

Riemannian quantum circuit optimization based on matrix product operators

Isabel Nha Minh Le^{1,2}, Shuo Sun^{1,2}, and Christian B. Mendl^{1,2,3}

¹Technical University of Munich, School of Computation, Information and Technology, Boltzmannstr. 3, 85748 Garching, Germany

²Munich Center for Quantum Science and Technology (MCQST), Schellingstr. 4, 80799 Munich, Germany

³Technical University of Munich, Institute for Advanced Study, Lichtenbergstr. 2a, 85748 Garching, Germany

We significantly enhance the simulation accuracy of initial Trotter circuits for Hamiltonian simulation of quantum systems by integrating first-order Riemannian optimization with tensor network methods. Unlike previous approaches, our method imposes no symmetry assumptions, such as translational invariance, on the quantum systems. This technique is scalable to large systems through the use of a matrix product operator representation of the reference time evolution propagator. Our optimization routine is applied to various spin chains and fermionic systems described by the transverse-field Ising Hamiltonian, the Heisenberg Hamiltonian, and the spinful Fermi-Hubbard Hamiltonian. In these cases, our approach achieves a relative error improvement of up to four orders of magnitude for systems of 50 qubits, although our method is also applicable to larger systems. Furthermore, we demonstrate the versatility of our method by applying it to molecular systems, specifically lithium hydride, achieving an error improvement of up to eight orders of magnitude. This proof of concept highlights the potential of our approach for broader applications in quantum simulations.

1 Introduction

Simulating the time evolution of a quantum system was the original motivation behind quantum computers and remains one of their most natural applications [1–3], particularly in quantum chemistry and quantum many-body physics [4–6]. A widely used method for this purpose is Trotterization [7, 8], where the time propagator $U_t = e^{-iHt}$ is approximated using lower-body operators. This approach is straightforward to implement and comes in various orders, each with different circuit depths and error scalings [8, 9]. However, achieving high accuracy for long-time simulations necessitates deep quantum circuits. This raises some interesting questions:

- Can the same quantum circuit layout provided by Trotterization achieve better accuracy?
- And is it possible to optimize the quantum gates using only classical computation so that expensive quantum resources are only needed for the (optimized) Hamiltonian simulation?

These questions have been previously explored by (i) considering specific parameterizations of the quantum gates [10–16], and (ii) directly interpreting the quantum gates as unitary matrices in a brickwall circuit [17–20]. In this work, we focus on the latter approach and thereby circumvent possible dependencies of the optimization on the chosen parameterization.

In Ref. [17], Riemannian optimization and tensor network methods are combined to simultaneously optimize all quantum gates under the constraint of unitarity for small translationally-invariant quan-

Isabel Nha Minh Le: isabel.le@tum.de
Christian B. Mendl: christian.mendl@tum.de

tum systems using an explicit full-rank matrix reference of U_t . Due to the translational invariance, the same optimized quantum gates can simulate the quantum dynamics of corresponding larger systems. In contrast, Refs. [18, 19] employ a local quantum gate update inspired by tensor network methods to directly optimize larger quantum systems. To this end, they approximate U_t using a fine Trotterization for a suitable time step t , allowing it to be expressed efficiently as a matrix product operator (MPO). The classically optimized quantum circuit can then repeatedly be executed on quantum hardware to enable the simulation of quantum dynamics for longer times $t' \gg t$.

This work introduces a novel alternative quantum circuit optimization approach by integrating the ideas and methods of Refs. [17–19]. Specifically, we focus on quantum systems where a Trotter step can be represented as a brickwall circuit and introduce a simultaneous-update optimization method to enhance quantum gates under the unitarity constraint. To achieve this, we adapt the gradient-based ADAM optimizer [21] to the framework of Riemannian optimization on the complex Stiefel manifold, i.e., the manifold of unitary matrices. We approximate the time evolution reference operator U_t to target larger quantum systems by employing a higher-order Trotterization MPO and evaluate the cost function and corresponding gradient utilizing tensor network methods. Our Riemannian optimization approach updates the quantum gates of the quantum circuit simultaneously in contrast to the local gate updates used in Refs. [18, 19]. This also leads to a potential promotion to second-order Riemannian optimization – something that is not possible in the context of local gate updates. Furthermore, the presented approach does not assume any symmetry invariance of the quantum system to be simulated as in Ref. [17].

While numerous scalable methods for quantum circuit compression have been proposed [10–20], our approach distinguishes itself by framing the problem within the context of differential geometry. This geometric perspective not only facilitates efficient optimization but also potentially enables access to intrinsic, hardware-independent measures of circuit complexity [22, 23]. Notably, the shortest

path between unitaries under a Riemannian metric can be defined via the geodesic equation, offering a principled benchmark for circuit optimality. Such geometric tools can also be used in potential alternative optimization strategies [24]. Importantly, two circuits may yield comparable fidelity yet differ substantially in their geometric complexity – a distinction our Riemannian framework could, in principle, capture, unlike methods that do not account for the manifold structure of the unitary group.

We apply the presented method to exemplary spin chains and fermionic systems. Specifically, we conduct numerical experiments on systems of 50 qubits for a non-integrable case of the Ising Hamiltonian and a Heisenberg model, explicitly considering non-translationally invariant systems.

For fermionic systems, we utilize the fermionic swap network [25] and optimize a one-dimensional spinful Fermi-Hubbard model of 50 spin orbitals, as well as a paradigmatic example of lithium hydride (LiH) for the molecular Hamiltonian.

Related research. Concurrently with our research, two independent studies have been published that combine the Riemannian ADAM optimizer, automatic differentiation, and tensor network methods to optimize a brickwall circuit [26, 27]. The first study optimizes a quantum state represented as a matrix product state, while we directly optimize the time evolution operator, allowing for an arbitrary choice of initial quantum state. The second work explicitly compiles the brickwall circuit into single-qubit and CNOT gates. In contrast, we view the compilation as an excluded step and do not specify a fixed native gate set. Furthermore, we analytically implement the required gradient without relying on additional software packages for automatic differentiation.

The remaining paper is structured as follows: We first introduce the optimization problem in Section 2, by presenting the quantum circuit layout, defining the cost function and its gradient, formulating the underlying optimization problem, and introducing the utilized initialization methods. We then give a brief recap on Riemannian optimization and adapt the ADAM optimizer to the relevant case of the complex Stiefel manifold in Section 3.

Furthermore, we present the tensor network methods utilized to efficiently represent the unitary reference and to evaluate the cost function and its gradient in Section 4. We additionally point out how these methods can be used within an alternative parameterization-based approach. In Section 5, we show the results of the conducted numerical simulations for various (i) spin chains, and (ii) fermionic systems. Finally, we give a conclusion and outlook.

2 Optimization problem

Quantum circuit layout. We consider a so-called brickwall circuit layout built from L layers of two-qubit gates, where the ℓ -th layer of the quantum circuit W^ℓ has n_ℓ gates. We denote the i -th gate in layer W^ℓ as G_i^ℓ and the concatenation of layers i to j as $W^{i:j}$, i.e.,

$$W^\ell = \bigotimes_{i=1}^{n_\ell} G_i^\ell \text{ and } W^{i:j} = \prod_{\ell=i}^j W^\ell.$$

Finally, we call the overall brickwall circuit $W = W^{1:L}$.

Cost function. A common way [17, 18] to quantify how well a quantum circuit W of N qubits approximates a reference unitary U_{ref} is given by the normalized Frobenius norm:

$$\begin{aligned} \mathcal{C}_F(U_{\text{ref}}, W) &= \frac{1}{2d} \|U_{\text{ref}} - W\|_F^2 \\ &= \frac{1}{2d} \text{Tr} \left([U_{\text{ref}} - W]^\dagger [U_{\text{ref}} - W] \right) \\ &= 1 - \frac{1}{d} \text{Re} \text{Tr} \left(U_{\text{ref}}^\dagger W \right), \end{aligned}$$

where $d = 2^N$. The trace $\text{Tr}(U_{\text{ref}}^\dagger W)$ is the overlap between the reference unitary and a brickwall circuit and plays a crucial role in the presented method. Using the tensor networks notation, this

overlap can be diagrammatically expressed as

$$\mathcal{T}(W) = \text{Tr}(U_{\text{ref}}^\dagger W) \quad (1)$$

The Frobenius norm is sensitive to global phases, rendering it more restrictive than necessary for many practical applications. In scenarios where the focus lies on the time evolution of a specific observable, global phases have no physical consequence. In case the global phase is of importance, its value can easily be extracted and corrected within a postprocessing step. Consequently, relaxing the requirement for the quantum circuit to determine a fixed global phase can simplify the optimization landscape. Within our study, we make use of a less stringent cost function allowing for different global phases, namely the Hilbert-Schmidt test [28] given by

$$\mathcal{C}_{HS}(U_{\text{ref}}, W) = 1 - \frac{1}{d^2} |\mathcal{T}(W)|^2. \quad (2)$$

$\mathcal{C}_{HS}$ is directly related to the fidelity between two states $U_{\text{ref}}|\psi\rangle$ and $W|\psi\rangle$ averaged over the Haar distribution [28], and hence quantifies how close the unitaries U_{ref} and W are (up to a global phase). The two cost functions $\mathcal{C}_F$ and $\mathcal{C}_{HS}$ are related via the following inequality:

$$\begin{aligned} \mathcal{C}_{HS}(U_{\text{ref}}, W) &= 1 - \frac{1}{d^2} \left([\text{Re} \mathcal{T}(W)]^2 - [\text{Im} \mathcal{T}(W)]^2 \right) \\ &\leq 1 - [1 - \mathcal{C}_F(U_{\text{ref}}, W)]^2 \\ &\leq 2\mathcal{C}_F(U_{\text{ref}}, W). \end{aligned}$$

For simplicity, we denote $\mathcal{C} = \mathcal{C}_{HS}$, and its variational part is given by

$$\begin{aligned} \mathcal{L}(U_{\text{ref}}, W) &= -|\mathcal{T}(W)|^2 \\ &= -[\text{Re} \mathcal{T}(W)]^2 - [\text{Im} \mathcal{T}(W)]^2. \end{aligned} \quad (3)$$

Formulation of optimization problem. The objective is to optimize a set of unitary two-qubit gates $G = \{G_i\}_{i \in \text{ind}(W)}$ such that the resulting optimized brickwall circuit W minimizes its distance to the reference U_{ref} . The underlying optimization problem can hence be expressed as

$$G_{\text{opt}} = \arg \min_{G \in \mathcal{U}(4)^{\times N}} \mathcal{C}(U_{\text{ref}}, W). \quad (4)$$

In the latter, $\mathcal{U}(4)$ denotes the set of unitary 4×4 -matrices:

$$\mathcal{U}(4) = \{G_g \in \mathbb{C}^{4 \times 4} | G_g^\dagger G_g = \mathbb{1}_4\}, \quad (5)$$

which can be viewed as a particular case of the complex Stiefel manifold.

Euclidean gradient of the cost function.

The presented optimization problem shall be solved by means of gradient-based optimization and therefore, computing derivatives of the cost function is essential. While we will later see that the Euclidean gradient alone is not suitable for our Riemannian optimization method, in this part of the paper, we first present the relevant Euclidean gradient. A detailed calculation is given in Appendix A.

The gradient of $\mathcal{C}$ and $\mathcal{L}$ are equal up to a constant factor, and hence the quantity of interest is given by $\nabla \mathcal{L}$. By using the Wirtinger formalism [29], the partial derivatives with respect to an entry of W are given by

$$\partial_{W_{jk}} \text{Re } \mathcal{T}(W) = \frac{U_{jk}^*}{2} \quad \text{and} \quad \partial_{W_{jk}} \text{Im } \mathcal{T}(W) = \frac{U_{jk}^*}{2i}.$$

The partial derivative of $\mathcal{L}$ with respect to W_{jk} can then be computed as

$$\partial_{W_{jk}} \mathcal{L} = -[\mathcal{T}(W)U_{jk}]^*.$$

By applying the chain rule of the Wirtinger formalism, the partial derivatives of $\mathcal{L}$ with respect to a specific gate are given by

$$\partial_{G_i^\ell} \mathcal{L} = -\mathcal{T}(W)^* \cdot \partial_{G_i^\ell} \mathcal{T}(W),$$

where we denote the number of layers in the brickwall circuit with L and the number of gates in layer

ℓ with n_ℓ . Finally, the overall gradient can be obtained as

$$\nabla \mathcal{L} = -2\mathcal{T}(W) \cdot \partial_{G_i^\ell} [\mathcal{T}(W)]_{\substack{\ell=1, \dots, L \\ i=1, \dots, n_\ell}}^* \cdot \quad (6)$$

In order to compute $\nabla \mathcal{L}$, it is necessary to evaluate both $\mathcal{T}(W)$ and $\partial_{G_i^\ell} \mathcal{T}(W)$. Conveniently, by computing $\mathcal{T}(W)$, the cost function $\mathcal{C}$ given in Eq. (2) can be directly extracted.

Quantum circuit initialization.

Previous analyses have shown that a good initialization can significantly enhance quantum circuit optimization [17–19]. If the brickwall circuit to be optimized has L layers, the initialization is chosen as follows:

- We first determine if and, if yes, which orders of (repeated) Trotterization match the number of layers L . If several Trotter circuits of different orders match L , we choose the one that yields the best approximation error. We consider Trotterizations of orders two and four.
- If the brickwall circuit does not match any (repeated) Trotterization, we test whether it can be initialized as the concatenation of a second-order and a fourth-order Trotter circuit. To this end, the Trotter time steps of both Trotterizations, t_1 and t_2 , have to be chosen, such that $t = t_1 + t_2$ for the target simulation time t . We choose t_1 and t_2 such that they minimize the approximation error.
- Lastly, we also allow for extending the above initialization circuits by layers consisting of identity gates to be able to initialize any brickwall circuit of arbitrary L .

3 Riemannian optimization on the complex Stiefel manifold

As described above, the goal is to optimize a set of two-qubit gates under the constraint of unitarity, which means working on the complex Stiefel manifold. One effective method for such optimization is the gradient-based Riemannian optimization, which does not necessitate any specific parameterization of the unitary gates. Recently, this approach

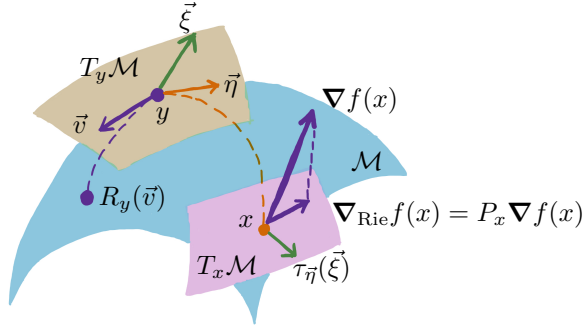


Figure 1: Visualization of concepts on the Riemannian manifold $\mathcal{M}$. A point $y \in \mathcal{M}$ is “moved” in the direction of the vector $\vec{v} \in T_y \mathcal{M}$ via the retraction R . A vector $\vec{\xi}$ is moved in the direction of a vector $\vec{\eta} \in T_y \mathcal{M}$ to the tangent space $T_x \mathcal{M}$ in point $x \in \mathcal{M}$ using the vector transport τ . Lastly, the Riemannian gradient $\nabla_{\text{Rie}} f(x)$ can be obtained as the projected Euclidean gradient $\nabla f(x)$ through the projector P .

has been explicitly utilized for various optimization tasks in quantum technologies [17, 26, 27, 30–37]. In the following, we will provide a brief overview of the basic concepts required for Riemannian optimization of brickwall circuits. For a more detailed introduction to these concepts, we refer interested readers to Refs. [38–40]. Furthermore, we adapt the well-known ADAM optimizer [21] to the setting of Riemannian optimization.

3.1 Concepts of Riemannian geometry

Intuitively, a manifold $\mathcal{M}$ can be thought of as a curved space that, at any point, $x \in \mathcal{M}$, can be locally described by a Euclidean space, which is isomorphic to a vector space called the *tangent space* $T_x \mathcal{M}$. The collection of all points in the manifold and the corresponding tangent spaces, $(x, T_x \mathcal{M})$, is called the *tangent bundle* $T\mathcal{M}$. If each tangent space is equipped with an inner product $\langle \cdot, \cdot \rangle_x : T_x \mathcal{M} \rightarrow T_x \mathcal{M}$, $\mathcal{M}$ and $\langle \cdot, \cdot \rangle$ form a *Riemannian manifold*. When naively moving a point $x \in \mathcal{M}$ along the direction of a vector $\vec{v}_x \in T_x \mathcal{M}$, the resulting point will, in general, lie in $T_x \mathcal{M}$ and not be an element of $\mathcal{M}$ anymore. A *retraction*

$$R : T\mathcal{M} \rightarrow \mathcal{M}, R_x(\vec{v}_x) = y$$

is a smooth map from the tangent bundle into the manifold, and hence $R_x(v_x)$ corresponds to moving

a point x in the direction of v_x within the manifold. Moreover, a so-called *vector transport*

$$\tau_{\vec{\eta}_x} : T_x \mathcal{M} \rightarrow T_y \mathcal{M}, \tau_{\vec{\eta}_x}(\vec{v}_x) = \vec{v}_y$$

is a smooth map that moves a vector $\vec{v}_x \in T_x \mathcal{M}$ to another tangent space $T_y \mathcal{M}$ in point $y \in \mathcal{M}$, where $\vec{\eta}_x \in T_x \mathcal{M}$ is the direction of transport. Consider a Riemannian manifold $\mathcal{M}$ and a function $f : \mathcal{M}^{\times n} \rightarrow \mathbb{R}$ with inputs $x = \{x_i\}_{i=1}^n$ from a product manifold. The so-called *Riemannian gradient* can be obtained from the Euclidean gradient $\nabla f(x)$ by projecting the latter entry-wise onto $T_x \mathcal{M}^{\times n}$ via some suitable projection P :

$$\nabla_{\text{Rie}} f(x) = P_x \nabla f(x) \in T_x \mathcal{M}^{\times n}.$$

An overview of these concepts is visualized in Fig. 1.

3.2 Example of the complex Stiefel manifold

As already mentioned, $\mathcal{U}$ defined in Eq. (5) is a manifold, and the tangent space in a point $G \in \mathcal{U}$ can be parameterized by a set of complex anti-Hermitian matrices

$$T_G \mathcal{U} = \{GA : A \in \mathbb{C}^{m \times m}, A^\dagger = -A\}.$$

Furthermore, a Riemannian metric can be defined on each tangent space $T_G \mathcal{U}$ by

$$\langle \cdot, \cdot \rangle : T_G \mathcal{U} \times T_G \mathcal{U} \rightarrow \mathbb{R}, \quad \langle X, Y \rangle_G = \text{Tr}(X^\dagger Y), \quad (7)$$

with which $\mathcal{U}$ can be identified as a Riemannian manifold and – more specifically – a particular case of the complex Stiefel manifold. Note that the cost function defined in Section 2 involves $\mathcal{T}(W) = \langle U_{\text{ref}}, W \rangle$. A proper *retraction* of $\mathcal{U}$ is induced by the polar decomposition $A = UM$ with U being a unitary matrix and M being a positive semi-definite Hermitian matrix. When denoting the corresponding unitary part as $\pi(A) = U$, a vector $\vec{\xi} \in T_G \mathcal{U}$ can be retracted into the unitary manifold via

$$R^{\text{polar}} : T\mathcal{U} \rightarrow \mathcal{U}, \quad R_G^{\text{polar}}(\vec{\xi}) = \pi(G + \vec{\xi}). \quad (8)$$

A general vector $\vec{\eta} \in \mathbb{C}^{m \times m}$ can be *projected onto the tangent space* $T_G \mathcal{U}$ by means of the linear pro-

jector

$$\begin{aligned} P_G(\vec{\eta}) &= \vec{\eta} - \frac{1}{2}G(\vec{\eta}^\dagger G + G^\dagger \vec{\eta}) \\ &= G \text{skew}(G^\dagger \vec{\eta}). \end{aligned} \quad (9)$$

Utilizing Eq. (9), the map

$$\tau_{\vec{\eta}}(\vec{\xi}) = P_{R_G^{\text{polar}}(\vec{\eta})}(\vec{\xi}) \quad (10)$$

is a valid *vector transport on $\mathcal{U}$* . In other words, a vector $\vec{\xi} \in T_G \mathcal{M}$ in the tangent space corresponding to $G \in \mathcal{M}$ is transported in the direction of $\vec{\eta} \in T_G \mathcal{M}$ by projecting it onto the tangent space corresponding to the retracted directional vector $R_G^{\text{polar}}(\vec{\eta}) \in \mathcal{U}$ using Eq. (9).

When optimizing a brickwall circuit W consisting of $|W|$ two-qubit gates $\{G_i\}_{i \in \text{ind}(W)}$, the cost function f is defined on the product manifold $\mathcal{U}^{\times |W|}$

$$f : \mathcal{U}^{\times |W|} \rightarrow \mathbb{R}.$$

The corresponding tangent space is given by the direct sum of the individual tangent spaces

$$T_W \mathcal{U}^{\times |W|} = \bigoplus_{i \in \text{ind}(W)} T_{G_i}(\mathcal{U}).$$

Using the same projector as given in Eq. (9) the corresponding *Riemannian gradient* can be computed via

$$\nabla_{\text{Rie}} f(W) = P_W \nabla f(W). \quad (11)$$

Lastly, the overall retraction corresponds to applying the retraction in Eq. (8) to the individual quantum gates and tangent vectors.

3.3 Adapting the ADAM optimizer to the Riemannian setting of the complex Stiefel manifold

In the following, we make use of the previously introduced concepts to adapt the first-order gradient-based ADAM optimizer [21] to the Riemannian optimization setting on the manifold of unitary operations [41]. ADAM aims to minimize an objective function f via the component-wise update rule

$$x_{t+1} \leftarrow x_t - \alpha m_t / \sqrt{v_t},$$

where $\alpha \in \mathbb{R}$ is the learning rate, and m_t and v_t are the first and second momenta in each iteration t , which can be obtained from the gradient ∇f as

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla f,$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) (\nabla f)^2,$$

where usually $\beta_1 = 0.9$ and $\beta_2 = 0.99$ are chosen.

Using the Riemannian gradient. To constrain ADAM to the unitary manifold, firstly, ∇f has to be replaced by $\nabla_{\text{Rie}} f$, from which follows that v_t is computed from the inner product $\langle \nabla_{\text{Rie}} f, \nabla_{\text{Rie}} f \rangle_{x_t}$. Additionally, some further modifications need to be made.

Vector transport of momentum. m_t is computed by adding the scaled Riemannian gradient $\nabla_{\text{Rie}} f \in T_{x_t} \mathcal{M}$ to the scaled previous momentum $m_{t-1} \in T_{x_{t-1}} \mathcal{U}$, i.e., two quantities from different tangent spaces. Therefore, it is necessary to transport m_{t-1} to the tangent space of the current point. The modified Riemannian momenta updates are then given by

$$\tilde{m}_t = \beta_1 \tau_{\tilde{m}_t}(\tilde{m}_{t-1}) + (1 - \beta_1) \nabla_{\text{Rie}} f,$$

$$\tilde{v}_t = \beta_2 \tilde{v}_{t-1} + (1 - \beta_2) \langle \nabla_{\text{Rie}} f, \nabla_{\text{Rie}} f \rangle_{x_t}.$$

Retraction of updated step. Since the Riemannian momenta determine the update direction, the updated point x_{t+1} will lie in the tangent space of the former point x_t . Therefore, the retraction given in Eq. (8) must be applied to obtain valid new parameters within $\mathcal{M}$. The modified update rule is hence given by

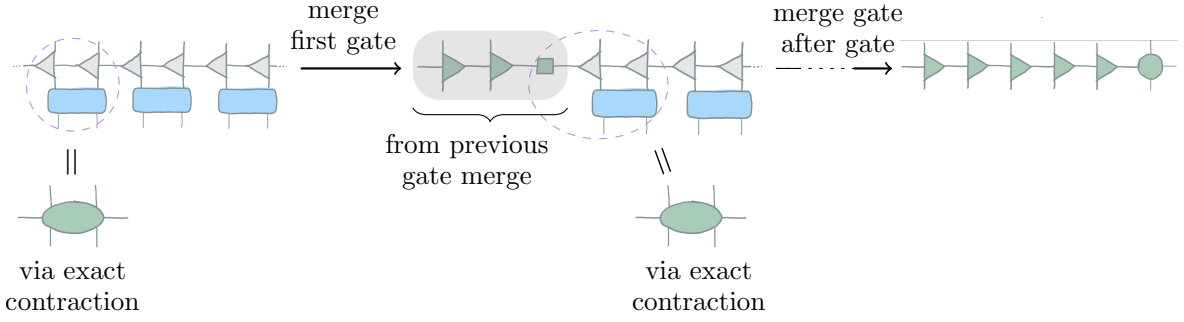
$$\tilde{x}_{t+1} \leftarrow R_{\tilde{x}_t}^{\text{polar}} \left(\tilde{x}_t - \alpha \tilde{m}_t / \sqrt{\tilde{v}_t} \right) \in \mathcal{M}.$$

Note that the ADAM optimizer and its Riemannian adaption are invariant under gradient rescaling.

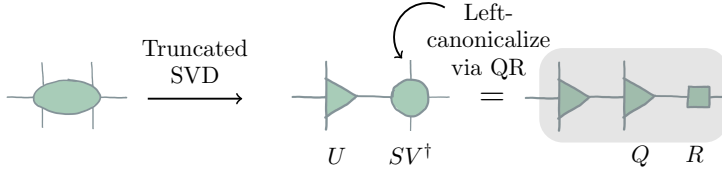
4 Tensor networks methods for quantum circuit optimization

4.1 Merging a brickwall layer of two-qubit gates into a matrix product operator

A fundamental computational step in previous and the presented methods involves merging a brick-



(a) A brickwall layer can be merged into a right-canonicalized MPO via a left-to-right-sweep. The canonical center of the MPO is shifted accordingly during the left-to-right-sweep. The final MPO is right canonicalized.

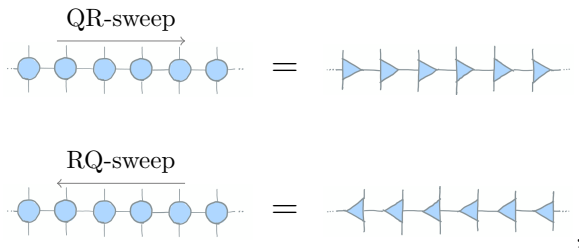


(b) How to merge a two-qubit gate into the corresponding local MPO tensors.

Figure 2: Illustration of how to merge a brickwall layer into an MPO in a left-to-right-sweep using the tensor network framework. Similarly, a right-to-left-sweep can be used, which is not explicitly illustrated here.

wall layer of two-qubit gates into an MPO. This operation arises in key contexts such as approximating the time-evolution operator U_t as an MPO and computing both the cost function and its gradient. Given its central role, we now describe this procedure in more detail.

Recall that any MPO can be brought into left- or right-canonical form via a sweep of QR or RQ decompositions, respectively – performed from left to right or right to left across the MPO tensors, i.e.,



where $\blacktriangleright$ denotes a left-isometry and $\blacktriangleleft$ denotes a right-isometry.

The objective is to merge a brickwall layer of two-qubit gates into an MPO while simultaneously truncating the bond dimension to a maximum of $\chi_{\max}$. The procedure either begins with a right- or left-canonicalized MPO, where the canonical cen-

ter is located at the leftmost or rightmost tensor, respectively. For simplicity, we will only describe the method for an initial right-canonicalized MPO. The two-qubit gates are then sequentially merged from left to right as outlined by the following steps:

- The considered two-qubit gate is exactly contracted with the corresponding pair of local MPO tensors and some residual tensor R obtained from the previous quantum gate merge. If the considered quantum gate corresponds to the leftmost gate, R is ignored. This yields an intermediate higher-rank tensor K .
- The tensor K is then decomposed via a singular value decomposition (SVD), $K = USV^\dagger$. To optimally truncate the bond dimension, only the largest $\chi_{\max}$ singular values are retained. The updated left tensor is set to $K_1 = U$, which is left-isometric by construction, while the updated right tensor is given by $K_2 = \tilde{S}V^\dagger$, where $\tilde{S}$ denotes the truncated diagonal matrix of singular values.
- To restore a mixed-canonical form, the left-isometric component of K_2 is extracted via a QR decomposition, $K_2 = QR$, where Q is the new left-canonical tensor.

- The residual tensor R is then contracted with the next two-qubit gate and the subsequent pair of local MPO tensors. This effectively shifts the canonical center to the next site pair to be merged.

This procedure is applied sequentially to each two-qubit gate in the brickwall layer. After completing a full left-to-right sweep, all gates have been incorporated into the MPO, resulting in an updated MPO in left-canonical form. The subsequent brickwall layer can then be merged analogously via a right-to-left sweep. An illustration of the left-to-right merging process is provided in Fig. 2.

4.2 Matrix product operator representation of the reference time evolution operator

To solve the optimization problem formulated in Eq. (4), a reference operator representing the time evolution propagator $U_t = e^{-iHt}$ is required. While U_t can be computed for small systems via numerically exact diagonalization, this approach is infeasible for larger systems of interest due to the curse of dimensionality. Therefore, we choose to approximate U_t to a reasonable accuracy and efficiently express it as an MPO with a maximum bond dimension $\chi_{\max}$ [18, 19].

Computing the reference matrix product operator. For the optimization, it is essential to have an efficient and accurate representation of the time evolution operator $U_t = e^{-iHt}$ as a reference, which we will denote as U_{ref} . Depending on the number of qubits N , the references are generated in two different ways:

- For system sizes of $N \leq 12$, we can compute U_t numerically exactly as a full-rank matrix and decompose and truncate it to an MPO.
- For larger systems, we obtain an accurate approximation of U_t using n repetitions of a fourth-order Trotterization $U_{\text{Trotter}}^{\text{IV},n}$. For this purpose, n is chosen such that $U_{\text{Trotter}}^{\text{IV},n}$ has a negligible approximation error for the systems considered in this work. We then treat the resulting deep quantum circuit as a tensor network and contract it into an MPO [18, 19]. To

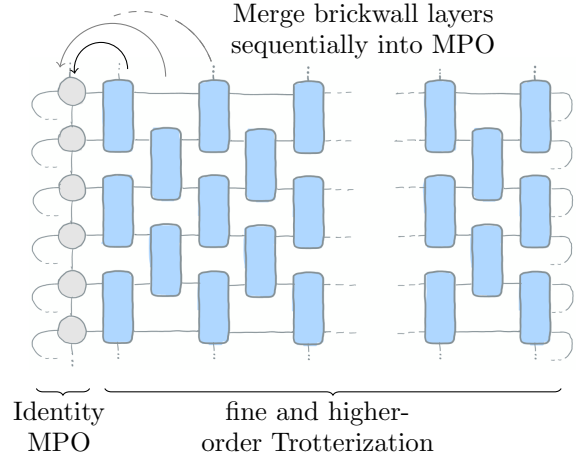


Figure 3: How to obtain the reference MPO via a suitable fine higher-order Trotterization for system sizes of $N \geq 12$. The corresponding deep Trotter circuit is interpreted as a tensor network, and its brickwall layers are sequentially merged into an initial identity MPO.

this end, we start with an initial identity MPO and sequentially merge the layers of $U_{\text{Trotter}}^{\text{IV},n}$ into it as explained in Fig. 3. The procedure of merging a brickwall layer into an MPO follows the same strategy discussed in Section 4.1.

In both cases, the bond dimensions of the MPO are truncated to some maximum value $\chi_{\max}$ [42], and we denote the resulting MPO by $U_{\chi_{\max}}$ to indicate this property. In practice, the accuracy of U_{ref} does not have to be arbitrarily high. Suppose the final error between the optimized quantum circuit and the reference is $\mathcal{C}_{\text{final}}$. In that case, it is sufficient to have a reference operator U_{ref} that approximates the exact time evolution U_t by

$$\mathcal{C}(U_t, U_{\text{ref}}) \approx \frac{\mathcal{C}_{\text{final}}}{10} = \epsilon_{\text{thres}}.$$

To further relax computational demands, a reasonable target error $\mathcal{C}_{\text{final}}$ is evaluated, and $U_{\chi_{\max}}$ is compressed to the smallest bond dimension $\chi \leq \chi_{\max}$ [19] for which it still holds that

$$\mathcal{C}(U_{\chi_{\max}}, U_{\chi}) \leq \epsilon_{\text{thres}}.$$

The compression is achieved by applying a singular value decomposition (SVD) sweep on $U_{\chi_{\max}}$ in mixed-canonical form, where the canonical center is always moved to the local tensor that is currently truncated [42].

Quantifying the reference approximation error. Several error sources lead to a deviation of U_{ref} from U_t .

- Trotterization error $\varepsilon_{\text{Trot}}$ when approximating U_t by $U_{\text{Trotter}}^{\text{IV},n}$ for systems with $N > 12$ qubits.
- Truncation error $\varepsilon_{\text{trunc}}$ when computing $U_{\chi_{\text{max}}}$.
- Compression error $\varepsilon_{\text{comp}}$ when compressing $U_{\chi_{\text{max}}}$ to U_{χ} , which is given by the chosen ϵ_{thres} .

To estimate the approximation error $\varepsilon_{\text{Trot}}$, we note that for local Hamiltonians, the error is expected to scale linearly in system size. Hence, $\varepsilon_{\text{Trot}}$ for large system sizes can be estimated from extrapolating the approximation errors of a Trotterization for a considered simulation time t for small enough systems, for which the exact full-rank matrix U_t can still be computed. In order to quantify the truncation error $\varepsilon_{\text{trunc}}$ of $U_{\chi_{\text{max}}}$, a series of MPOs $\{U_{\chi}\}_{\chi > \chi_{\text{max}}}$ is computed as described before. First, it is verified that increasing the bond dimension does not change the MPO significantly:

$$\mathcal{C}(U_{\chi}, U_{\chi+1}) < 10^{-10}.$$

The $\varepsilon_{\text{trunc}}$ of $U_{\chi_{\text{max}}}$ is then given by $\mathcal{C}(U_{\chi_{\text{max}}}, U_{\chi_{\text{max}}+1})$. Lastly, the compression error $\varepsilon_{\text{comp}}$ is determined by the choice of ϵ_{thres} . This study compares the optimization results with Trotterizations up to the fourth order. Hence, $\varepsilon_{\text{comp}}$ is chosen such that the reference MPO is accurate enough to reproduce the error scaling behavior of these Trotter circuits. We will accordingly document the error estimations of the U_{ref} used in the numerical simulations.

4.3 Tensor network methods for cost function and gradient evaluation

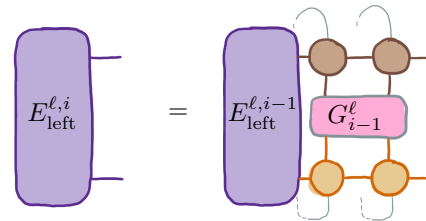
In the following, we present the tensor network methods used to evaluate both the gradient and the cost function presented in Section 2.

Computing the cost function in the context of tensor networks. When interpreting

the brickwall circuit as a tensor network, the partial derivatives $\partial_{G_i^{\ell}} \mathcal{T}(W)$ can be obtained by “cutting out” the respective quantum gate G_i^{ℓ} from Eq. (1). This is visualized in Fig. 4. The cost function is given by Eq. (2) with the overlap $\mathcal{T}(W)$ being representable by a tensor network contraction as given in Eq. (1). Given any partial derivative $\partial_{G_i^{\ell}} \mathcal{T}(W)$, $\mathcal{T}(W)$ can quickly be evaluated by contracting $\partial_{G_i^{\ell}} \mathcal{T}(W)$ with the corresponding “cut-out-gate” G_i^{ℓ} .

Computing the partial derivatives: Identifying tensor networks environments. To compute $\partial_{G_i^{\ell}} \mathcal{T}(W)$, we follow an approach similar to Ref. [19]. For the partial derivative $\partial_{G_i^{\ell}} \mathcal{T}(W)$ with respect to the i -th quantum gate in layer ℓ , we first compute a top environment MPO E_{top}^{ℓ} , which is obtained by merging all layers $\ell' = \ell + 1, \dots, L$ lying above the considered layer ℓ into the adjoint reference MPO $U_{\text{ref}}^{\dagger}$. Similarly, a bottom environment MPO E_{bottom}^{ℓ} is computed by merging all layers $\ell' = 1, \dots, \ell - 1$ below the considered layer ℓ . This leads to our layer ℓ being “sandwiched” by the two MPOs E_{top}^{ℓ} and E_{bottom}^{ℓ} as visualized in Fig. 4. To contract this resulting tensor network, two further environments are computed. The first results from contracting all tensors to the left of the considered gate G_i^{ℓ} , which we call the left environment $E_{\text{left}}^{\ell,i}$. Similarly, the right environment $E_{\text{right}}^{\ell,i}$ is obtained by contracting all tensors to the right of G_i^{ℓ} .

Computing the partial derivatives: Environment caching. For the efficient evaluation of all $\partial_{G_i^{\ell}} \mathcal{T}(W)$, it is helpful to cache some of the environments. Note that for a fixed layer ℓ , all left environments $E_{\text{left}}^{\ell,i}$ (right environments $E_{\text{right}}^{\ell,i}$) can be recursively obtained by a left-to-right (right-to-left) sweep, since



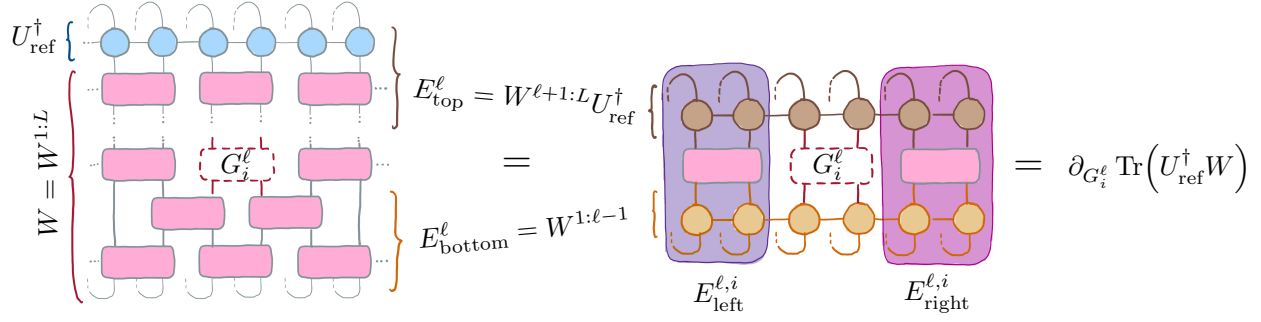
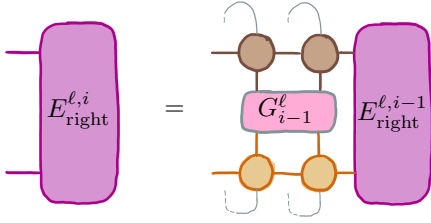


Figure 4: Diagrammatic sketch of $\partial_{G_i^\ell} \text{Tr}(U_{\text{ref}}^\dagger W)$, where G_i^ℓ is the i -th gate in the ℓ -th layer of the brickwall circuit. The partial derivative can be obtained by “cutting out” the considered gate G_i^ℓ . For the efficient evaluation, a top environment MPO E_{top}^ℓ and a bottom environment MPO E_{bottom}^ℓ are computed. From there, the left and right environment, $E_{\text{left}}^{\ell,i}$ and $E_{\text{right}}^{\ell,i}$, are evaluated. Finally, contracting the resulting tensor network can compute the partial derivative.

and



Using efficient tensor contractions, each evaluation of $E_{\text{left}}^{\ell,i}$ ($E_{\text{right}}^{\ell,i}$) has a cost that scales the most with $\mathcal{O}(D^3\chi^3)$, where in our case it is always $D = 2$. Since each two-qubit gate has to be merged once in order to compute all $E_{\text{left}}^{\ell,i}$ ($E_{\text{right}}^{\ell,i}$), the overall cost per gradient computation scales linearly in the number of gates $|W|$, i.e., $\mathcal{O}(|W|d^3\chi^3)$.

Furthermore, E_{top}^ℓ (E_{bottom}^ℓ) for all layers $\ell = 1, \dots, L$ in the brickwall circuit can be recursively computed by a top-to-bottom (bottom-to-top) sweep, since

$$E_{\text{top}}^{\ell-1} = E_{\text{top}}^\ell W^\ell$$

$$\Leftrightarrow$$

and

$$E_{\text{bottom}}^{\ell+1} = E_{\text{bottom}}^\ell W^\ell$$

$$\Leftrightarrow$$

with the initial environments $E_{\text{top}}^L = U_{\text{ref}}^\dagger$ and $E_{\text{bottom}}^1 = \text{id}_{\text{MPO}}$. To this end, a brickwall layer has to be merged into the previous environment MPO, which is achieved by the strategy discussed in Section 4.1. The contraction cost to merge a two-qubit gate with the corresponding local MPO tensors scales at most with $\mathcal{O}(D^5\chi^3)$. The resulting merged tensor needs to be split using a truncated SVD in order to retain the MPO form and a restricted bond dimension. Such an SVD contributes a cost of $\mathcal{O}([D^2\chi]^3)$. Since each gate has to be merged and split once into an MPO in order to compute all E_{bottom}^ℓ (E_{top}^ℓ), the overall cost within each gradient evaluation scales linearly in the number of gates $|W|$, i.e., $\mathcal{O}(|W|D^6\chi^3)$.

Full computation of the gradient $\nabla\mathcal{L}$. In practice, the computation of $\nabla\mathcal{L}$ consists of

- (i) Compute all partial derivatives $\partial_{G_i^\ell} \mathcal{T}(W)$.
- (ii) Compute the overlap $\mathcal{T}(W)$ from one of the partial derivatives.
- (iii) Compute $\nabla\mathcal{L}$ via Eq. (6).

Computing all $\partial_{G_i^\ell} \mathcal{T}(W)$ in step (i) can be achieved by the following strategy:

1. Perform a top-to-bottom sweep to compute and store all E_{top}^ℓ .
2. Perform a bottom-to-top sweep and consecutively compute all partial derivatives with respect to the gates in the layers $\ell = 1, \dots, L$.

- (a) At each layer ℓ , first compute E_{bottom}^ℓ from the previous $E_{\text{bottom}}^{\ell-1}$. Only the current and previous bottom environments must be stored during this sweep.
- (b) Perform a right-to-left sweep to compute and cache $E_{\text{right}}^{\ell,i}$ for all $i = n_\ell, \dots, 1$.
- (c) Perform a left-to-right-sweep to compute each $\partial_{G_i^\ell} \mathcal{T}(W)$ by first computing $E_{\text{left}}^{\ell,i}$ from the previous $E_{\text{left}}^{\ell,i-1}$ and then contracting the resulting tensor network. Only the previous and current left environments have to be stored in this sweep.

The last contraction in step (c) has a cost that scales at most with $\mathcal{O}(D^3\chi^3)$. Since this contraction has to be done for each two-qubit gate, the overall cost to evaluate the full gradient coming from this step scales with $\mathcal{O}(|W|D^3\chi^3)$. Taking into account the intermediate contractions to evaluate all environments, the cost to compute the gradient scales at most with $\mathcal{O}(|W|D^6\chi^3)$. An overview of the contraction costs per gradient evaluation is given in Table 1.

Table 1: Overview of contributing contraction costs per gradient evaluation with respect to the number of two-qubit gates $|W|$, the maximum bond dimension χ and the qubit dimension $D = 2$.

$E_{\text{top}}/E_{\text{bottom}}$	$\mathcal{O}(W D^6\chi^3)$
$E_{\text{left}}/E_{\text{right}}$	$\mathcal{O}(W D^3\chi^3)$
Final contraction	$\mathcal{O}(W D^3\chi^3)$

The procedure of computing one partial derivative is visualized for a specific gate G_i^ℓ in Fig. 4. The Riemannian gradient $\nabla_{\text{Rie}} \mathcal{L}$ is then obtained through the projection given in Eq. (9) as discussed in Section 3.

4.4 Enabling gradient-based parameter optimization

As discussed in the introduction, an alternative strategy for quantum circuit optimization involves parameterizing quantum gates [10–15]. While a geometric-aware Riemannian optimization approach is expected to perform better especially on difficult loss landscapes, parameterization-based

techniques are particularly effective when adapting to native hardware gate sets. For instance, certain quantum hardware platforms natively support two-qubit gates of the form

$$V(a, b, c) = e^{i(aXX+bYY+cZZ)}.$$

In such cases, it is advantageous to directly optimize the Weyl decomposition [43]:

$$G = (K_1 \otimes K_2) \cdot V(a, b, c) \cdot (K_3 \otimes K_4),$$

where each $K_i \in SU(2)$ and can be expressed as a sequence of single-qubit rotations, e.g.,

$$K_i = RZ(\theta_i) RY(\psi_i) RZ(\phi_i).$$

This yields a 15-parameter representation of each two-qubit gate:

$$\{\alpha_i\}_{i=1}^{15} = \{a, b, c\} \cup \{\theta_i, \psi_i, \phi_i\}_{i=1}^4.$$

Our scalable tensor network-based method for the evaluation of Eq. (6) presented in Section 4.3 can readily be used in the parameterized context as well. For a gradient-based Euclidean optimization, the gradient with respect to the parameters $\{\alpha_i\}$ is required. The latter can be obtained by a simple chain rule:

$$\frac{\partial \mathcal{L}}{\partial \alpha_i} = \left\langle \frac{\partial \mathcal{L}}{\partial G}, \frac{\partial G}{\partial \alpha_i} \right\rangle, \quad (12)$$

which intuitively corresponds to replacing the “cut-out-gate” in the tensor network picture with the corresponding inner derivative $\frac{\partial G}{\partial \alpha_i}$. For completeness, we have listed $\frac{\partial G}{\partial \alpha_i}$ for the case of the Weyl decomposition in Appendix B.

5 Numerical simulations

In the following, we consider Hamiltonians of (i) spin chains, and (ii) fermionic systems. We use the common notation of $(\sigma_i^1, \sigma_i^2, \sigma_i^3) = (X_i, Y_i, Z_i)$ for the vector of Pauli operators acting on the i -th qubit. Moreover, we denote in the context of second quantization the creation (annihilation) operator acting on orbital p as $a_p^\dagger$ (a_p) and the corresponding number operator as $n_p = a_p^\dagger a_p$.

While our method can generally be applied to non-translationally invariant systems, we mainly

consider systems with translational invariance for simplicity. However, we explicitly demonstrate this aspect for 10 disordered transverse-field Ising models on a one-dimensional chain with 20 sites. Furthermore, we show the optimization results for non-disordered transverse-field Ising and Heisenberg models on a one-dimensional chain with 50 sites although our method is also applicable to larger systems. Regarding the fermionic systems, we first show the optimization results for a non-disordered spinful Fermi-Hubbard model on a one-dimensional chain for 50 spin orbitals. As a proof of concept, we then demonstrate that our method can be applied to molecular Hamiltonians using the exemplary system of LiH. The utilized error measure $\mathcal{C}$ is given by Eq. (2).

5.1 Spin systems

5.1.1 Transverse-field Ising model (non-disordered).

As a first example, we consider the transverse-field Ising model Hamiltonian on a chain with $N = 50$ sites and open boundary conditions:

$$H^{\text{Ising}} = \sum_{i=1}^{N-1} JZ_i Z_{i+1} + \sum_{i=1}^N (gX_i + hZ_i),$$

with $J, g, h \in \mathbb{R}$. In this study, we set without loss of generality $J = 1$ and $g = 0.75$, specifically considering the non-integrable case $h = 0.6$. We compute U_{ref} as 20 repetitions of a fourth-order Trotterization, $U_{\text{Trotter}}^{\text{IV},20}$, for a simulation time of $t = 2$, which is further compressed to a threshold of ϵ_{thres} . An overview of the reference error estimation is given in Appendix C Table 2.

The two-qubit Trotterization gates are given by

$$G^{\text{Ising}} = e^{-i\Delta t(JZZ + \frac{g_1}{2}XI + \frac{g_2}{2}IX + \frac{h_1}{2}ZI + \frac{h_2}{2}IZ)},$$

where

- $g = g_1 = g_2$ and $h = h_1 = h_2$ for gates in the middle of the circuit,
- $g_2 = g = \frac{g_1}{2}$ and $h_2 = h = \frac{h_1}{2}$ for gates that act on the first qubit, and
- $g_1 = g = \frac{g_2}{2}$ and $h_1 = h = \frac{h_2}{2}$ for gates that act on the last qubit

due to the open boundary conditions.

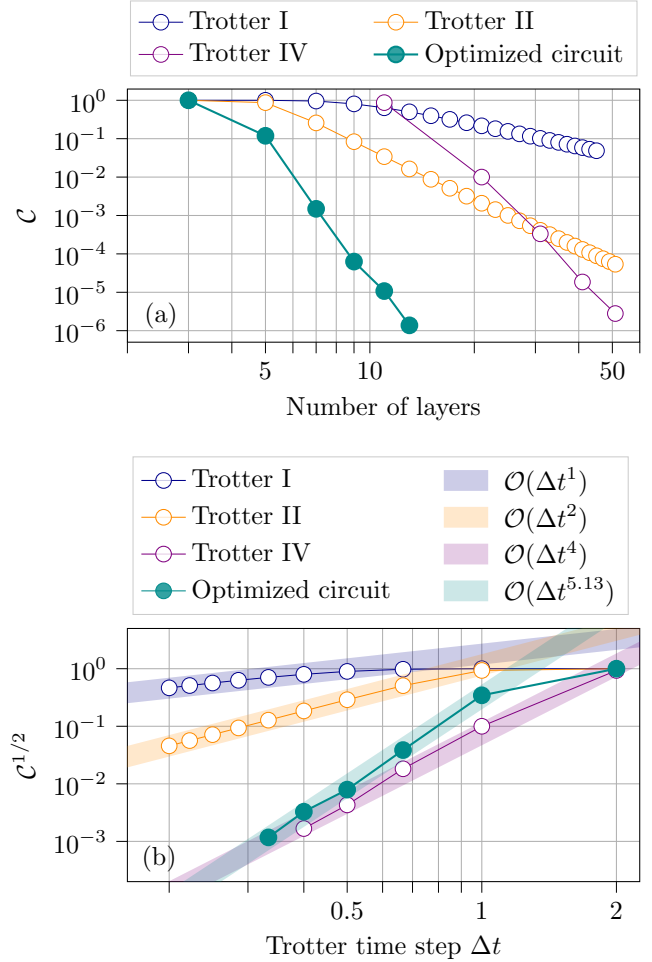


Figure 5: Results for the Ising model on a chain with $N = 50$ sites, $J = 1$, $g = 0.75$, $h = 0.6$, and $t = 2$. (a) The error of the Riemannian optimized brickwall circuit compared to that of different Trotter circuits. (b) The error scaling in Trotter time step Δt for the Riemannian optimized brickwall circuit compared to that of various Trotter circuits.

The quantum circuits that are optimized are initialized as Trotterizations of order two. The optimization results and their comparison to Trotterizations of various orders are shown in Fig. 5.

The Riemannian optimization method can increase the accuracy of each optimized quantum circuit with a relative error improvement, $\mathcal{C}_{\text{rel}} = \mathcal{C}_{\text{init}}/\mathcal{C}_{\text{opt}}$, of up to four orders of magnitude as shown in Fig. 5(a). For example, the brickwall circuit with 12 layers could be optimized to an accuracy of five repetitions of a fourth-order Trot-

terization, which would initially require 51 layers. Furthermore, since all considered initial quantum circuits correspond to a Trotterization of order two, we can investigate the error scaling behavior with respect to the discretized Trotter time step Δt . For naive Trotterization of k -th order, the Trotter error scales like $\mathcal{O}(\Delta t^k)$. For the optimized brickwall circuit, we observe a scaling behavior of $\mathcal{O}(\Delta t^{5.13})$, i.e., a better scaling behavior for $\Delta t \ll 1$ than a fourth-order Trotterization. This analysis is shown in Fig. 5(b).

5.1.2 Transverse-field Ising model (disordered).

Unlike previous Riemannian quantum circuit optimization methods [17], our new method can also be applied to systems without translational invariance. To demonstrate this aspect, we consider a disordered modification of the prior Ising model:

$$H^{\text{Ising,dis}} = \sum_{i=1}^{N-1} J_i Z_i Z_{i+1} + \sum_{i=1}^N (g_i X_i + h_i Z_i)$$

with $J_i, g_i, h_i \in \mathbb{R}$, where we uniformly sample $J_i \in [\frac{J}{2}, \frac{3J}{2}]$, $g_i \in [\frac{g}{2}, \frac{3g}{2}]$, $h_i \in [\frac{h}{2}, \frac{3h}{2}]$ and without loss of generality J, g, h are chosen as in the translationally invariant case. We generate 10 different disordered Ising models of $N = 20$ sites and optimize each. The results for this setting are visualized in Fig. 6, where the optimization result for the corresponding non-disordered system is additionally displayed for comparison. Our method can handle disordered systems and achieves optimization results similar to those in the previous non-disordered case.

5.1.3 Heisenberg model.

As a second spin system, we consider the Heisenberg Hamiltonian on a chain of $N = 50$ sites:

$$H^{\text{Heis}} = \sum_{i=1}^{N-1} \sum_{\alpha=1}^3 J^\alpha \sigma_i^\alpha \sigma_{i+1}^\alpha + \sum_{i=1}^N \sum_{\alpha=1}^3 h^\alpha \sigma_i^\alpha,$$

with $\vec{J}, \vec{h} \in \mathbb{R}^3$. In the numerical simulations, we consider the parameters $\vec{J} = (1, 1, -\frac{1}{2})$ and $\vec{h} = (\frac{3}{4}, 0, 0)$. We compute U_{ref} as 10 repetitions of a fourth-order Trotterization, $U_{\text{Trotter}}^{\text{IV},10}$, for a simulation time of $t = 0.5$, which is further compressed

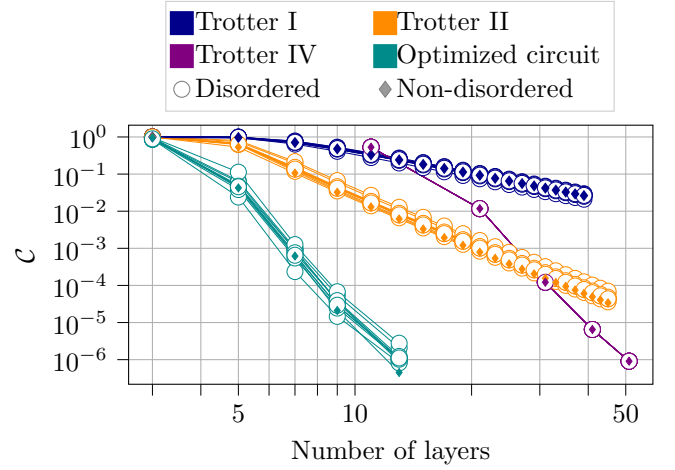


Figure 6: Results for the disordered Ising model on a one-dimensional chain of $N = 20$ sites. The corresponding non-disordered Ising model is plotted for reference for parameters $J = 1$, $g = 0.75$, $g = 0.6$, and $t = 2$. The parameters for the disordered models are uniformly sampled as $J_i \in [\frac{1}{2}J, \frac{3}{2}J]$, $g_i \in [\frac{1}{2}g, \frac{3}{2}g]$, and $h_i \in [\frac{1}{2}h, \frac{3}{2}h]$.

to a threshold of ϵ_{thres} . An overview of the reference error estimation is given in Appendix C Table 2. The two-qubit Trotterization gates in the considered case are given by

$$G^{\text{Heis}} = e^{-i\Delta t \left(\frac{h_1}{2} XI + \frac{h_2}{2} IX + \sum_{\alpha=1}^3 J^\alpha \sigma_i^\alpha \sigma_{i+1}^\alpha \right)},$$

where

- $h_1 = \frac{3}{4} = h_2$ for quantum gates that do not act on any edge qubits,
- $h_2 = \frac{3}{4} = \frac{h_1}{2}$ for quantum gates that act on the first qubit, and
- $h_1 = h = \frac{h_2}{2}$ for quantum gates that act on the last qubit

due to the open boundary conditions.

We optimize various quantum circuits that are initialized as the most accurate Trotterization for the considered number of layers. In our case, these are Trotter circuits of orders two and four as well as concatenations of both. The numerical results are presented in Fig. 7. While the error improvement is not as significant as for the previous transverse-field Ising model, the Riemannian optimization increases the accuracy of all optimized quantum circuits to up to one order of magnitude.

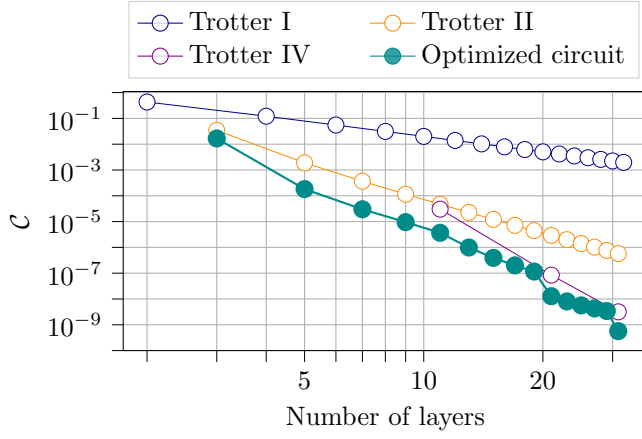


Figure 7: Results for the Heisenberg model on a chain with $N = 50$ sites, $\vec{J} = (1, 1, -\frac{1}{2})$, $\vec{h} = (\frac{3}{4}, 0, 0)$, and $t = 0.25$.

5.2 Fermionic systems

When considering fermionic systems, their anti-symmetrical nature needs to be taken into account, e.g., utilizing second quantization. This study examines two types of fermionic Hamiltonians formulated in second quantization. A suitable fermion-to-qubit mapping such as the Jordan-Wigner transformation [44, 45] must be applied to map the corresponding operators. Furthermore, fermionic swaps must be considered when simulating longer-range interactions using only nearest-neighbor two-qubit gates. Ref. [25] has introduced the fermionic swap network to simulate a Trotter step of a fermionic system using a brickwall circuit layout. In the following, we introduce the fermionic systems we considered for our numerical simulations and provide further details on the Trotter implementation in Appendix D.

5.2.1 Spinful Fermi-Hubbard model.

As a first fermionic system, we consider the one-dimensional spinful Fermi-Hubbard model given by

$$H^{\text{FH}} = - \sum_{\langle pq \rangle, s} T \left(a_{ps}^\dagger a_{qs} + a_{qs}^\dagger a_{ps} \right) + \frac{1}{2} \sum_p V n_{p\uparrow} n_{p\downarrow},$$

where $\langle pq \rangle$ denotes the pairs of adjacent spatial orbitals on a chain, and $s \in \{\uparrow, \downarrow\}$ denotes the spin. Within the numerical simulations, we consider $T = 1$ and $V = 4$. We compute U_{ref} as 10 repetitions of a fourth-order Trotterization, $U_{\text{Trotter}}^{\text{IV},10}$,

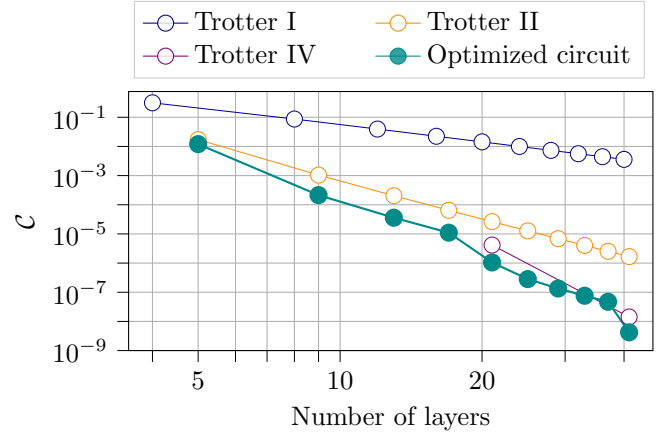


Figure 8: Results for the one-dimensional spinful Fermi-Hubbard model with $N = 50$ spin orbitals, $T = 1$, $V = 4$, and $t = 0.3$.

for a simulation time of $t = 0.3$, which is further compressed to a threshold of ϵ_{thres} as listed in Appendix C Table 2. To implement a Trotter step, a fermionic swap network is required, for which we provide more details in Appendix D. We start from quantum circuits initialized in Trotter steps of the second or fourth order as well as their concatenations, where the best Trotterization is chosen for a given number of layers. The optimization results are visualized in Fig. 8.

Again, the Riemannian optimization method can improve the simulation accuracy for each optimized quantum circuit. The presented results show a relative error improvement of up to a factor of 6.

5.2.2 Molecular Hamiltonian

Finally, we showcase a proof of concept on how the presented method could be applied to a chemical system. The general molecular Hamiltonian in second quantization is given by

$$H^{\text{mol}} = \underbrace{\sum_{p,q=1}^N t_{pq} a_p^\dagger a_q}_{T} + \underbrace{\sum_{p,q,r,s=1}^N v_{pqrs} a_p^\dagger a_q^\dagger a_r a_s}_{V},$$

where the scalar coefficients h_{pq} and h_{pqrs} are the one- and two-electron integrals. Various techniques can be employed to “diagonalize” the Coulomb interaction term V [46–49]. For the presented numerical experiment, we utilize the so-called *double*

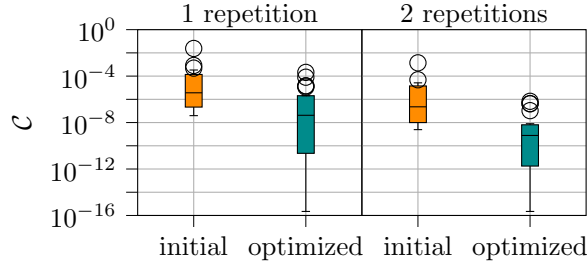


Figure 9: Riemannian optimization results for LiH. The optimization was performed on 21 diagonal molecular Hamiltonian terms. For each term, a brickwall circuit initialized as one repetition (left panel) or two repetitions (right panel) of a second-order Trotter step is optimized.

factorization [49], within which the two-body interaction term can be decomposed into diagonal terms via

$$V = \sum_{p,q} S_{pq} a_p^\dagger a_q + \sum_{\ell=1}^{N_{\text{rot}}} \sum_{i,j}^{\rho_\ell} \frac{\lambda_i^{(\ell)} \lambda_j^{(\ell)}}{2} n_i^{(\ell)} n_j^{(\ell)},$$

where

$$n_i^{(\ell)} = \sum_{p,s=1}^N U_{si}^{(\ell)} a_p^\dagger a_s U_{pi}^{(\ell)} = a_{\psi_i^{(\ell)}}^\dagger a_{\psi_i^{(\ell)}},$$

corresponds to the number operator in a rotated basis, $\psi_i^{(\ell)} = \sum_{p=1}^N U_{pi}^{(\ell)} \phi_p$. By doing so, H^{mol} can be decomposed into Givens rotations and N_{rot} Hamiltonian terms

$$H_{\text{diag}}^{\text{mol},(\ell)} = \sum_{i,j} T_{ij}^{(\ell)} a_{\psi_i^{(\ell)}}^\dagger a_{\psi_j^{(\ell)}} + V_{ij}^{(\ell)} n_i^{(\ell)} n_j^{(\ell)},$$

where the two-body interaction $V^{(\ell)}$ is diagonal. A Trotter step for each $H_{\text{diag}}^{\text{mol},(\ell)}$ can be implemented following the fermionic swap network introduced in Ref. [25]. Again, we leave further details on the fermionic swap network in Appendix D. As a proof of concept, we consider the molecule LiH with six orbitals and a simulation time $t = 1$. We decompose its molecular Hamiltonian into $N_{\text{rot}} = 21$ sets of rotations and diagonal Hamiltonians $\{H_{\text{diag}}^{\text{mol},(\ell)}\}_{\ell=1}^{N_{\text{rot}}}$, of which we optimize each. We start from an initial quantum circuit corresponding to one and two Trotter steps of order 2. The optimization results are visualized in Fig. 9.

We observe that the simulation accuracy could be increased for each considered quantum circuit.

In the best cases, the optimized errors become negligible on the order of numerical precision ($\approx 10^{-16}$). However, we do not claim any generalization of these results to larger systems, as the considered molecule is a simple example intended to serve as a proof of concept. In each optimization case, a relative error improvement of at least one order of magnitude could be achieved, with the best relative error improvement being eight orders of magnitude. The median relative error improvement is about two orders of magnitude.

6 Conclusion

In this work, we present a novel approach that combines Riemannian optimization and tensor network methods to significantly enhance the simulation accuracy of initial Trotter circuits for the time evolution of various quantum systems. Our technique is purely classical, eliminating the need for costly quantum resources. To address the challenge of dimensionality in larger systems, we employ an MPO approximation for the reference time evolution operator. This requires careful selection of the maximum bond dimension and simulation time to ensure computational feasibility while maintaining accurate MPO representation of the actual time evolution. Our method is versatile and does not impose any symmetry constraints on the quantum systems. We demonstrate its effectiveness by optimizing ten disordered transverse-field Ising model instances. Additionally, we achieve error improvements of up to four orders of magnitude for one-dimensional systems such as the non-disordered transverse-field Ising model, the Heisenberg model, and the spinful Fermi-Hubbard model. We further showcase the applicability of our method to molecular Hamiltonians, using LiH as a proof of concept, where we achieve error improvements of up to eight orders of magnitude.

To tackle larger molecular systems, efficient methods to approximate the time evolution operator as an MPO are crucial [50, 51]. We leave a more detailed analysis of this aspect for future work. Our current method employs first-order Riemannian optimization. However, recent results [52] have shown a tremendous advantage of

using second-order Riemannian optimization over first-order methods in the context of quantum circuit compression. Therefore, in future research, we plan to explore second-order optimization methods for brickwall circuits, such as the trust-region algorithm [17, 52].

Acknowledgments

We thank Joe Gibbs for valuable discussions about his recent work in Ref. [19]. We further thank Yu Wang for his input on tensor network methods. The Riemannian ADAM optimizer was implemented based on the ADAM implementation of `qiskit v0.18` [53]. All tensor network operations were implemented using `jax.numpy` [54]. S.S. acknowledges support from the BMW group. The authors gratefully acknowledge the computational and data resources provided by the Leibniz Supercomputing Centre (www.lrz.de).

Data availability

The implementation used in this project, as well as the numerical results and a script to generate the figures in this publication, are available under <https://github.com/INMLe/rqcopt-mpo>.

Authors contribution

I.L. implemented the tensor network and optimization framework and performed the numerical simulations. S.S. implemented the double factorization of the molecular Hamiltonian. I.L. wrote the manuscript with contributions from all authors. C.M. supervised the project.

References

- [1] Richard P Feynman. “Simulating physics with computers”. In Feynman and computation. **Pages 133–153**. cRc Press (2018).
- [2] Seth Lloyd. “Universal quantum simulators”. **Science** **273**, 1073–1078 (1996).
- [3] Christof Zalka. “Simulating quantum systems on a quantum computer”. **Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences** **454**, 313–322 (1998).
- [4] Bela Bauer, Sergey Bravyi, Mario Motta, and Garnet Kin-Lic Chan. “Quantum algorithms for quantum chemistry and quantum materials science”. **Chemical Reviews** **120**, 12685–12717 (2020).
- [5] Yudong Cao, Jonathan Romero, Jonathan P. Olson, Matthias Degroote, Peter D. Johnson, Mária Kieferová, Ian D. Kivlichan, Tim Menke, Borja Peropadre, Nicolas P. D. Sawaya, et al. “Quantum chemistry in the age of quantum computing”. **Chemical Reviews** **119**, 10856–10915 (2019).
- [6] Jens Eisert, Mathis Friesdorf, and Christian Gogolin. “Quantum many-body systems out of equilibrium”. **Nature Physics** **11**, 124–130 (2015).
- [7] Grant Kluber. “Trotterization in quantum theory” (2023). [arXiv:2310.13296](https://arxiv.org/abs/2310.13296).
- [8] Andrew M. Childs, Yuan Su, Minh C. Tran, Nathan Wiebe, and Shuchen Zhu. “Theory of Trotter error with commutator scaling”. **Physical Review X** **11**, 011020 (2021).
- [9] Ansgar Schubert and Christian B. Mendl. “Trotter error with commutator scaling for the Fermi-Hubbard model”. **Physical Review B** **108**, 195105 (2023).
- [10] Refik Mansuroglu, Timo Eckstein, Ludwig Nützel, Samuel A. Wilkinson, and Michael J. Hartmann. “Variational Hamiltonian simulation for translational invariant systems via classical pre-processing”. **Quantum Science and Technology** **8**, 025006 (2023).
- [11] Refik Mansuroglu, Felix Fischer, and Michael J. Hartmann. “Problem-specific classical optimization of Hamiltonian simulation”. **Physical Review Research** **5**, 043035 (2023).
- [12] Maurits S. J. Tepaske, Dominik Hahn, and David J. Luitz. “Optimal compression of quantum many-body time evolution operators into brickwall circuits”. **SciPost Physics** **14**, 073 (2023).
- [13] Conor Mc Keever and Michael Lubasch. “Classically optimized Hamiltonian simu-

- lation”. *Physical Review Research* **5**, 023146 (2023).
- [14] Conor Mc Keever and Michael Lubasch. “Towards adiabatic quantum computing using compressed quantum circuits”. *PRX Quantum* **5**, 020362 (2024).
- [15] Niall Robertson, Albert Akhriev, Jiri Vala, and Sergiy Zhuk. “Approximate quantum compiling for quantum simulation: A tensor network based approach”. *ACM Transactions on Quantum Computing* **6** (2025).
- [16] Yuxuan Zhang, Roeland Wiersema, Juan Carrasquilla, Lukasz Cincio, and Yong Baek Kim. “Scalable quantum dynamics compilation via quantum machine learning” (2024). [arXiv:2409.16346](#).
- [17] Ayse Kotil, Rahul Banerjee, Qunsheng Huang, and Christian B. Mendl. “Riemannian quantum circuit optimization for Hamiltonian simulation”. *Journal of Physics A: Mathematical and Theoretical* **57**, 135303 (2024).
- [18] Luke Causer, Felix Jung, Asimpunya Mitra, Frank Pollmann, and Adam Gammon-Smith. “Scalable simulation of nonequilibrium quantum dynamics via classically optimized unitary circuits”. *Physical Review Research* **6**, 033062 (2024).
- [19] Joe Gibbs and Lukasz Cincio. “Deep circuit compression for quantum dynamics via tensor networks”. *Quantum* **9**, 1789 (2025).
- [20] Baptiste Anselme Martin, Thomas Ayrat, François Jamet, Marko J. Rančić, and Pascal Simon. “Combining matrix product states and noisy quantum computers for quantum simulation”. *Physical Review A* **109**, 062437 (2024).
- [21] P. Kingma Diederik. “Adam: A method for stochastic optimization” (2014). [arXiv:1412.6980](#).
- [22] Mark R. Dowling and Michael A. Nielsen. “The geometry of quantum computation” (2006). [arXiv:0701004](#).
- [23] Michael A. Nielsen, Mark R. Dowling, Mile Gu, and Andrew C. Doherty. “Quantum computation as geometry”. *Science* **311**, 1133–1135 (2006).
- [24] Dylan Lewis, Roeland Wiersema, Juan Carrasquilla, and Sougato Bose. “Geodesic algorithm for unitary gate design with time-independent Hamiltonians”. *Physical Review A* **111**, 052618 (2025).
- [25] Ian D. Kivlichan, Jarrod McClean, Nathan Wiebe, Craig Gidney, Alán Aspuru-Guzik, Garnet Kin-Lic Chan, and Ryan Babbush. “Quantum simulation of electronic structure with linear depth and connectivity”. *Physical Review Letters* **120**, 110501 (2018).
- [26] David Rogerson and Ananda Roy. “Quantum circuit optimization using differentiable programming of tensor network states” (2024). [arXiv:2408.12583](#).
- [27] Yuchen Guo and Shuo Yang. “Efficient quantum circuit compilation for near-term quantum advantage”. *EPJ Quantum Technology* **12**, 69 (2025).
- [28] Sumeet Khatri, Ryan LaRose, Alexander Poremba, Lukasz Cincio, Andrew T. Sornborger, and Patrick J. Coles. “Quantum-assisted quantum compiling”. *Quantum* **3**, 140 (2019).
- [29] Kelvin Koor, Yixian Qiu, Leong Chuan Kwek, and Patrick Rebentrost. “A short tutorial on Wirtinger calculus with applications in quantum information” (2023). [arXiv:2312.04858](#).
- [30] Ilia A. Luchnikov, Alexander Ryzhov, Sergey N. Filippov, and Henni Ouerdane. “QGOpt: Riemannian optimization for quantum technologies”. *SciPost Physics* **10**, 079 (2021).
- [31] Ilia A. Luchnikov, Mikhail E. Krechetov, and Sergey N. Filippov. “Riemannian geometry and automatic differentiation for optimization problems of quantum physics and quantum technologies”. *New Journal of Physics* **23**, 073006 (2021).
- [32] Emiliano Godinez-Ramirez, Richard M. Milbradt, and Christian B. Mendl. “Riemannian approach to the Lindbladian dynamics of a locally purified tensor network”. *Physical Review A* **112**, 012224 (2025).
- [33] Roeland Wiersema and Nathan Killoran. “Optimizing quantum circuits with Riemannian gradient flow”. *Physical Review A* **107**, 062421 (2023).
- [34] Shah Nawaz Ahmed, Fernando Quijandría, and Anton Frisk Kockum. “Gradient-descent

- quantum process tomography by learning Kraus operators”. *Physical Review Letters* **130**, 150402 (2023).
- [35] Ar A. Melnikov, Alena A. Termanova, Sergey V. Dolgov, Florian Neukart, and M. R. Perelshtein. “Quantum state preparation using tensor networks”. *Quantum Science and Technology* **8**, 035027 (2023).
- [36] V. Akshay, Ar Melnikov, A. Termanova, and M. R. Perelshtein. “Tensor networks based quantum optimization algorithm” (2024). [arXiv:2404.15048](#).
- [37] A. Termanova, Ar Melnikov, E. Mamenchikov, N. Belokonev, S. Dolgov, A. Berezhutskii, R. Ellerbrock, C. Mansell, and M. R. Perelshtein. “Tensor quantum programming”. *New Journal of Physics* **26**, 123019 (2024).
- [38] Markus Hauru, Maarten Van Damme, and Jutho Haegeman. “Riemannian optimization of isometric tensor networks”. *SciPost Physics* **10**, 040 (2021).
- [39] P.-A. Absil, Robert Mahony, and Rodolphe Sepulchre. “Optimization algorithms on matrix manifolds”. *Princeton University Press*. (2008).
- [40] Alan Edelman, Tomás A. Arias, and Steven T. Smith. “The geometry of algorithms with orthogonality constraints”. *SIAM Journal on Matrix Analysis and Applications* **20**, 303–353 (1998).
- [41] Gary Bécigneul and Octavian-Eugen Ganea. “Riemannian adaptive optimization methods” (2018). [arXiv:1810.00760](#).
- [42] Ulrich Schollwöck. “The density-matrix renormalization group in the age of matrix product states”. *Annals of Physics* **326**, 96–192 (2011).
- [43] Barbara Kraus and J. Ignacio Cirac. “Optimal creation of entanglement using a two-qubit gate”. *Physical Review A* **63**, 062309 (2001).
- [44] Ian Duck, E. C. G. Sudarshan, and Arthur S. Wightman. “Pauli and the spin-statistics theorem”. *American Journal of Physics* **67**, 742–746 (1999).
- [45] Pascual Jordan and Eugene P. Wigner. “About the Pauli exclusion principle”. *Zeitschrift für Physik* **47**, 631–651 (1928).
- [46] Ryan Babbush, Nathan Wiebe, Jarrod McClean, James McClain, Hartmut Neven, and Garnet Kin-Lic Chan. “Low-depth quantum simulation of materials”. *Physical Review X* **8**, 011044 (2018).
- [47] Steven R. White. “Hybrid grid/basis set discretizations of the Schrödinger equation”. *The Journal of Chemical Physics* **147** (2017).
- [48] Maxine Luo and J. Ignacio Cirac. “Efficient simulation of quantum chemistry problems in an enlarged basis set”. *PRX Quantum* **6**, 010355 (2025).
- [49] Mario Motta, Erika Ye, Jarrod R McClean, Zhendong Li, Austin J Minnich, Ryan Babbush, and Garnet Kin-Lic Chan. “Low rank representations for quantum simulation of electronic structure”. *npj Quantum Information* **7**, 83 (2021).
- [50] Sebastian Paeckel, Thomas Köhler, Andreas Swoboda, Salvatore R. Manmana, Ulrich Schollwöck, and Claudius Hubig. “Time-evolution methods for matrix-product states”. *Annals of Physics* **411**, 167998 (2019).
- [51] Maarten Van Damme, Jutho Haegeman, Ian McCulloch, and Laurens Vanderstraeten. “Efficient higher-order matrix product operators for time evolution”. *SciPost Physics* **17**, 135 (2024).
- [52] Fabian Putterer, Max M. Zumpe, Isabel Nha Minh Le, Qunsheng Huang, and Christian B. Mendl. “High performance contraction of quantum circuits for Riemannian optimization” (2025). [arXiv:2506.23775](#).
- [53] “ADAM implementation by qiskit v0.18” (2021).
- [54] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. “JAX: composable transformations of Python+NumPy programs” (2018).

A Gradient computation using the Wirtinger formalism

Within this work, we compute the gradient using the Wirtinger formalism [29] that can also be applied to non-holomorphic functions. In the following, we give a short summary of the latter and provide further details on the gradient calculation.

Consider complex-valued smooth functions $f, g : \mathbb{C} \rightarrow \mathbb{C}$ and a complex number $z = x + iy$ with $x, y \in \mathbb{R}$. The Wirtinger derivatives are defined as follows:

$$\partial_z f(z) = \frac{1}{2} (\partial_x f - i \partial_y f), \quad (13a)$$

$$\partial_{z^*} f(z) = \frac{1}{2} (\partial_x f + i \partial_y f), \quad (13b)$$

where $\partial_x f$ and $\partial_y f$ are the conventional partial derivatives when interpreting f as function $f : \mathbb{R}^2 \rightarrow \mathbb{C}$, $f(x, y) = f(x + iy)$. Furthermore, the following *chain rule* holds:

$$\partial_z (f \circ g) = (\partial_w f \circ g) \partial_z g + (\partial_{w^*} f \circ g) \partial_z g^*, \quad (14)$$

where “ $\circ$ ” denotes function composition and $w = g(z)$. By definition, we can express the gradient ∇f as

$$\nabla f(z) = 2(\partial_z f(z))^*, \quad (15)$$

where $(*)$ denotes the complex conjugate.

In the following, we will give a more detailed computation of the Euclidean gradient $\nabla \mathcal{L}$. By using the presented Wirtinger formalism, the partial derivatives with respect to an entry of W are given by

$$\partial_{W_{jk}} \text{Re } \mathcal{T}(W) = \frac{1}{2} \partial_{W_{jk}} \text{Tr}(U^\dagger W + W^\dagger U) = \frac{U_{jk}^*}{2},$$

and

$$\partial_{W_{jk}} \text{Im } \mathcal{T}(W) = \frac{1}{2i} \partial_{W_{jk}} \text{Tr}(U^\dagger W - W^\dagger U) = \frac{U_{jk}^*}{2i},$$

where we have used that

$$\partial_{B_{\gamma\tau}} \text{Tr}(A^\dagger B) = \sum_{\alpha, \beta} A_{\beta\alpha}^* B_{\beta\alpha} \delta_{\beta\gamma} \delta_{\alpha\tau} = A_{\gamma\tau}^*,$$

and

$$\partial_{B_{\gamma\tau}} \text{Tr}(B^\dagger A) = \partial_{B_{\gamma\tau}} \sum_{\alpha, \beta} B_{\beta\alpha}^* A_{\beta\alpha} = 0.$$

For convenience, we repeat the loss function given in the main text in Eq. (3):

$$\mathcal{L} = -[\text{Re } \mathcal{T}(W)]^2 - [\text{Im } \mathcal{T}(W)]^2$$

Using the chain rule given by Eq. (14) and the previous results, the partial derivative of $\mathcal{L}$ with respect to a matrix element W_{jk} can be computed as

$$\begin{aligned} \partial_{W_{jk}} \mathcal{L} &= -2 \text{Re } \mathcal{T}(W) \cdot \partial_{W_{jk}} \text{Re } \mathcal{T}(W) \\ &\quad - 2 \text{Im } \mathcal{T}(W) \cdot \partial_{W_{jk}} \text{Im } \mathcal{T}(W) \\ &= -\text{Re } \mathcal{T}(W) \cdot U_{jk}^* + \text{Im } \mathcal{T}(W) \cdot i U_{jk}^* \\ &= -[\mathcal{T}(W) U_{jk}]^*. \end{aligned}$$

By again applying the chain rule given in Eq. (14), the partial derivatives of $\mathcal{L}$ with respect to a specific gate are given by

$$\begin{aligned}
\partial_{G_i^\ell} \mathcal{L} &= \partial_W \mathcal{L} \partial_{G_i^\ell} W = \sum_{jk} \partial_{W_{jk}} \mathcal{L} \cdot \partial_{G_i^\ell} W_{jk} \\
&= - \sum_{jk} \mathcal{T}(W)^* U_{kj}^\dagger \partial_{G_i^\ell} W_{jk} \\
&= - \mathcal{T}(W)^* \partial_{G_i^\ell} \sum_{jk} U_{kj}^\dagger W_{jk} \\
&= - \mathcal{T}(W)^* \cdot \partial_{G_i^\ell} \mathcal{T}(W),
\end{aligned}$$

where we denote the number of layers in the brickwall circuit with L and the number of gates in layer ℓ with n_ℓ . Finally, the overall gradient can be obtained via Eq. (15) as

$$\nabla \mathcal{L} = -2 \mathcal{T}(W) \cdot \partial_{G_i^\ell} [\mathcal{T}(W)]_{\substack{\ell=1,\dots,L \\ i=1,\dots,n_\ell}}^* \cdot \quad (16)$$

B Jacobians of the Weyl decomposition

For convenience, we state the Weyl decomposition [43] again:

$$G = (K_1 \otimes K_2) \cdot V(a, b, c) \cdot (K_3 \otimes K_4),$$

with

$$K_i = R_Z(\theta_i) R_Y(\psi_i) R_Z(\phi_i),$$

and

$$V(a, b, c) = e^{i(aXX+bYY+cZZ)}$$

The Jacobian with respect to $a, b, c \in \mathbb{R}$ is given by:

$$\begin{aligned}
\frac{\partial G}{\partial a} &= iK_1 \otimes K_2 \cdot XX \cdot V(a, b, c) \cdot K_3 \otimes K_4 \\
\frac{\partial G}{\partial b} &= iK_1 \otimes K_2 \cdot YY \cdot V(a, b, c) \cdot K_3 \otimes K_4 \\
\frac{\partial G}{\partial c} &= iK_1 \otimes K_2 \cdot ZZ \cdot V(a, b, c) \cdot K_3 \otimes K_4.
\end{aligned}$$

The Jacobian of K_j with respect to the corresponding angles $\theta_j, \psi_j, \phi_j \in \mathbb{R}$ is given by:

$$\begin{aligned}
\frac{\partial K_j}{\partial \theta_j} &= -\frac{i}{2} Z K_j \\
\frac{\partial K_j}{\partial \psi_j} &= -\frac{i}{2} R_Z(\theta_j) Y R_Y(\psi_j) R_Z(\psi_j) \\
\frac{\partial K_j}{\partial \phi_j} &= -\frac{i}{2} R_Z(\theta_j) R_Y(\psi_j) Z R_Z(\phi_j).
\end{aligned}$$

The Jacobian $\frac{\partial G}{\partial \alpha}$ for $\alpha = \theta_j, \psi_j, \phi_j$ can directly be obtained from the latter equations.

Table 2: Overview of reference approximation for the different Hamiltonians.

	Reference approximation	Trotter error	Trunc. error	Compr. error
<i>Ising (20 qubits)</i>	$U_{\text{Trotter}}^{\text{IV},20} (t = 2.0)$	$1 \cdot 10^{-11}$	$5 \cdot 10^{-15}$	$9 \cdot 10^{-8}$
<i>Ising (50 qubits)</i>	$U_{\text{Trotter}}^{\text{IV},20} (t = 2.0)$	$4 \cdot 10^{-11}$	$2 \cdot 10^{-14}$	$2 \cdot 10^{-9}$
<i>Heisenberg (50 qubits)</i>	$U_{\text{Trotter}}^{\text{IV},10} (t = 0.25)$	$4 \cdot 10^{-11}$	$3 \cdot 10^{-14}$	$3 \cdot 10^{-11}$
<i>Fermi-Hubbard (20 qubits)</i>	$U_{\text{Trotter}}^{\text{IV},10} (t = 0.3)$	$5 \cdot 10^{-11}$	$2 \cdot 10^{-11}$	$5 \cdot 10^{-11}$
<i>Fermi-Hubbard (50 qubits)</i>	$U_{\text{Trotter}}^{\text{IV},10} (t = 0.3)$	$2 \cdot 10^{-10}$	$5 \cdot 10^{-11}$	$5 \cdot 10^{-11}$

C Overview of reference approximation for the different Hamiltonians

Several error sources lead to a deviation of the reference MPO and the exact time evolution operator as described in Section 4.2. For the numerical simulations conducted in this work, we summarize these contributions in Table 2.

D Fermionic swap network

In this work, Trotter circuits of first-, second-, and fourth-order are considered. Given a Hamiltonian of the form $H = \sum_{i=1}^k H_i$, these Trotterizations are given by [7]

$$\begin{aligned}
 U_1(t) &= \prod_{i=1}^k e^{-iH_i t}, \\
 U_2(t) &= \prod_{i=1}^k e^{-iH_i t/2} \prod_{i=k}^1 e^{-iH_i t/2}, \\
 U_4(t) &= [U_2(s_2 t)]^2 [U_2((1 - 4s_2)t)] [U_2(s_2 t)]^2,
 \end{aligned}$$

where in the latter $s_2 = (4 - 4^{1/3})^{-1}$.

To implement a Trotter circuit for fermionic Hamiltonians, each orbital p is represented by a qubit i_p and the fermionic operators $a^\dagger, a, n$ need to be mapped to corresponding qubit operators [25]. Using the Jordan-Wigner transformation [44, 45] for this purpose results in possibly long-range Pauli-Z strings. However, if the qubits representing the involved orbitals p, q are adjacent, i.e., $i_q = i_p + 1$, the time-evolution of a single summand in V can be implemented by two-qubit gates. To this end, a fermionic swap network architecture can be employed [25], where fermionic swaps

$$f_{\text{swap}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

are utilized suitably.

D.1 Spinful Fermi-Hubbard model

For simplicity, we state the considered one-dimensional spinful Fermi-Hubbard Hamiltonian again:

$$H^{\text{FH}} = - \sum_{\langle pq \rangle, s} T_{pq} \left(a_{ps}^\dagger a_{qs} + a_{qs}^\dagger a_{ps} \right) + \frac{1}{2} \sum_p V_p n_{p\uparrow} n_{p\downarrow}.$$

As can be seen, interactions of spin orbitals (ps) and (qs) for $p = q + 1$ as well as interactions between spin orbitals ($p \uparrow$) and ($p \downarrow$) need to be considered. To simulate a Trotter step, we order the spin orbitals as follows (example of 5 spatial orbitals):



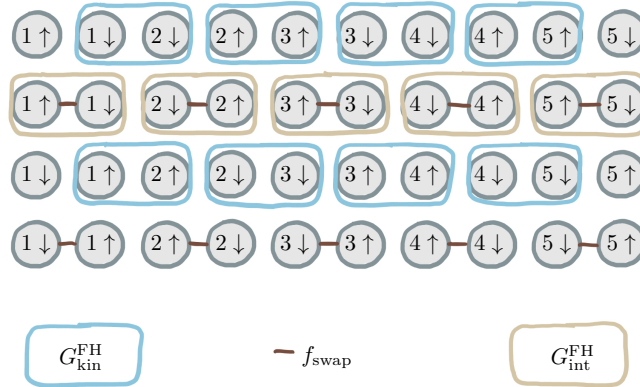
We then construct the swap network as follows: The first and third layer simulate the kinetic hopping terms, for which the two-qubit gates V_{kin} after Jordan-Wigner transformation are given by

$$G_{\text{kin}}^{\text{FH}}(i_p, i_q) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(T_{pq}\Delta t) & i \sin(T_{pq}\Delta t) & 0 \\ 0 & i \sin(T_{pq}\Delta t) & \cos(T_{pq}\Delta t) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The second layer simulates the on-site interactions while simultaneously swapping the corresponding orbitals, for which the two-qubit gates V_{int} after applying Jordan-Wigner transformation are given by

$$G_{\text{int}}^{\text{FH}}(i_p, i_q) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -e^{-iV_{pq}\Delta t} \end{pmatrix}.$$

Note that for first-order Trotterization, an additional layer that reverses the previous fermionic swaps given by needs to be applied, which is not necessary for even-order Trotterizations. The fermionic swap network for a first-order Trotter step of a system with 10 spin orbitals is then given by:



D.2 Molecular Hamiltonian

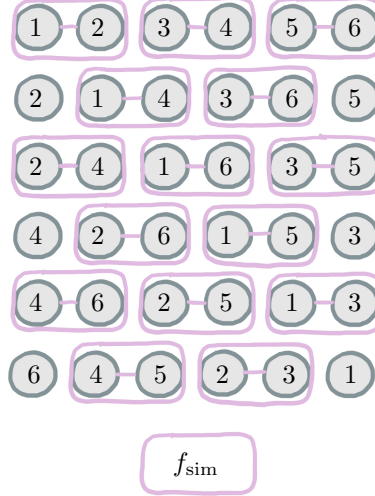
Ref. [25] presented a fermionic swap network to implement a Trotter step of a molecular Hamiltonian with diagonal interaction term given by

$$H = \sum_{pq} T_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{p \neq q} V_{pq} n_p n_q,$$

where p, q denote the orbitals. In this case, the interaction and kinetics between two neighboring orbitals, as well as their fermionic swapping, can be jointly simulated by the following two-qubit gate:

$$f_{\text{sim}}(i_p, i_q) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -i \sin(T_{pq}\Delta t) & \cos(T_{pq}\Delta t) & 0 \\ 0 & \cos(T_{pq}\Delta t) & -i \sin(T_{pq}\Delta t) & 0 \\ 0 & 0 & 0 & -e^{-iV_{pq}\Delta t} \end{pmatrix}$$

The following fermionic swap network implements a Trotter step of first-order for 6 orbitals, where the layers for reversing the orbital ordering are neglected for simplicity:



E Comparison of local and simultaneous gate updates

Instead of updating all quantum gates simultaneously within a single optimization step, alternative “sweeping” approaches update one gate at a time while keeping the remaining quantum circuit fixed [18, 19]. In both methods, $\partial_{G_i^\ell} \mathcal{T}(W)$ must be evaluated, as illustrated in Fig. 4. These partial derivatives contribute to the full gradient in Riemannian optimization. In the sweeping method, however, a specific gate G_i^ℓ is updated as

$$\tilde{G}_i^\ell = q_{\text{polar}} \left[\partial_{G_i^\ell} \mathcal{T}(W)^\dagger \right],$$

where $q_{\text{polar}}(A)$ denotes the unitary matrix $Q \in \mathcal{U}$ obtained from the polar decomposition $A = QP$, with $P \in \mathbb{C}^{4 \times 4}$ being a Hermitian positive semi-definite matrix. There are additional aspects in which the two optimization methods differ. In the local update method, there is flexibility in the order of sweeping through the quantum gates. It has been observed that different sweeping orders impact the optimization [18]; however, depending on the chosen local optimization method, not all sweeping orders can be implemented efficiently [19]. Riemannian optimization updates all quantum gates simultaneously, thereby avoiding potential issues arising from a non-optimal sweeping order. However, Riemannian optimization requires an appropriate selection of the learning rate, which is not necessary for local updates. While the ADAM optimizer adjusts the learning rate during the optimization process, an appropriate initial choice can influence the quality of the optimization.

To compare both optimization methods, we define one optimization step of the local sweep method as a top-to-bottom or bottom-to-top sweep, where each layer’s quantum gates are updated in a back-and-forth manner, i.e., left-to-right-to-left. The corresponding sweeping order is illustrated in Fig. E1. This

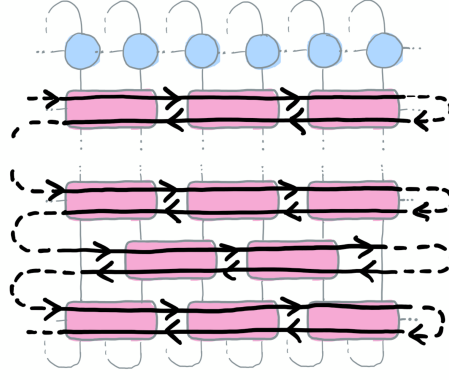


Figure E1: The chosen top-to-bottom sweeping order that defines one optimization step within the local update method. Similarly, a bottom-to-top sweep defines another optimization step. After one optimization step, each quantum gate was updated twice.

results in each quantum gate being updated twice per step, in contrast to the single update within the Riemannian approach. From a computational perspective, the bottleneck for Riemannian optimization is the computation of the gradient, whereas for the sweeping method, it is the local updates. Since both computations utilize a similar environment caching, the asymptotic computational scaling is the same. In the following, we consider two stopping criteria for the optimization: (i) maximum number of iterations and (ii) early stopping. In the first case, the optimization is terminated after a maximum number of iterations n_{iter} , and the optimized error for both methods is compared. In the second case, the optimization is halted if the relative deviation of the optimization cost between two steps i and $i - n$ is smaller than the tolerance given by

$$2 \cdot \left| \frac{\mathcal{C}_{i-n} - \mathcal{C}_i}{\mathcal{C}_{i-n} + \mathcal{C}_i} \right| \leq 10^{-5},$$

where the distance between the two compared optimization steps is adjusted with the number of iterations, $n = \lceil 0.01i \rceil$. For example, if the optimizer has proceeded for fewer than 100 iterations, the last two points are compared; if it has proceeded for 1000 steps, the last and the tenth last points are compared. If one method does not converge within n_{iter} steps, we determine the number of iterations at which the convergence error of the other method is found. This is indicated with (*). We compare the sweeping method with the Riemannian method following the described procedure for two systems of a spinful

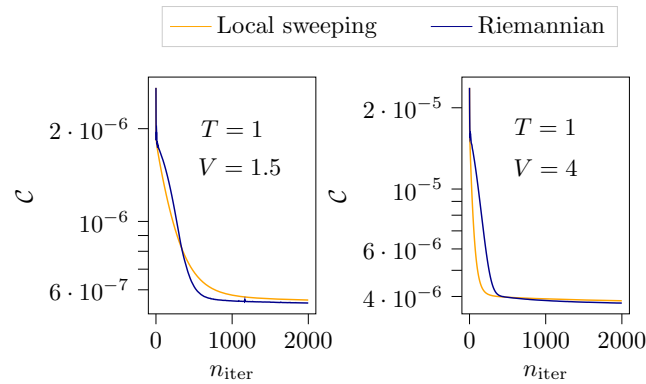


Figure E2: Loss curve comparison for a brickwall circuit with 17 layers. The considered systems are spinful Fermi-Hubbard models on 20 qubits with different interaction strengths.

Table 3: Convergence comparison between local sweeping method and Riemannian optimization for a spinful Fermi-Hubbard system on a chain of 20 qubits. ($\ddagger$) indicates that the convergence criteria was not met within $n_{\text{iter,max}}$. In these cases, n_{iter} was identified as the number of steps required to obtain the convergence error of the other method.

$T = 1$ and $V = 1.5$							
n_{layers}	<i>Local</i> (<i>early stopping</i>)		<i>Riemannian</i> (<i>early stopping</i>)		<i>Local</i> (<i>max. iteration</i>)		<i>Riemannian</i> (<i>max. iteration</i>)
	n_{iter}	Error	n_{iter}	Error	$n_{\text{iter,max}}$	Error	Error
5	25	$4.73 \cdot 10^{-4}$	62	$4.73 \cdot 10^{-4}$	100	$4.73 \cdot 10^{-4}$	$4.73 \cdot 10^{-4}$
9	1872 $\ddagger$	$1.09 \cdot 10^{-5}$	1091	$1.09 \cdot 10^{-5}$	2000	$1.09 \cdot 10^{-5}$	$1.81 \cdot 10^{-5}$
14	1308 $\ddagger$	$1.87 \cdot 10^{-6}$	799	$1.87 \cdot 10^{-6}$	2000	$1.85 \cdot 10^{-6}$	$1.81 \cdot 10^{-6}$
17	—	—	914	$5.54 \cdot 10^{-7}$	2000	$5.56 \cdot 10^{-7}$	$3.43 \cdot 10^{-7}$
21	886	$9.53 \cdot 10^{-9}$	544	$9.53 \cdot 10^{-9}$	2000	$9.13 \cdot 10^{-9}$	$8.85 \cdot 10^{-9}$
41	310	$7.04 \cdot 10^{-11}$	817	$7.04 \cdot 10^{-11}$	2000	$5.62 \cdot 10^{-11}$	$6.97 \cdot 10^{-11}$

$T = 1$ and $V = 4$							
n_{layers}	<i>Local</i> (<i>early stopping</i>)		<i>Riemannian</i> (<i>early stopping</i>)		<i>Local</i> (<i>max. iteration</i>)		<i>Riemannian</i> (<i>max. iteration</i>)
	n_{iter}	Error	n_{iter}	Error	$n_{\text{iter,max}}$	Error	Error
5	11	$4.37 \cdot 10^{-3}$	50	$4.37 \cdot 10^{-3}$	100	$4.37 \cdot 10^{-3}$	$4.37 \cdot 10^{-3}$
9	560 $\ddagger$	$7.74 \cdot 10^{-5}$	373	$7.74 \cdot 10^{-5}$	2000	$7.73 \cdot 10^{-5}$	$7.73 \cdot 10^{-5}$
14	1111 $\ddagger$	$1.28 \cdot 10^{-5}$	577	$1.28 \cdot 10^{-5}$	2000	$1.26 \cdot 10^{-5}$	$1.23 \cdot 10^{-5}$
17	593 $\ddagger$	$3.96 \cdot 10^{-6}$	541	$3.96 \cdot 10^{-6}$	2000	$3.86 \cdot 10^{-6}$	$3.78 \cdot 10^{-6}$
21	686 $\ddagger$	$3.13 \cdot 10^{-7}$	518	$3.13 \cdot 10^{-7}$	2000	$2.77 \cdot 10^{-7}$	$2.69 \cdot 10^{-7}$
41	578	$1.43 \cdot 10^{-9}$	225	$1.43 \cdot 10^{-9}$	2000	$1.29 \cdot 10^{-9}$	$1.33 \cdot 10^{-9}$

Fermi-Hubbard chain on 20 qubits with interaction strengths $V = 1.5$ and $V = 4$ and $T = 1$. Our findings are documented in Table 3.

We observe a generally similar convergence behavior with nearly identical final errors after $n_{\text{iter,max}}$ update steps and only a small variance in the required update steps when considering early stopping. An exemplary loss curve comparison for both Fermi-Hubbard models and a quantum circuit with 17 layers is depicted in Fig. E2.

Note that a one-to-one comparison of both methods is challenging since the local sweeping method generally does not have a natural global update step. Here, we chose a generous definition that allows the local sweeping method to update each gate twice per global update step, whereas one Riemannian optimization step updates each quantum gate only once. While both methods exhibit comparable optimization behavior, it is noteworthy that our method can, in principle, be extended to second-order Riemannian optimization [17, 52], which requires a more computationally expensive evaluation of the Hessian but likely necessitates fewer optimization steps and is more robust against local minima.