Approximate Quantum Compiling for Quantum Simulation: A Tensor Network based approach

Niall F. Robertson[1], Albert Akhriev[1], Jiri Vala[2,3] and Sergiy Zhuk[1]

[1] *IBM Quantum, IBM Research Europe - Dublin, IBM Technology Campus, Dublin 15, Ireland*
[2] *Maynooth University, Maynooth, Ireland*
[3] *Tyndall National Institute, Cork, Ireland*

Abstract

We introduce *AQCtensor*, a novel algorithm to produce short-depth quantum circuits from Matrix Product States (MPS). Our approach is specifically tailored to the preparation of quantum states generated from the time evolution of quantum many-body Hamiltonians. This tailored approach has two clear advantages over previous algorithms that were designed to map a generic MPS to a quantum circuit. First, we optimize all parameters of a parametric circuit at once using Approximate Quantum Compiling (AQC) - this is to be contrasted with other approaches based on locally optimizing a subset of circuit parameters and "sweeping" across the system. We introduce an optimization scheme to avoid the so-called "orthogonality catastrophe" - i.e. the fact that the fidelity of two arbitrary quantum states decays exponentially with the number of qubits - that would otherwise render a global optimization of the circuit impractical. Second, the depth of our parametric circuit is constant in the number of qubits for a fixed simulation time and fixed error tolerance. This is to be contrasted with the linear circuit Ansatz used in generic algorithms whose depth scales linearly in the number of qubits. For simulation problems on 100 qubits, we show that *AQCtensor* thus achieves at least an order of magnitude reduction in the depth of the resulting optimized circuit, as compared with the best generic MPS to quantum circuit algorithms. We demonstrate our approach on simulation problems on Heisenberg-like Hamiltonians on up to 100 qubits and find optimized quantum circuits that have signficantly reduced depth as compared to standard Trotterized circuits.

1   Introduction

Tensor Networks are powerful tools that can be used to tackle a range of problems in various fields including chemistry [1, 2], machine learning [3, 4] and quantum physics [5, 6]. In recent years, there have been several proposals to tackle problems in these fields by combining tensor networks with quantum computing by e.g. finding an approximate solution to a ground state problem using Matrix Product States (MPS) and then preparing the corresponding quantum state on a quantum computer for further optimization [7]. Similar strategies for quantum machine learning have also been proposed [8]. These schemes thus motivate the design of efficient algorithms that decompose a generic Matrix Product State into a quantum circuit [9, 10, 11, 12] - the advantages and limitations of the known approaches were discussed in detail in [9]. For example, the MPS to quantum circuit algorithm described in [11] requires the compilation of a unitary matrix into 1 and 2-qubit gates - while the number of qubits that this unitary acts on is logarithmic in the bond dimension χ of the MPS, the number of gates required to compile this unitary is exponential in the number of qubits [13] in general and quadratic in χ. It was also discussed in [9] that the performance of the approach described in [10] did not improve in a meaningful way when more layers were added to the Ansatz, thus restricting the algorithm to apply to Matrix Product States with very low bond dimensions. It was finally concluded in [9] that the approach with the best performance was a novel combination of the variational algorithm proposed in [12] with the approach of [10].

All of these algorithms, including the one found to have the best performance, use a "staircase circuit" with a linear structure - see Figure 1. The depth of a "staircase circuit" scales linearly with the number of qubits, thus the minimum depth of even the *least* expressive 100-qubit parametric circuit of this type

would already be deeper than some of the deepest circuits that have been executed to date [14]. By using a more expressible Ansatz such as the brickwork circuit in Figure 1, we obtain a circuit with significantly more expressivity but with the same circuit depth. As we will show, using the brickwork circuit also allows to use a "smart-initialisation" scheme in the optimization procedure. We note that the linear scaling of the depth with qubit number of the staircase circuit is a necessary requirement if one's aim is to prepare a quantum state whose lightcone increases with the number of qubits. However, if one considers a Matrix Product State with a bond dimension χ that is constant in the number of qubits, then it is not necessary to increase the circuit depth as one increases the number of qubits. An example of such a scenario is the preparation of a quantum state generated by the time evolution of a local Hamiltonian at a fixed time - the main focus of this work. We will show that our $AQCtensor$ algorithm can be used
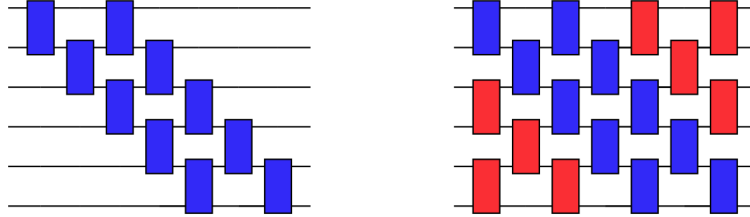

Figure 1: Staircase circuit (left) vs brickwork circuit (right): the staircase circuit has two "staircase layers". Its circuit depth increases linearly with the number of qubits. The brickwork circuit has the same depth as the staircase circuit but has more gates and thus more expressivity. The additional gates in the brickwork circuit (shown in red) do not increase the circuit depth with respect to the staircase circuit.

to significantly reduce the depth of quantum circuits used to simulate the time evolution of quantum many-body systems, even beyond the regime of classical simulability. The workflow to achieve this is shown in Figure 2 and discussed briefly below.

First, one uses a classical algorithm to store a time-evolved quantum state classically as a Matrix Product State. For a one-dimensional system with local interactions one can use e.g. TEBD [15, 16] to prepare the time-evolved MPS, while for models in higher dimensions or with long-range interactions one can use other methods such as the TDVP algorithm [17, 18] or the $W^{I,II}$ method [19]. The bond dimension χ typically increases exponentially with the time that is simulated, thus one eventually reaches a time t at which the classical capabilities are saturated. Second, one uses $AQCtensor$ to find a short-depth circuit that prepares this state. Finally, one then applies additional time steps, e.g. using Trotterization, to prepare a state that, if stored as a Matrix Product State, would have a higher bond dimension than the classical computing resources could store. We will show that this procedure leads to a quantum circuit with a depth which is significantly lower than if one only used a Trotterization scheme without applying $AQCTensor$, or if one used one of the previously introduced MPS to quantum circuit mappings.

We note that a number of schemes to optimize parametric circuits on a quantum device to find circuits that simulate time evolution with depths lower than Trotterized circuits have been proposed [20, 21, 22, 23]. While these approaches offer a number of insights, particularly regarding error bounds [22], each of them suffers from a number of issues such as convergence, runtime and limited device connectivity. As a result, it has been argued that such quantum-variational approaches are not practical for use on near term quantum hardware [24]. Our approach, however, performs the optimization procedure *classically* and hence avoids these issues, allowing us to run the algorithm on 100 qubits - see main text.

We briefly comment on $AQCTensor$ in the context of other compilation algorithms for the purposes of Hamiltonian simulation. $AQCTensor$ can be applied in tandem with these approaches as a procedure to prepare some non-trivial initial state. Consider for example the works [25, 26] which use Matrix Product Operators to optimize for the full unitary representing the time evolution operator. One can apply the workflow presented here to generate a short depth circuit that prepares the state at the largest time allowed by the available classical resources, followed by the application of the optimized circuits representing the time evolution operator. We point out that $AQCTensor$ finds an optimal circuit with respect to the initial state *and* the time evolution operator - using both methods simultaneously would thus be advantageous. Although we note that the classical resources required to optimize for the full
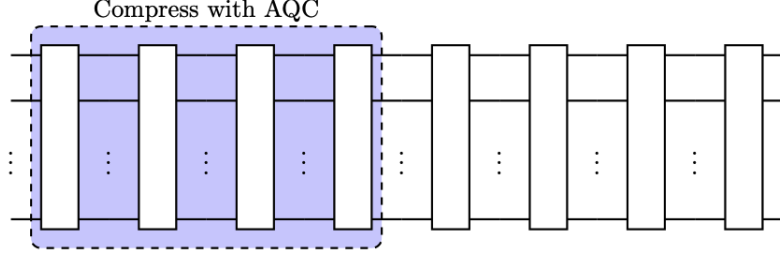
Figure 2: Schematic of our approach: Trotterisation is applied classically (purple box) using TEBD and then a Matrix Product State implementation of Approximate Quantum Compiling is applied to compress this part of the circuit. Standard Trotterisation is then applied on a quantum device afterwards to simulate longer times, i.e. times that are beyond what is possible classically.

unitary are quadratically larger than those required to optimize for the quantum state. One can adopt the same approach for other simulation algorithms such as QDrift [27] and multiproduct formulas [28, 29].

The structure of this work is as follows: in section 2 we review the classical and quantum time evolution algorithms based on Trotterization as well as a brief introduction to Matrix Product States and how they are used in this work. In section 3 we describe the *AQCtensor* algorithm, leaving some details to the appendix. In section 4 we present the results of our approach applied to spin chains with 100 qubits - see Figure 9 where the performance of the optimized circuits are compared with standard Trotterized circuits. We discuss the application of the method to models in higher dimensions and with longer range interactions in Appendix A - see Figures 11 and 12. We conclude in section 5 with a discussion of extensions of this work and how it can be combined with other methods.

2 Background

2.1 Time Evolution

We first consider a family of spin chains with Hamiltonians of the form:

$$H_{XYZ} = -\sum_{i=0}^{L-2} \left(\alpha_i S_i^x S_{i+1}^x + \beta_i S_i^y S_{i+1}^y + \Delta_i S_i^z S_{i+1}^z\right) + \sum_{i=0}^{L-1} h_i S_i^z, \tag{1}$$

where S^x, S^y and S^z are written in terms of Pauli matrices as $S^x = \frac{\sigma^x}{2}$, $S^y = \frac{\sigma^y}{2}$ and $S^z = \frac{\sigma^z}{2}$. We will consider models with long range interactions in appendix A. Note that we can recover the XXZ and XXX models - prototypical examples of 1D integrable models - by setting $\alpha_i = \beta_i = 1$ or $\alpha_i = \beta_i = \Delta_i = 1$ respectively. The dynamical behaviour of these models has been studied extensively [30], including on a quantum computer [31, 32].

The time evolution of a quantum state $|\psi(t)\rangle$ is governed by the Schrödinger equation:

$$|\psi(t)\rangle = e^{-iH_{XYZ}t} |\psi(0)\rangle \tag{2}$$

where $|\psi(0)\rangle$ is the wavefunction at time $t = 0$. We will consider the initial state to be a product state such as e.g. the Néel state, written as: $|\uparrow\downarrow\uparrow\downarrow ... \uparrow\downarrow\rangle$ where $\uparrow$ and $\downarrow$ represent up and down spins respectively. The Néel state for n spins is simply implemented on n qubits as $|1010...10\rangle$.

The time evolution operator $U(t) \equiv e^{-iH_{XYZ}t}$ can be executed as a quantum circuit in a resource efficient way; we first define $h_{i,i+1} \equiv \alpha_i S_i^x S_{i+1}^x + \beta_i S_i^y S_{i+1}^y + \Delta_i S_i^z S_{i+1}^z$ and then rewrite the Hamiltonian in (1) as $H_{XYZ} = H_1 + H_2 + \sum_{i=0}^{L-1} h_i S_i^z$ where $H_1 = -\sum_{i \text{ odd}} h_{i,i+1}$ and $H_2 = -\sum_{i \text{ even}} h_{i,i+1}$. Next we define the unitary $U_{j,j+1}(dt) = e^{ih_{j,j+1}dt}$ for some time step dt. We can then define a symmetric, second-order
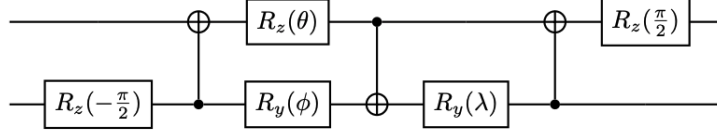
Figure 3: Implementation of two site operator $e^{i(\alpha S^x \otimes S^x + \beta S^y \otimes S^y + \Delta S^z \otimes S^z)}$ as a quantum circuit. We have the correspondences $\theta = \frac{\pi}{2} - \frac{1}{2}\Delta$, $\phi = \frac{1}{2}\alpha - \frac{\pi}{2}$ and $\lambda = \frac{\pi}{2} - \frac{1}{2}\beta$.
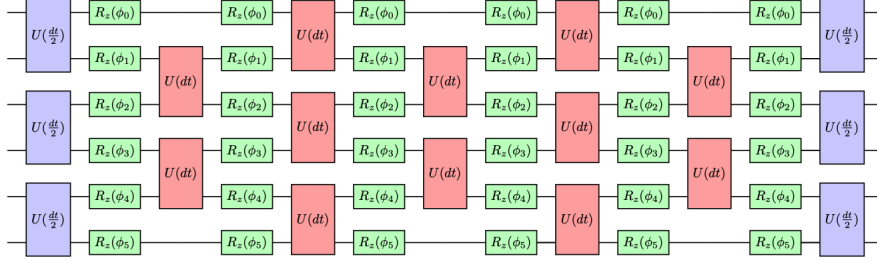

Figure 4: Second order Trotter circuit acting on six qubits. The single qubit rotation gates are shown in green and implement the terms of the form $e^{-\frac{1}{2}ih_j S^z dt}$ in equation (3). These rotation gates take angles given by $\phi_i = h_i dt$. The purple two-qubit gates implement the half Trotter steps - i.e. terms of the form $U_{2j-1,2j}(\frac{dt}{2})$ in equation (3). The red two-qubit gates implement full Trotter steps given by $U_{2j-1,2j}(dt)$.

Suzuki-Trotter time evolution operator $\mathcal{U}_{\text{trott}}(dt)$ in the following way:

$$\mathcal{U}_{\text{trott}}(dt) = \prod_{j=0}^{L/2-1} U_{2j,2j+1}\left(\frac{dt}{2}\right) \prod_{j=0}^{L-1} e^{-\frac{1}{2}ih_j S^z dt} \prod_{j=1}^{L/2-1} U_{2j-1,2j}(dt) \prod_{j=0}^{L-1} e^{-\frac{1}{2}ih_j S^z dt} \prod_{j=0}^{L/2-1} U_{2j,2j+1}\left(\frac{dt}{2}\right) \tag{3}$$

The Suzuki-Trotter time evolution operator $\mathcal{U}_{\text{trott}}(dt)$ is an approximation of the exact time evolution operator $U(t) \equiv e^{-iH_{XYZ}t}$, in particular we have:

$$\mathcal{U}_{\text{trott}}(dt) = e^{-iH_{XYZ}t} + O(dt^3) \tag{4}$$

As discussed in [31], each $U_{j,j+1}(dt)$ can be implemented by the quantum circuit with just three CNOTs as in Figure 3. The full second-order Suzuki-Trotter time evolution operator $\mathcal{U}_{\text{trott}}(dt)$ can be implemented as a quantum circuit in Figure 4. Note that each "half Trotter step" $U_{2j,2j+1}\left(\frac{dt}{2}\right)$ in equation (3) can be fused into a full Trotter step from the previous layer, so the half steps only appear at the beginning and at the end of the circuit - see the purple gates in Figure 4.

2.2 Matrix Product States

In this section we briefly describe the properties of Matrix Product States. For more details, we refer the reader to more in depth reviews [6, 33, 34]. An arbitrary quantum state on n qubits can be written in terms of complex variables $c_{j_1,...,j_n}$, the number of which scales as 2^n:

$$|\psi\rangle = \sum_{\{j_1,...,j_n\}} c_{j_1,...,j_n} |j_1,...,j_n\rangle \tag{5}$$

where the sum is over all configurations of the binary variables $j_1,...,j_n$. The bipartite entanglement entropy of an arbitrary quantum state picked at random from Hilbert space satisfies a volume law which is distinct from area law entanglement in which case the entanglement entropy of two regions after the bipartition of the system is proportional to the area of the boundary of the system. A small subset of states in Hilbert space satisfies an area law. The coefficients $c_{j_1,...,j_n}$ of such states have a certain structure that we can take advantage of to study classically. Any state $|\psi\rangle$ can be written in the following way:

$$c_{j_1,...,j_n} = A_{j_1}^{(1)} \cdot A_{j_2}^{(2)} ... \cdot A_{j_n}^{(n)} \tag{6}$$
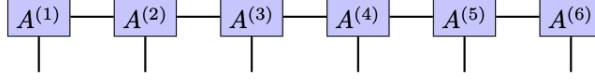
4

Figure 5: Graphical representation of an MPS. There are two matrices $A^{(i)}$ for each qubit at position i.
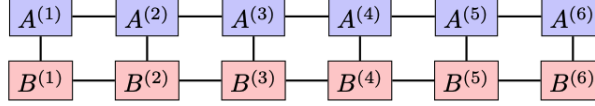

Figure 6: The inner product $\langle \psi_1 | \psi_2 \rangle$ of two Matrix Product States - see equations (9) and (6).

where the A_j are $\chi_j \times \chi_{j+1}$ dimensional matrices. Quantum states of the form (6) are known as Matrix Product States (MPS). The maximum value of χ_j is referred to as the bond dimension of the MPS. We can represent an MPS graphically as in Figure 5. The bond dimension χ_j can be seen as a measure of the entanglement between the two subsystems when a bipartition is made at qubit j. Therefore, states in Hilbert space that satisfy an area law - and therefore have a low bond dimension in their MPS representation - can be efficiently stored as Matrix Product States. States that satisfy a volume law will have a bond dimension that is exponential in the number of qubits. We will consider in this work the non-trivial dynamics governed by equation (2). The bipartite entanglement entropy of a ground state of a one-dimensional Hamiltonian that has a gap between its ground state and its excited state is independent of the size of the subsystems [35]. The ground state of such a system - and hence the initial state in our setup - can be efficiently stored as an MPS. One can then use an algorithm such as TEBD (Time Evolving Block Decimation) [15, 16] or TDVP [17, 18] to update the MPS as a function of time to study the dynamics of the system. However, the entanglement entropy of the state increases linearly with time, hence the bond dimension χ that is required to keep the error constant diverges exponentially with time, thus motivating the use of a quantum computer.

3 Algorithm

Approximate quantum compiling (AQC) involves the design of a parametric quantum circuit with fixed depth - the parameters are then adjusted to bring it as close as possible to the target, where "close" is defined via some carefully chosen metric, see below. A parametric circuit is constructed from $CNOT$ blocks, each of which has four parameters - see Figure 7. The position of the $CNOT$ blocks in the parameterised circuit can be customised to suit the particular target circuit that one is interested in. Here we are interested in finding a circuit that implements the unitary time evolution operator, which is written in equation (2) for the case of a one-dimensional spin chain. For this family of models, we thus consider a structure inspired by the second-order Trotter circuit in Figure 4. Recall that each block $U(dt)$ in Figure 4 represents the 2-qubit sub-circuit with three $CNOT$s in Figure 3; it is therefore natural to consider a circuit Ansatz with sub-circuits each with three $CNOT$ blocks as in Figure 8, such that the circuit Ansatz mimics the structure of the second order Trotter circuits. We denote the number of repetitions of $CNOT$ blocks in each layer by 3, and the number of layers by l - the parameterised circuit in Figure 8 corresponds to $n = 4$ qubits, $l = 2$ layers and $b = 3$ $CNOT$ blocks in each layer. Note that when we take $b = 3$, the total $CNOT$ depth of the parameterised circuit with l layers is equal to the $CNOT$ depth of a Trotter circuit with l Trotter steps - see Figure 4. In Figure 8 there are three rotation gates acting on each qubit at the beginning of the circuit. If the initial state of the parametric circuit is $|0\rangle$ then the initial rotation gate $R_z(\theta)$ is redundant but would act non-trivially on more general initial states.
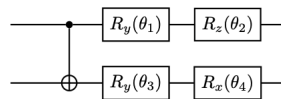

Figure 7: $CNOT$ block - each block corresponds to one of the purple two-qubit gates in Figure 8.
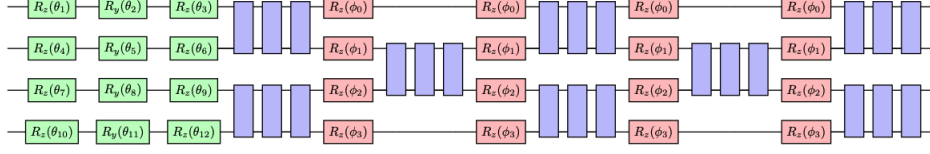
Figure 8: Parameterised circuit inspired by the structure of the second order Suzuki-Trotter circuit in Figure 4. The angles of the red single qubit rotation gates are given by $\phi_i = h_i dt$. The angles of the green single qubit gates are parameters to be optimized. The purple two-qubit gates are $CNOT$ blocks shown in Figure 7, each with an independent set of parameters $\theta_1, \theta_2, \theta_3, \theta_4$ so that there is a total of 108 free parameters in this circuit - 12 in the green gates and 96 in the purple gates. Note that in each triplet of purple $CNOT$ blocks, the direction of the $CNOT$ gate in the outer two blocks should be reversed in order to match exactly the $CNOT$ structure of the Suzuki-Trotter circuit in Figure 4.

Algorithm 1 *AQCtensor* algorithm
___
    Input Hamiltonian H, initial state $|\psi(0)\rangle$, parametric circuit $V(\theta_0)$
    Output Optimized parametric circuit with appended Trotter steps: $(U_{trott}(dt))^k V(\theta)$
1. Apply classical algorithm (e.g. TEBD) to generate state $|\psi(t)\rangle$
2. Minimize cost function in (8) with respect to parameters θ to find optimized parameters θ_0
3. Append k additional Trotter steps with time step dt to $V(\theta_0)$ to obtain the circuit $(U_{trott}(dt))^k V(\theta_0)$

One can define the distance between the target and parameterised circuit via a number of different metrics. Here we define this distance based on the Hilbert-Schmidt test:

$$C_{hs}^{\text{state}} = 1 - |\langle 0| V^\dagger(\theta) |\psi_t\rangle|^2 \tag{7}$$

where $V(\theta)$ is the unitary operator corresponding to the parametric circuit. Note that here we are considering the application of AQC to state preparation as opposed to full circuit compilation. More precisely, this means that our cost function is designed such that it is minimised when the action of $V(\theta)$ on the initial state produces a state that is as close as possible to a target state $|\psi_t\rangle$ (up to some global phase). This is in contrast to the situation where one starts with some target *circuit* U and the cost function is designed to bring the full matrix $V(\theta)$ as close as possible to U.

It was pointed out in [36] that the gradient of the cost function in (7) vanishes exponentially in the number of qubits. While this is primarily a problem when training on a quantum device - which is not the case in the *AQCtensor* framework - we consider problem sizes of up to to 100 qubits, thus the exponentially vanishing gradient of C_{hs}^{state} can lead to impractically long training times even when using classical methods to train the circuit parameters. This is related to the so-called "orthogonality catastrophe" whereby the overlap of two quantum states in Hilbert space vanishes exponentially in the number of qubits, resulting in $C_{hs}^{\text{state}} \approx 1$ for a large region of parameter space. Here we will briefly introduce two methods to overcome this issue - more details can be found in appendix B. The first method is based on the concept of "local cost functions" [36, 37] and involves the addition of terms to C_{hs}^{state}, so that our algorithm uses a cost function of the form:

$$C_L^{(1)}(\alpha) = 1 - |\langle 0| V^\dagger(\theta) |\psi_t\rangle|^2 - \alpha \sum_{j=1}^{n} |\langle 0| X_j V^\dagger(\theta) |\psi_t\rangle|^2 \tag{8}$$

where α is a parameter that can be tuned throughout the optimization procedure. We discuss in appendix B how the additional term on the RHS of (8) arises from expanding and truncating the full expression for the local cost function introduced in [36]. The second method that we introduce is "Trotter-initialization" - whereby we tune the initial values of the circuit parameters, i.e. at the beginning of the optimization procedure, to take values such that the parametric circuit initially corresponds exactly to the Trotter circuit. This has the advantage that performing even one step of the optimization procedure ensures that our parametric circuit has a higher fidelity with the exactly time-evolved state than a standard Trotter circuit. Note that the "Trotter-initialization" method is only possible when the $CNOT$-structure of the

parametric circuit matches that of the Trotter circuit, as is the case for the circuit in Figure 8.

Note that each term in (8) is an overlap of quantum states, and since the overlap of two MPS can be calculated very efficiently, the architecture of Matrix Product States can be leveraged to calculate the cost function and solve the approximate quantum compilation problem for large numbers of qubits. Consider for example two quantum states $|\psi_1\rangle$ and $|\psi_2\rangle$:

$$
\begin{aligned}
|\psi_1\rangle &= \sum_{\{j_1,...,j_n\}} c^{(1)}_{j_1,...,j_n} |j_1,...,j_n\rangle \\
|\psi_2\rangle &= \sum_{\{j_1,...,j_n\}} c^{(2)}_{j_1,...,j_n} |j_1,...,j_n\rangle
\end{aligned}
\tag{9}
$$

As discussed in section 2.2, for weakly entangled states the coefficients $c^{(1)}_{j_1,...,j_n}$ and $c^{(2)}_{j_1,...,j_n}$ can be represented efficiently as Matrix Product States:

$$
\begin{aligned}
c^{(1)}_{j_1,...,j_n} &= A^{(1)}_{j_1} \cdot A^{(2)}_{j_2} ... \cdot A^{(n)}_{j_n} \\
c^{(2)}_{j_1,...,j_n} &= B^{(1)}_{j_1} \cdot B^{(2)}_{j_2} ... \cdot B^{(n)}_{j_n}
\end{aligned}
\tag{10}
$$

The fidelity of two MPS:

$$
f(|\psi_1\rangle, |\psi_2\rangle) = || \langle \psi_1 | \psi_2 \rangle ||^2
\tag{11}
$$

can be calculated efficiently by contracting the Tensor Network - see Figure 6. One has a choice regarding the definitions of the states $|\psi_1\rangle$ and $|\psi_2\rangle$ when evaluating the cost function in (8). One can for example define $|\psi_1\rangle = V(\theta)|0\rangle$ and $|\psi_2\rangle = |\psi_t\rangle$ or $|\psi_1\rangle = |0\rangle$ and $|\psi_2\rangle = V^\dagger(\theta)|\psi_t\rangle$. The latter approach has the advantage that, when the parameters θ are close to their optimal values then both $|\psi_1\rangle$ and $|\psi_2\rangle$ are weakly entangled states, thus allowing us to reduce the errors induced from the truncation of the bond dimension. We thus adopt this approach in our simulations, but we point out that for some other families of Hamiltonians this may not necessarily be the optimal strategy.

4   Results

We now present our results of the simulations using the *AQCtensor* algorithm. In this section we consider the application of *AQCtensor* to the family of Hamiltonians defined in equation (1) and we leave the discussion of longer range Hamiltonians to appendix A. We consider two paradigms; firstly we compare Trotterized circuits and optimized circuits of equal depth and measure their respective fidelities with the "ground-truth" circuit, i.e. the Matrix Product State corresponding to the state generated by a Trotterized unitary with a very small time step dt. We will show that the fidelities of the optimized parametric circuits are significantly and consistently larger than the Trotterized circuits with equal circuit depth (left panel of Figure 9). Secondly, we compare Trotterized circuits and optimized circuits with similar fidelities and compare their circuit depths (right panel of Figure 9). We find that the optimized circuits can achieve roughly equal fidelities to the Trotterized circuits with half the circuit depth. We consider three Hamiltonians in the class of models defined in equation (1): 1) randomly generated coefficients 2) the XXX Hamiltonian and 3) the XXZ Hamiltonian - see the caption in Figure 9. We define the following notation:

- $|a_1\rangle$: The state generated by the optimized parametric circuit in Figure 8.

- Number of layers l in Ansatz: in Figure 8 there are $l = 2$ layers.

- $|t_1\rangle$: The state generated by the Trotter circuit in Figure 4. The time step used to generate this state is given by the total time divided by the number of layers/Trotter steps.

- Number of Trotter steps: the analogue of the number of layers in the Ansatz circuits. In Figure 4 there are 3 Trotter steps.

- $|t^{(gt)}\rangle$: the "ground truth" generated by classical Tensor Network simulations of deep Trotter circuits, i.e. small time steps. We take a time step dt that is 10 times smaller than the time step used to generate $|t_1\rangle$.
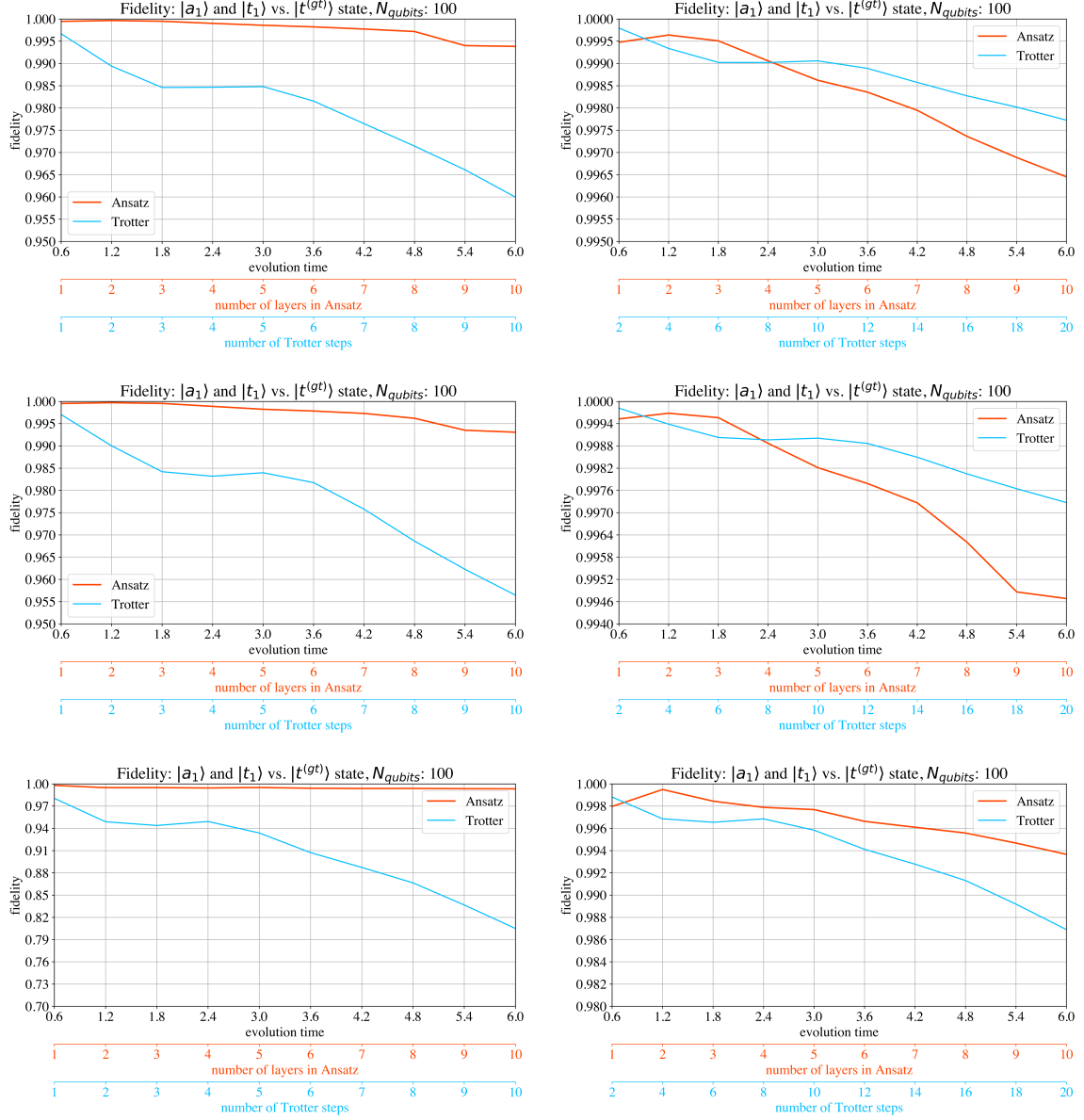
Figure 9: Top panel: $H = -\sum_{i=0}^{L-1} \left(\alpha_i S_i^x S_{i+1}^x + \beta_i S_i^y S_{i+1}^y + \Delta_i S_i^z S_{i+1}^z \right)$ with $\alpha_i, \beta_i, \Delta_i$ taking randomly generated values between 0.375 and 1.125. Middle panel: $H = -\sum_{i=0}^{L-1} \left(0.75\, S_i^x S_{i+1}^x + 0.75\, S_i^y S_{i+1}^y + 0.75\, S_i^z S_{i+1}^z \right)$. Lower panel: $H = -\sum_{i=0}^{L-1} \left(0.75\, S_i^x S_{i+1}^x + 0.75\, S_i^y S_{i+1}^y + 1.5\, S_i^z S_{i+1}^z \right)$. In the left panels, the accuracy of the optimized parametric circuit Ansatz is compared with the accuracy of the Suzuki-Trotter circuits with the same circuit depth. On the right, the Suzuki-Trotter circuits have twice the circuit depth as the optimized parametric circuit yet have comparable accuracy. The maximum number of optimization iterations was set to 30.

The simulations were run on Xeon E5-2699, v4, 2.2 GHz machine with 44/88 cores/threads and took several hours each. We used the Qiskit Aer MPS simulator - we believe that runtime can be decreased significantly by improving the performance of this simulator. We used the L-BFGS classical optimizer to minimize the cost function in our simulations. We found that convergence of the cost function was enhanced by prepending the L-BFGS optimizer with a few steps of the ADAM optimizer.

5 Discussion

This work demonstrates the power of combining state-of-the art classical simulation methods with quantum computing. Rather than viewing quantum computing and tensor networks as competing approaches, our *AQCtensor* algorithm allows one to combine the power of both of these tools. While this work focused on Matrix Product States and a family of one-dimensional spin chains, future work could apply a similar workflow to higher dimensional models, where more sophisticated tensor network algorithms are used to optimize quantum circuits. Our work also demonstrates the effect of using a tailored approach to MPS-quantum circuit mappings. By using such an approach we found a much shallower circuit that generated the time-evolved MPS than if we had used one of the generic algorithms discussed in [9]. We focused here on optimizing circuits for time evolution, but one could imagine applying a similar tailored approach to the MPS/quantum approaches to ground-state search and quantum machine learning described in [7, 8]. The circuit depth reductions achieved by *AQCTensor* allow for the reduction of hardware noise due to the reduced number of quantum gates in the optimized circuit. We note that, in addition to the error reduction achieved by reducing the circuit depth, one can *also* apply error mitigation methods [38, 14] to the optimized circuit to further reduce hardware noise. We also point out that the *AQCTensor* workflow is still likely to be applicable in the fault tolerant era since many algorithms such as Quantum Phase Estimation require a good initial state preparation.

As discussed in section 3, there is no unique definition of the pair of states $|\psi_1\rangle$ and $|\psi_2\rangle$ that are contracted to obtain the cost function in (8). Each choice of a particular pair of states corresponds to a particular order of bond dimension truncations. We used one particular choice for our extensive study of one-dimensional spin chains, namely: $|\psi_1\rangle = |0\rangle$ and $|\psi_2\rangle = V^\dagger(\theta)|\psi_t\rangle$. However, for other Hamiltonians with long-range interactions or in higher dimensions, one may find that a different approach works better. We lay the groundwork for the exploration of the application of *AQCTensor* to these more complex models in appendix A where we present some initial results on a model defined on a 2D decorated hexagonal lattice as well as a spin chain with next to nearest neighbour interactions. We demonstrate that the *AQCTensor* workflow can still be applied successfully to these models. Future work can consider how to optimize the method for these more challenging cases. In addition to improving these MPS-based methods, one could also consider using higher dimensional tensor networks, i.e. beyond MPS, within the *AQCTensor* workflow to further improve its performance when applied to higher dimensional Hamiltonians.

Finally, we briefly comment on how one could combine our approach to optimizing circuits for time-evolution with other approaches. The workflow in Figure 2 allows us to find short depth circuits with low Trotter error that simulate the time-evolved state up until the latest time that the classical resources available can handle. To mitigate the Trotter errors arising from the remaining part of the circuit which has not been optimized, one could use other techniques such as multiproduct formulas [29, 28] - this will be the focus of future work. Additionally, one could consider using additional more resource-intensive tensor-network techniques to optimize the circuits at later times using the unitary compilation techniques described in [26, 25].

A *AQCTensor* beyond 1D models

Here we consider two additional models to the family of Hamiltonians defined in equation (1). First, we consider a next to nearest neighbour Hamiltonian with Heisenberg interactions:

$$H_{nnn} = -\sum_{i=0}^{L-2} \left(\alpha_i S_i^x S_{i+1}^x + \beta_i S_i^y S_{i+1}^y + \Delta_i S_i^z S_{i+1}^z\right) - \sum_{i=0}^{L-3} \left(\alpha_i S_i^x S_{i+2}^x + \beta_i S_i^y S_{i+2}^y + \Delta_i S_i^z S_{i+2}^z\right) + \sum_{i=0}^{L-1} h_i S_i^z,$$

(12)

We take $\alpha_i = \beta_i = \Delta_i = 0.75$ for the purposes of our numerical studies below. Second, we consider a model defined on the decorated hexagonal lattice with Hamiltonian:

$$H_{hex} = -\sum_{i,j \in e}^{L-2} \left(\alpha_i S_i^x S_j^x + \beta_i S_i^y S_j^y + \Delta_i S_i^z S_j^z\right) + \sum_{i=0}^{L-1} h_i S_i^z,$$

(13)

where the sum is over the edges e in the graph such that i, j are nearest neighbours on the decorated hexagonal lattice in Figure 10. For this model, we take $\alpha_i = \beta_i = 0.5$ and $\Delta_i = 1.0$. We consider an array of 2×2 decorated hexagons which has a total of 35 qubits. We apply MPS-based methods to this two-dimensional model by ordering the qubits as shown on the figure and 'unrolling' the graph onto a one-dimensional spin chain. This has the effect of introducing long-range interactions - for example, qubits labelled 20 and 30 are nearest neighbours in Figure 10 but will be separated by a distance of 10 on the one-dimensional chain to which they are mapped. In Figure 11, we compare the fidelities with the ground truth circuit $|t^{(gt)}\rangle$ of the standard Trotterized circuits and the optimized circuits. It is observed that *AQCTensor* significantly improves the fidelities for both the next to nearest neighbour model on 40 qubits (left panel) and the 2×2 decorated hexagonal model on 35 qubits (right panel). The cost function in both cases is evaluated using the choice $|\psi_1\rangle = |0\rangle$ and $|\psi_2\rangle = V^\dagger(\theta)|\psi_t\rangle$, as discussed in the main text. In Figure 12, we plot the convergence of the fidelity as a function of the number of iterations in the optimization for the Hamiltonians in both (12) and (13). We compare this convergence profile for two different values of the bond dimension in the MPS, $\chi = 50$ and $\chi = 100$. The results demonstrate that the optimization results have converged even at these very modest values of χ. This is unsurprising as the states $|\psi_1\rangle = |0\rangle$ and $|\psi_2\rangle = V^\dagger(\theta)|\psi_t\rangle$ are both weakly entangled, particularly when the parameters θ are close to their optimal values.
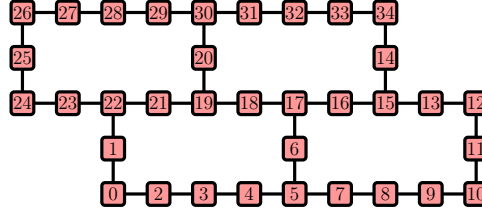

Figure 10: The 2x2 decorated hexagon graph on which the Hamiltonian in (13) is defined. To simulate this model with Matrix Product States, one first maps the two-dimensional topology to the chain by labelling each of the 35 qubits with an integer. This results in the introduction of long-range interactions.

B Circuit optimization: cost function and Trotter-initialisation

As discussed in the main text, the *AQCtensor* framework involves the design of a parametric quantum circuit with a specified $CNOT$ structure. The parametric circuit is written in terms of so-called $CNOT$ blocks [13] - defined as a $CNOT$ gate followed by single qubit rotations (see Figure 7). Formally, a block with a $CNOT$ gate acting on a "control" qubit j and "target" qubit k is written as $\mathrm{CU}_{jk}(\theta_1, \theta_2, \theta_3, \theta_4)$ and for a specified $CNOT$ structure one can then write down a fully parameterised circuit as:

$$\begin{aligned}V_{ct}(\boldsymbol{\theta}) =& \mathrm{CU}_{ct(L)}\left(\theta_{3n+4L-3}, \ldots, \theta_{3n+4L}\right) \cdots \mathrm{CU}_{ct(1)}\left(\theta_{3n+1}, \ldots, \theta_{3n+4}\right) \\ & [R_z\left(\theta_1\right) R_y\left(\theta_2\right) R_z\left(\theta_3\right)] \otimes \cdots \otimes [R_z\left(\theta_{3n-2}\right) R_y\left(\theta_{3n-1}\right) R_z\left(\theta_{3n}\right)]\end{aligned}$$

(14)

The goal of AQC is to tune the parameters θ to minimise the cost function under consideration. As discussed in section 3, when the $CNOT$ structure of the parametric circuit matches that of the second-order
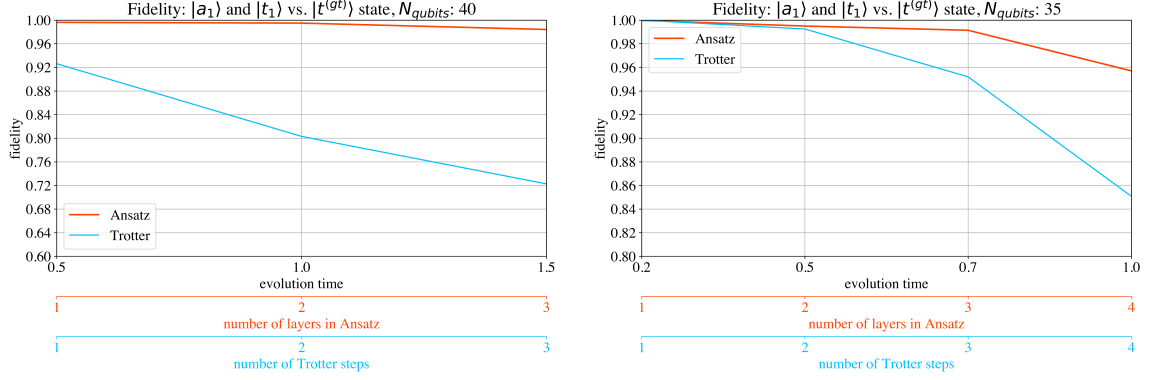
Figure 11: Fidelities between the ground-truth circuits $|t^{(gt)}\rangle$ and the circuits optimized with *AQCTensor*. Left panel: the next to nearest neighbour Hamiltonian in (12) with $\alpha_i = \beta_i = \Delta_i = 0.75$. Right panel: The Hamiltonian in (13) with $\alpha_i = \beta_i = 0.5$ and $\Delta_i = 1.0$ with nearest neighbour couplings on the decorated hexagon lattice in Figure 10.
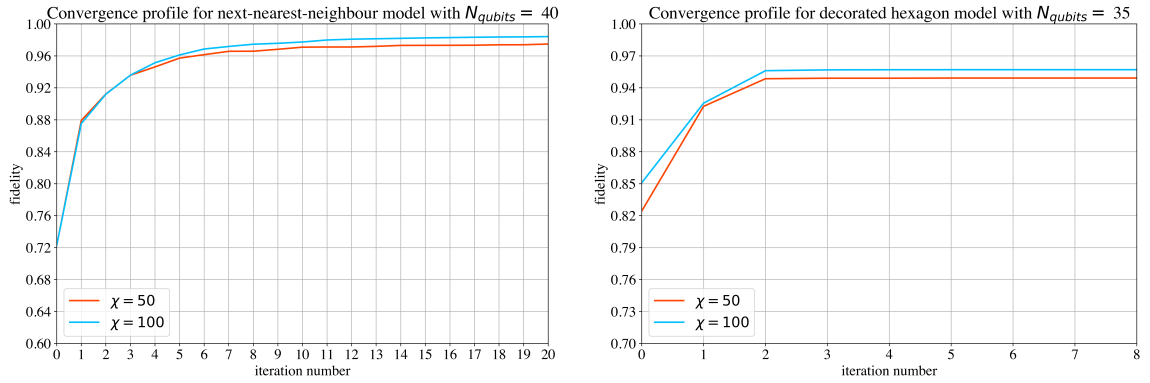

Figure 12: The fidelities of the optimized circuits as a function of the iteration number in the optimization procedure. Left panel: the next to nearest neighbour Hamiltonian in (12) at time $t = 1.5$. Right panel: the decorated hexagonal Hamiltonian in (13) at time $t = 1.0$.

Trotter circuit - as is the case for Figures 8 and 4 - we can initialise the parameters such that the two circuits are initially identical, and hence such that the state $V(\theta)|0\rangle$ is precisely the Trotter state with time step dt equal to the total simulation time divided by the number of layers l in the circuit Ansatz. This is to allow the the optimization procedure to start from an initial state which is likely to be closer to the target state than a state generated from a random initialisation of θ. More precisely, for a given group of 3 $CNOT$-block repetitions acting on qubits i and $i + 1$ - see Figure 8 - we tune the parameters of these blocks such that their combination produces the same 2-qubit unitary as that in Figure 3 with $\theta = \frac{\pi}{2} - \frac{1}{2}\Delta_i$, $\phi = \frac{1}{2}\alpha_i - \frac{\pi}{2}$ and $\lambda = \frac{\pi}{2} - \frac{1}{2}\beta_i$ where α_i, β_i and Δ_i correspond to the coefficients in the Hamiltonian H_{XYZ} in equation (1).

We now turn our attention to the cost function in equation (8). It was pointed out in [36] that the gradient of the cost function in (7) vanishes exponentially in the number of qubits. This observation lead to the distinction between global and local cost functions; local cost functions have only polynomially vanishing gradients in some cases of interest - see [36, 39, 37] for details. As was shown in [37], the Hilbert-Schmidt test - which is a global cost function - can be turned into a local one by adding several "bit-flip" terms which increases the magnitude of the variance of the gradient:

$$C_{lhs}^{\text{state}} = 1 - |\langle 0| V^\dagger(\theta) |\psi_t\rangle|^2 - \left(\frac{n-1}{n}\right) \sum_{j=1}^n |\langle 0| X_j V^\dagger(\theta) |\psi_t\rangle|^2$$
$$- \left(\frac{n-2}{n}\right) \sum_{j<k} |\langle 0| X_j X_k V^\dagger(\theta) |\psi_t\rangle|^2 - ... - \frac{1}{n} \sum_{j<k<l<...} |\langle 0| X_j X_k X_l...V^\dagger(\theta) |\psi_t\rangle|^2$$

(15)

11

One can define a truncated version of C_{lhs}^{state} to get:

$$
\begin{aligned}
C_L^{(k)}(\alpha_1,...,\alpha_k) =& 1 - |\langle 0| V^\dagger(\theta) |\psi_0\rangle|^2 - \alpha_1 \sum_{j=1}^n |\langle 0| X_j V^\dagger(\theta) |\psi_0\rangle|^2 \\
& - \alpha_2 \sum_{j<k} |\langle 0| X_j X_k V^\dagger(\theta) |\psi_0\rangle|^2 - ... - \alpha_k \sum_{j_1<...<j_k} |\langle 0| X_{j_1}...X_{j_k} V^\dagger(\theta) |\psi_0\rangle|^2
\end{aligned}
\tag{16}
$$

It was shown in [37], that when one takes the trivial parametric circuit Ansatz:

$$
V(\theta) = \otimes_{j=1}^n e^{-i\theta_j \frac{\sigma_x}{2}}
\tag{17}
$$

and the trivial target state:

$$
|\psi_0\rangle = |0\rangle
\tag{18}
$$

the variance of the gradient of $C_L^{(k)}(\alpha_1,...,\alpha_k)$ increases by a constant multiplicative factor for every "bit-flip" term in equation (16), i.e. we have:

$$
\text{Var}\left[\frac{\partial C_L^{(k)}}{\partial \theta_j}\right] \propto \left(\frac{3}{8}\right)^{n-k-1}
\tag{19}
$$

While this is a rigorous result for the particularly simple circuit Ansatz and target state defined here, we do not have such a proof for the general case. However, the result suggests that the cost function in (16) may converge faster than the cost function in (7) for more general cases of interest and we thus use it as a heuristic method - see [37] for more details on how we tune α throughout the optimization. The cost of calculating $C_L^{(k)}(\alpha_1,...,\alpha_k)$ and its gradient increases with each k so in practice we take $k=1$, i.e. we have dropped all terms with more than one NOT operators X_i in (16), thus leading to the expression in (8).

C   Acknowledgements

References

[1] Lucas O Wagner, Thomas E Baker, E Miles Stoudenmire, Kieron Burke, and Steven R White. Kohn-sham calculations with the exact functional. *Physical Review B*, 90(4):045109, 2014.

[2] EM Stoudenmire, Lucas O Wagner, Steven R White, and Kieron Burke. One-dimensional continuum electronic structure with the density-matrix renormalization group and its implications for density-functional theory. *Physical review letters*, 109(5):056402, 2012.

[3] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. *Advances in Neural Information Processing Systems*, 29, 2016.

[4] Tom Vieijra, Laurens Vanderstraeten, and Frank Verstraete. Generative modeling with projected entangled-pair states. *arXiv preprint arXiv:2202.08177*, 2022.

[5] Román Orús. Tensor networks for complex quantum systems. *Nature Reviews Physics*, 1(9):538–550, 2019.

[6] Shi-Ju Ran, Emanuele Tirrito, Cheng Peng, Xi Chen, Luca Tagliacozzo, Gang Su, and Maciej Lewenstein. *Tensor network contractions: methods and applications to quantum many-body systems*. Springer Nature, 2020.

[7] Manuel S Rudolph, Jacob Miller, Danial Motlagh, Jing Chen, Atithi Acharya, and Alejandro Perdomo-Ortiz. Synergistic pretraining of parametrized quantum circuits via tensor networks. *Nature Communications*, 14(1):8367, 2023.

[8] William Huggins, Piyush Patil, Bradley Mitchell, K Birgitta Whaley, and E Miles Stoudenmire. Towards quantum machine learning with tensor networks. *Quantum Science and technology*, 4(2):024001, 2019.

[9] Manuel S Rudolph, Jing Chen, Jacob Miller, Atithi Acharya, and Alejandro Perdomo-Ortiz. Decomposition of matrix product states into shallow quantum circuits. *Quantum Science and Technology*, 9(1):015012, 2023.

[10] Shi-Ju Ran. Encoding of matrix product states into quantum circuits of one-and two-qubit gates. *Physical Review A*, 101(3):032310, 2020.

[11] Sheng-Hsuan Lin, Rohit Dilip, Andrew G Green, Adam Smith, and Frank Pollmann. Real-and imaginary-time evolution with compressed quantum circuits. *PRX Quantum*, 2(1):010342, 2021.

[12] Matan Ben Dov, David Shnaiderov, Adi Makmal, and Emanuele G Dalla Torre. Approximate encoding of quantum states using shallow circuits. *arXiv preprint arXiv:2207.00028*, 2022.

[13] Liam Madden and Andrea Simonetto. Best approximate quantum compiling problems. *ACM Transactions on Quantum Computing*, 3(2):1–29, 2022.

[14] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout Van Den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, et al. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

[15] Johannes Hauschild and Frank Pollmann. Efficient numerical simulations with tensor networks: Tensor network python (tenpy). *SciPost Physics Lecture Notes*, page 005, 2018.

[16] Guifré Vidal. Efficient simulation of one-dimensional quantum many-body systems. *Physical review letters*, 93(4):040502, 2004.

[17] Jutho Haegeman, J. Ignacio Cirac, Tobias J. Osborne, Iztok Pižorn, Henri Verschelde, and Frank Verstraete. Time-dependent variational principle for quantum lattices. *Phys. Rev. Lett.*, 107:070601, Aug 2011.

[18] Jutho Haegeman, Christian Lubich, Ivan Oseledets, Bart Vandereycken, and Frank Verstraete. Unifying time evolution and optimization with matrix product states. *Phys. Rev. B*, 94:165116, Oct 2016.

[19] Michael P. Zaletel, Roger S. K. Mong, Christoph Karrasch, Joel E. Moore, and Frank Pollmann. Time-evolving a matrix product state with long-ranged interactions. *Phys. Rev. B*, 91:165112, Apr 2015.

[20] Joe Gibbs, Kaitlin Gili, Zoë Holmes, Benjamin Commeau, Andrew Arrasmith, Lukasz Cincio, Patrick J Coles, and Andrew Sornborger. Long-time simulations for fixed input states on quantum hardware. *npj Quantum Information*, 8(1):135, 2022.

[21] Cristina Cirstoiu, Zoe Holmes, Joseph Iosue, Lukasz Cincio, Patrick J Coles, and Andrew Sornborger. Variational fast forwarding for quantum simulation beyond the coherence time. *npj Quantum Information*, 6(1):1–10, 2020.

[22] Christa Zoufal, David Sutter, and Stefan Woerner. Error bounds for variational quantum time evolution. *Physical Review Applied*, 20(4):044059, 2023.

[23] Kishor Bharti and Tobias Haug. Quantum-assisted simulator. *Physical Review A*, 104(4):042418, 2021.

[24] Alexander Miessen, Pauline J Ollitrault, and Ivano Tavernelli. Quantum algorithms for quantum dynamics: a performance study on the spin-boson model. *Physical Review Research*, 3(4):043212, 2021.

[25] Maurits S. J. Tepaske, Dominik Hahn, and David J. Luitz. Optimal compression of quantum many-body time evolution operators into brickwall circuits. *SciPost Phys.*, 14:073, 2023.

[26] Conor Mc Keever and Michael Lubasch. Classically optimized hamiltonian simulation. *Phys. Rev. Res.*, 5:023146, Jun 2023.

[27] Earl Campbell. Random compiler for fast hamiltonian simulation. *Physical review letters*, 123(7):070503, 2019.

[28] Almudena Carrera Vazquez, Daniel J Egger, David Ochsner, and Stefan Woerner. Well-conditioned multi-product formulas for hardware-friendly hamiltonian simulation. *Quantum*, 7:1067, 2023.

[29] Sergiy Zhuk, Niall F Robertson, and Sergey Bravyi. Trotter error bounds and dynamic multi-product formulas for hamiltonian simulation. *Physical Review Research*, 6(3):033309, 2024.

[30] Lorenzo Piroli, Balázs Pozsgay, and Eric Vernier. From the quantum transfer matrix to the quench action: the loschmidt echo in xxz heisenberg spin chains. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(2):023106, 2017.

[31] Adam Smith, MS Kim, Frank Pollmann, and Johannes Knolle. Simulating quantum many-body dynamics on a current digital quantum computer. *npj Quantum Information*, 5(1):1–13, 2019.

[32] Nathan Keenan, Niall F Robertson, Tara Murphy, Sergiy Zhuk, and John Goold. Evidence of kardar-parisi-zhang scaling on a digital quantum simulator. *npj Quantum Information*, 9(1):72, 2023.

[33] Ulrich Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of physics*, 326(1):96–192, 2011.

[34] Glen Evenbly. A practical guide to the numerical implementation of tensor networks i: Contractions, decompositions, and gauge freedom. *Frontiers in Applied Mathematics and Statistics*, 8:806549, 2022.

[35] Matthew B Hastings. An area law for one-dimensional quantum systems. *Journal of statistical mechanics: theory and experiment*, 2007(08):P08024, 2007.

[36] Sumeet Khatri, Ryan LaRose, Alexander Poremba, Lukasz Cincio, Andrew T Sornborger, and Patrick J Coles. Quantum-assisted quantum compiling. *Quantum*, 3:140, 2019.

[37] Niall F Robertson, Albert Akhriev, Jiri Vala, and Sergiy Zhuk. Escaping barren plateaus in approximate quantum compiling. *arXiv preprint arXiv:2210.09191*, 2022.

[38] Ewout Van Den Berg, Zlatko K Minev, Abhinav Kandala, and Kristan Temme. Probabilistic error cancellation with sparse pauli–lindblad models on noisy quantum processors. *Nature physics*, 19(8):1116–1121, 2023.

[39] Marco Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nature communications*, 12(1):1–12, 2021.