sum_{k=1}^{n_{\text{eig}}} a_k e^{-iE_k t} |E_k\rangle. \quad (4)$$

Thus it follows that if $|\psi_0\rangle$ has non-zero overlap with n_{eig} energy eigenstates of H , so does $e^{-iHt}|\psi_0\rangle$ for all future times. It follows that the subspace spanned by the initial state $|\psi_0\rangle$ and its future evolution, is equivalent to the n_{eig} dimensional subspace \mathcal{S}_{ψ_0} spanned by $\{|E_k\rangle\}_{k=1}^{n_{\text{eig}}}$. Therefore to find a compilation of U that captures its action on $e^{-iHt}|\psi_0\rangle$ (for all times t) it suffices to find a compilation of U on \mathcal{S}_{ψ_0} . We stress that the eigenstates $\{|E_k\rangle\}_{k=1}^{2^n}$ need not be ordered, and therefore \mathcal{S}_{ψ_0} is not necessarily low lying in energy.

A No-Free-Lunch Theorem for quantum machine learning introduced in ref. ²⁸ proves that to perfectly learn the action of a unitary on a d -dimensional space requires d training pairs. In the context of fsVFF, we are interested in learning the action of a unitary on an n_{eig} -dimensional subspace. Since the unitary is block diagonal, one can directly apply this NFL theorem to the subspace of interest. Therefore n_{eig} training pairs are required to learn the unitary's action on this subspace. (Note, we assume here that the training states are not entangled with an additional register. It was shown in ref. ²⁹ that using entangled training data can reduce the required number of training states. In fact, this more powerful method is used by the VFF algorithm. However, producing such entangled training data requires additional qubits and two-qubit gates and therefore is less NISQ-friendly.)

The No-Free-Lunch theorem therefore implies that n_{eig} states are required to learn U on $|\psi_0\rangle$ (assuming leakage due to Trotter error is negligible). In general these states may be freely chosen from \mathcal{S}_{ψ_0} . Here a convenient choice in training states would be $|\psi_0\rangle$ and its Trotter evolutions, that is the set $\{U^k|\psi_0\rangle\}_{k=1}^{n_{\text{eig}}}$. Motivated by these observations, we define our cost function for fsVFF as

$$C_{\text{fsVFF}} := 1 - \frac{1}{n_{\text{eig}}} \sum_{k=1}^{n_{\text{eig}}} |\langle \psi_0 | (V^\dagger)^k U^k | \psi_0 \rangle|^2, \quad (5)$$

where similarly to VFF we use a diagonal ansatz $V(\mathbf{a}, \Delta t) := W(\boldsymbol{\theta})D(\boldsymbol{\gamma}, \Delta t)W(\boldsymbol{\theta})^\dagger$. This cost quantifies the overlap between the initial state evolved under U for k time steps, $U^k|\psi_0\rangle$, and the initial state evolved under the trained unitary for k time steps, $WD^k W^\dagger|\psi_0\rangle$, averaged over n_{eig} time steps. Assuming we have access to the unitary that prepares the state $|\psi_0\rangle$, the state overlaps can be

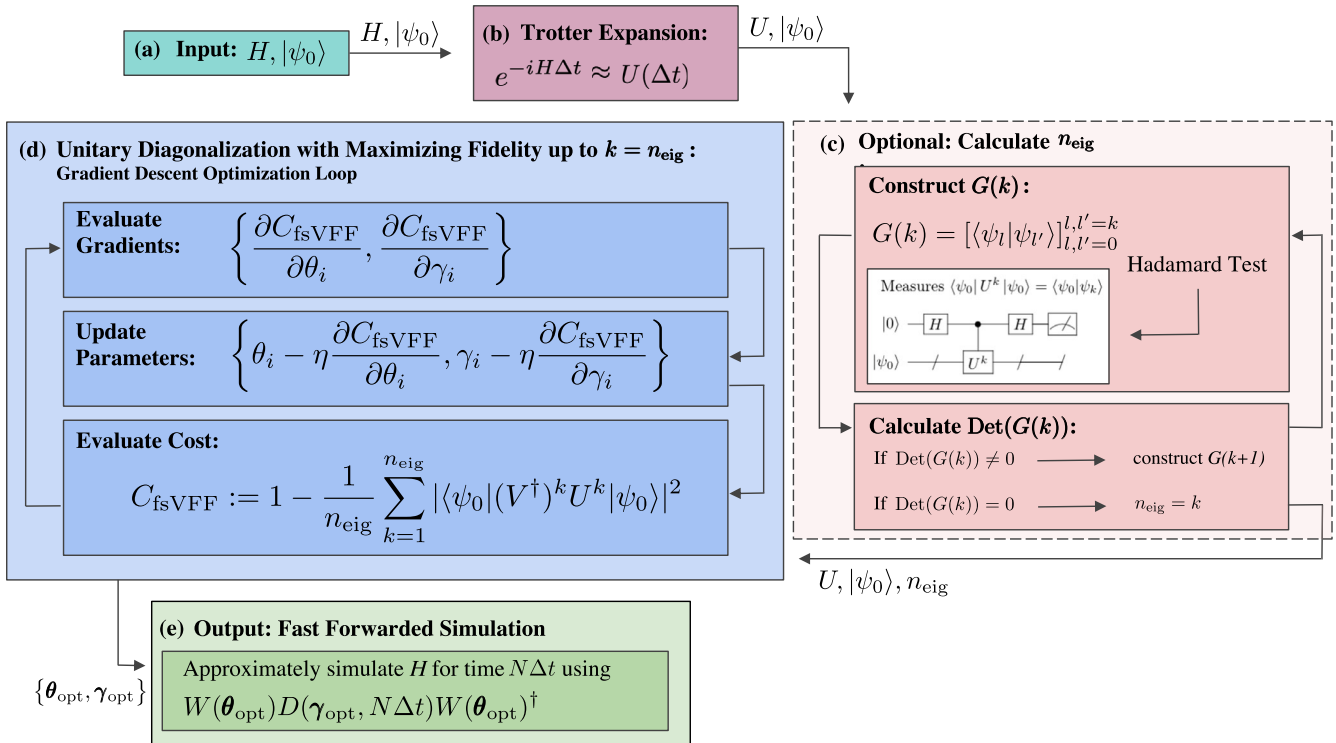


Fig. 1 The fsVFF Algorithm. **a** An input Hamiltonian and an initial input state are necessary **(b)** to create a single time-step Trotterized unitary, $U(\Delta t)$ and **(c)** to calculate the number of eigenstates that have non-zero overlap with the initial state. The value of n_{eig} can be calculated by constructing a matrix of state overlaps $U^k|\psi_0\rangle$ and increasing the matrix dimension until the determinant is zero. **d** The unitary is then variationally diagonalized into the form, $V(\alpha, \Delta t) = W(\theta)D(\gamma, \Delta t)W^\dagger(\theta)$. The cost function C_{fsVFF} is minimized with a classical optimizer (e.g., gradient descent), where the parameters θ and γ are updated. **e** The optimal parameters θ_{opt} and γ_{opt} are then used to implement a fast-forwarded simulation with the diagonalized unitary form.

measured using n qubits, via a circuit that performs a Loschmidt echo³⁰. Therefore C_{fsVFF} can be evaluated using only n qubits. This is half as many as standard VFF, opening up the possibility of performing larger simulations on current hardware.

It is important to note that while the exact time-evolved state $\exp(-iHt)|\psi_0\rangle$ is perfectly confined to the initial subspace, the approximate evolution induced by $U(\Delta t)$ allows for leakage from the initial subspace³⁴. Thus the subspace spanned by $\{U^k|\psi_0\rangle\}_{k=1}^{n_{\text{eig}}}$ in general does not perfectly overlap with $\{|E_k\rangle\}_{k=1}^{n_{\text{eig}}}$. However, by reducing Δt and considering higher order Trotter approximations^{4,35}, this leakage can be made arbitrarily small. In Supplementary Note 1, we prove that in the limit that leakage from the initial subspace is negligible, C_{fsVFF} is faithful. That is, we show that the cost vanishes, $C_{\text{fsVFF}} = 0$, if and only if the fidelity of the fast-forwarded simulation is perfect,

$$F_\tau = |\langle \psi_0 | W^\dagger D^\dagger W U^\tau | \psi_0 \rangle|^2 = 1, \quad (6)$$

for all times τ . Note, that the reverse direction is trivial. If $F_\tau = 1$ for all τ , then $C_{\text{fsVFF}} = 0$.

Similar to the VFF cost, the fsVFF cost is noise resilient in the sense that incoherent noise should not affect the global optimum of the function. This is proven for a broad class of incoherent noise models using the results of ref. ³⁰ in Supplementary Note 2.

Nonetheless, it is only possible to measure C_{fsVFF} if the unitary $U^{n_{\text{eig}}}$ can be implemented comfortably within the coherence time of the QC. Additionally, the number of circuits required to evaluate C_{fsVFF} in general scales with n_{eig} . Given these two restrictions, fsVFF is limited to simulating quantum states that have non-zero overlap with a non-exponential number of eigenstates. Consequently, we advocate using fsVFF to simulate states with $n_{\text{eig}} = \text{poly}(n)$. Crucially these states need *not* be low lying and therefore our approach is more widely applicable than the

Subspace Variational Quantum Simulator (SVQS) algorithm²⁰, which simulates fixed low energy input states.

While C_{fsVFF} was motivated as a natural choice of cost function to learn the evolution induced by a target unitary on a fixed initial state, it is a global cost³⁶ and hence it encounters what is known as a *barren plateau* for large simulation sizes^{33,36-46}. In Supplementary Note 3 we suggest an alternative local version of the cost to mitigate such trainability issues.

Calculating n_{eig}

In this section, we present an algorithm to calculate n_{eig} and therefore determine the number of training states required to evaluate C_{fsVFF} . Before doing so, we remark that in the presence of degeneracies in the spectrum of H , the eigenvectors corresponding to degenerate eigenvalues are not unique. Therefore, in this case, the number of eigenstates with non-zero overlap with $|\psi_0\rangle$ depends on how the eigenvectors corresponding to degenerate eigenvalues are chosen. However, as detailed in Supplementary Note 1, to learn the action of U on $|\psi_0\rangle$, what matters is the number of eigenstates overlapped by $|\psi_0\rangle$ corresponding to *unique* eigenvalues. This is equivalent to the number of linearly independent states in the set \mathcal{V}_∞ where $\mathcal{V}_k := \{|\psi_l\rangle\}_{l=0}^k$ with $|\psi_l\rangle := U(\Delta t)^l|\psi_0\rangle$. The subspace $\mathcal{K}_k(U, \psi_0)$ spanned by \mathcal{V}_k is known as the Krylov subspace associated with the operator U and vector $|\psi_0\rangle$ ⁴⁷. In the limit in which Trotter error is negligible $\mathcal{K}_\infty(U, \psi_0) = \mathcal{S}_{\psi_0}$ and therefore, n_{eig} is equivalently the dimension of the Krylov subspace $\mathcal{K}_\infty(U, \psi_0)$.

To determine the dimension of $\mathcal{K}_\infty(U, \psi_0)$ we can utilize the fact that the determinant of the Gramian matrix of a set of vectors (i.e., the matrix of their overlaps) is zero if and only if the vectors are linearly dependent. The Gramian corresponding to \mathcal{V}_k is given

by

$$G(k) = \begin{pmatrix} \langle \psi_0 | \psi_0 \rangle & \langle \psi_0 | \psi_1 \rangle & \cdots & \langle \psi_0 | \psi_k \rangle \\ \langle \psi_1 | \psi_0 \rangle & \langle \psi_1 | \psi_1 \rangle & \cdots & \langle \psi_1 | \psi_k \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \psi_k | \psi_0 \rangle & \langle \psi_k | \psi_1 \rangle & \cdots & \langle \psi_k | \psi_k \rangle \end{pmatrix}. \quad (7)$$

If $\text{Det}(G(k)) \neq 0$, then the vectors in \mathcal{V}_k are linearly independent and therefore span *at least* a $k+1$ dimensional subspace. Conversely, if $\text{Det}(G(k)) = 0$, the set \mathcal{V}_k contains linear dependencies and the subspace they span is less than $k+1$ dimensional. Therefore, if we can find k_{\min} , the smallest k such that $\text{Det}(G(k)) = 0$, then (noting that $G(k)$ is a $k+1$ dimensional matrix) we know that k_{\min} is the largest number of linearly independent vectors spanned by \mathcal{V}_{∞} . That is, k_{\min} is the dimension of $\mathcal{K}_{\infty}(U, \psi_0)$ and so we have that $n_{\text{eig}} = k_{\min}$.

The overlaps $\langle \psi_l | \psi_{l'} \rangle$ for any l and l' can be measured using the Hadamard Test, shown in Fig. 1, and thus the Hadamard test can be used to determine $G(k)$ on quantum hardware. Since the Gramian here contains two symmetries, hermiticity and the invariance $\langle \psi_l | \psi_{l'} \rangle = \langle \psi_0 | U^{-l} U^{l'} | \psi_0 \rangle = \langle \psi_0 | U^{l'-l} | \psi_0 \rangle = \langle \psi_0 | \psi_{l-l'} \rangle$, we only have to calculate the first row of the matrix $G(k)$ on the quantum computer.

In summary, our proposed algorithm to determine n_{eig} consists of the following loop. Starting with $k = 1$,

1. Construct $G(k)$ using the Hadamard test.
2. Calculate (classically) $\text{Det}(G(k))$.
If $\text{Det}(G(k)) = 0$, terminate the loop and conclude that $n_{\text{eig}} = k$.
If $\text{Det}(G(k)) \neq 0$, increase $k \rightarrow k+1$ and return to step 1.

This is shown schematically in Fig. 1.

While it is beneficial to learn n_{eig} to determine how many training states are required to perfectly learn the diagonalization on the subspace spanned by the initial state and its future evolution, we stress that it is not strictly necessary for the successful implementation of fsVFF. One could always train on an increasing number of states and study the convergence of an observable of interest. More concretely, one could train on k states and then use the resultant diagonalization to compute the evolution of a particular observable as a function of time. For $k < n_{\text{eig}}$ the trajectory of the observable will alter as k is increased. However, for $k \geq n_{\text{eig}}$ increasing k further will no longer change the trajectory of the observable because it will have already converged on the true trajectory. Using this approach, n_{eig} need not be already known to implement fsVFF. We demonstrate this method in Section “Numerical Simulations”.

Summary of algorithm

The fixed state Variational Fast Forwarding algorithm (fsVFF) is summarized in Fig. 1. We start with an initial state $|\psi_0\rangle$ that we wish to evolve under the Hamiltonian H .

1. The first step is to approximate the short time evolution using a single step Trotter approximation U .
2. This Trotter approximation can be used to find an approximation for n_{eig} , using the method outlined in Section “Calculating n_{eig} ”.
3. Equipped with a value for n_{eig} , we then variationally search for a diagonalization of U over S_{ψ_0} using C_{fsVFF} , Eq. (5). At each iteration step the gradient of the cost with respect to a parameter θ_i is measured on the quantum computer for a fixed set of parameters using the analytic expressions for $\partial_{\theta_i} C_{\text{fsVFF}}$ provided in Supplementary Note 4. These gradients are used to update the parameters using a classical optimizer, such as those in refs. 48–50. The output of the

optimization loop is the set of parameters that minimize C_{fsVFF} ,

$$\{\boldsymbol{\theta}_{\text{opt}}, \mathbf{y}_{\text{opt}}\} = \arg \min_{\boldsymbol{\theta}, \mathbf{y}} C_{\text{fsVFF}}(\boldsymbol{\theta}, \mathbf{y}). \quad (8)$$

4. Finally, the state $|\psi_0\rangle$ can be simulated for time $T = N\Delta t$ using the circuit

$$W(\boldsymbol{\theta}_{\text{opt}}) D(\mathbf{y}_{\text{opt}}, N\Delta t) W(\boldsymbol{\theta}_{\text{opt}})^\dagger. \quad (9)$$

That is, by simply multiplying the parameters \mathbf{y}_{opt} in the diagonalized unitary by a constant number of iterations N .

In Supplementary Note 5, we show that the total simulation fidelity, in the limit that leakage is small, is expected to scale sub-quadratically with the number of fast-forwarding time steps N . Thus, if the minimal cost from the optimization loop is sufficiently small, we expect the fsVFF algorithm to allow for long, high fidelity simulations.

Randomized training

While the fsVFF cost as stated in Eq. (5) has n_{eig} terms, this does not necessarily mean that the number of circuits required to evaluate it also scales with n_{eig} . Analogous to mini-batch gradient descent methods popular for the training of classical neural networks, we can use only a small random selection of the total training dataset per gradient evaluation, yet over the whole optimization the total training set will be fully explored many times over. Therefore, instead of restricting ourselves to a discrete set of training states, which requires setting the size of the training set to be equal to or greater than n_{eig} , we can instead randomly select our training states from a continuum. This has the added advantage that it is then unnecessary to explicitly compute n_{eig} .

This approach results in a modified cost function of the form

$$\tilde{C}_{\text{fsVFF}} := 1 - \frac{1}{|R|} \sum_{r \in R} |\langle \psi_0 | V(-rt_{\max}) U(rt_{\max}) | \psi_0 \rangle|^2 \quad (10)$$

where $V(t) = WD(t)W^\dagger$ is the fsVFF ansatz, $U(t)$ is a Trotter approximation for the short time unitary evolution, and the elements of the set R are randomly generated numbers from the interval $[-1, 1]$. The gradients of the cost function are smaller when the unitary acts close to the identity operation, so to maintain stronger gradients it is advantageous for the elements of R to be slightly biased towards the edges of the interval. Specifically, in our numerics to test this approach, the absolute magnitude of r was raised to the power of 0.75. Although this approach does not require an a priori calculation of n_{eig} , there is a caveat that t_{\max} needs to be large enough to get sufficient separation of the training states so they are not functionally identical. This alternative training setup is potentially a yet more NISQ friendly variant, as the unitary does not need to be decomposed into the form $U(\Delta t)^{n_{\text{eig}}}$, as required in Eq. (5), and therefore allows for shorter depth circuits.

Hardware implementation

In this section we demonstrate that fsVFF can be used to implement long time simulations on quantum hardware. Specifically, we simulate the XY spin chain, which has the Hamiltonian

$$H_{\text{XY}} := \sum_{j=1}^{n-1} X_j X_{j+1} + Y_j Y_{j+1}, \quad (11)$$

where X_j and Y_j are Pauli operators on the j th qubit. In what follows, we first present results showing that we can determine n_{eig} for an initial state $|\psi_0\rangle$ using the method described in Section “Calculating n_{eig} ”. We then demonstrate that the fsVFF cost can be trained to find an approximate diagonalization of H_{XY} on the

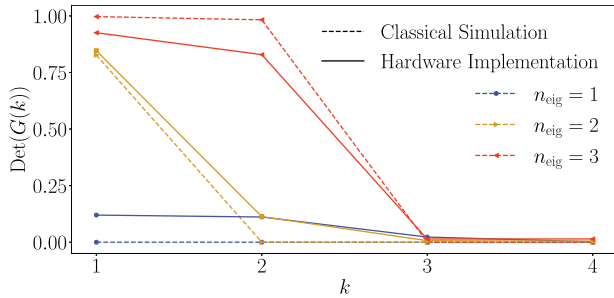


Fig. 2 Gramian Determinant Calculation. Here we plot the determinant of the Gramian matrix, $\text{Det}(G)$, for G measured on the Honeywell quantum computer (solid) and simulated classically (dashed) for a 2-qubit XY spin chain. Specifically we looked at states with non-zero overlap with $k = 1$ (blue), $k = 2$ (yellow) and $k = 3$ (red) eigenstates. For both sets of data $\text{Det}(G(n_{\text{eig}})) \approx 0$, demonstrating the effectiveness of the method for determining n_{eig} that we introduce in Section “Calculating n_{eig} ”. For the Honeywell implementation we used 1000 measurement samples per circuit.

subspace spanned by $|\psi_0\rangle$ and its future evolution under H_{XY} . We finally use this diagonalization to perform a long time fast forwarded simulation. In all cases we focus on a two qubit chain, i.e., $n = 2$, and we approximate its evolution operator using a first-order Trotter approximation.

The 2-qubit XY Hamiltonian has the eigenvectors $\{|00\rangle, \frac{1}{\sqrt{2}}(|10\rangle + |01\rangle), \frac{1}{\sqrt{2}}(|10\rangle - |01\rangle), |11\rangle\}$, corresponding to the eigenvalues $\{0, 1, -1, 0\}$. As proof of principle, we tested the algorithm for determining n_{eig} on the states $|00\rangle$ (corresponding to $n_{\text{eig}} = 1$), $|10\rangle$ ($n_{\text{eig}} = 2$) and $\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$ ($n_{\text{eig}} = 3$). As described in Section “Calculating n_{eig} ”, the n_{eig} of these states can be found by calculating $\text{Det}(G(k))$ for increasing values of k since, as k is increased, the determinant first equals 0 when $k = n_{\text{eig}}$.

To verify this for the states considered here, we first determine G using a classical simulator. As seen in Fig. 2, in this case $\text{Det}(G(k))$ exactly equals 0 when $k = n_{\text{eig}}$. We then measured G on Honeywell’s quantum computer. Although on the real quantum device gate noise and sampling errors are introduced, the results reproduce the classical results reasonably well. Namely, at the correct value of k , $\text{Det}(G(k))$ drastically reduces and approximately equals 0. Thus, we have shown that it is possible to determine n_{eig} for an initial state by measuring G on quantum hardware.

We tested the training step of the algorithm on IBM and Rigetti’s quantum computers, specifically *ibmq_toronto* and Aspen-8. For the purpose of implementing a complete simulation, we chose to focus on simulating the evolution of the state $|\psi_0\rangle = |10\rangle$. As discussed in the previous section, this state overlaps with $n_{\text{eig}} = 2$ eigenstates.

To diagonalize H_{XY} on the 2-dimensional subspace spanned by $|10\rangle$ and its future evolution, we used a hybrid quantum-classical optimization loop to minimize C_{fsVFF} . For a state with $n_{\text{eig}} = 2$ the cost C_{fsVFF} , Eq. (5), uses two training states $\{U(\Delta t)^k |\psi_0\rangle\}_{k=1,2}$ where $U(t)$ is the first-order Trotter approximation of H_{XY} . On the IBM quantum computer we evaluated the full cost function for each gradient descent iteration. However, the time available on the Aspen-8 device was limited, so to speed up the rate of optimization we evaluated the overlap on just one of the two training states per iteration, alternating between iterations (instead of evaluating the overlaps on both training states every iteration). To allow the movement through parameter space to use information averaged over the two timesteps, whilst only using a single training state per cost function evaluation, momentum was added to the gradient updates⁵¹.

To take advantage of the fact that more compact ansätze are viable for fsVFF, we variationally searched for a short depth ansatz, tailored to the target problem. Specifically, we started training

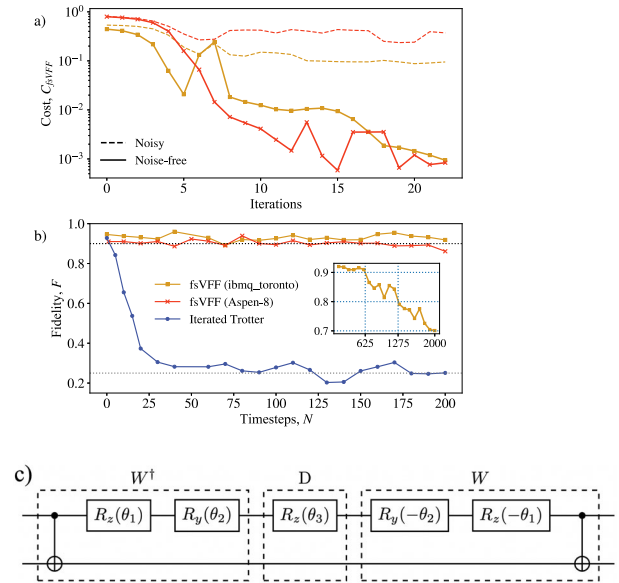


Fig. 3 Hardware Implementation. **a** The 2-qubit parameterized quantum circuit shown in **c** was trained to diagonalize $U(\Delta t)$, a first order Trotter expansion of the 2-qubit XY Hamiltonian with $\Delta t = 0.5$, in the subspace spanned by $|10\rangle$ and its future evolution. The dashed line plots the noisy cost as measured on *ibmq_toronto* (yellow) and Aspen-8 (red) using 30,000 samples per circuit. The solid line indicates the equivalent noise-free cost that was calculated on a classical simulator. **b** The initial state $|\psi_0\rangle = |10\rangle$ is evolved forwards in time on the *ibmq_rome* quantum computer using the iterated Trotter method (blue) and using fsVFF with the optimum parameters found on *ibmq_toronto* (yellow) and Aspen-8 (red). The quality of the simulation is evaluated by plotting the fidelity $F = \langle \psi | \rho | \psi \rangle$ between the evolved state and exact evolution. The gray dotted line at $F = 0.25$ represents the overlap with the maximally mixed state. The black dotted line denotes a threshold fidelity at $F = 0.9$. The inset shows the fast-forwarding of the ansatz trained on *ibmq_toronto* on a longer timescale, where the fidelity dropped below 0.9 (0.8) at 625 (1275) timesteps. All simulation data was taken using 8192 samples per circuit. **c** The ansatz used to diagonalize the 2-qubit XY Hamiltonian in the subspace of initial state $|10\rangle$ for the implementation on Rigetti and IBM’s quantum computers. Here $R_j(\theta) = \exp(-i\theta\sigma_j/2)$ for $j = x, y, z$.

with a general 2-qubit unitary and then during training the structure was minimized by pruning unnecessary gates. In Fig. 3c, we show the circuit for the optimal ansatz obtained using the method. The ansatz requires one CNOT gate and two single qubit gates for W and only one R_z rotation for D . This is a substantial compression on the most general two qubit ansatz for W which requires 3 CNOTs and 15 single qubit rotations and the most general 2 qubit ansatz for D which requires 2 R_z rotations and one 2-qubit ZZ rotation (though in the case of the XY Hamiltonian this may be simplified to only 2 R_z rotations⁵²).

Figure 3a shows the fsVFF cost function versus the number of iterations for the implementations on *ibmq_toronto* (yellow) and Aspen-8 (red). The dashed line indicates the noisy cost value obtained from the quantum computer. To evaluate the quality of the optimization, we additionally classically compute the true cost (indicated by the solid lines) using the parameters found on *ibmq_toronto* and Aspen-8. While the noisy cost saturates at around 10^{-1} , we obtained a minimum noise-free cost of the order 10^{-3} . The two orders of magnitude difference between the noisy and the noise-free cost is experimental evidence that the cost function is noise resilient on extant quantum hardware.

Finally we took the two sets of parameters found from training on *ibmq_toronto* and Aspen-8, and used them to implement a fast-forwarded simulation of the state $|10\rangle$ on *ibmq_rome*. To

evaluate the quality of the fast forwarding we calculated the fidelity, $F(N) = \langle \psi(N) | \rho(N) | \psi(N) \rangle$, between the density matrix of the simulated state, $\rho(N)$, after N timesteps, and the exact time evolved state, $|\psi(N)\rangle$, at time $T = N\Delta t$.

Quantum State Tomography⁵³ was used to reconstruct the density matrix. An n -dimensional density matrix ρ can be decomposed into the Pauli product basis as $\rho = \boldsymbol{\eta} \cdot \boldsymbol{\sigma}^{(n)}$. Here $\boldsymbol{\sigma}^{(n)}$ is a 4^n dimensional vector composed of the elements of the n -qubit Pauli group $P_n = \{\sigma_I, \sigma_X, \sigma_Y, \sigma_Z\}^{\otimes n}$ and $\boldsymbol{\eta}$ is the corresponding vector of Pauli weights, i.e., $\eta_k = \frac{1}{2^n} \text{Tr}(\sigma_k^{(n)} \rho)$. The values η_k were computed on the quantum device using 8192 shots, then used to classically calculate the fidelity $F(N)$. Since $\boldsymbol{\eta}$ is computed on the quantum hardware this naturally induces some additional noise; however, as single qubit rotations have a relatively high fidelity (average U3 error of qubits 0 & 1 on ibmq_bogota during usage was 4.60×10^{-4}) and read-out errors are small (average readout assignment error of qubits 0 & 1 on ibmq_bogota during usage was 4.60×10^{-2}) we expect this effect to be small compared to simulation errors. Moreover, these errors affect the fast-forwarded and iterated Trotter simulations equally.

As shown by the plots of $F(N)$ in Fig. 3b, fsVFF significantly outperforms the iterated Trotter method. Let us refer to the time before the simulation falls below an error threshold δ as the high fidelity time. Then the ratio of the high fidelity time for fsVFF (T_δ^{FF}) and for standard Trotterization (T_δ^{Trot}) is a convenient measure of simulation performance,

$$R_\delta^{\text{FF}} = T_\delta^{\text{FF}} / T_\delta^{\text{Trot}}. \quad (12)$$

A simulation can be said to have been successfully *fast-forwarded* if $R_\delta^{\text{FF}} > 1$. The iterated Trotter method dropped below a simulation infidelity threshold of $\delta = 1 - F = 0.1(0.2)$ after 4 (8) timesteps. In comparison, fsVFF maintained a high fidelity for 625 (1275) timesteps. Thus we achieved a simulation fast-forwarding ratio of $R_{0.1}^{\text{FF}} = 156$ ($R_{0.2}^{\text{FF}} = 159$).

Numerical simulations

We further validate fsVFF's performance by testing it on a simulator of a noisy quantum computer. The noise levels on the simulator are lower than those experienced on current devices and hence these results are indicative of the performance of the algorithm in the near future as hardware improves.

For these numerics we diagonalize the evolution of the 4 qubit XY Hamiltonian, in the subspace spanned by the domain wall state $|\psi_0\rangle = |1100\rangle$ and its future evolution. This space spans 5 energy eigenstates of the XY Hamiltonian and so we use the training states $\{U(\Delta t)^k |\psi_0\rangle\}_{k=1}^5$. Here $U(\Delta t)$ is chosen to be a second-order Trotter–Suzuki decomposition for the evolution operator under H_{XY} with $\Delta t = 0.5$. The noise model used was based upon the IBM architecture.

To construct the ansatz for the diagonalizing unitary, W , we developed an adaptive technique, similar to that proposed in refs. 13,54, to evolve the discrete circuit structure, as well as optimize the rotation parameters using gradient descent. This method tends to produce shallower circuits than the ones obtained with fixed ansatz approaches. It is also less prone to get stuck in local minima. Since, the XY Hamiltonian is particle number conserving we further use only particle number conserving gates. This reduces the number of parameters in W , as well as minimizing the leakage out of the symmetry sector when the circuit is executed with a noisy simulator. Additional details on this adaptive learning method are provided in Supplementary Note 6. The ansatz for D , as in our 2 qubit hardware implementation, simply consisted of R_z rotations on each qubit.

The result of the training is shown in the inset of Fig. 4. The noisy cost was measured by the noisy quantum simulator, whereas the noise-free cost is calculated simultaneously but in the absence of any noise. The significant separation between the

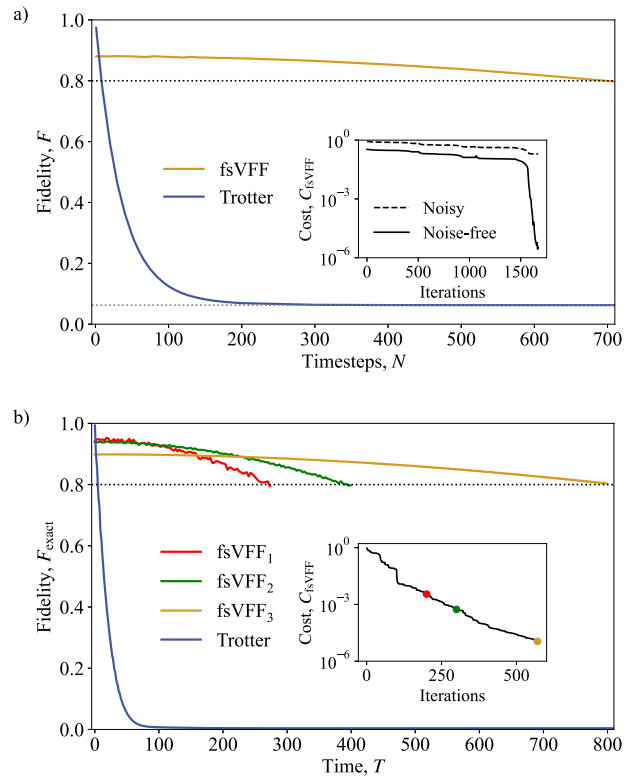


Fig. 4 Numerical Simulations. a Noisy Training and Fast-Forwarding of the 4 qubit XY Hamiltonian. The inset shows the cost curve as the ansatz is evolved and optimized to diagonalize the 4 qubit XY Hamiltonian in the 5-dimensional subspace spanned by initial state $|1100\rangle$ and its future evolution. The final circuit found by the learning algorithm for the diagonalizing unitary, W , had 50 CNOT gates. The main plot evaluates the fast-forwarding performance of the trained ansatz, in comparison to the iterated-Trotter evolved state. The fidelity is calculated against the ideal state found in simulation using the iterated Trotter method in the absence of noise, $F(N) = \langle \psi(N) | \rho | \psi(N) \rangle$ with $|\psi(N)\rangle = U(\Delta t)^N |\psi_0\rangle$. The black dotted line highlights a threshold value $F = 0.8$. The gray dotted line at $F = 1/2^4$ represents the overlap with the maximally mixed state. **b** Training and Fast-Forwarding of the 8 qubit Hubbard Model. The inset shows the cost as it is iteratively minimized using an adaptive ansatz. Various quality diagonalizations are indicated by the colored circles. In the main figure, we plot the fidelity between the simulated state and the exact evolution as a function of time. The red, green and yellow lines denote fsVFF simulations using the corresponding quality diagonalization shown in the inset. In blue we plot the fidelity of the iterated-Trotter simulation, $F_{\text{exact}}(T) = \langle \psi(T) | \rho_{\text{trot}} | \psi(T) \rangle$ with $|\psi(T)\rangle = e^{-iHT} |\psi_0\rangle$ and ρ_{trot} the simulated iterated Trotter state.

noisy and noise-free cost again demonstrates the noise resilience of the VFF algorithm. After successfully training the cost, the fast-forward performance was then evaluated. Using the same noise model, the output density matrix of the iterated-Trotter state and the fast-forwarded state was compared against the iterated-Trotter state in the absence of noise, with the fidelity between the two states plotted. As shown in Fig. 4, the fast-forwarded evolution significantly outperforms the iterated-Trotter evolution, with the former's fidelity dropping below 0.8 after 700 timesteps, compared to only 8 steps of the latter. Thus we achieved a fast forwarding ratio of $R_{0.2}^{\text{FF}} = 87.5$.

Finally, to probe the scalability and the breadth of applicability of the fsVFF algorithm we performed a larger (noiseless) numerical implementation of the algorithm on the Fermi–Hubbard model. Specifically, we considered the 1D Fermi–Hubbard Hamiltonian on

an L -site lattice with open boundary conditions:

$$H_{\text{FH}} = -J \sum_{j=1}^{L-1} \sum_{\sigma=\uparrow,\downarrow} c_{j,\sigma}^\dagger c_{j+1,\sigma} + \text{h.c.} + U \sum_{j=1}^L n_{j,\uparrow} n_{j,\downarrow}. \quad (13)$$

Here, $c_{j,\sigma}$ ($c_{j,\sigma}^\dagger$) denotes fermionic creation (annihilation) operator at site j for each of the two spin states $\sigma = \uparrow, \downarrow$ and $n_{j,\sigma} = c_{j,\sigma}^\dagger c_{j,\sigma}$ is a particle number operator. The total number of fermions with a spin σ is given by $N_\sigma = \sum_j n_{j,\sigma}$. The term with coefficient J in Eq. (13) represents a single-fermion nearest-neighbor hopping and the term with coefficient U introduces on-site repulsion. The Hamiltonian preserves particle numbers N_\uparrow and N_\downarrow .

In our numerical studies, we choose $L=4$ (which requires 8 qubits to simulate) and $J=1$, $U=2$ as well as $N_\uparrow = N_\downarrow = 2$ (half filling). The initial state is chosen to be a superposition of $n_{\text{eig}} = 5$ eigenvectors of H_{FH} in the particle sector $N_\uparrow = N_\downarrow = 2$. Similar to our noisy simulations of the XY model, we utilize an adaptive ansatz for W that is made out of gates that preserve particle number N_\uparrow and N_\downarrow . The ansatz for D takes the form of Eq. (16), where we only allow for single- Z terms in Eq. (17). For this numerical result, we trained using the full exponentiation of the Hamiltonian as the evolution operator, with no Trotter error.

In the inset of Fig. 4b we show the cost function as it is iteratively minimized. We then test the performance on a noisy simulator based upon a fully connected 8-qubit trapped-ion device²³. As shown in Fig. 4b, small final cost values typically require deeper circuits to achieve, the optimum diagonalization to use depends on the length of time one wishes to simulate. At short times, a larger final cost function value performs better since this corresponds to a shorter ansatz which experiences less noise. However, to simulate longer times, a higher quality diagonalization is required, with the additional noise induced by increased circuit depth resulting in a relatively small decrease in fidelity. As shown in Fig. 4b, we find that the fast forwarding corresponding to an optimized cost of 1.1×10^{-5} maintained a fidelity of greater than 0.8 for $T < 800$. In contrast, the iterated Trotter method drops below 0.8 for $T > 4.6$ and hence we here achieve a fast forwarding ratio of $R_{0.2}^{\text{FF}} = 174$.

To demonstrate the viability of the batched training method outlined in Section “Randomized Training”, we diagonalized the 5

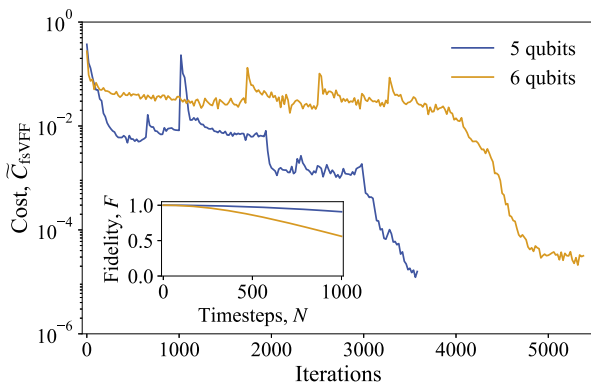


Fig. 5 Randomized Training. The 5 (6) qubit XY Hamiltonian with initial state $|11100\rangle(|111000\rangle)$ is diagonalized using the cost function Eq. (10), using only 2 training states per cost function evaluation to learn the evolution within the 9 (12) dimensional subspace. The final circuit found by the learning algorithm for the diagonalizing unitary, W , had 32 (134) CNOT gates. After completion of the randomized training, the Hamiltonians were fast-forwarded, with the fidelity evaluated in comparison to the noiseless Trotter-iterated state $U(\Delta t)^N |\psi_0\rangle$ as shown in the inset.

(6) qubit XY Hamiltonian with initial state $|11100\rangle(|111000\rangle)$, which has an $n_{\text{eig}} = 9(12)$. For both training curves shown in Fig. 5, only 2 training states per cost function evaluation were used. In both cases, we trained with the unitary $U(t/6)^6$ where U was the second order Trotter–Suzuki operator, and $t_{\text{max}} = 1$. The cost was successfully minimized to 10^{-5} in both cases, and a noiseless simulation error of less than 10^{-2} was maintained for over 100 time steps on fast forwarding.

Compared to its predecessor VFF, the fsVFF algorithm halves the width of circuits required to diagonalize an n -qubit Trotterized unitary from $2n$ to n qubits. Additionally, fsVFF does not need to generate the (potentially highly noise inducing) $2n$ Bell pairs required by VFF. In particular, on a device with limited connectivity this would require $2n$ non-local CNOT gates and so potentially a large number of (hard-to-implement) SWAP gates. These circuit simplifications all contribute to making the fast-forwarding of Hamiltonians more feasible on near-term quantum hardware.

Another less obvious but equally important advantage of fsVFF over VFF is that it can utilize shallower diagonalizing circuits. Since in fsVFF we only need to learn to reproduce the target unitary on the n_{eig} dimension subspace explored by the initial state, there is a larger set of possible diagonalizations that can be found. This is in contrast to VFF which must reproduce the target on the full 2^n dimensional Hilbert space. Thus the set of solution unitaries for fsVFF is much larger than VFF, greatly increasing the probability that a good solution can be found using a short depth ansatz. Shorter depth ansätze are both less resource intensive to implement and easier to optimize. For example, if using a gradient-based method, the number of cost evaluations to evaluate the gradient scales with the number of parameters (and so the depth) of the ansatz.

To demonstrate this advantage of fsVFF, we compare the performance of both fsVFF and VFF to fast-forward the same Hamiltonian, $H = \sum_{i=1}^4 X_i X_{i+1} + Y_i Y_{i+1} + \frac{1}{2} Z_i Z_{i+1}$. For fsVFF the initial state is chosen to be $|\psi_0\rangle = |10000\rangle$. Both algorithms use the same ansatz design with hardware-efficient layers composed with particle number conserving gates. As shown in Fig. 6, with an 8 layer ansatz, fsVFF during training was able to achieve a $C_{\text{fsVFF}} \approx 10^{-6}$, which is shown in the inset of Fig. 6 to translate into a successful fast forwarding of the initial state. In contrast, when

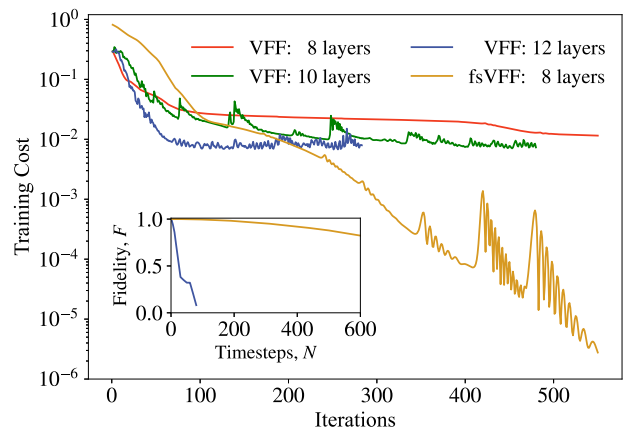


Fig. 6 fsVFF versus VFF training and fast forwarding. The 5-qubit XXZ Hamiltonian $H = \sum_{i=1}^4 X_i X_{i+1} + Y_i Y_{i+1} + \frac{1}{2} Z_i Z_{i+1}$ is diagonalized using both fsVFF and VFF. Both algorithms use the same hardware efficient layered ansatz and optimizer. For fsVFF the initial state is $|\psi_0\rangle = |10000\rangle$. The cost function as it is iteratively minimized using fsVFF/VFF with various depth ansätze as indicated in the legend. The inset shows the fast forwarded fidelity for the best diagonalizations learnt by fsVFF and VFF respectively. Here, fsVFF was restricted to only 8 layers and yet still dramatically outperforms VFF, even when VFF is allowed a deeper circuit of 12 layers.

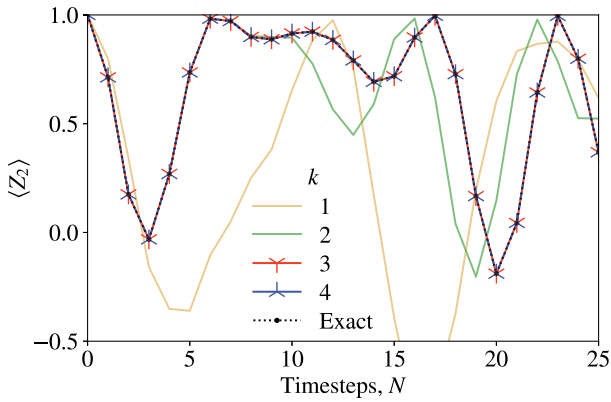


Fig. 7 Observable Convergence for Increasing Training States. The 5 qubit XY Hamiltonian is trained on an increasing number of training states, for the initial state $|\psi_0\rangle = |10000\rangle$. For each instance, C_{fsVFF} was trained down to 10^{-10} on k training states, then the fsVFF ansätze $V(N\Delta t)_k = WD(N\gamma)W^\dagger$ were used to fast-forward the initial state. For the Pauli operator Z_2 , we plot the expectation value of each fast-forwarded state. For $k \geq 3$ the expectation exactly reproduces the value produced by iterated Trotter evolved state, shown by the black dotted line.

using the same optimizer and layered ansatz, even after increasing the depth of W to 12 layers, VFF is unable to find a successful diagonalization, with the cost C_{VFF} plateauing around 10^{-2} . As shown in the inset of Fig. 6, the corresponding diagonalization was not able to simulate the Trotter evolution for any length of time.

When performing fsVFF on too few training states, the cost function C_{fsVFF} may decrease arbitrarily small whilst not generalizing outside of this training region. As the number of training states is increased, a transition will occur at the point of sufficient training states, where expectation values calculated with respect to the fast-forwarded state converge to the exact value. The Quantum No Free Lunch theorem^{28,29} implies that this transition will occur at $k = n_{\text{eig}}$ training states. This convergence provides a method of implementing fsVFF without needing to pre-compute n_{eig} .

This behavior is demonstrated in Fig. 7. Here we train on an increasing number of training states, $\{U(\Delta t)^j|\psi_0\rangle\}_{j=1}^k$, for $k = 1, 2, 3, 4$. For each of the 4 instances we trained down to $C_{\text{fsVFF}} = 10^{-10}$ and use the saved ansätze to compute the expectation of the Pauli operator Z_2 as a function of time, which we compare to the true value produced with the iterated Trotter method. For an insufficient number of training states C_{fsVFF} can be made arbitrarily small but this does not translate to an accurate simulation outside the training region. The number of training states needed can be inferred by noting the value of k such that increasing k ceases to substantially change the fast forwarded expectation values.

In this case, as shown in Fig. 7, we need $k \geq 3$. Given that here the initial state is $|\psi_0\rangle = |10000\rangle$ corresponding to $n_{\text{eig}} = 5$, this is fewer states than predicted by the Quantum No-Free-Lunch theorem. This discrepancy can perhaps be explained by the fact that the overlap of $|\psi_0\rangle$ with two of its eigenstates is small. Specifically, we have $|\psi_0\rangle = \sum_{i=1}^5 \alpha_i |E_i\rangle$ with $\{|\alpha_i|^2\}_{i=1}^5 = \{0.33, 0.25, 0.25, 0.083, 0.083\}$ such that the combined amplitudes of two of the eigenvectors account for only $1/6^{\text{th}}$ of the total amplitude weight. Thus, these results suggest that the number of training states required can be less than the number of eigenvectors with a non-zero overlap with the initial state, as long as these amplitudes are sufficiently small.

Energy estimation

The diagonalization obtained from the optimization stage of fsVFF, $W(\theta_{\text{opt}})D(\mathbf{v}_{\text{opt}}, \Delta t)W(\theta_{\text{opt}})^\dagger$, implicitly contains approximations of the eigenvalues and eigenvectors of the Hamiltonian of the system of interest. In this section we discuss methods for extracting the energy eigenstates and eigenvalues from a successful diagonalization and implement them on quantum hardware.

The energy eigenvectors that the initial state $|\psi_0\rangle$ overlaps can be determined by the following simple sampling method. The first step is to apply W^\dagger to the initial state $|\psi_0\rangle$. In the limit of perfect learning and vanishing Trotter error, this gives

$$W(\theta_{\text{opt}})^\dagger|\psi_0\rangle = \sum_{k=1}^{n_{\text{eig}}} a_k |v_k\rangle \quad (14)$$

where $a_k = \langle E_k|\psi_0\rangle$ and $\{|v_k\rangle\}_{k=1}^{n_{\text{eig}}}$ is a set of computational basis states. The energy eigenstates that the initial state $|\psi_0\rangle$ overlaps are then found by applying $W(\theta_{\text{opt}})$ to any of the states obtained from measuring $W(\theta_{\text{opt}})^\dagger|\psi_0\rangle$ in the computational basis, that is $\{|E_k\rangle\}_{k=1}^{n_{\text{eig}}} = \{W(\theta_{\text{opt}})|v_k\rangle\}_{k=1}^{n_{\text{eig}}}$.

Extracting the energy eigenvalues from D is more subtle. Firstly, as WDW^\dagger and U , even in the limit of perfectly minimizing the cost C_{fsVFF} , may disagree by a global phase ϕ , at best we can hope to learn the difference between, rather than absolute values, of the energy eigenvalues of H . For simple cases, where the diagonal ansatz D is composed of a polynomial number of terms, these energy value differences may be extracted directly by rewriting D in the computational basis. For example, in our hardware implementation $D(\gamma) = \exp(-i\frac{\gamma\Delta z_1}{2}) \otimes \mathbb{1}$ and therefore the difference in energy between the two eigenvalues that the state $|\psi_0\rangle$ has non-zero overlap with is given by $\gamma_{\text{opt}} + \frac{k\pi}{\Delta t}$. Here k is an integer correcting for the arbitrary phase arising from taking the log of D that can be determined using the method described in ref. 22. Using this approach, we obtain 1.9995 and 2.0019 from the training on IBM and Rigetti respectively, in good agreement with the theoretically expected value of 2. For more complex cases, this simple post-processing method will become intractable and an algorithmic approach will be necessary.

Quantum Phase Estimation (QPE)⁵³ and Quantum Eigenvalue Estimation (QEE)⁵⁵ are fault tolerant quantum algorithms for estimating the eigenvalues of a unitary operation. However, their implementation on current quantum devices is limited by the reliance on the execution of controlled unitaries from ancillary qubits. These controlled unitaries require many entangling gates, and introduce too much noise to be realized for large scale systems on current hardware. Once an evolution operator has been diagonalized in the subspace of an initial state, fsVFF can be used to significantly reduce the circuit depth of QPE and QEE, as shown in Fig. 8. In this manner, fsVFF provides a NISQ friendly means of estimating the eigenvalues within a subspace of a Hamiltonian.

To demonstrate the power of fsVFF to reduce the depth of QPE, we perform QPE using the diagonalization obtained from training on IBM's quantum computer. Specifically, we consider the input eigenvector $|E_1\rangle := \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. This is one of the eigenvectors overlapped by the input state of our earlier hardware implementation, $|\psi_0\rangle = |10\rangle$. We then consider evolving $|E_1\rangle$ under H_{XY} for a time step of $\Delta t = 1/8$. Since the energy of the state $|E_1\rangle$ equals 1, we expect this to result in a phase shift of $e^{2\pi i/8}$ being applied to $|E_1\rangle$. We implemented QPE and fsVFF-enhanced QPE to measure this phase using the circuits shown in Fig. 8. We chose to measure to 3 bits of precision and therefore the output should be the measurement 001 with probability one. As Fig. 9 shows, it appears that the standard QPE implementation was unable to discern this phase. In contrast, when fsVFF was used to reduce the circuit depth, the output distribution was strongly peaked at the correct state.

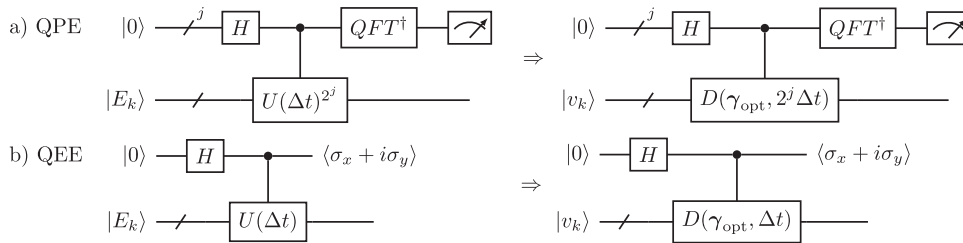


Fig. 8 Energy estimation circuits. **a** and **b** show circuit diagrams depicting the enhancement of QPE/QEE using fsVFF. A circuit depth reduction is achieved through replacing $U(\Delta t)$ with $D(\gamma_{\text{opt}}, \Delta t)$, and removing the need to prepare an eigenstate in favor of a computational basis state, $|v_k\rangle = W^\dagger|E_k\rangle$. QPE relies on implementing controlled unitaries of the form $U(\Delta t)^{2^j}$ and therefore replacing these with $D(\gamma_{\text{opt}}, 2^j \Delta t)$ results in an exponential reduction in circuit depth.

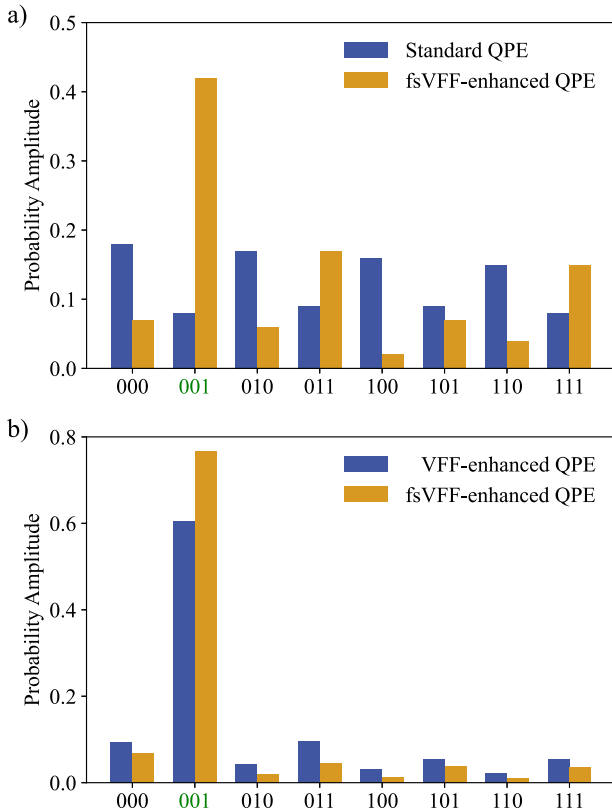


Fig. 9 Quantum Phase Estimation. **a** Hardware Implementation. Using the 2-qubit diagonalization found from training on `ibmq_toronto`, QPE was performed on `ibmq_boeblingen` on the eigenvector $|E_1\rangle := \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. A phase of $e^{\frac{2\pi i}{8}}$ is applied, so the measured output should be 001 with probability 1. The variation distance from the target probability distribution when using fsVFF-enhanced QPE was 0.578, compared to 0.917 using standard QPE. **b** Noisy Simulation comparing fsVFF and VFF. A noisy simulation of both VFF-enhanced and fsVFF-enhanced QPE is performed and compared. The same eigenvector and phase is applied as in **a**). The variation distance from the target probability distribution when using fsVFF-enhanced QPE was 0.233, compared to 0.394 using VFF-enhanced QPE.

VFF can also be used to enhance QPE. In Fig. 9 we compare the effect of noise on fsVFF and VFF by implementing fsVFF and VFF enhanced QPE. In both cases the optimization was performed using the Qiskit "FakeVigo" noise model. During the training routine, the lowest noise-free cost achieved by fsVFF and VFF respectively were 3.56×10^{-5} and 3.63×10^{-6} . For fsVFF, the ansatz used was the same as in Fig. 3c; for VFF, the ansatz used for W was a single Givens rotation, and D had an Rz gate on each

qubit. As shown in Fig. 9b, both algorithms were able to find the correct phase applied with high probability. However, even though VFF was able to achieve a lower noise free cost compared to fsVFF, when QPE was performed fsVFF achieved a 40.1% lower variation distance from the target probability distribution. Here, the ansatz for W requires one CNOT for fsVFF compared to two CNOTs in the VFF ansatz, however in the QPE execution we do not need to apply W so we do not get a benefit from this circuit reduction. The more significant effect here comes from the simpler D ansatz, requiring only one Rz gate for fsVFF compared to two Rz gate for VFF.

Quantum eigenvalue estimation (QEE) requires only one ancillary qubit, a single implementation of e^{-iHt} , and no Quantum Fourier Transform and therefore is less resource intensive than QPE. Nonetheless, we can again, as shown in Fig. 8, use fsVFF as a pre-processing step to reduce the circuit depth.

We tested this on the 3-qubit XY Hamiltonian by first performing fsVFF on a quantum simulator with the initial state $|\psi_0\rangle = |110\rangle$. Having obtained an approximate diagonalization, we determined the eigenstates using the sampling method described earlier. Figure 10a shows the results of the measurement of $W(\theta_{\text{opt}})^\dagger|\psi_0\rangle$, with four strong peaks corresponding to the four eigenvectors in this subspace.

Figure 10 shows the results of QEE implemented on `ibmq_boeblingen`. We use the basis states found from the sampling method as our inputs to reduce the depth of the circuit, and remove the need to use the time-series method originally proposed for extracting the eigenvalues, as we could calculate the eigenvalues individually by inputting their corresponding eigenvectors. A value of $\Delta t = 1$ was used so the phase calculated directly matched the eigenvalue. After removing a global phase, QEE had accurately found the eigenvalues of the four eigenvectors, with a mean-squared error from the true values of 4.37×10^{-3} .

DISCUSSION

In this work, we demonstrated that despite the modest size and noise levels of the quantum hardware that is currently available, it is possible to perform long time dynamical simulations with a high fidelity. Specifically, we have introduced fsVFF, an algorithm for NISQ simulations, which we used to simulate a 2-qubit XY-model spin chain on the Rigetti and IBM quantum computers. We achieved a fidelity of at least 0.9 for over 600 time steps. This is a 150-fold improvement on the standard iterated Trotter approach, which had a fidelity of less than 0.9 after only 4 time steps. Moreover, our numerical simulations of the 4 qubit XY model and 8 qubit Fermi–Hubbard model achieved fast-forwarding ratios of 87.5 and 174 respectively, indicating the viability of larger implementations in the near future as hardware improves.

Central to the success of the fsVFF algorithm is the fact that it is tailored to simulating a particular fixed initial state rather than an arbitrary initial state. By sacrificing generality and focusing on this less demanding task, we showed that it is possible to substantially

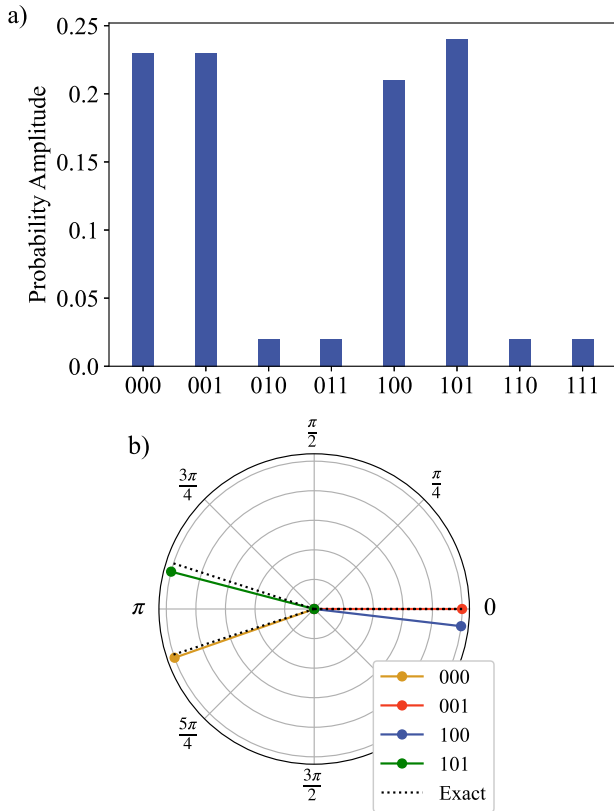


Fig. 10 Eigenvalue Estimation. **a** Determining the eigenstates overlapped by the initial state: The 3-qubit XY Hamiltonian was diagonalized on a quantum simulator in the subspace of initial state $|\psi_0\rangle = |110\rangle$ to obtain θ_{opt} and \mathbf{y}_{opt} . Here we show the output of measuring $W(\theta_{\text{opt}})^\dagger |\psi_0\rangle$ in the computational basis on `ibmq_boeblingen`. The 4 non-zero states correspond to the 4 eigenvectors overlapped by $|\psi_0\rangle$. **b** Eigenvalue Estimation using QEE. Here we show the result of implementing QEE (using fsVFF as a pre-processing step) on `ibmq_santiago` to calculate the eigenvalues of the eigenvectors in the subspace overlapped by $|011\rangle$. The solid yellow, red, blue and green lines represent the eigenvalues obtained for the $|000\rangle$, $|001\rangle$, $|100\rangle$ and $|101\rangle$ states, with exact corresponding energies of $\{-2.828, 0, 0, 2.828\}$, indicated by the dotted lines. The eigenvalues are plotted as phases since for $\Delta t = 1$ there is a one to one correspondence.

reduce the algorithmic resources as compared to the previously proposed VFF algorithm. In particular, fsVFF only requires finding a diagonalization of a short-time evolution unitary on the subspace spanned by the eigenstates of the Hamiltonian with non-zero overlap with the initial state. When this subspace is much smaller than the total Hilbert space, fsVFF can utilize much simpler ansätze than VFF, which requires finding a diagonalization over the entire Hilbert space. This is demonstrated in Fig. 6 where an 8 layered number-preserving hardware efficient ansatz proved sufficient to diagonalize the XXZ Hamiltonian for the initial state $|10000\rangle$ but 12 layers was insufficient to find a full diagonalization using VFF. Additionally, the fsVFF algorithm requires only n qubits rather than the $2n$ qubits needed by VFF, thus opening up larger implementations on smaller devices.

The complexity of the fsVFF algorithm is determined by n_{eig} , the dimension of the subspace the diagonalization needs to be learnt on. This determines the amount of training data and depth of circuits required for training, as well as the complexity of the diagonalization ansätze. However, we stress that this limits the generality of the algorithm (i.e., the set of initial states it can be used to simulate) but not its scalability (the size of system it can be applied to). Moreover, to mitigate these limitations of the fsVFF

algorithm, one could investigate using error mitigation methods to manage the added noise associated with larger n_{eig} . It would also be worth developing problem inspired ansätze, utilizing the symmetry properties of the target system, to reduce the scaling of the complexity of the ansatz.

The fsVFF algorithm, similarly to VFF, is fundamentally limited by the initial Trotter error approximating the short time evolution of the system. The Variational Diagonalization Hamiltonian (VHD) algorithm²² may be used to remove this error. However, like VFF, VHD is designed to simulate any possible initial state. There are a number of different approaches inspired by fsVFF that could be explored for reducing the resource requirements of the VHD algorithm by focusing on simulating a particular initial state. Such a "fixed state VHD" algorithm would allow for more accurate long time simulations on NISQ hardware.

More generally, our work highlights the trade off between the universality of an algorithm and the resources required to implement it. One can imagine a number of alternative ways in which the universality of an algorithm can be sacrificed, without significantly reducing its utility, in order to make it more NISQ friendly. For example, one is often interested in studying the evolution of a particular observable of interest, rather than all possible observables. It would be interesting to investigate whether a fixed-observable fsVFF could further reduce the resources required to implement long time high fidelity simulations. More broadly, an awareness of this trade off may prove useful beyond dynamical simulation for the ongoing challenge of adapting quantum algorithms to the constraints of NISQ hardware.

METHODS

Ansatz

The fsVFF algorithm, similarly to VFF, employs an ansatz of the form

$$V(\mathbf{a}, \Delta t) = W(\theta)D(\mathbf{y}, \Delta t)W^\dagger(\theta), \quad (15)$$

to diagonalize the initial Trotter unitary $U(\Delta t)$. Here $W(\theta)$ is a quantum circuit that approximately rotates the standard basis into the eigenbasis of H , and $D(\mathbf{y})$ is a diagonal unitary that captures the (exponentiated) eigenvalues of H . A generic diagonal operator D can be written in the form

$$D(\mathbf{y}, \Delta t) = \prod_{\mathbf{q}} e^{i\mathbf{y}_{\mathbf{q}}Z^{\mathbf{q}}\Delta t}, \quad (16)$$

where $\mathbf{y}_{\mathbf{q}} \in \mathbb{R}$ and we use the notation

$$Z^{\mathbf{q}} = Z_1^{q_1} \otimes \dots \otimes Z_n^{q_n}, \quad (17)$$

with Z_j the Pauli Z operator acting on qubit j . While Eq. (16) provides a general expression for a diagonal unitary, for practical ansätze it may be desirable to assume that the $Z^{\mathbf{q}}$ operators are local operators and the product contains a polynomial number of terms, i.e., is in $\mathcal{O}(\text{poly}(n))$. There is more flexibility in the construction of the ansätze for W since these are generic unitary operations. A natural choice might be to use a hardware-efficient ansatz⁵⁶ or an adaptive ansatz^{13,54}.

One of the main advantages of fsVFF is that diagonalization is only necessary over the subspace spanned by the initial state and its future evolution, rather than the entire Hilbert space which will be significantly larger. To outperform standard VFF, it is in our interest to take advantage of this small subspace to find compact ansätze.

The two main impeding factors we wish to minimize to aid diagonalization are error rates and optimization time. Therefore, when searching for ansätze, our priorities are to minimize the number of CNOT gates required (the noisiest component in the ansätze) and the number of rotation parameters. There is, however, a trade off between expressibility of the ansatz and its

trainability. There needs to be enough freedom in the unitary to map the required eigenvectors to the computational basis but generically highly expressive ansätze exhibit barren plateaus³³.

For systems with symmetries and/or systems that are nearby perturbations of known diagonalizable systems, it may be possible to find a fully expressive, compact ansatz by inspection. This is the case for a simple 2-qubit XY Hamiltonian, as discussed in Section “Hardware implementation”.

More generally, it can be challenging to analytically find compact but sufficiently expressible ansätze. Nonetheless, it is possible to variationally update the ansatz structure and thereby systematically discover simple structures. One straightforward approach is to use a layered ansatz where each layer initializes to the identity gate^{57,58}. The ansatz can be optimized until it plateaus, redundant single qubit gates removed, then another layer can be appended and the process repeats. Alternatively, more sophisticated discrete optimization techniques may be used to variationally search the space of ansätze.

Gradient formulae

Gradient-based optimizers were used to produce the numerical results, as these have been shown to improve the convergence of variational quantum algorithms⁵⁹, and the gradient can be evaluated using the same depth circuit as the cost function via the parameter shift rule⁶⁰.

The fsVFF ansatz is defined in Eq. (1), and for convenience we define $C_{\text{fsVFF}} := C_{\text{fsVFF}}(U, WDW^\dagger)$. The parameter shifted operator $V_{\pm} (V_{I-})$ is generated from the original operator $V(\theta)$ by the addition of an extra $\frac{\pi}{2} (-\frac{\pi}{2})$ rotation about a given parameter's rotation axis:

$$V_{I\pm} := V(\theta_{I\pm}) \text{ with } (\theta_{I\pm})_i := (\theta_i)_{\pm} \pm \frac{\pi}{2} \delta_{i,I}. \quad (18)$$

The partial derivative of C_{fsVFF} with respect to parameter θ_i in the diagonalizing unitary $W(\theta)$ is given by

$$\begin{aligned} \frac{\partial C_{\text{fsVFF}}}{\partial \theta_i} = & \frac{1}{2} (C_{\text{fsVFF}}(U, W_{I+} DW^\dagger) \\ & - C_{\text{fsVFF}}(U, W_{I-} DW^\dagger) \\ & + C_{\text{fsVFF}}(U, WD(W_{I+})^\dagger) \\ & - C_{\text{fsVFF}}(U, WD(W_{I-})^\dagger)) \end{aligned} \quad (19)$$

The partial derivative of C_{fsVFF} with respect to parameter γ_j in the diagonal unitary $D(\gamma)$ is given by

$$\begin{aligned} \frac{\partial C_{\text{fsVFF}}}{\partial \gamma_j} = & \frac{1}{n_{\text{eig}}} \sum_{k=1}^{n_{\text{eig}}} \frac{k}{2} (C_{\text{fsVFF}}^{(k)}(U, WD_{I+} W^{\dagger}) \\ & - C_{\text{fsVFF}}^{(k)}(U, WD_{I-} W^{\dagger})) \end{aligned} \quad (20)$$

where

$$C_{\text{fsVFF}}^{(k)} := 1 - |\langle \psi_0 | WD^k W^{\dagger} U^k | \psi_0 \rangle|^2 \quad (21)$$

The derivation for these formulae is provided in Supplementary material.

DATA AVAILABILITY

The authors declare that the main data supporting the findings of this study are available within the article and its Supplementary Information files. Extra data sets are available upon reasonable request.

CODE AVAILABILITY

The code supporting the findings of this study are available upon reasonable request.

Received: 14 July 2021; Accepted: 23 August 2022;

Published online: 19 November 2022

REFERENCES

- Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
- Arute, F. et al. Observation of separated dynamics of charge and spin in the fermi-hubbard model. Preprint at <https://arxiv.org/abs/2010.07965> (2020).
- Lloyd, S. Universal quantum simulators. *Science* 1073–1078, <https://science.sciencemag.org/content/273/5278/1073> (1996).
- Sornborger, A. & Stewart, E. D. Higher-order methods for simulations on quantum computers. *Phys. Rev. A* **60**, 1956 (1999).
- Low, G. H. & Chuang, I. L. Hamiltonian simulation by qubitization. *Quantum* **3**, 163 (2019).
- Berry, D. W., Childs, A. M., Cleve, R., Kothari, R. & Somma, R. D. Simulating hamiltonian dynamics with a truncated taylor series. *Phys. Rev. Lett.* **114**, 090502 (2015).
- Cerezo, M. et al. Variational quantum algorithms. *Nat. Rev. Phys.* **3**, 625–644 (2021).
- Endo, S., Cai, Z., Benjamin, S. C. & Yuan, X. Hybrid quantum-classical algorithms and quantum error mitigation. *J. Phys. Soc. Jpn* **90**, 032001 (2021).
- Bharti, K. et al. Noisy intermediate-scale quantum algorithms. *Rev. Modern Phys.* **94**, 015004 (2022).
- Peruzzo, A. et al. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **5**, 1–7 (2014).
- Farhi, E., Goldstone, J. & Gutmann, S. A quantum approximate optimization algorithm. Preprint at <https://arxiv.org/abs/1411.4028> (2014).
- McClellan, J. R., Romero, J., Babbush, R. & Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *N. J. Phys.* **18**, 023023 (2016).
- Khatri, S. et al. Quantum-assisted quantum compiling. *Quantum* **3**, 140 (2019).
- LaRose, R., Tilkku, A., O'Neel-Judy, É., Cincio, L. & Coles, P. J. Variational quantum state diagonalization. *npj Quantum Inf.* **5**, 1–10 (2019).
- Arrasmith, A., Cincio, L., Sornborger, A. T., Zurek, W. H. & Coles, P. J. Variational consistent histories as a hybrid algorithm for quantum foundations. *Nat. Commun.* **10**, 1–7 (2019).
- Cerezo, M., Poremba, A., Cincio, L. & Coles, P. J. Variational quantum fidelity estimation. *Quantum* **4**, 248 (2020).
- Li, Y. & Benjamin, S. C. Efficient variational quantum simulator incorporating active error minimization. *Phys. Rev. X* **7**, 021050 (2017).
- Endo, S., Sun, J., Li, Y., Benjamin, S. C. & Yuan, X. Variational quantum simulation of general processes. *Phys. Rev. Lett.* **125**, 010501 (2020).
- Yao, Y.-X. et al. Adaptive variational quantum dynamics simulations. Preprint at <https://arxiv.org/abs/2011.00622> (2020).
- Heya, K., Nakanishi, K. M., Mitarai, K. & Fujii, K. Subspace variational quantum simulator. Preprint at <https://arxiv.org/abs/1904.08566> (2019).
- Cirstoiu, C. et al. Variational fast forwarding for quantum simulation beyond the coherence time. *npj Quantum Inf.* **6**, 1–10 (2020).
- Commeau, B. et al. Variational hamiltonian diagonalization for dynamical quantum simulation. Preprint at <https://arxiv.org/abs/2009.02559> (2020).
- Trout, C. J. et al. Simulating the performance of a distance-3 surface code in a linear ion trap. *N. J. Phys.* **20**, 043038 (2018).
- Benedetti, M., Fiorentini, M. & Lubasch, M. Hardware-efficient variational quantum algorithms for time evolution. Preprint at <https://arxiv.org/abs/2009.12361> (2020).
- Bharti, K. & Haug, T. Quantum-assisted simulator. *Phys. Rev. A* **104**, 042418 (2021).
- Lau, J. W. Z., Bharti, K., Haug, T. & Kwek, L. C. Quantum assisted simulation of time dependent hamiltonians. Preprint at <https://arxiv.org/abs/2101.07677> (2021).
- Haug, T. & Bharti, K. Generalized quantum assisted simulator. Preprint at <https://arxiv.org/abs/2011.14737> (2020).
- Poland, K., Beer, K. & Osborne, T. J. No free lunch for quantum machine learning. Preprint at <https://arxiv.org/abs/2003.14103> (2020).
- Sharma, K. et al. Reformulation of the no-free-lunch theorem for entangled data sets. Preprint at <https://arxiv.org/abs/2007.04900> (2020).
- Sharma, K., Khatri, S., Cerezo, M. & Coles, P. J. Noise resilience of variational quantum compiling. *N. J. Phys.* **22**, 043006 (2020).
- Sim, S., Johnson, P. D. & Aspuru-Guzik, A. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. *Adv. Quantum Technol.* **2**, 1900070 (2019).
- Nakaji, K. & Yamamoto, N. Expressibility of the alternating layered ansatz for quantum computation. *Quantum* **5**, 434 (2021).
- Holmes, Z., Sharma, K., Cerezo, M. & Coles, P. J. Connecting ansatz expressibility to gradient magnitudes and barren plateaus. *PRX Quantum* **3**, 010313 (2022).
- Sahinoglu, B. & Somma, R. D. Hamiltonian simulation in the low energy subspace. Preprint at <https://arxiv.org/abs/2006.02660> (2020).
- Suzuki, M. Generalized trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems. *Commun. Math. Phys.* **51**, 183–190 (1976).

36. Cerezo, M., Sone, A., Volkoff, T., Cincio, L. & Coles, P. J. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.* **12**, 1–12 (2021).
37. McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R. & Neven, H. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* **9**, 1–6 (2018).
38. Cerezo, M. & Coles, P. J. Higher order derivatives of quantum neural networks with barren plateaus. *Quantum Sci. Technol.* **6**, 035006 (2021).
39. Arrasmith, A., Cerezo, M., Czarnik, P., Cincio, L. & Coles, P. J. Effect of barren plateaus on gradient-free optimization. *Quantum* **5**, 558 (2021).
40. Holmes, Z. et al. Barren plateaus preclude learning scramblers. *Phys. Rev. Lett.* **126**, 190501 (2021).
41. Volkoff, T. & Coles, P. J. Large gradients via correlation in random parameterized quantum circuits. *Q. Sci. Technol.* **6**, 025008 (2021).
42. Sharma, K., Cerezo, M., Cincio, L. & Coles, P. J. Trainability of dissipative perceptron-based quantum neural networks. Preprint at <https://arxiv.org/abs/2005.12458> (2020).
43. Pesah, A. et al. Absence of barren plateaus in quantum convolutional neural networks. *Phys. Rev. X* **11**, 041011 (2021).
44. Uvarov, A. & Biamonte, J. D. On barren plateaus and cost function locality in variational quantum algorithms. *J. Phys. A Math. Theor.* **54**, 245301 (2021).
45. Marrero, C. O., Kieferova, M. & Wiebe, N. Entanglement induced barren plateaus. Preprint at <https://arxiv.org/abs/2010.15968> (2020).
46. Patti, T. L., Najafi, K., Gao, X. & Yelin, S. F. Entanglement devised barren plateau mitigation. *Phys. Rev. Res.* **3**, 033090 (2021).
47. Krylov, A. On the numerical solution of equation by which are determined in technical problems the frequencies of small vibrations of material systems. *News Acad. Sci. USSR* **7**, 491–539 (1931).
48. Kübler, J. M., Arrasmith, A., Cincio, L. & Coles, P. J. An adaptive optimizer for measurement-frugal variational algorithms. *Quantum* **4**, 263 (2020).
49. Arrasmith, A., Cincio, L., Somma, R. D. & Coles, P. J. Operator sampling for shot-frugal optimization in variational algorithms. Preprint at <https://arxiv.org/abs/2004.06252> (2020).
50. Sweke, R. et al. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum* **4**, 314 (2020).
51. Defazio, A. Understanding the role of momentum in non-convex optimization: practical insights from a Lyapunov analysis. Preprint at <https://arxiv.org/abs/2010.00406> (2020).
52. Lieb, E., Schultz, T. & Mattis, D. Two soluble models of an antiferromagnetic chain. *Ann. Phys.* **16**, 407–466 (1961).
53. Nielsen, M. A. & Chuang, I. L. *Quantum computation and quantum information* (Cambridge University Press, 2000).
54. Bilkis, M., Cerezo, M., Verdon, G., Coles, P. J. & Cincio, L. A semi-agnostic ansatz with variable structure for quantum machine learning. Preprint at <https://arxiv.org/abs/2103.06712> (2021).
55. Somma, R. D. Quantum eigenvalue estimation via time series analysis. *N. J. Phys.* **21**, 123025 (2019).
56. Kandala, A. et al. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* **549**, 242–246 (2017).
57. Grant, E., Wossnig, L., Ostaszewski, M. & Benedetti, M. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum* **3**, 214 (2019).
58. Skolik, A., McClean, J. R., Mohseni, M., van der Smagt, P. & Leib, M. Layerwise learning for quantum neural networks. *Quant. Mach. Intel.* **3**, 1–11 (2021).
59. Harrow, A. W. & Napp, J. C. Low-depth gradient measurements can improve convergence in variational hybrid quantum-classical algorithms. *Physical Review Letters* **126**, 140502 (2021).
60. Mitarai, K., Negoro, M., Kitagawa, M. & Fujii, K. Quantum circuit learning. *Physical Review A* **98**, 032309 (2018).

ACKNOWLEDGEMENTS

J.G. and K.G. acknowledge support from the U.S. Department of Energy (DOE) through a quantum computing program sponsored by the Los Alamos National

Laboratory (LANL) Information Science & Technology Institute. Z.H., B.C. and P.J.C. acknowledge support from the LANL ASC Beyond Moore's Law project. Z.H. acknowledges subsequent support from the Mark Kac Fellowship. We acknowledge the LANL Laboratory Directed Research and Development (LDRD) program for support of A.S. and initial support of B.C. under project number 20190065DR as well as L.C. under project number 20200022DR. A.A. was supported by the U.S. Department of Energy (DOE), Office of Science, Office of High Energy Physics QuantISED program under Contract No. DE-AC52-06NA25396. L.C. and P.J.C. were also supported by the U.S. DOE, Office of Science, Basic Energy Sciences, Materials Sciences and Engineering Division, Condensed Matter Theory Program. This research used quantum computing resources provided by the LANL Institutional Computing Program, which is supported by the U.S. Department of Energy National Nuclear Security Administration under Contract No. 89233218CNA000001. This research used additional quantum computational resources supported by the LANL ASC Beyond Moore's Law program and by the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

AUTHOR CONTRIBUTIONS

The project was conceived by Z.H., L.C., P.J.C., and A.S. The manuscript was written by Z.H., J.G., K.G., L.C., P.J.C., and A.S. The quantum hardware implementations were performed by J.G. and K.G. The numerical simulations were performed by J.G., B.C., A.A. and L.C. The analytic calculations were performed by Z.H. with input from A.A., P.J.C. and A.S.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41534-022-00625-0>.

Correspondence and requests for materials should be addressed to Joe Gibbs or Zoë. Holmes.

Reprints and permission information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2022