Quantum optimal control using tensor networks

Vincent Van Duong
Center for Quantum Phenomena, Department of Physics, New York University, 726 Broadway, New York, New York 10003, USA

Dries Sels
Center for Quantum Phenomena, Department of Physics, New York University, 726 Broadway, New York, New York 10003, USA and
Center for Computational Quantum Physics, Flatiron Institute, 162 5th Avenue, New York, NY 10010, USA

(Dated: August 29, 2024)

Optimal control problems suffer from the curse of dimensionality: they require searching exponentially large spaces. Meanwhile, a tensor network is potent for compressing and manipulating exponentially large data. Inspired by this, we show that a tensor network architecture can efficiently solve quantum optimal control problems by expressing the objective function in the tensor train (TT) format and conducting approximate inference of its optimal control. Our architecture is model-free, parsimonious, and derivative-free, making it suitable for solving a broad class of control problems. We argue that a tensor network architecture of the objective function is particularly natural when controlling low-entangled quantum states. This paper includes examples of single-qubit and many-body control with high fidelity.

I. INTRODUCTION

The high-fidelity manipulation of quantum states is crucial for sensing, computing, and information processing [1]. Optimal control theory provides a framework that ensures that the manipulation of these devices is robust, efficient, and amenable to real-world settings with dissipative losses, coherent errors, and hardware constraints [2]. Quantum optimal control applies the framework to quantum systems where one must implement a protocol to accomplish a desired task. This paper is particularly interested in quantum state preparation, in which one steers an initial state to a target state by manipulating external fields called controls.

Determining the optimal control is typically impossible because the search space suffers from the curse of dimensionality. To see why, consider a simple finite deterministic process with d steps, a 0 or 1 at each step. The process forms a decision tree with 2^d leaves. A bitstring x of length d encodes a particular control, and we associate a cost $F[x]$ for the control sequence. The cost quantifies our ability to perform the desired task, such as preparing a quantum state. At worst, determining the optimal control x along the decision tree requires searching all 2^d outcomes. Searching within a space that grows exponentially with problem size is infeasible, especially if a single evaluation of the objective $F[x]$ is expensive. This affliction is called the curse of dimensionality. Therefore, approximate methods for evaluating $F[x]$ and inferring its global optima are necessary.

This paper shows that when we approximate the cost function by a tensor train, we can approximately solve the optimal control problem with polynomial scaling. Moreover, our architecture is parsimonious, derivative-free, and model-free. Our methods provide an alternative pathway to solve optimal control problems with fewer resources.

Although we present a new method for controlling a quantum system, others exist and shall serve as valuable comparisons. One successful technique is called *gradient ascent pulse engineering* (GRAPE) [3]. GRAPE requires both an explicit model of the quantum dynamics and differentiating the cost function with respect to its controls. Computing these derivatives can become computationally expensive while lacking convergence guarantees to a global optimum. The *chopped random basis* (CRAB) method circumvents evaluating derivatives with Simplex methods [4]. However, CRAB depends on the user to choose the basis of functions, thereby introducing an inductive bias. Reinforcement learning provides tools to avoid specifying a model or derivatives [5]. Reinforcement learning is known to be *model-free* because only cost function evaluations are required. Techniques such as Q-learning and policy gradient have, indeed, successfully solved quantum optimal control problems [6–8]. Despite its success, reinforcement learning typically requires a large volume of data for training, which can be prohibitively expensive if data acquisition requires experiments or high-performance-computing simulations.

In contrast, the tensor network architecture that we introduce has three immediate computational advantages:

1. It is model-free. Specifying the physical system and dynamics is unnecessary for learning the cost function—we only need samples from the function.

2. It is parsimonious. By storing the cost function using the tensor train format, we are able to manipulate it with sub-exponentially many resources.

3. It is derivative-free. The tensor train format allows us to search its minimal entry using tools inspired by quantum tensor networks.

The main numerical techniques involve the *tensor cross interpolation* algorithm and optimization algorithms tai-

lored to tensor trains [9, 10]. Finally, we argue that a tensor train architecture naturally encodes the cost function associated with low-entangled quantum states. Therefore, our physics-inspired technique benefits from literature on tensor networks and matrix product states [11–16].

II. OPTIMAL CONTROL OF A QUANTUM SYSTEM

A. Continuous-time

We review optimal control for quantum systems in continuous time. Consider a quantum state $|\psi(t)\rangle$. Let H_0 be the drift Hamiltonian. Let H_a be a set of control Hamtilonians, and $h_a(t)$ be their corresponding controls. The state's evolution is determined by quantum dynamics,

$$\begin{aligned} H(t) =& H_0 + \sum_a h_a(t) H_a, \\ |\psi(t)\rangle =& \mathcal{T} e^{-i \int_0^t dt' H(t')} |\psi_0\rangle . \end{aligned} \tag{1}$$

Crucially, the state's evolution $|\psi(t)\rangle$ depends on controls $h_a(t)$ and one can attempt to steer the quantum state with a judicious choice for them. Fig. 1 shows how one steers some initial quantum state into another using dynamics.

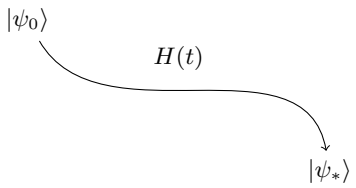

FIG. 1. Quantum state preparation using a controlled Hamiltonian $H(t)$. A protocol steered the initial state $|\psi_0\rangle$ into the target state $|\psi_*\rangle$.

In optimal control, we define our inability to perform the desired task using a cost functional of the controls, and our goal is to minimize it. Formally, the optimal control $h^*(t)$ to the problem is

$$h^*(t) = \underset{h(t)}{\operatorname{argmin}} F[h(t)], \tag{2}$$

where $F[h(t)]$ is the scalar-valued cost functional of the controls $h(t)$. Forms for $F[h(t)]$ include expectation values of the evolved state, but fidelity, entropy, or other relevant metrological properties can be included.

Solving the aforementioned optimization problem is typically non-trivial because dynamics are controlled by Eq. (1). Moreover, additional regularity assumptions on $h(t)$ constrain the optimization problem. A brute force search is infeasible since the search space of functions is uncountably infinite. The next steps will show how to tame the infinite search spaces.

B. Discrete-time

In the previous subsection, we formulated the optimal control problem continuously. This section lays out its equivalent problem statement in discretized time, which our solution technique uses. Moreover, we comment on the relationship between the discrete-time quantum optimal control problem and variational quantum algorithms.

The critical step is to write the control $h(t)$ as a piecewise constant control signal. For a evolution interval $[0, T]$, we partition it into N intervals $[t_k, t_{k+1}]$. (If N is sufficiently large, we can approximate any continuous almost everywhere function.) In what follows, we partition $[0, T]$ in N equal-sized time steps $\delta t = T/N$. The control $h(t)$ becomes a vector $h = [h_0, \ldots, h_{N-1}]$ defined by the constant values it takes on:

$$h(t) = h_k \quad \text{for} \quad t_k \le t < t_{k+1}. \tag{3}$$

What effect does this approximation have on the original problem? The evolution takes the form of N controlled unitary gates,

$$|\psi(T)\rangle = U_{N-1} \cdots U_1 U_0 |\psi_0\rangle , \tag{4}$$

where $U_k = e^{-iH(t_k)\delta t}$. The upshot is a transformation from a cost functional to a cost function: $F[h(t)] \to F[h] = F[h_0, \ldots, h_{N-1}]$. The previous optimal control problem Eq. (2) is recast as finding the global optimum of a standard function,

$$h^* = \underset{h_0,\ldots,h_{N-1}}{\operatorname{argmin}} F[h_0, \ldots, h_{N-1}], \tag{5}$$

which is more amenable to numerical optimization.

An important class of cost functions $F[h]$ are expectation values. We show the tensor network diagram for this problem in Fig. 2. For example, the cost function for preparing the lowest eigenvalue states is

$$F[h] = \langle \psi(T)| O |\psi(T)\rangle , \tag{6}$$

where $\psi(T)$ is the state evolution under control h and O is the target observable, akin to a variational quantum algorithm (VQA) [17]. However, a priori, it is unclear whether a tensor network well approximates F because the cost landscape could be "irregular." Indeed, gradients along h can vanish exponentially with the system size, a phenomenon called barren plateaus [18]. Therefore, $F[h]$ is hard to optimize using gradient-based methods, and its evaluations can be expensive in experiments or numerically. Determining its global optima remains an open question. It is known that the entanglement of the underlying quantum system causes these afflictions, so we will restrict ourselves to controlling quantum systems that entangle sufficiently slowly to improve the regularity of the cost landscape [19].

This section formulated the optimal control problem by minimizing the function Eq. (5) instead of its equivalent functional. The following section shows that a tensor train naturally describes F when dealing with a low-entangled quantum system. The advantage of storing the

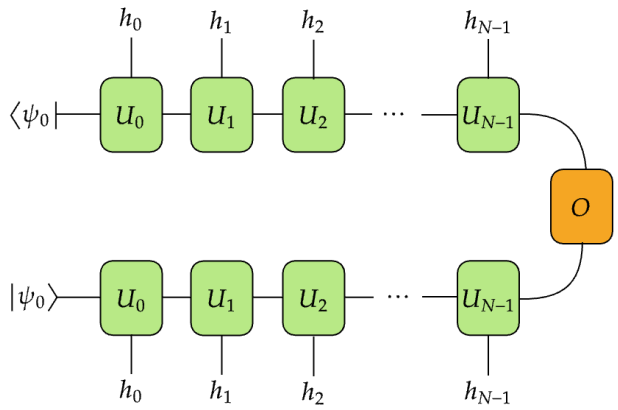

FIG. 2. Tensor network diagram of a cost function $F[h] = \langle\psi(T)|\, O\, |\psi(T)\rangle$. The cost function is an expectation value conditioned on the classical control field $h_0, \ldots, h_{N-1}$.

cost function as a tensor train is its efficiency for performing approximate inference and, in particular, finding an approximate optimum.

C. Controlling matrix product states

In the previous sections, we reviewed the optimal control of arbitrary quantum systems. This section discusses the importance of a quantum system that admits a matrix product state (MPS) representation. We argue that the cost function $F[h]$ admits a tensor train (TT) representation shown in Fig. 2, provided the underlying quantum state is accurately expressed as a MPS.

A MPS represents a state with low entanglement entropy, such as the ground state of gapped short-ranged Hamiltonian in one dimension [20]. A characteristic quantity of a MPS is its bond dimension, typically labeled as χ. The bond dimension quantifies the entanglement entropy between two sites. At worst, the bond dimension of a MPS increases exponentially during evolution because interactions between particles generate entropy. Therefore, a state lacks a MPS representation if its bond dimension grows too fast. We suggest section 6.6.2 of [21] for a more detailed analysis of errors.

Now, consider the controlled dynamics of a MPS. Why should $F[h]$ admit a tensor train (TT) representation? At worst, two things can happen. First, the matrix product operator (MPO) bond dimension between two successive unitary controls U_k and U_{k+1} scales as the product of their bond dimensions. E.g., if U_k and U_{k+1} have a maximum bond dimension of χ, then the composite operator $U_{k+1}U_k$ has at most bond dimension χ^2. Second, as for chaotic systems, χ can grow exponentially with time. Therefore, F will likely have an efficient tensor network representation provided that its underlying MPS bond dimension grows slowly from the first to the last control.

Another heuristic reason $F[h]$ should admit a tensor train representation is that it is a function of a compressed latent representation of the system dynamics, namely, that of the MPS. Therefore, the cost function F is expressable as a function of latent variables. The following section shows how to use tensor networks to learn a latent representation of the cost function $F[h]$.

III. ARCHITECTURE

A. Quantics tensor train

This section shows how to embed the cost function as a quantics tensor train (TT) using the digital signal processing technique called quantization. In simple terms, we quantize the control signal using classical bits x and use these bits as binary inputs to the cost function $F[x]$. Our scheme maps the continuum of controls to a countable – but exponentially large – set of controls we can manipulate.

Without loss of generality, we choose to quantize a single control field $h(t)$. By bounding the control, say, $h(t) \in [0, 1)$, we can represent the signal with classical bits of 0 and 1's. Using the discrete-time approximation from sub-section II B, we quantize the continuous signal:

$$h_k = \sum_{l=0}^{L} x_k^{(l)} 2^{-l}. \tag{7}$$

The bitstring $(x_k^{(0)}, x_k^{(1)}, \ldots, x_k^{(L)})$ is the binary representation of the control h_k at timestep k. A control $h(t)$ that contains N timesteps and quantized at level L becomes a bitstring x that contains $d = LN$ classical bits:

$$x = \left[(x_0^{(0)}, \cdots, x_0^{(L)}), \ldots, (x_{N-1}^{(0)}, \cdots, x_{N-1}^{(L)})\right]. \tag{8}$$

The number of control sequences in this embedding is 2^d. Fig. 3 shows how two different controls, A and B, are quantized by bitstrings. This embedding is particularly natural as it preserves the time ordering. For example, bang-bang protocols are encoded with this structure by setting $L = 1$. In that case, a 0 or 1 corresponds to turning the control field off or on. The advantage of increasing L and N is a higher precision: for sufficiently large L and N, any piece-wise constant function $h(t)$ admits a bitstring approximation.

Next, we tensorize the cost function F as follows,

$$F[x] = F^{x_1, \ldots, x_d}, \tag{9}$$

where $x = [x_1, \ldots, x_d]$ as defined in Eq. (8). Fig. 4 shows the equivalent tensor network diagram.

Now that we expressed $h(t)$ as a binary string x, our optimal control problem is recast as:

$$x^* = \operatorname*{argmin}_x F[x]. \tag{10}$$

But this minimization is challenging since F contains 2^d entries to search from. Our workaround involves learning a tensor train representation of $F[x]$, where we can

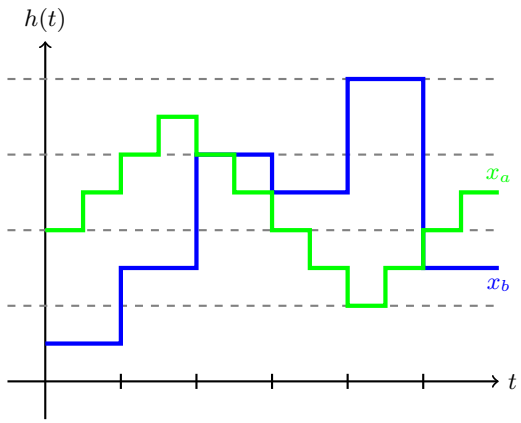

FIG. 3. Bitstrings x_A and x_B represent two choices of controls. The control x_A contains a higher quantization level and sampling rate than x_B.

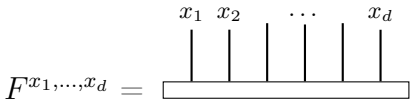

FIG. 4. Tensor network representation of the cost function $F[x]$, where x is the quantized control signal. The tensor network diagram visualizes the tensor's index structure, corresponding to control sequences.

use efficient approximate inference techniques to obtain a candidate solution.

B. Tensor cross interpolation

This section reviews the tensor cross interpolation algorithm (TCI), a tool to compress a high-dimensional tensor as a tensor train. Most of this section is unoriginal, and we refer readers to [22] for a comprehensive introduction. TCI is a compression algorithm analogous to storing a quantum state as a MPS. The interpolation reduces the number of entries required to store $F[x_1, \ldots, x_d]$ from $\mathcal{O}(2^d)$ to $\mathcal{O}(d)$, offering a parsimonious representation of the tensor. Moreover, the number of evaluations required to perform the interpolation scales at most as

$$\mathcal{O}(d^2 n r^2), \tag{11}$$

where n is the number of TCI sweeps, and r is the tensor train's bond dimension. The tensor network diagram corresponding to the tensor train approximation is found in Fig. 5.

The tensor cross interpolation consists of expressing a function as a tensor train,

$$F[x_1, \ldots, x_d] \approx \sum_{\mathbf{s}} X^{(1)}(x_1, s_1) X^{(2)}(s_1, x_1, s_2) \cdots \cdots X^{(d)}(s_{d-1}, x_d), \tag{12}$$

where the matrices X are commonly referred to as its tensor train cores and $\mathbf{s} = s_1, \ldots, s_{d-1}$ are its internal indices, which range from $s_i = 1, \ldots, r$. A characteristic quantity for the tensor train is its bond dimension r. A tensor train with a higher bond dimension means large entropy, and correlations are stored in the function. More specifically, the bond dimension r is the rank between tensor train cores, thereby regulating the expressivity of the approximation. The chief advantage of the tensor train format is its parsimony: we reduce the number of entries from 2^d to $2r^2 d$.

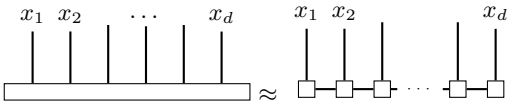

FIG. 5. Approximating the tensor network for $F[x]$ (left) as a tensor train (right). The lower squares correspond to tensor train cores $X^{(1)}, X^{(2)}, \ldots, X^{(d)}$. The intermediate edges represent bonds $s_1, s_2, \ldots, s_{d-1}$.

The TCI algorithm consists of repeatedly sampling $F[x]$ to learn a good tensor train approximation to the tensor $F[x]$. TCI is a generalization of the interpolative matrix decomposition introduced by [23]. Similarly to DMRG, learning the TT cores consists of sweeping across the tensor core sites $1, \ldots, d$ and updating its "pivots." Increasing the number of sweeps n improves the interpolative accuracy. Crucially, TCI requires the ability to evaluate specific entries of $F[x]$ exactly, meaning one has to run an experiment or simulate the quantum system. By restricting ourselves to many-body quantum systems described by a MPS, we can use efficient time evolution algorithms, such as time evolution block decimation (TEBD) [24].

We offer several interpretations of the TT approximation of the cost function. The TT cores introduce latent variables that quantify the dependence between successive control indices. That is, they describe quantum dynamics. The TT ansatz is also natural because we assume the cost function is well approximated by a binary tree structure – successive controls are strongly dependent. In contrast, distant controls behave independently, as quantified by the bond dimension r.

Let us also comment on the entanglement entropy of the tensor train architecture. In analogy with quantum mechanics, the entanglement entropy between two subsystems, A and B, quantifies the amount of correlation between the subsystems [25]. Although $F[x]$ is not a quantum state, we normalize it to treat it as a quasi-probability distribution over classical bits. In that way, the von Neumann entropy S of the objective function quantifies dependence between controls. A simple diagnostic of the architecture consists of determining the entanglement entropy between the two half-chains of F, which we label as subsystem A and B. Fig. 6 illustrates cutting the tensor train into two. Labeling the indices of subsystem A as $x_1, \ldots, x_k$ and B as $x_{k+1}, \ldots, x_d$ al-

lows us to perform the analysis. The control landscape is highly unpredictable if the entropy between the two half-chains is large. Conversely, if the entropy is small, then the cost landscape is highly structured, and one should use an exploitative strategy to find the global optimum.

In practice, we calculate the cost function's entropy by normalizing the tensor $F[x]$, treating it as a quantum state vector. The state is orthogonalized at the center index $k = \lceil \frac{d}{2} \rceil$, whereby we perform a singular value decomposition centered at k and calculate its von Neumann entropy S between the two subsystems.

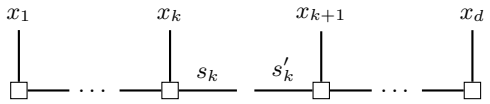

FIG. 6. Cutting the tensor train F into two halves allows us to probe entanglement between the controls. Half-chain A consists of the first k controls $x_1, \ldots, x_k$. The second half-chain B consists of the remaining controls $x_{k+1}, \ldots, x_d$. Intermediate bonds s_k and s'_k represent the additional indices required to reconstruct the full tensor train.

Of course, not every function $F^{x_1,\ldots x_d}$ can be expressed as a tensor train, but evidence shows it can interpolate across a large class of physical quantities. For example, tensor cross interpolation successfully calculated Brillouin zone integrals, evaluated Feynman integrals that suffer a sign problem, and interpolated Besov functions [26–28]. The success of the tensor train and the tensor cross interpolation algorithm is overwhelmingly positive.

C. Searching the tensor train

This section reviews the Optima-TT algorithm for performing approximate inference of a tensor train's global optimum. The algorithm was first introduced in [29], and its computational complexity scales as

$$\mathcal{O}(dKr^2), \tag{13}$$

where d is the number of indices, r is the bond dimension, and K is the number of candidates kept. Crucially, it scales polynomially with d. The algorithm's idea is to interpret $F[x]$ as a probability distribution and perform maximum a posterior (MAP) estimation, which approximates a global optimum. $F[x]$ must be in a tensor train format; otherwise, running the algorithm would be prohibitively expensive. We discuss several parallels between the Optima-TT algorithm and "rollout" techniques typically used in reinforcement learning [30].

We now summarize the Optima-TT algorithm. Fig. 7 shows a sketch of the algorithm. A key advantage to the tensor train format is our ability to "marginalize" indices. For example, we obtain the marginalized cost function by cutting the tensor train into two pieces and summing over the remaining degrees of freedom:

$$F[x_{1:k}] = \sum_{x_{k+1},\ldots,x_d} F[x]. \tag{14}$$

Here, k is the last index of the first chain. By Eq. (12), summations over individual indices are efficient because the TT format is a factorization. When k is small enough, we can feasibly search the marginalized cost function, which contains only 2^k entries. We then prune the tree for $F[x]$ by keeping the $K > 1$ best paths of the reduced $F[x_{1:k}]$. In the next iteration, we, once again, consider the marginalized cost function, but this time, we roll out more indices $F[x_{1:k'}]$, where $k' > k$. We prune the tree by keeping the best K best paths of the reduced cost function. Continuing this algorithm until all indices are exhausted obtains K candidate root-to-leaf paths, of which we choose the optimal one. This procedure is used in Optima-TT and provides a strategy for performing approximate inference on the global optima for the tensor $F[x]$.

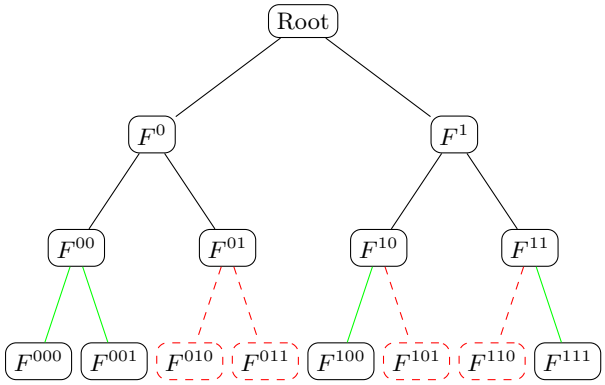

FIG. 7. Sketch of Optima-TT algorithm for searching the optima of the tensor $F^{x_1,\ldots,x_d}$. Suppose we fix the number of nodes to keep to be $K = 4$. First, we expand the tree until there are $K = 4$ nodes: $F^{00}, F^{00}, F^{10}, F^{11}$. Then, we expand the tree again, pruning all but the 4 best nodes, say, $F^{000}, F^{001}, F^{100}, F^{111}$. The pruned edges are red, and the kept edges are green. In practice, we recursively roll out the tree from the remaining nodes and prune until we exhaust all indices.

Optima-TT is similar to the rollout and Monte Carlo Tree Search (MCTS) algorithms used for RL and optimal control, and they have been successful in various settings. However, the ordering of indices of the Optima-TT algorithm is somewhat arbitrary. We currently find the control exploitative or greedy, favoring the initial indices towards the optimum. This problem is akin to the exploitation vs. exploration trade-off in reinforcement learning. One may need to incur short-term penalties before reaching the correct solution. To encourage exploration, we could marginalize a random permutation of the indices instead of sequentially and still obtain an approximate global optimum, but we leave this for a future direction of research. Indeed, a successful modification

of Optima-TT called PROTES was implemented by [31], where one begins to sample highly peaked areas of the probability distribution.

IV. PERFORMANCE

This section provides examples of quantum optimal control problems solved by our tensor network technique. We used the QTTOperations Julia library, built on top of the ITensor software, to perform the tensor cross interpolation and search algorithms [32]. We performed all numerics on an Apple MacBook Air M1 laptop with 8 GB of memory.

A. Two-level system

We verify our technique by controlling a two-level quantum system like a qubit with an external field. We use the initial state $|\psi_0\rangle$ and rotate it to $|\psi_*\rangle$ after a time T, with dynamics generated by the controlled Hamiltonian,

$$H[h(t)] = g\sigma_z + h(t)\sigma_x. \tag{15}$$

The standard Pauli matrices are the operators σ_x, σ_y, and σ_z. Our technique finds solutions for $h(t)$ when the protocol duration T is sufficiently long.

In our first example, we verify that our architecture can discover the optimal protocol for rotating $|\psi_0\rangle = |+z\rangle$ into $|\psi_*\rangle = |+x\rangle$. The shortest-time protocol consists of two steps: (1) fire $h(t)$ at maximum until the spin points along the equator and (2) turn off $h(t)$ until the spin points towards $+x$. The corresponding pulse durations, T_1 and T_2, satisfy

$$\begin{aligned} T_1 &= \frac{1}{2}\left(g^2+h^2\right)^{-1/2}\arccos\left(-g^2/h^2\right) \\ h &= g\cos(2gT_2) + (h^2-g^2)^{1/2}\sin(2gT_2). \end{aligned} \tag{16}$$

Note that these expressions assume $h \geq g > 0$. We can now benchmark these times against the protocols discovered by our architecture. In what follows, we set $|h(t)| \leq 4$ and $g = 1$. We attempt to perform the rotation for a fixed evolution time T. The cost functional to this problem becomes

$$F[h(t)] = -\langle\psi(T)|\,\sigma_x\,|\psi(T)\rangle\,, \tag{17}$$

which attains its minimum of -1 if and only if the terminal state is $|+x\rangle$. We proceed with our optimization strategy by quantizing $h(t)$ as a bitstring x, performing tensor cross interpolation on $F[x]$, and searching the optimal control x^*. Fig. 8 benchmarks our ability to discover optimal protocols for various evolution times T. The results indicate that inside $T \in [0.85, 1.25]$, we can complete the desired qubit rotation with excellent fidelity. At long times, the tensor train format faces difficulty in approximating the cost landscape. Note that $T = 0.85$ coincides with Eq. (16), since $T_1+T_2 = 0.198071+0.659058 = 0.857129$. In Fig. 9, we show that the protocol discovered by our method matches theory.

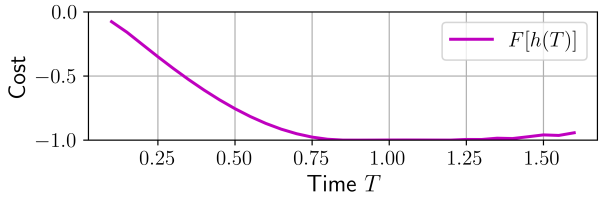

FIG. 8. Performance of the architecture for various evolution times T. Protocols are discovered inside the interval $[0.85, 1.25]$. At early times, the control is insufficiently strong to perform the desired rotation. The tensor cross interpolation or searching algorithm struggles at long times because the protocol duration is long. We set the time-step at $\delta t = 0.05$ and the quantization level at $L = 2$. For TCI, we set the number of sweeps at $n = 3$, the maximum bond dimension of $r = 100$, and the maximum error of 10^{-5}. We search using Optima-TT with the number of kept nodes at $K = 2048 = 2^{11}$.

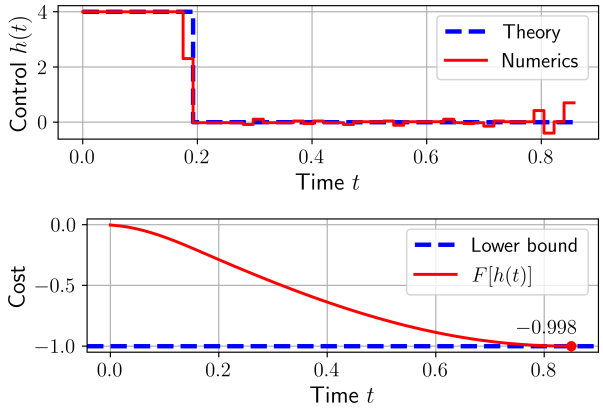

FIG. 9. Protocol discovered for $T = 0.857129$ coincides with theory. Our method achieves a cost of $F = -0.998$, within 0.2% of the lower bound -1. The control jumps at $t \approx 0.2$, near the value for $T_1 = 0.198071$. We quantized the control using $N = 50$ time steps with a quantization level $L = 8$ so that the number of bits is $d = 400$. For TCI, we set the number of sweeps to $n = 3$, the maximum bond dimension to $r = 100$, and the maximum error to 10^{-5}. TCI truncated the tensor train's bond dimension to $r = 5$. During TCI, the number of function calls of $F[x]$ was 39192, much smaller than its $d = 2^{400} \simeq 10^{120}$ entries. Finally, we used $K = 2048 = 2^{11}$ number of kept nodes during Optima-TT. (Not shown: remarkably, by decreasing the number of time steps to $N = 10$ and the quantization level to $L = 2$, we achieved similar performance with only 1415 function calls.)

The second control problem consists of rotating $|-z\rangle$ to $|+z\rangle$ using the same controlled Hamiltonian. We can formulate this problem by finding the optimal control

$h(t)$ that minimizes the loss functional

$$F[h(t)] = -\langle\psi(T)|\,\sigma_z\,|\psi(T)\rangle\,. \tag{18}$$

Similarly, the cost function attains its minimum value -1 if and only if the terminal state is $|+z\rangle$. In Fig. 10, we diagnose our ability to perform the desired rotation at various evolution times T.

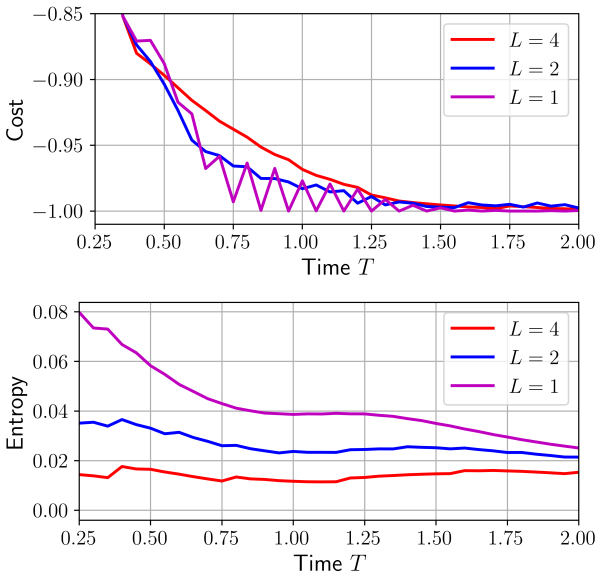

FIG. 10. Diagnostics of the architecture's performance at various protocol durations T and quantization levels L. (Upper plot) The state's terminal cost $F[h(T)]$. All quantization levels L perform the rotation, as indicated by achieving $F = -1$. But surprisingly, the most straightforward architecture ($L = 1$) discovers the shortest time protocol. (Lower plot) The architecture's entropy $S(T)$ for various evolution times T. The entropy is equal to the von Neuman entropy between the half-chains of the tensor train of F. Entropy is largest at short times. The fact that the entropy is relatively small is a sign that the cost landscape contains structure.

Some results for the $|-z\rangle$ to $|+z\rangle$ protocol is surprising. TCI and Optima-TT succeed in learning the cost function and performing approximate inference. However, it does so more accurately using a smaller quantization level of L. For example, when $L = 1$, such as a bang-bang protocol, the protocol duration T is smaller than $L = 2, 4$. That means that when L is large, TCI may overfit the cost landscape or that Optima-TT finds an incorrect optimum. However, that means fewer parameters are needed to interpolate the cost function.

We can understand the protocol by focusing on a particular solution for the $|-z\rangle$ to $|+z\rangle$ control problem. We display the $T = 1.25$ protocol, cost function, and trajectory in Fig. 11. The protocol consists of firing the control at $h = -4$ for a fixed time and letting the qubit rotate into the $+z$ with non-trivial minor corrections. If one had used a greedy approach to the control problem, one would not have discovered this protocol because the control incurred short-term costs to accomplish the task. Our protocol exemplifies the tension between exploration and exploitation in control problems.

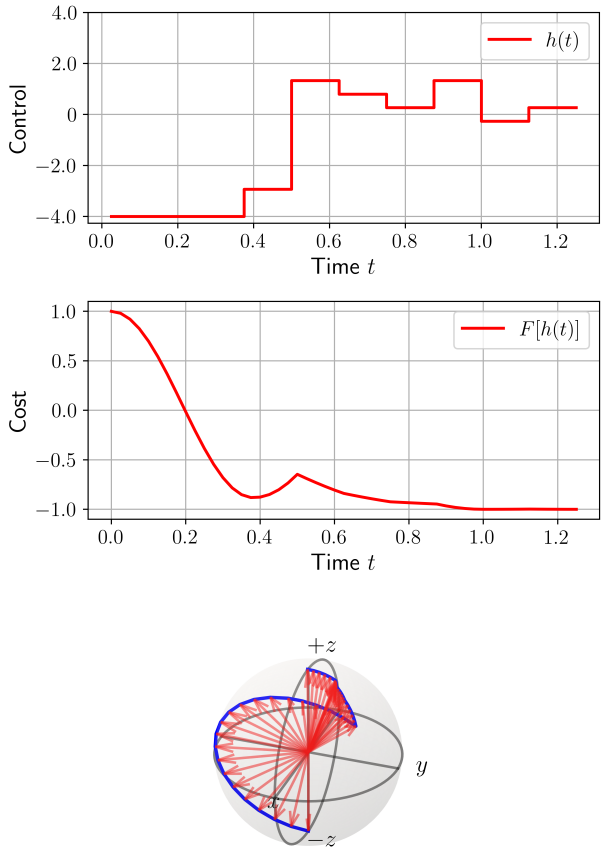

FIG. 11. A solution to the single qubit optimal control problem for $T = 1.25$. (Top) Protocol discovered using the technique. The control field $h(t)$ is quantized as a bitstring x ($N = 10$ and $L = 4$) and the cost function $F[x]$ contains 2^{40} entries. To perform TCI, we set the number of sweeps as $n = 4$, the maximum bond dimension as $r = 100$, and the maximum error cutoff as 10^{-9}. To perform TT-Optima, we set the number of samples to keep as $K = 2048$. (Middle) The cost function decreases and then incurs short-term penalties before steering into the target state. The rise in costs displays our technique's ability to discover non-greedy protocols. (Bottom) Bloch sphere diagram of the qubit's trajectory.

B. Mixed-field Ising chain

In this subsection, we apply the technique for controlling a many-body quantum system: the mixed-field Ising model on a ring. This system is more complicated because interactions increase the state's entropy. The tensor cross interpolation could have difficulty creating a latent space that describes the cost function. We find ev-

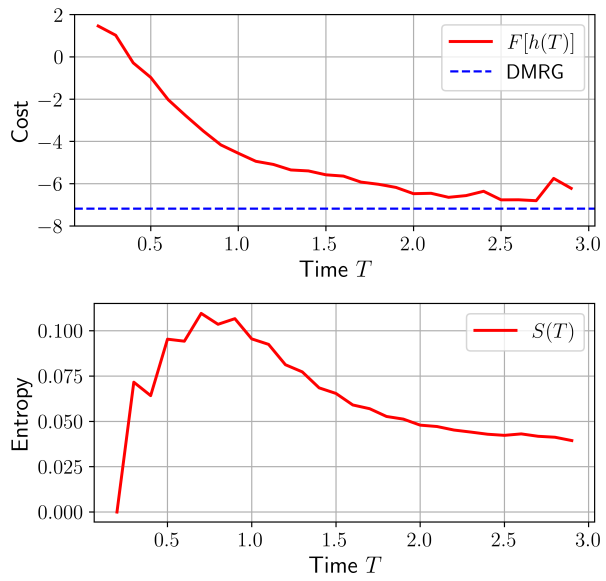

FIG. 12. Architecture diagnostic. (Upper plot) The state's terminal cost $F[h(T)]$ for various evolution times T. TCI learned the cost function for each evolution time, and Optima-TT was run to obtain the optimal control. This system is more challenging to control; it achieves the best cost of $F = -6.80$ at evolution time $T = 2.7$, which is within 6% of the target ground state energy of -7.17, as obtained with DMRG. At early times, the control has insufficient energy to perform the rotation, while at late times, the system's interactions compete with our ability to control it. (Lower plot) The architecture's entropy $S(T)$ for various evolution times T. The entropy is equal to the von Neuman entropy between the half-chains of the tensor train. In the time region where no control exists, the architecture's entropy is largest, signaling that TCI requires many degrees of freedom to parameterize the cost function.

idence that our tensor train technique can efficiently find approximate protocols for preparing many-body quantum states with up to 98% single-site fidelity.

The controlled Hamiltonian of interest is

$$H[h(t)] = J\sum_{j=1}^{L_{\text{sys}}}\sigma_j^z\sigma_{j+1}^z + g\sum_{j=1}^{L_{\text{sys}}}\sigma_j^z + h(t)\sum_{j=1}^{L_{\text{sys}}}\sigma_j^x, \quad (19)$$

where $L_{\text{sys}} = 6$ is the number of particles, $J = -1$, $g = -1$, and $h(t)$ is the external control. Our task is to rotate the ground state of $H[h = 2]$ to the ground state of $H[h = -2]$ using the control $h(t) \in [-4, 4]$ in an evolution time T. Reinforcement learning has solved this control problem using Q-learning, which we use to benchmark results [7]. For this optimal control problem, we assign the cost functional

$$F[h(t)] = \langle\psi(T)|\, H\, |\psi(T)\rangle\,. \quad (20)$$

Here, $H = H[h = 2]$ is the Hamiltonian for the target ground state, and $|\psi(T)\rangle$ is the terminal state after evolving under the protocol $h(t)$ for a duration T. We approximate the cost functional $F[h(t)]$ using the tensor cross interpolation and perform an approximate inference of its optimum. In Fig. 12, we show our method's ability to approximate the optimal control using bang-bang ($L = 1$) protocols for various evolution times T. Our results show that the terminal state can discover protocols close to the ground state with few resources. Moreover, the entropy of the tensor train approximation to the cost function is relatively stable, showing that the control landscape contains structure.

Let us focus on the optimal control for an evolution time $T = 2.7$. We show how the state evolves during the protocol in Fig. 13. Tensor cross interpolation approximates the control landscape with only 6359 function calls and a maximum bond dimension of $r = 16$, even though the cost function $F[x]$ contains $2^{27} \approx 10^8$ entries.

V. CONCLUSION

Our work shows how the tensor train format can interpolate across the cost function associated with quantum optimal control problems. When this cost function admits the tensor train format, its optima can be approximated with algorithms developed for tensor trains on a digital computer. Our results teach us that optimal control problems that may suffer from the curse of dimensionality can be approximately solved using tensor networks. The technique offers an alternative method of solving control problems because it is (1) model-free, (2) parsimonious, and (3) derivative-free.

More precisely, our method can learn optimal protocols to prepare states for systems of a single qubit and that of the one-dimensional mixed field Ising model on a ring. Furthermore, our work provides evidence that bang-bang protocols are often sufficient to perform optimal control.

In this paper, we ignored quantum systems that can not be efficiently simulated on digital devices. (Using a digital computer, we evaluated the cost function $F[x]$ by simulation.) In future work, we aim to run the quantum circuit in an experiment when performing the tensor cross-interpolation algorithm, thereby creating a feedback loop. Once the tensor train approximation of the control problem's cost function is learned, it can be optimized offline on a digital computer. We hope this method allows us to train variational quantum algorithms or perform QAOA with significantly fewer shots. In the context of Floquet engineering, one could optimize the periodic driving for a desired task.

This work bridges theoretical advancements of the quantics tensor train format and optimal control. We hope that tensor networks when applied to control problems, provide a new approximate solution to overcoming the curse of dimensionality.

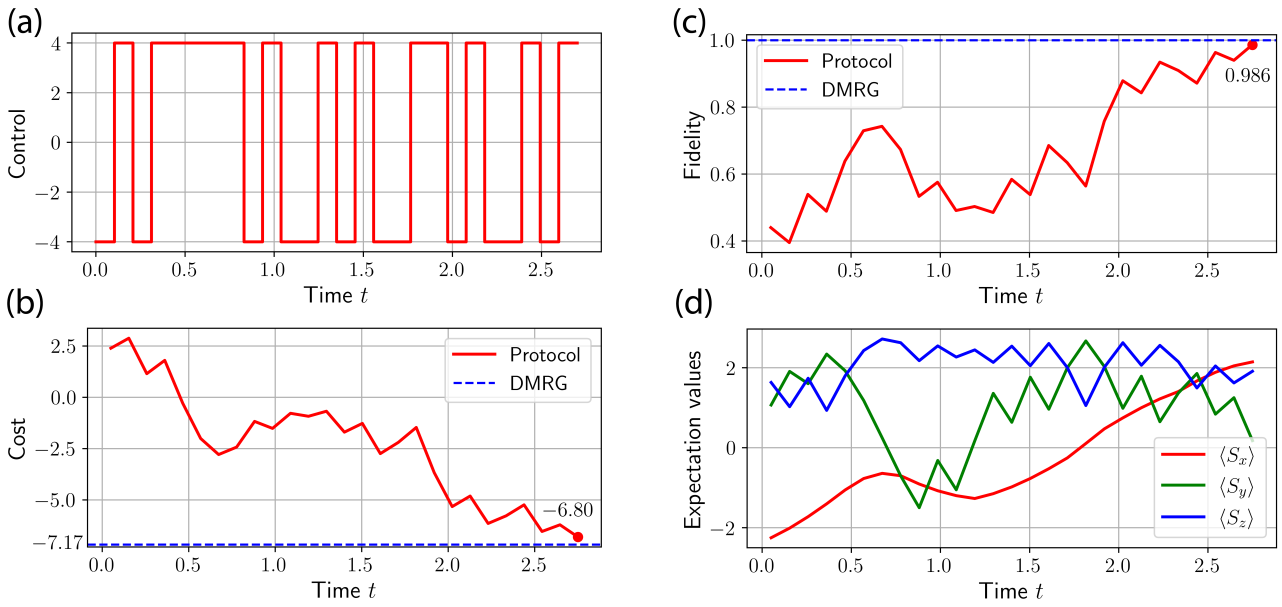

FIG. 13. A solution to the mixed field Ising model control problem for $T = 2.7$. (a) Protocol discovered using the technique. The control field $h(t)$ is quantized ($N = 27$ and $L = 1$) and the cost function $F[x]$ contains 2^{27} entries. To perform TCI, we set the number of sweeps as $n = 2$, the maximum bond dimension as $r = 100$, and the maximum error cutoff as 10^{-3}. During TCI, the bond dimension took a maximum value of 16 and 6359 function calls. To perform TT-Optima, we set the number of samples to keep as $K = 2048$. (b) Cost function as the protocol steers the system into the target state. (c) Single site fidelity (compared to the DMRG ground state) during the protocol. Single site fidelity is calculated with $|\langle\psi_*|\psi(T)\rangle|^{2/L_{\text{sys}}}$, where $|\psi_*\rangle$ is the DMRG ground state and L_{sys} is the number of sites. (d) Expectation values of the system's trajectory during the protocol. The protocol correctly rotates the system's $\langle S_x \rangle$ component, but other quantities tend to fluctuate.

VI. ACKNOWLEDGMENTS

The authors are grateful for ongoing support through ...

[1] Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018). URL https://doi.org/10.22331/q-2018-08-06-79.

[2] Werschnik, J. & Gross, E. Quantum optimal control theory. *Journal of Physics B: Atomic, Molecular and Optical Physics* **40**, R175 (2007).

[3] Khaneja, N., Reiss, T., Kehlet, C., Schulte-Herbrüggen, T. & Glaser, S. J. Optimal control of coupled spin dynamics: design of nmr pulse sequences by gradient ascent algorithms. *Journal of Magnetic Resonance* **172**, 296–305 (2005). URL https://www.sciencedirect.com/science/article/pii/S1090780704003696.

[4] Doria, P., Calarco, T. & Montangero, S. Optimal control technique for many-body quantum dynamics. *Physical review letters* **106**, 190501 (2011).

[5] Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction* (MIT press, 2018).

[6] Zhang, X.-M., Wei, Z., Asad, R., Yang, X.-C. & Wang, X. When does reinforcement learning stand out in quantum control? a comparative study on state preparation. *npj Quantum Information* **5**, 85 (2019).

[7] Bukov, M. *et al.* Reinforcement learning in different phases of quantum control. *Phys. Rev. X* **8**, 031086 (2018). URL https://link.aps.org/doi/10.1103/PhysRevX.8.031086.

[8] Metz, F. & Bukov, M. Self-correcting quantum many-body control using reinforcement learning with tensor networks. *Nature Machine Intelligence* **5**, 780–791 (2023).

[9] Oseledets, I. & Tyrtyshnikov, E. Tt-cross approximation for multidimensional arrays. *Linear Algebra and its Applications* **432**, 70–88 (2010). URL https://www.sciencedirect.com/science/article/pii/S0024379509003747.

[10] Chertkov, A., Ryzhakov, G., Novikov, G. & Oseledets, I. Optimization of functions given in the tensor train format. *arXiv preprint arXiv:2209.14808* (2022).

[11] Orús, R. Tensor networks for complex quantum systems. *Nature Reviews Physics* **1**, 538–550 (2019).

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

[13] Ran, S.-J. *et al.* *Tensor network contractions: methods and applications to quantum many-body systems* (Springer Nature, 2020).

[14] Cirac, J. I., Perez-Garcia, D., Schuch, N. & Verstraete, F. Matrix product states and projected entangled pair states: Concepts, symmetries, theorems. *Reviews of Modern Physics* **93**, 045003 (2021).

[15] Khoromskij, B. N. O(dlogn)-quantics approximation of n-d tensors in high-dimensional numerical modeling. *Constructive Approximation* **34**, 257–280 (2011). URL http://dx.doi.org/10.1007/s00365-011-9131-1.

[16] Oseledets, I. V. Tensor-train decomposition. *SIAM Journal on Scientific Computing* **33**, 2295–2317 (2011).

[17] Cerezo, M. *et al.* Variational quantum algorithms. *Nature Reviews Physics* **3**, 625–644 (2021).

[18] McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R. & Neven, H. Barren plateaus in quantum neural network training landscapes. *Nature Communications* **9** (2018). URL http://dx.doi.org/10.1038/s41467-018-07090-4.

[19] Ortiz Marrero, C., Kieferová, M. & Wiebe, N. Entanglement-induced barren plateaus. *PRX Quantum* **2**, 040316 (2021). URL https://link.aps.org/doi/10.1103/PRXQuantum.2.040316.

[20] Hastings, M. B. Solving gapped hamiltonians locally. *Phys. Rev. B* **73**, 085115 (2006). URL https://link.aps.org/doi/10.1103/PhysRevB.73.085115.

[21] Paeckel, S. *et al.* Time-evolution methods for matrix-product states. *Annals of Physics* **411**, 167998 (2019). URL http://arxiv.org/abs/1901.05824. ArXiv:1901.05824 [cond-mat, physics:quant-ph].

[22] Savostyanov, D. V. Quasioptimality of maximum-volume cross interpolation of tensors. *Linear Algebra and its Applications* **458**, 217–244 (2014). URL https://www.sciencedirect.com/science/article/pii/S0024379514003711.

[23] Cheng, H., Gimbutas, Z., Martinsson, P.-G. & Rokhlin, V. On the compression of low rank matrices. *SIAM J. Sci. Comput.* **26**, 1389–1404 (2005). URL https://api.semanticscholar.org/CorpusID:2146696.

[24] Vidal, G. Efficient simulation of one-dimensional quantum many-body systems. *Phys. Rev. Lett.* **93**, 040502 (2004). URL https://link.aps.org/doi/10.1103/PhysRevLett.93.040502.

[25] Nielsen, M. A. & Chuang, I. L. *Cambridge series on information and the natural sciences: Quantum computation and quantum information* (Cambridge University Press, Cambridge, England, 2000).

[26] Ritter, M. K. *et al.* Quantics tensor cross interpolation for high-resolution parsimonious representations of multivariate functions. *Phys. Rev. Lett.* **132**, 056501 (2024). URL https://link.aps.org/doi/10.1103/PhysRevLett.132.056501.

[27] Núñez Fernández, Y. *et al.* Learning feynman diagrams with tensor trains. *Phys. Rev. X* **12**, 041018 (2022). URL https://link.aps.org/doi/10.1103/PhysRevX.12.041018.

[28] Ali, M. & Nouy, A. Approximation theory of tree tensor networks: Tensorized univariate functions. *Constructive Approximation* **58**, 463–544 (2023).

[29] Zheltkov, D. A. & Osinsky, A. Global optimization algorithms using tensor trains. In *Large-Scale Scientific Computing: 12th International Conference, LSSC 2019, Sozopol, Bulgaria, June 10–14, 2019, Revised Selected Papers 12*, 197–202 (Springer, 2020).

[30] Tesauro, G. & Galperin, G. On-line policy improvement using monte-carlo search. In Mozer, M., Jordan, M. & Petsche, T. (eds.) *Advances in Neural Information Processing Systems*, vol. 9 (MIT Press, 1996). URL https://proceedings.neurips.cc/paper_files/paper/1996/file/996009f2374006606f4c0b0fda878af1-Paper.pdf.

[31] Batsheva, A., Chertkov, A., Ryzhakov, G. & Oseledets, I. Protes: probabilistic optimization with tensor sampling. *Advances in Neural Information Processing Systems* **36**, 808–823 (2023).

[32] Fishman, M., White, S. R. & Stoudenmire, E. M. The ITensor Software Library for Tensor Network Calculations. *SciPost Phys. Codebases* 4 (2022). URL https://scipost.org/10.21468/SciPostPhysCodeb.4.