Towards improved sample complexity using a quantum model based SAC

I. Khalid[1], C. A. Weidner[2], E. A. Jonckheere[3], S. G. Schirmer[4], F. Langbein[5]

[1,5]Cardiff University, [2]University of Bristol, [3]University of Southern California, [4]Swansea University

August 2022

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part
  $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part
  $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{-i \int_0^t dt' \; H_u(t')\right\}$

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part
  $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{-i \int_0^t dt' \; H_u(t')\right\}$
- In a sense, an inverse problem. Given some target $U(t)$, we try to engineer a trajectory $U(0) \xrightarrow{u} U(t)$, from some initial state $U(0)$ using only the control scheme $u(t)$

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{-i \int_0^t dt' \; H_u(t')\right\}$
- In a sense, an inverse problem. Given some target $U(t)$, we try to engineer a trajectory $U(0) \xrightarrow{u} U(t)$, from some initial state $U(0)$ using only the control scheme $u(t)$
- Why?

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{-i \int_0^t dt' \; H_u(t')\right\}$
- In a sense, an inverse problem. Given some target $U(t)$, we try to engineer a trajectory $U(0) \xrightarrow{u} U(t)$, from some initial state $U(0)$ using only the control scheme $u(t)$
- Why?
  - Realize more accurate/robust gates (quantum operations)

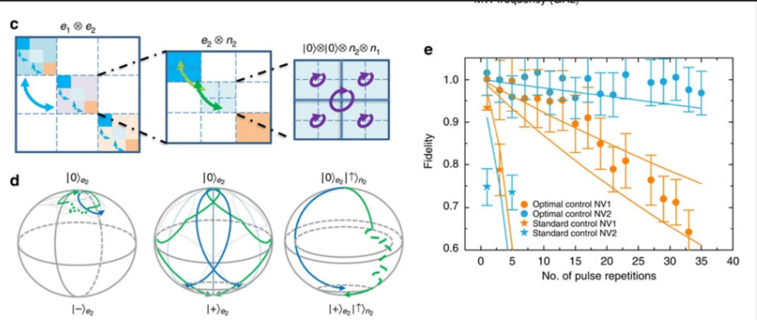
Reminder: Utility of quantum control

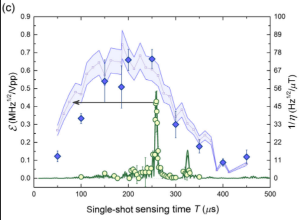
- Control: System = Controllable part + Not controllable part
  $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{-i \int_0^t dt' \; H_u(t')\right\}$
- In a sense, an inverse problem. Given some target $U(t)$, we try to engineer a trajectory $U(0) \xrightarrow{u} U(t)$, from some initial state $U(0)$ using only the control scheme $u(t)$
- Why?
  - Realize more accurate/robust gates (quantum operations)
  - Prepare appropriate few/many body system states (enable quantum science and technology)

Reminder: Utility of quantum control

- Control: System = Controllable part + Not controllable part
  $H(t) = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u$
- general goal find some control path $\{u(t)\}_{\mathcal{C}}$ that allows us to realize $U(t) = \mathcal{T} \exp\left\{ -i \int_0^t dt' \; H_u(t') \right\}$
- In a sense, an inverse problem. Given some target $U(t)$, we try to engineer a trajectory $U(0) \xrightarrow{u} U(t)$, from some initial state $U(0)$ using only the control scheme $u(t)$
- Why?
  - Realize more accurate/robust gates (quantum operations)
  - Prepare appropriate few/many body system states (enable quantum science and technology)
  - metrology applications

Diamond spins ↑↑↓





[1]

[2]

[1]F. Dolde et. al. (Nature 2014)
[2]F. Poggiali et. al. (PRX 2018)

Definitions

- Constraint 1: Transmon H (control structure H_u fixed) (Magesan et. al. 2020)

$$
\begin{aligned}
H(t)/\hbar &= H_0 + \sum_{u \in \mathcal{C}} u(t) H_u && (1)\\
&= \sum_{j=1}^{2} \omega_j \hat{b}_j^\dagger \hat{b}_j + \frac{\delta_j}{2} \hat{b}_j^\dagger \hat{b}_j (\hat{b}_j^\dagger \hat{b}_j - 1) && (2)\\
& J \sum_{j=1}^{2} (\hat{b}_j^\dagger \hat{b}_{j+1} + \hat{b}_j \hat{b}_{j+1}^\dagger) + \sum_{j=1}^{2} \Delta_j(t)(\hat{b}_j + \hat{b}_j^\dagger)
\end{aligned}
$$

Definitions

- Constraint 1: Transmon H (control structure H_u fixed) (Magesan et. al. 2020)

$$H(t)/\hbar = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u \tag{1}$$

$$= \sum_{j=1}^{2} \omega_j \hat{b}_j^\dagger \hat{b}_j + \frac{\delta_j}{2} \hat{b}_j^\dagger \hat{b}_j (\hat{b}_j^\dagger \hat{b}_j - 1) \tag{2}$$

$$J \sum_{j=1}^{2} (\hat{b}_j^\dagger \hat{b}_{j+1} + \hat{b}_j \hat{b}_{j+1}^\dagger) + \sum_{j=1}^{2} \Delta_j(t)(\hat{b}_j + \hat{b}_j^\dagger)$$

- Constraint 2: Piece-wise constant pulse control $u_j(t_i) = \Delta_j(t_i)$ where $t_i = t_0 + T/N$ for some N timesteps

Definitions

- Constraint 1: Transmon H (control structure H_u fixed) (Magesan et. al. 2020)

$$H(t)/\hbar = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u \tag{1}$$

$$= \sum_{j=1}^{2} \omega_j \hat{b}_j^\dagger \hat{b}_j + \frac{\delta_j}{2} \hat{b}_j^\dagger \hat{b}_j (\hat{b}_j^\dagger \hat{b}_j - 1) \tag{2}$$

$$J \sum_{j=1}^{2} (\hat{b}_j^\dagger \hat{b}_{j+1} + \hat{b}_j \hat{b}_{j+1}^\dagger) + \sum_{j=1}^{2} \Delta_j(t)(\hat{b}_j + \hat{b}_j^\dagger)$$

- Constraint 2: Piece-wise constant pulse control $u_j(t_i) = \Delta_j(t_i)$ where $t_i = t_0 + T/N$ for some N timesteps
- Constraint 3: Schrödinger or Lindblad evolution. We want a gate V via pulse trajectory E_Δ

Definitions

- Constraint 1: Transmon H (control structure H_u fixed) (Magesan et. al. 2020)

$$H(t)/\hbar = H_0 + \sum_{u \in \mathcal{C}} u(t) H_u \tag{1}$$

$$= \sum_{j=1}^{2} \omega_j \hat{b}_j^\dagger \hat{b}_j + \frac{\delta_j}{2} \hat{b}_j^\dagger \hat{b}_j (\hat{b}_j^\dagger \hat{b}_j - 1) \tag{2}$$

$$J \sum_{j=1}^{2} (\hat{b}_j^\dagger \hat{b}_{j+1} + \hat{b}_j \hat{b}_{j+1}^\dagger) + \sum_{j=1}^{2} \Delta_j(t)(\hat{b}_j + \hat{b}_j^\dagger)$$

- Constraint 2: Piece-wise constant pulse control $u_j(t_i) = \Delta_j(t_i)$ where $t_i = t_0 + T/N$ for some N timesteps
- Constraint 3: Schrödinger or Lindblad evolution. We want a gate V via pulse trajectory E_Δ
- Formal Goal:

$$\boldsymbol{\Delta}^* = \operatorname*{arg\,max}_{\boldsymbol{\Delta}=[\Delta_1,\ldots,\Delta_m], m \le N} \mathcal{F}(E_{\boldsymbol{\Delta}}, V) \tag{3}$$

MDP (Model Free)

- states, next states, actions, rewards $(\mathcal{S}, \mathcal{S}', \mathcal{A}, \mathcal{R})$

MDP (Model Free)

- states, next states, actions, rewards $(\mathcal{S}, \mathcal{S}', \mathcal{A}, \mathcal{R})$
- Objective function

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) \right] \tag{4}$$

MDP (Model Free)

- states, next states, actions, rewards $(\mathcal{S}, \mathcal{S}', \mathcal{A}, \mathcal{R})$
- Objective function

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) \right] \tag{4}$$

- Assign

$$\mathbf{a}_k = \Delta_k \tag{5a}$$

$$\mathbf{s}_k = E_{\mathbf{\Delta}_{:k}} = E_{\Delta_k} \cdots E_{\Delta_2} \cdot E_{\Delta_1} \tag{5b}$$

$$\mathbf{r}_k = \mathcal{F}(E_{\mathbf{\Delta}_{:k}}, V) \tag{5c}$$

MDP (Model Free)

- states, next states, actions, rewards $(\mathcal{S}, \mathcal{S}', \mathcal{A}, \mathcal{R})$
- Objective function

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) \right] \tag{4}$$

- Assign

$$\mathbf{a}_k = \Delta_k \tag{5a}$$

$$\mathbf{s}_k = E_{\boldsymbol{\Delta}_{:k}} = E_{\Delta_k} \cdots E_{\Delta_2} \cdot E_{\Delta_1} \tag{5b}$$

$$\mathbf{r}_k = \mathcal{F}(E_{\boldsymbol{\Delta}_{:k}}, V) \tag{5c}$$

- Note reward $\mathbf{r}_k$ sampled by taking action $\mathbf{a}_k$ after observing $\mathbf{s}_k$ in $\mathbf{r}_k, \mathbf{s}_{k+1} \sim \mathcal{E}(\mathbf{a}_k, \mathbf{s}_k)$

MDP (Model Free)

- states, next states, actions, rewards $(\mathcal{S}, \mathcal{S}', \mathcal{A}, \mathcal{R})$
- Objective function

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) \right] \tag{4}$$

- Assign

$$\mathbf{a}_k = \Delta_k \tag{5a}$$

$$\mathbf{s}_k = E_{\boldsymbol{\Delta}_{:k}} = E_{\Delta_k} \cdots E_{\Delta_2} \cdot E_{\Delta_1} \tag{5b}$$

$$\mathbf{r}_k = \mathcal{F}(E_{\boldsymbol{\Delta}_{:k}}, V) \tag{5c}$$

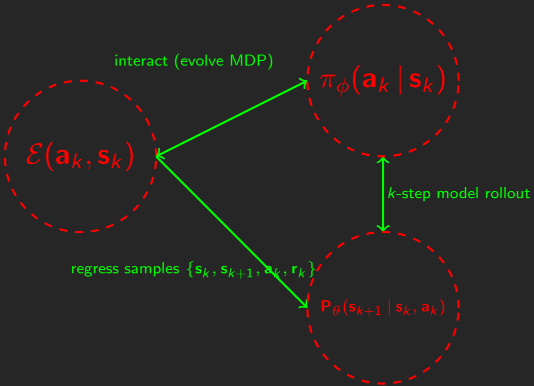
- Note reward $\mathbf{r}_k$ sampled by taking action $\mathbf{a}_k$ after observing $\mathbf{s}_k$ in $\mathbf{r}_k, \mathbf{s}_{k+1} \sim \mathcal{E}(\mathbf{a}_k, \mathbf{s}_k)$
- So, far only two things interacting: policy $\pi(\mathbf{a}_k \mid \mathbf{s}_k)$ and $\mathcal{E}(\mathbf{a}_k, \mathbf{s}_k)$

Model addition

- Suppose the agent is now able to store and interact with an internal representation of $\mathcal{E}$. Let's denote that with $\hat{\mathbf{P}}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k) = \hat{\mathcal{E}}_\theta$.

Model addition

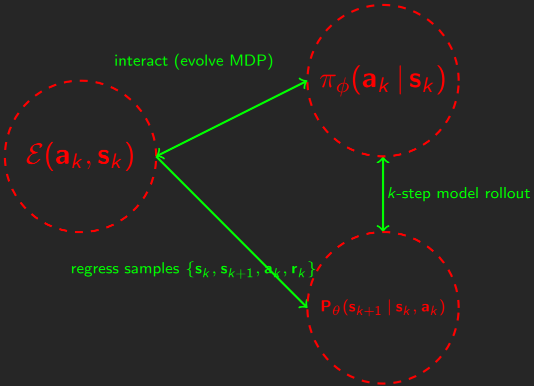
- Suppose the agent is now able to store and interact with an internal representation of $\mathcal{E}$. Let's denote that with $\hat{\mathbf{P}}_\theta(\mathbf{s}_{k+1} \mid \mathbf{s}_k, \mathbf{a}_k) = \hat{\mathcal{E}}_\theta$.



- A simple picture

Model addition

- Suppose the agent is now able to store and interact with an internal representation of $\mathcal{E}$. Let's denote that with $\hat{\mathbf{P}}_\theta(\mathbf{s}_{k+1} \mid \mathbf{s}_k, \mathbf{a}_k) = \hat{\mathcal{E}}_\theta$.



- A simple picture
- Have a choice to go function approximation route for model or ODE solver

Why RL?

Why RL?

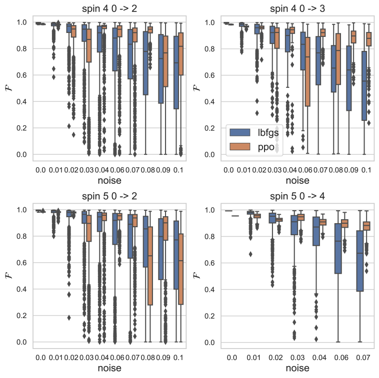
- Noise robust optimal controllers (Khalid et. al. CDC 2021)

Why RL?

- Noise robust optimal controllers (Khalid et. al. CDC 2021)

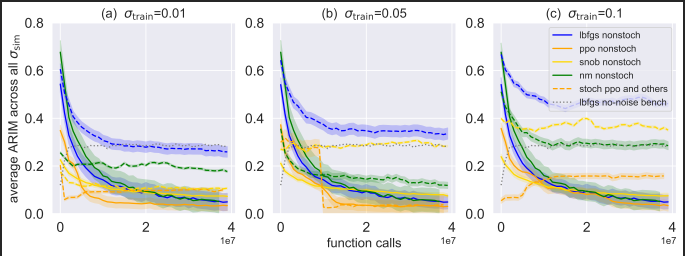
Why RL?

- Noise robust optimal controllers (Khalid et. al. CDC 2021)



Why RL?

- Noise robust optimal controllers (Khalid et. al. CDC 2021)
- as $\mathbb{E}[\mathcal{F}]$ is optimized (Khalid et. al. 2022 arxiv:2207.07801)

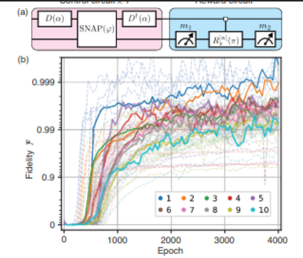


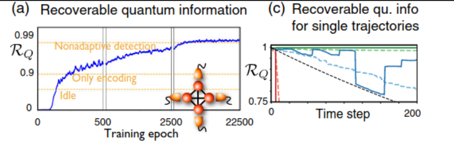
Why RL?

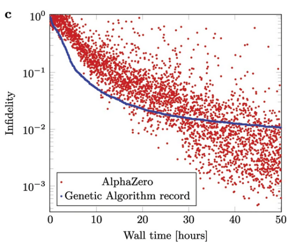
- Black box optimization (we don't need a model)
  - Note not adaptive!
  - Controller solution is a point in the solution space and not a function.
  - It does not *react* to perturbations

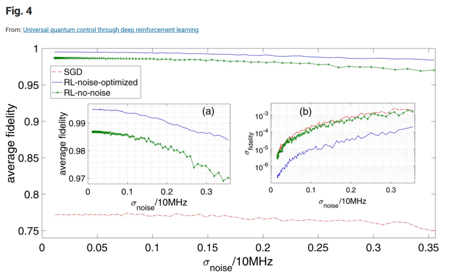
Why RL?

- Several other results in the past 3-4 years









cite[3]

[3] TL: Sivak… (PRX 2022) TR: Fösel… (PRX 2018) BL: Dalgaard… (Nature 2020) BR: Niu… (Nature 2018)

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity
  1. order of 10^6 $\mathcal{F}$ queries vs. standard order of 10^3

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity
  1. order of 10^6 $\mathcal{F}$ queries vs. standard order of 10^3
  2. especially compared to quantum model based methods e.g. GRAPE/BFGS, (neural ODE) other methods that try to obtain controllers by optimizing the fixed model or Hamiltonian (assumed optimal)

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity
  1. order of 10^6 $\mathcal{F}$ queries vs. standard order of 10^3
  2. especially compared to quantum model based methods e.g. GRAPE/BFGS, (neural ODE) other methods that try to obtain controllers by optimizing the fixed model or Hamiltonian (assumed optimal)
- Question: Can we combine model based methods in the RL framework above to improve sample complexity?

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity
  1. order of 10^6 $\mathcal{F}$ queries vs. standard order of 10^3
  2. especially compared to quantum model based methods e.g. GRAPE/BFGS, (neural ODE) other methods that try to obtain controllers by optimizing the fixed model or Hamiltonian (assumed optimal)
- Question: Can we combine model based methods in the RL framework above to improve sample complexity?
- Idea: encode the physical intuition/expertise of the physicist by proposing a δ-perfect Hamiltonian and then using RL expectation-maximization to improve the model and find the optimal controller in tandem

Unresolved issues

- How much of the state space $\mathcal{S}$ assumed observable is really observable? (ignore for now)
- High Sample complexity
  1. order of 10^6 $\mathcal{F}$ queries vs. standard order of 10^3
  2. especially compared to quantum model based methods e.g. GRAPE/BFGS, (neural ODE) other methods that try to obtain controllers by optimizing the fixed model or Hamiltonian (assumed optimal)
- Question: Can we combine model based methods in the RL framework above to improve sample complexity?
- Idea: encode the physical intuition/expertise of the physicist by proposing a δ-perfect Hamiltonian and then using RL expectation-maximization to improve the model and find the optimal controller in tandem
- at the moment constraint is a δ-perfect Hamiltonian where δ has to be small

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.
  1. maximize entropy $\mathbb{E}_\pi[\log \pi(\mathbf{a} \mid \mathbf{s})]$ and expected fidelity $\mathbb{E}[\mathcal{F}]$ simultaneously

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.
  1. maximize entropy $\mathbb{E}_\pi[\log \pi(\mathbf{a} \mid \mathbf{s})]$ and expected fidelity $\mathbb{E}[\mathcal{F}]$ simultaneously
  2. make the policy π_θ able to use data generated by past policies $\{\pi_{\theta'}\}_{\theta'}$ more effectively by making it *off-policy*

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.
  1. maximize entropy $\mathbb{E}_{\pi}[\log \pi(\mathbf{a} \mid \mathbf{s})]$ and expected fidelity $\mathbb{E}[\mathcal{F}]$ simultaneously
  2. make the policy π_θ able to use data generated by past policies $\{\pi_{\theta'}\}_{\theta'}$ more effectively by making it *off-policy*
- How? This is original Soft-Actor Critic (Haarnoja et. al. 2018)

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.
  1. maximize entropy $\mathbb{E}_\pi[\log \pi(\mathbf{a} \mid \mathbf{s})]$ and expected fidelity $\mathbb{E}[\mathcal{F}]$ simultaneously
  2. make the policy π_θ able to use data generated by past policies $\{\pi_{\theta'}\}_{\theta'}$ more effectively by making it *off-policy*
- How? This is original Soft-Actor Critic (Haarnoja et. al. 2018)
  1. The new objective function is,

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) + \alpha \log \pi(\mathbf{a}_k \mid \mathbf{s}_k)\right] \tag{6}$$

Towards better SC RL controllers (Model-Free version)

Through

- Better exploration of the solution space through a compound RL objective function.
  1. maximize entropy $\mathbb{E}_{\pi}[\log \pi(\mathbf{a} \mid \mathbf{s})]$ and expected fidelity $\mathbb{E}[\mathcal{F}]$ simultaneously
  2. make the policy π_θ able to use data generated by past policies $\{\pi_{\theta'}\}_{\theta'}$ more effectively by making it *off-policy*
- How? This is original Soft-Actor Critic (Haarnoja et. al. 2018)
  1. The new objective function is,

$$J(\pi) = \sum_{k=0}^{N} \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \mathcal{E}_\pi} \left[\gamma^k r_k(\mathbf{s}_k, \mathbf{a}_k) + \alpha \log \pi(\mathbf{a}_k \mid \mathbf{s}_k) \right] \quad (6)$$

  2. optimal convergence to π^* after repeated Bellman iterations for the tabular case ensures asymptotic performance reaches that of other Model-free methods like DDPG, PPO

... Model Based version

- How to improve further?

... Model Based version

- How to improve further?
- Assume generalization properties in training data generated by the $\pi_\phi - \mathcal{E}$ interactions

... Model Based version

- How to improve further?
- Assume generalization properties in training data generated by the $\pi_\phi - \mathcal{E}$ interactions
- train a black box *NN* model $\mathbf{P}_\theta(\mathbf{s}_{\mathrm{k}+1} \,|\, \mathbf{s}_{\mathrm{k}}, \mathbf{a}_{\mathrm{k}})$ to infer underlying dynamics and then exploit this model

... Model Based version

- How to improve further?
- Assume generalization properties in training data generated by the $\pi_\phi - \mathcal{E}$ interactions
- train a black box *NN* model $\mathbf{P}_\theta(\mathbf{s}_{\mathrm{k}+1} \,|\, \mathbf{s}_\mathrm{k}, \mathbf{a}_\mathrm{k})$ to infer underlying dynamics and then exploit this model
- This is effectively done by Janner (2019) combining PETS (probabilistic ensemble trajectory sampling) (Chua 2018) with SAC.

... Model Based version

- How to improve further?
- Assume generalization properties in training data generated by the $\pi_\phi - \mathcal{E}$ interactions
- train a black box *NN* model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k)$ to infer underlying dynamics and then exploit this model
- This is effectively done by Janner (2019) combining PETS (probabilistic ensemble trajectory sampling) (Chua 2018) with SAC.
- Generalization bound on the model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k)$ error ϵ_m yields the lower bound on k-branched rollouts (Thm 4.2.)

$$J(\pi) \geq J(\pi)_{\mathrm{branch}} - 2r_{\max}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma^2)+\frac{\gamma^k+2}{(1-\gamma)}}\epsilon_\pi + \frac{k}{1-\gamma}(\epsilon_m + 2\epsilon_\pi)\right] \tag{7}$$

Aside: What is PETS?

- Essentially from model predictive controller (MPC) control theory

Aside: What is PETS?

- Essentially from model predictive controller (MPC) control theory
- In Chua et. al.'s case, the MPCs are a B-bunch (bootstraps) of NN parametrized gaussians $\{\mathcal{N}(f_\theta^{(B)}(\mu), f_\theta^{(B)}(\sigma^2))\}_B$

Aside: What is PETS?

- Essentially from model predictive controller (MPC) control theory
- In Chua et. al.'s case, the MPCs are a B-bunch (bootstraps) of NN parametrized gaussians $\{\mathcal{N}(f_\theta^{(B)}(\mu), f_\theta^{(B)}(\sigma^2))\}_B$
- The goal is optimal action sequence acquisiton that is done using the cross entropy method (CEM, Botev et. al. 2013) by solving the following problem

$$\mathbf{s}_{t:t+T} = \arg\max_{\mathbf{a}_{t:t+T}} \mathbb{E}_{f_\theta}\left[\sum_{i=t}^{t+T} \mathbf{r}(\mathbf{s}_i, \mathbf{a}_i)\right] \tag{8}$$

Aside: What is PETS?

- Essentially from model predictive controller (MPC) control theory
- In Chua et. al.'s case, the MPCs are a B-bunch (bootstraps) of NN parametrized gaussians $\{\mathcal{N}(f_\theta^{(B)}(\mu), f_\theta^{(B)}(\sigma^2))\}_B$
- The goal is optimal action sequence acquisiton that is done using the cross entropy method (CEM, Botev et. al. 2013) by solving the following problem

$$\mathbf{s}_{t:t+T} = \underset{\mathbf{a}_{t:t+T}}{\arg\max}\, \mathbb{E}_{f_\theta}\left[\sum_{i=t}^{t+T} \mathbf{r}(\mathbf{s}_i, \mathbf{a}_i)\right] \tag{8}$$

- CEM is an importance sampling approximator. In it the cross entropy is maximized by the MLE for most natural exponential families which $\mathcal{N}$ and a randomization over various guesses converges to the optimizer of (8)

- We guess a set of Hamiltonians $\{H\}_B$ close to the true Hamiltonian H^* generating the unitary dynamics and use an ODE solver (standard or neural). This becomes our model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k)$ where $\theta = \{H\}_B$.

Alternative model has a baked in ansatz

- We guess a set of Hamiltonians $\{H\}_B$ close to the true Hamiltonian H^* generating the unitary dynamics and use an ODE solver (standard or neural). This becomes our model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \mid \mathbf{s}_k, \mathbf{a}_k)$ where $\theta = \{H\}_B$.
- We use the SU(N) parameterization: $H_i = \sum_{p=1}^{\dim(\mathrm{span}(SU(N)))} \alpha_p \sigma_p$ where σ_p are the generalized pauli matrices $\mathbb{1} \otimes \cdots \otimes \sigma_x$ and so on...

Alternative model has a baked in ansatz

- We guess a set of Hamiltonians $\{H\}_B$ close to the true Hamiltonian H^* generating the unitary dynamics and use an ODE solver (standard or neural). This becomes our model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k)$ where $\theta = \{H\}_B$.
- We use the SU(N) parameterization: $H_i = \sum_{p=1}^{\dim(\mathrm{span}(SU(N)))} \alpha_p \sigma_p$ where σ_p are the generalized pauli matrices $\mathbb{1} \otimes \cdots \otimes \sigma_x$ and so on...
- We train the set using PETS by regressing onto the states or rewards.

Alternative model has a baked in ansatz

- We guess a set of Hamiltonians $\{H\}_B$ close to the true Hamiltonian H^* generating the unitary dynamics and use an ODE solver (standard or neural). This becomes our model $\mathbf{P}_\theta(\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k)$ where $\theta = \{H\}_B$.
- We use the SU(N) parameterization: $H_i = \sum_{p=1}^{\dim(\mathrm{span}(SU(N)))} \alpha_p \sigma_p$ where σ_p are the generalized pauli matrices $\mathbb{1} \otimes \cdots \otimes \sigma_x$ and so on...
- We train the set using PETS by regressing onto the states or rewards.
- We found that state regression has a better transmon-dependent loss landscape than predicted fidelity regression

Results: Step 1: With $\mathbf{P}_\theta(\mathbf{s}_{k+1} \mid \mathbf{s}_k, \mathbf{a}_k)$ being perfect, how many real samples does it take?

Target gate is the CNOT

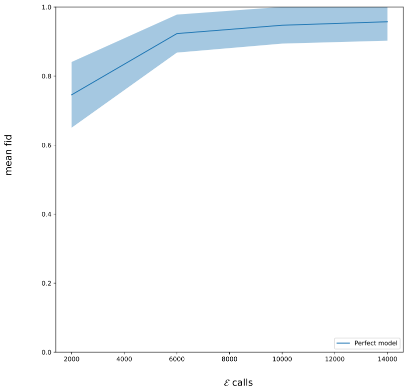


Figure: Sanity test and optimal lower bound (baseline) for the MBSAC

Understanding the exploitation-exploration tradeoff for the quantum control setting in the perfect model setting

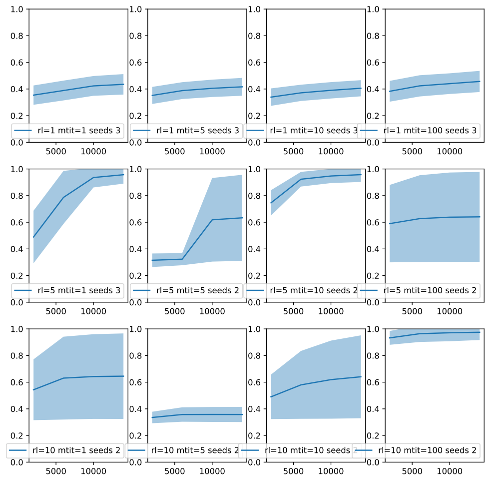


Figure: branch rollout length (rl) k (model explore) tradeoffs w.r.t. training per k model train iterations (mtit)

Step 2: How good is the δ-perfect model? What is the effect of δ on sample complexity?

- Maybe not the smartest way to pick δ but we choose it to be partially adversarial

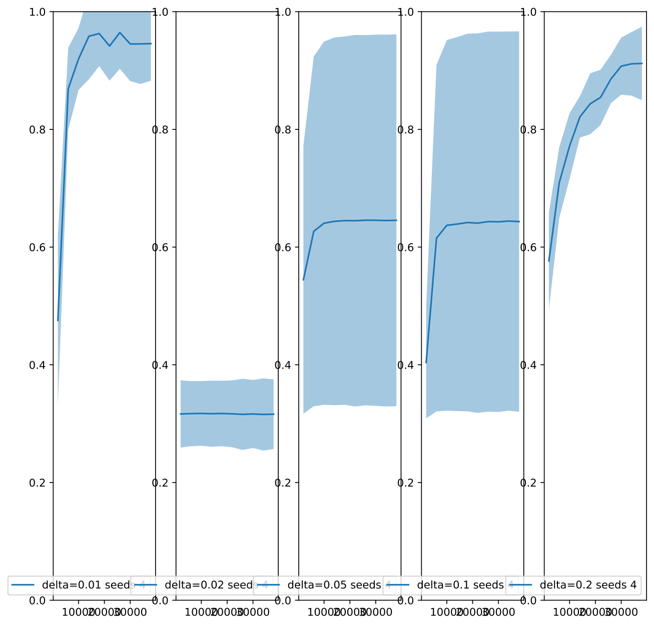
$$\Lambda = \|F(H^*(\Delta_i)) - F(H_2(\Delta_i))\| \approx \delta \ \forall i \tag{9}$$

Step 2: How good is the δ-perfect model? What is the effect of δ on sample complexity?

- Maybe not the smartest way to pick δ but we choose it to be partially adversarial

$$\Lambda = \|F(H^*(\Delta_i)) - F(H_2(\Delta_i))\| \approx \delta \ \forall i \tag{9}$$

- We minimize Λ w.r.t. H_2.



1.0
0.8
0.6
0.4
0.2
0.0
10000
20000
30000
delta=0.01 seeds 4
delta=0.02 seeds 4
delta=0.05 seeds 4
delta=0.1 seeds 4
delta=0.2 seeds 4

Step 3: Now train and contrast

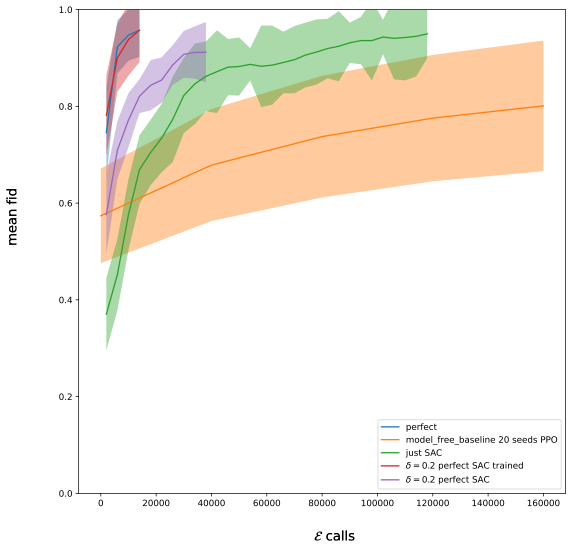


Figure: Various

Improving the definition of δ-perfection

Use Burgath et. al. (2022) upper bound

$$\|U^* - U_2\| \leq \left\| \int_0^T dsH^*(s) - H_2(s) \right\| \left(1 + \left\| \int_0^T dsH^*(s) \right\| + \left\| \int_0^T dsH_2(s) \right\| \right) \tag{10}$$

set $\|U^* - U_2\|$ as fidelity difference δ? and solve the above relaxed optimization problem?

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?
  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good. $>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.
- The ΔU loss is better than the ΔF loss function.

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.
- The ΔU loss is better than the ΔF loss function.
- Step 3 (learning the model in the toy setting of step 1 and more generally) Effect on sample complexity for the δ-perfect ham...?

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.
- The ΔU loss is better than the ΔF loss function.
- Step 3 (learning the model in the toy setting of step 1 and more generally) Effect on sample complexity for the δ-perfect ham...?

  Get an improvement.

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.
- The ΔU loss is better than the ΔF loss function.
- Step 3 (learning the model in the toy setting of step 1 and more generally) Effect on sample complexity for the δ-perfect ham...?

  Get an improvement.
- Effect on sample complexity for the general case...

Takeaways

- Step 0 GOAL: REDUCING RL CONTROL SAMPLE COMPLEXITY. Have preliminary results for toy Transmon setting
- Step 1 How good is the perfect model to establish a baseline?

  A. Very good.$>$ x10 for training the NN policy π_θ
- Step 2 How good is the δ-perfect model? What is the effect of δ on sample complexity?

  Stalling for some δ-perfect H. But a noticeable increase in sample complexity is observed as δ is increased.
- The ΔU loss is better than the ΔF loss function.
- Step 3 (learning the model in the toy setting of step 1 and more generally) Effect on sample complexity for the δ-perfect ham...?

  Get an improvement.
- Effect on sample complexity for the general case...
  TODO

Future work / Improvements

- Tomography guarantees are available

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians
- An application of Hoeffding's inequality for learning the model is possible. NN results available recently but not sure how applicable (NTK but not in the infinite width limit)

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians
- An application of Hoeffding's inequality for learning the model is possible. NN results available recently but not sure how applicable (NTK but not in the infinite width limit)
- Full learning (not δ-perfect).

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians
- An application of Hoeffding's inequality for learning the model is possible. NN results available recently but not sure how applicable (NTK but not in the infinite width limit)
- Full learning (not δ-perfect).
- Different Hamiltonians

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians
- An application of Hoeffding's inequality for learning the model is possible. NN results available recently but not sure how applicable (NTK but not in the infinite width limit)
- Full learning (not δ-perfect).
- Different Hamiltonians
- Multiple Gates

Future work / Improvements

- Tomography guarantees are available
- Learning guarantees (Flammia) for a restricted class of Hamiltonians
- An application of Hoeffding's inequality for learning the model is possible. NN results available recently but not sure how applicable (NTK but not in the infinite width limit)
- Full learning (not δ-perfect).
- Different Hamiltonians
- Multiple Gates
- Some of above with Noisy Lindblad dynamics and shot coarse-graining