Sample-efficient Model-based Reinforcement Learning for Quantum Control

Irtaza Khalid[1,†], Carrie Weidner[2], S. G. Schirmer[3], Edmond Jonckheere[4], Frank C. Langbein[1]

[1]Cardiff University, [2]QETLabs, University of Bristol, [3]Swansea University, [4]University of Southern California

[†]khalidmi@cardiff.ac.uk

https://qyber.black/

TL;DR

1. Our model-based reinforcement learning (RL) algorithm **reduces the sample complexity** for time-dependent noisy quantum gate control tasks by at least **an order of magnitude** over model-free RL.
2. The model is a differentiable ordinary differential equation (ODE) [1] within our **Learnable Hamiltonian Model-Based Soft-Actor Critic [2, 3]** (LH-MBSAC) algorithm.
3. We encode a **partially characterised Hamiltonian** in the model and only learn the time-independent term.
4. The **learned model can be leveraged** to further optimize RL controllers using GRAPE [4].
5. LH-MBSAC is a step towards bridging the gap between theoretical and experimental quantum control by reducing the experimental resource requirements for RL control.

Quantum Control Problem

We use the master equation [5] to model the noisy gate control problem. Control functions $\mathbf{u}(t)$ are piecewise constant in time in the propagator superoperator $\mathbf{E}$,

$$\mathbf{E}(t, \mathbf{u}(t)) := \mathbf{E}(\mathbf{u}_m) = \prod_{l=1}^{m} \exp\left(-\frac{i}{\hbar}\Delta t \mathbf{G}(t_l, \mathbf{u}(t_l))\right) \tag{1}$$

for m fixed timesteps of size $\Delta t = T/N$ where T is a final time with maximum number of timesteps N; $\mathbf{G}$ is the open/closed system dynamics' generator. The control problem is

$$\mathbf{u}_m^* = \underset{\mathbf{u}_m=[u_1,\ldots,u_m]\in\mathbb{X}, m\le N}{\arg\max} \overbrace{\mathrm{Tr}\left[\mathbf{\Phi}(\mathbf{E}(\mathbf{u}_m))^\dagger \mathbf{\Phi}(\mathbf{E}_{\text{target}})\right]}^{\text{Fidelity } \mathcal{F}\in[0,1]} \tag{2}$$

where $\mathbf{\Phi}(\mathbf{E})$ is the Choi form [6] of $\mathbf{E}$ estimated using ancilla assisted process tomography [7] using $O(3^L)$ binomial observables for an L-qubit system. The Hamiltonian is parametrised in the Pauli basis with learnable coefficients ζ. We also assume the control Hamiltonians (H_c) to be known.

$$H_\zeta(\mathbf{u}(t), t) = \sum_{l=1}^{n^2} \zeta_l P_l + H_c(\mathbf{u}(t), t) \tag{3}$$

Model-based Reinforcement Learning

The control problem in Eq. (2) can be formulated as a Markov Decision Problem (MDP) by sequentially constructing the control amplitudes as actions, using the propagator as the state with the reward being the fidelity $\mathcal{F}$:

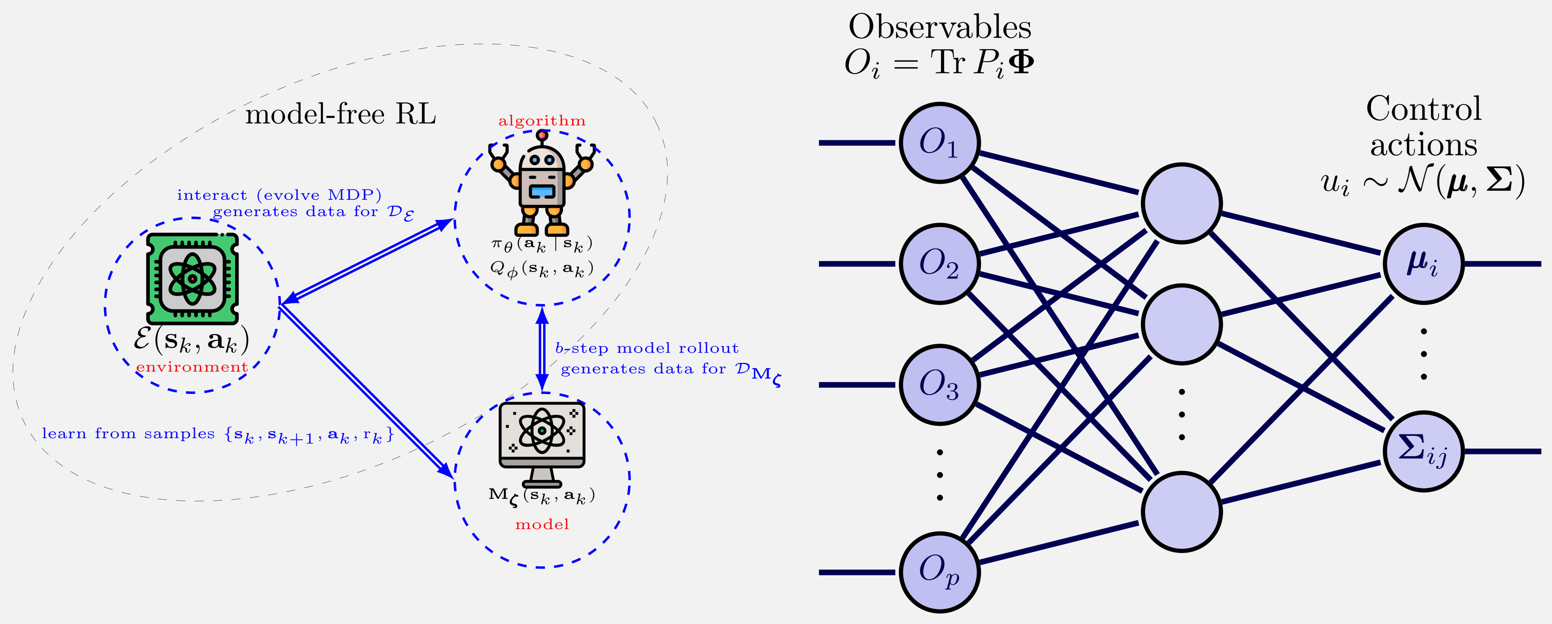
$$\mathbf{a}_k = u_k, \tag{4a}$$

$$\mathbf{s}_k = \prod_{l=1}^{k} \exp\left(-\frac{i}{\hbar}\Delta t \mathbf{G}(t_l, u_l)\right), \tag{4b}$$

$$\mathrm{r}_k = \mathcal{F}(\mathbf{E}(\mathbf{u}_k), \mathbf{E}_{\text{target}}). \tag{4c}$$

The model $\mathbf{M}_\zeta$ is a differentiable ODE whose generator is interpretable and has the form given by H_ζ in Eq. (3). It is used to make propagator predictions and is trained using MDP data D collected from the controllable system (environment) $\mathcal{E}$ by minimizing

$$L_{\text{model}}(D) = \sum_{D} \left(\mathbf{M}_\zeta(\mathbf{s}_k, k) - \mathbf{s}_{k+1}\right)^2. \tag{5}$$



(a) Model-based RL **(b) Policy function** $\pi_\theta(\mathbf{a}_k \mid \mathbf{s}_k)$

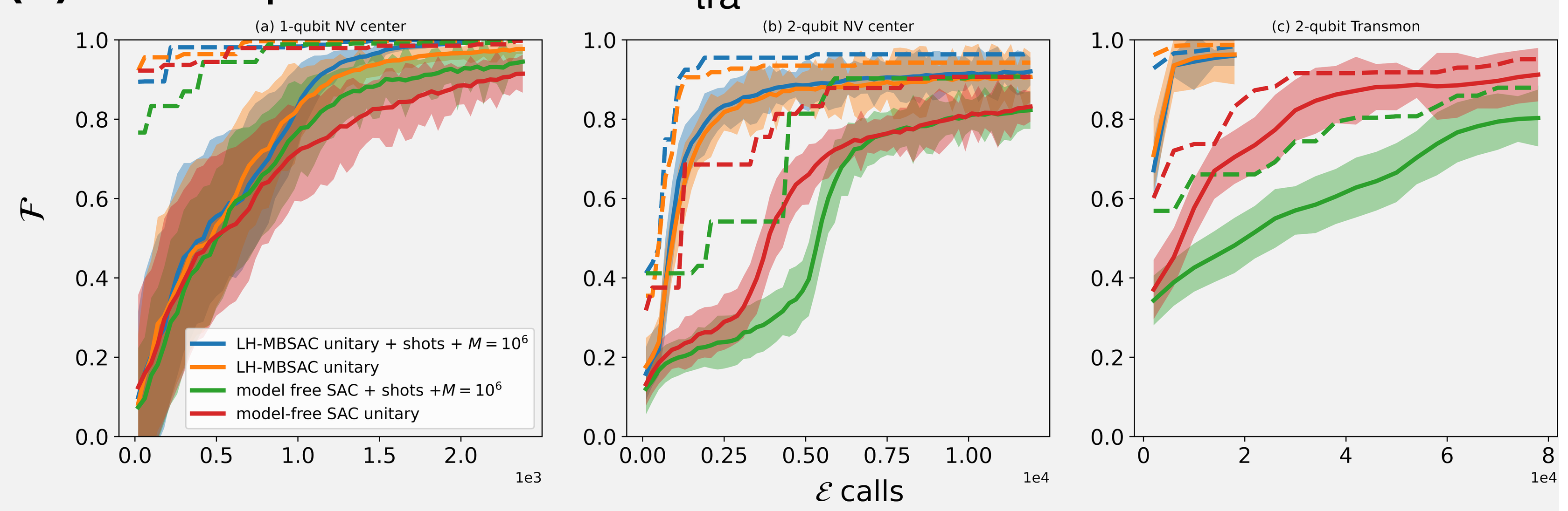
(a) In model-based RL, an agent π_θ interacts with the controllable system (environment) to collect data $\mathbf{s}_k, \mathbf{s}_{k+1}, \mathbf{a}_k, \mathrm{r}_k$ in model-free fashion. These data are utilised to train the model $\mathbf{M}_\zeta(\mathbf{s}_k, \mathbf{a}_k)$ until a quality measure plateaus, indicating training completion. Lastly, synthetic data are generated through a b-step rollout with π_θ interacting with the $\mathbf{M}_\zeta$ b times to train π_θ. **(b)** The policy function gets the propagator in $\mathbf{\Phi}$ form as input and outputs a distribution of next-step actions.

References

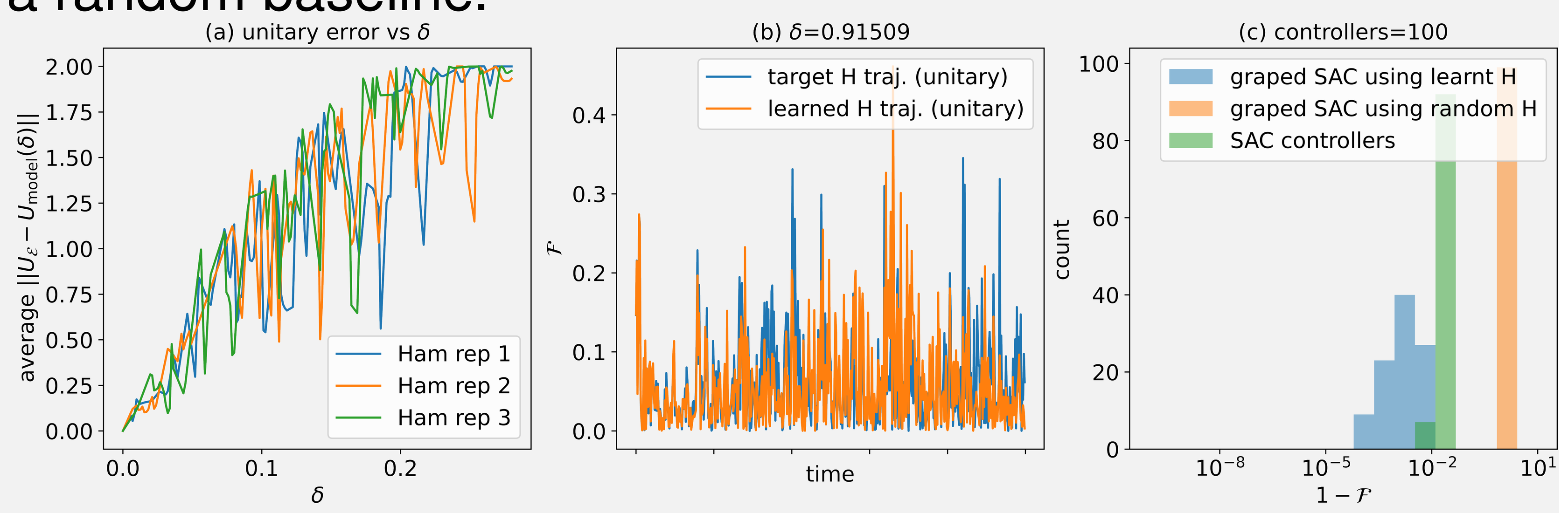
[1] RTQ Chen et al. Neural ordinary differential equations. In *NeurIPS*, volume 31, 2018.
[2] T Haarnoja et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Int. Conf. Machine Learning*, pages 1861–1870. PMLR, 2018.
[3] M Janner et al. When to trust your model: Model-based policy optimization, 2019. arXiv:1906.08253.
[4] N Khaneja et al. Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms. *J Magnetic Resonance*, 172(2):296–305, 2005.
[5] HP Breuer et al. *The theory of open quantum systems*. Oxford University Press, 2002.
[6] MD Choi. Completely positive linear maps on complex matrices. *Linear algebra and its applications*, 10(3):285–290, 1975.
[7] JB Altepeter et al. Ancilla-assisted quantum process tomography. *Phys. Rev. Lett.*, 90:193601, May 2003.

Results

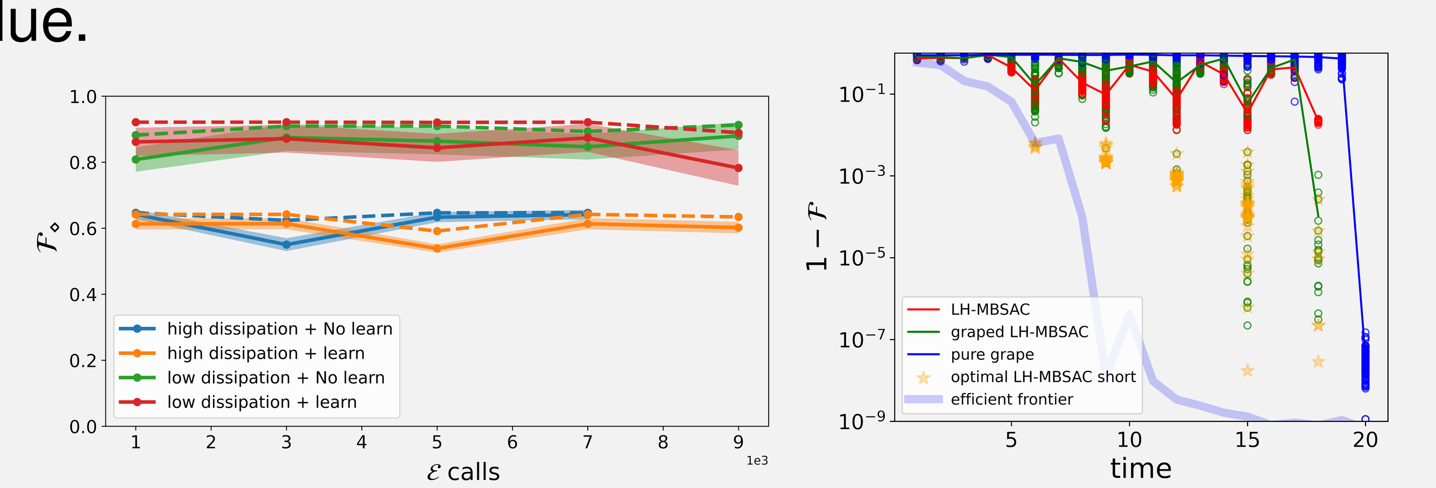
Sample complexity improvement: Fidelity $\mathcal{F}$ of a Hadamard gate for **(a)** a single-qubit nitrogen vacancy (NV) center; a CNOT gate for **(b)** a two-qubit NV center H_{NV} and **(c)** a two-qubit Transmon $H_{\text{tra}}^{(2)}$ as a function of $\mathcal{E}$ calls.



Leveraging the learned model: (a) A non-linear relationship between unitary model prediction error and model error δ shown for the two-qubit transmon. **(b)** $\delta \neq 0$? No problem: ODE trajectories are close but not identical. **(c)** Despite (b), using GRAPE with $\mathbf{M}_\zeta$ significantly improves $\mathcal{F}$ compared to a random baseline.



On open systems and finding short time pulses: (a) Sample complexity for low/high decoherence regimes. Learning decoherence + $\mathbf{M}_\zeta$ or just $\mathbf{M}_\zeta$ yield equivalent performances. **(b)** Optimal short pulses by truncating RL pulse parameters with the Pareto optimal frontier highlighted in blue.



(a) (b)