Look Beneath the Surface: Exploiting Fundamental Symmetry for Sample-Efficient Offline Reinforcement Learning

Anonymous Authors[1]

Abstract

Offline reinforcement learning (RL) offers an appealing approach to real-world tasks by learning policies from pre-collected datasets without interacting with the environment. However, the performance of existing offline RL algorithms heavily depends on the scale and state-action space coverage of datasets. Real-world data collection is often expensive and uncontrollable, leading to small and narrowly covered datasets and posing significant challenges for practical deployments of offline RL. In this paper, we provide a new insight that leveraging the fundamental symmetry of system dynamics can substantially enhance offline RL performance under small datasets. Specifically, we propose a Time-reversal symmetry (T-symmetry) enforced Dynamics Model (TDM), which establishes consistency between a pair of forward and reverse latent dynamics. TDM provides both well-behaved representations for small datasets and a new reliability measure for OOD samples based on compliance with the T-symmetry. These can be readily used to construct a new offline RL algorithm (TSRL) with less conservative policy constraints and a reliable latent space data augmentation procedure. Based on extensive experiments, we find TSRL achieves great performance on small benchmark datasets with as few as 1% of the original samples, which significantly outperforms the recent offline RL algorithms in terms of data efficiency and generalizability.

1. Introduction

The recently emerged offline reinforcement learning (RL) provides a new paradigm to learn policies from pre-collected datasets without the need of interacting with the environments (Levine et al., 2020; Fujimoto et al., 2018; Kumar et al., 2019). This is particularly desirable for solving practical tasks, as interacting with real-world systems can be costly or risky, and high-fidelity simulators are also hard to build (Zhan et al., 2022). However, existing offline RL methods have high requirements on the size and quality of offline datasets in order to achieve reasonable performance. When such requirements are not met, these algorithms may suffer from severe performance drop, as illustrated in Figure 1. Current offline RL algorithms are trained and validated on benchmark datasets (e.g., D4RL (Fu et al., 2020)) that contain millions of transitions for simple tasks. Whereas under realistic settings, it is often impractical or costly to collect such a large amount of data, and the real datasets might only narrowly cover the state-action space. Clearly, learning reliable policies from small datasets with partial coverage has become one of the most pressing challenges for successful real-world deployments of offline RL.

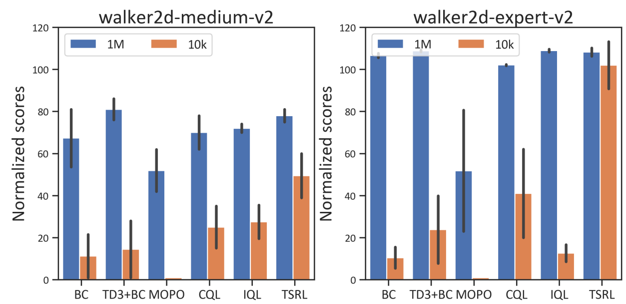

Figure 1: Performance of existing offline RL methods and our proposed TSRL on the D4RL-MuJoCo-v2 Walker2d medium and expert datasets (Fu et al., 2020) when reducing the number of samples from 1M (full dataset) to 10k (1%).

Unfortunately, sample-efficient design considerations have been largely overlooked in the majority of previous offline RL studies. Pessimism is universally adopted in existing offline RL methods and various forms of data-related regularizations have been applied to combat the distributional shift and (Kumar et al., 2019; Fujimoto et al., 2019) exploitation error accumulation issues, such as conservatively restricting policy deviation from the behavioral data (Kumar et al., 2019; Fujimoto et al., 2019; Wu et al., 2019; Fuji-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

moto & Gu, 2021), regularizing value function on out-of-distribution (OOD) samples (Kumar et al., 2020; Kostrikov et al., 2021a; Xu et al., 2022b; Bai et al., 2021), learning policy on a pessimistic MDP (Yu et al., 2020; Kidambi et al., 2020; Zhan et al., 2022), or adopting strict in-sample learning (Kostrikov et al., 2021b; Brandfonbrener et al., 2021; Xu et al.; 2022a). In many of these methods, the full coverage assumption plays an important role in their theoretical performance guarantees (Kumar et al., 2019; Le et al., 2019; Chen & Jiang, 2019), which assumes the dataset to contain all state-action pairs in the induced distribution of the policy. Obviously, most of the state-action space will become OOD areas under a small dataset. Applying strict data-related regularizations will inevitably cause severe performance degradation and poor generalization. Consequently, it is important to rethink what is essential in policy learning with small datasets. In other words, what is the fundamental or invariant information that can be used to facilitate policy learning, without being conservatively confined by the limited data?

In this paper, we provide a new insight that exploiting the fundamental symmetries in the system dynamics can substantially enhance the performance of offline RL with small datasets. Specifically, we consider the time-reversal symmetry (also called *T-symmetry*), which is one of the most fundamental properties discovered in classical and quantum mechanics (Elliott & Dawber, 1979; Lamb & Roberts, 1998). It suggests that the underlying laws of physics should not change under the time-reversal transformation: $t \rightarrow -t$ (Lamb & Roberts, 1998; Bluman & Kumei, 2013). Specifically, we are interested in an extended form of T-symmetry due to its simplicity and universality in physical systems.

Based on these intuitions, we develop a physics-informed T-symmetry enforced Dynamics Model (*TDM*) to learn a well-behaved and generalizable dynamics model with small datasets. TDM enforces the extended T-symmetry between a pair of latent space forward and reverse dynamics submodels, which are modeled as first-order ordinary differential equation (ODE) systems to extract fundamental dynamics patterns in data. TDM provides both well-behaved representations for small datasets and a new reliability measure for OOD samples based on compliance with the T-symmetry. These can be used to construct a highly data-efficient offline RL algorithm, which we call T-Symmetry regularized offline RL (*TSRL*). Specifically, TSRL uses the T-symmetry regularized representations learned in TDM to facilitate value function learning. Furthermore, the deviation on latent actions and the consistency with T-symmetry specified in TDM actually provide another perspective to detect unreliable or non-generalizable samples, which can serve as a new set of policy constraints to replace the highly restrictive OOD regularizations in existing offline RL algorithms. Lastly, a reliable latent space data augmentation scheme based on compliance with the T-symmetry is also applied to further remedy the limited size of training data. With these designs, TSRL performs surprisingly well compared with the state-of-the-art offline RL algorithms on reduced-size D4RL benchmark datasets with even as few as 1% of the original samples. To the best of the authors' knowledge, this is the first offline RL method that demonstrates promising performance on extremely small datasets.

2. Preliminaries

Offline reinforcement learning. We consider the standard Markov decision process (MDP) setting (Sutton & Barto, 2018), which is represented as a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \rho, \gamma\}$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $r(s, a)$ is a scalar reward function, $\mathcal{P}$ is the transition dynamics, ρ is the initial state distribution, and $\gamma \in (0, 1)$ is a discount factor. The objective of RL is to learn a policy $\pi(a|s)$ by maximizing the expected cumulative discounted return $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t))]$, which is typically approximated by a value function $Q(s, a)$ using some function approximators, such as deep neural networks. The Q-function is typically learned by minimizing the squared Bellman error:

$$Q = \arg\min_{Q} \mathbb{E}\left[\left(Q\left(s, a\right) - \mathcal{B}^\pi \hat{Q}(s, a)\right)^2\right] \tag{1}$$

where $\hat{Q}$ denotes a target Q-function, which is a delayed copy of the current Q-function; $\mathcal{B}^\pi$ is the Bellman operator, which is often used as the Bellman evaluation operator $\mathcal{B}^\pi \hat{Q}(s, a) = r(s, a) + \gamma \mathbb{E}_{a' \sim \pi} \hat{Q}\left(s', a'\right)$ in many RL algorithms.

Under the offline RL setting, we are provided with a fixed dataset $\mathcal{D} = \{(s_0, a_0, r_0, s_1, \cdots)^{(i)}\}_{i=1}^N$ without any chance of further environment interactions. Directly applying standard online RL methods in the offline setting suffers from severe value overestimation, due to counterfactual queries on OOD data and the resulting extrapolation errors (Levine et al., 2020; Kumar et al., 2019; Fujimoto et al., 2019). To avoid this issue, a widely used offline RL framework adopts the following behavior regularization scheme which regularizes the divergence between the learned policy π and the behavior policy π_β of the dataset $\mathcal{D}$:

$$\pi = \arg\max_{\pi} \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi(\cdot|s)} \left[Q(s, a) - \mathrm{D}\left(\pi(\cdot \mid s) \| \pi_\beta(\cdot \mid s)\right)\right] \tag{2}$$

where $\mathrm{D}(\cdot\|\cdot)$ is some divergence measures, which can have either an explicit (Kumar et al., 2019; Fujimoto & Gu, 2021) or implicit form (Fujimoto et al., 2019; Wang et al., 2020). Although straightforward, existing behavior regularization methods have been shown to be over-conservative (Kumar et al., 2020; Li et al., 2022) due to the restrictive regularization with respect to the behavior policy in data, which may suffer from notable performance drop under small datasets.

Time-reversal symmetry in dynamical systems. Most real-world dynamical systems with state measurement $\mathbf{x} \in \Omega$ on some phase space Ω can be modeled or approximated by the system of non-linear first-order ordinary differential equations (ODEs) as $\frac{d\mathbf{x}}{dt} = F(\mathbf{x})$, where F is some general non-linear, at least $\mathcal{C}^1$-differentiable vector-valued function. First-order ODE systems are said to be time-reversal symmetric if there is an invertible transformation $\Gamma : \Omega \mapsto \Omega$, that reverses the direction of time (Lamb & Roberts, 1998; Huh et al., 2020): $d\Gamma(\mathbf{x})/dt = -F(\Gamma(\mathbf{x}))$. If we define a time evolution operator $U_{\Delta t} : \Omega \mapsto \Omega$ as $U_{\Delta t} : \mathbf{x}(t) \mapsto U_{\Delta t}(\mathbf{x}(t)) = \mathbf{x}(t + \Delta t)$. Then T-symmetry implies that $\Gamma \circ U_\tau = U_{-\tau} \circ \Gamma$. In other words, the reversing of the forward time evolution of an arbitrary state should be equal to the backward time evolution of the reversed state.

Extending T-symmetry for more generic MDP settings. In our discrete-time MDP setting, we have $\mathbf{x} = (s, a)$. We can slightly abuse the notations and denote $\dot{s} = \frac{ds}{dt}$ as the time-derivative of the current state s, which can be approximated as the difference between the next and current states, i.e., $\dot{s} = s' - s$. For a dynamical system that satisfies T-symmetry, it suggests that if we learn a forward dynamics $F(s, a) = \dot{s}$ and a reverse dynamics $\tilde{G}(s', a') = -\dot{s}$ as a pair of first-order ODEs, we should have $F(s, a) = -\tilde{G}(s', a')$.

However, from a decision-making perspective, it is known that T-symmetry can sometimes be broken by irreversible actions or some special dynamic processes (e.g., frictional force against motion). Hence in this paper, we consider a more generic treatment by leveraging an alternative ODE reverse dynamics model $G(s', a) = -\dot{s}$ to establish the T-symmetry with the forward dynamics, i.e., enforcing $F(s, a) = -G(s', a)$. Note that $G(s', a)$ is now defined on the next state s' and the current action a, rather than the next action a', thus is not impacted if the next action is irreversible. Moreover, $G(s', a)$ shares a similar form as the widely used inverse dynamics model in many RL and IL studies (Yang et al., 2019; Jiang et al., 2020; Wang et al., 2021), except that we require it to be an ODE system. This extended T-symmetry provides a more fundamental and almost universally held property in discrete-time MDP systems. Its simplicity and fundamentalness make it an ideal property that we can leverage to construct a well-behaved data-driven dynamics model and a robust offline RL algorithm under small datasets.

3. T-Symmetry Enforced Dynamics Model

In this section, we present the detailed design of TDM, which is capable of learning a more fundamental and T-symmetry preserving dynamics from small datasets. The key ingredients of TDM are to embed a pair of latent forward and reverse dynamics as ODE systems, and further enforce their T-symmetry consistency. As illustrated in Figure 2, the proposed TDM consists the following components:

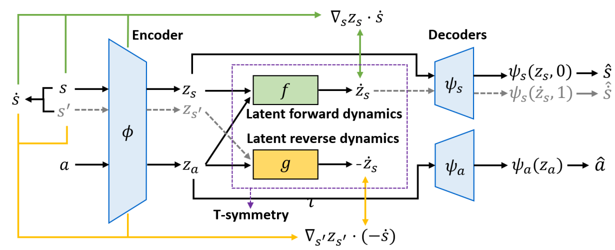

Figure 2: Overall architecture of the proposed TDM

Encoder and decoders. TDM implements a state-action encoder $\phi(s, a) = (z_s, z_a)$ and a pair of decoders $\psi_s(\cdot, \delta_s), \psi_a(z_a) = a$ that embed the state-action pair (s, a) into latent representations (z_s, z_a) and then map them back. Specifically, we require the state decoder $\psi_s(\cdot, \delta_s)$ to be capable of decoding both z_s and $\dot{z}_s$, where δ_s is an indicator to help the decoder to decide the target output, with $\delta_s = 0$ as decoding $z_s \to s$ and $\delta_s = 1$ as decoding $\dot{z}_s \to \dot{s}$. The encoder ϕ and decoders ψ_s and ψ_a induce the following reconstruction loss term for each state-action pair (s, a):

$$\ell_{rec}(s, a) = \|s - \psi_s(z_s, 0)\|_2^2 + \|a - \psi_a(z_a)\|_2^2 \tag{3}$$

Latent forward dynamics $f(z_s, z_a) = \dot{z}_s$. Inspired by the prior works that incorporate physics-informed information into dynamical systems modeling (Mezić, 2005; Brunton et al., 2016; Champion et al., 2019), we embed a discrete-time first-order ODE system to capture the latent forward dynamics $f(z_s, z_a) = \dot{z}_s$. Similar to $\dot{s}$, we write $\dot{z}_s = z_{s'} - z_s$ to denote the forward difference of the next and current latent state representations. Note that based on the chain-rule, we have $\dot{z}_s = \frac{dz_s}{dt} = \frac{\partial z_s}{\partial s} \cdot \frac{ds}{dt} = \nabla_s z_s \cdot \dot{s}$. To enforce the ODE property, we can introduce the following loss term for f:

$$\begin{aligned} \ell_{fwd}(s, a, s') &= \|(\nabla_s z_s)\dot{s} - \dot{z}_s\|_2^2 \\ &= \|\frac{\partial \phi(s, a)}{\partial s}\dot{s} - f(\phi(s, a))\|_2^2 \end{aligned} \tag{4}$$

Minimizing $\mathcal{L}_{fwd}$ ensures that the latent forward dynamics f correctly predicts the forward time evolution of latent states in the dynamical system. We also require the decoder $\psi_s(\cdot, \delta_{\dot{s}})$ to be able to decode $\dot{s}$ from $\dot{z}_s$ to ensure it is compatible with the ODE property, which implies the following loss:

$$\begin{aligned} \ell_{ds}(s, a, s') &= \|\dot{s} - \psi_s(\dot{z}_s, 1)\|_2^2 \\ &= \|\dot{s} - \psi_s(f(\phi(s, a)), 1)\|_2^2 \end{aligned} \tag{5}$$

Latent reverse dynamics $g(z_{s'}, z_a) = -\dot{z}_s$. We can further introduce a latent reverse dynamics $g(z_{s'}, z_a) = -\dot{z}_s$ in the model, which captures the reverse time evolution of the system in the latent space. Similar to the forward dynamics loss $\mathcal{L}_{fwd}$, we can write the reverse dynamics loss for g as:

$$\begin{aligned}\ell_{rvs}(s,a,s') &= \|(\nabla_{s'} z_{s'})(-\dot{s}) - (-\dot{z}_s)\|_2^2 \\ &= \|\frac{\partial \phi(s',a)}{\partial s'}(-\dot{s}) - g(\phi(s',a))\|_2^2\end{aligned} \tag{6}$$

T-symmetry regularization. The above latent forward and reverse dynamics f and g are learned to be two models, which may not necessarily satisfy the proposed extended T-symmetry. We can enforce the extended T-symmetry by requiring $f(z_s,z_a) = -g(z_{s'},z_a)$. To further couple the learning process of f and g, note that $z_{s'} = z_s + \dot{z}_s = z_s + f(z_s,z_a)$, which suggests $g(z_{s'},z_a) = g(z_s + f(z_s,z_a), z_a) = -\dot{z}_s = -f(z_s,z_a)$. This implies the following T-symmetry consistency loss:

$$\ell_{T\text{-}sym}(z_s,z_a) = \|f(z_s,z_a) + g(z_s + f(z_s,z_a), z_a)\|_2^2 \tag{7}$$

Above instance-wise T-symmetry consistency loss also provides an alternative measure for evaluating the reliability of a data sample. A state-action pair (s,a) with a large $\ell_{T\text{-}sym}(\phi(s,a))$ implies that this sample may not be well-explained by TDM or consistent with the fundamental symmetry of the system. This can be used to detect unreliable OOD samples in offline policy optimization as well as construct a new latent space data augmentation procedure, which will be discussed in later content.

Final learning objective of TDM. Finally, we can formulate the overall loss function of TDM as:

$$\begin{aligned}\mathcal{L}_{TDM} = \sum_{(s,a,s')\in\mathcal{D}} &[\ell_{rec} + \ell_{ds} + \ell_{fwd} + \ell_{rvs} \\ &+ \ell_{T\text{-}sym}](s,a,s') + \lambda_{L1}[\mathcal{L}_1(f) + \mathcal{L}_1(g)]\end{aligned} \tag{8}$$

where $\mathcal{L}_1(f)$ and $\mathcal{L}_1(g)$ are L1-norms of the parameters of f and g, and λ_{L1} is a scale parameter. L1 regularization is introduced to encourage learning parsimonious latent dynamics for f and g, which helps to improve model generalizability (Brunton et al., 2016; Champion et al., 2019).

Note that the proposed TDM is very different from the conventional dynamics models used in model-based RL (MBRL) methods (Janner et al., 2019; Yu et al., 2020; Kidambi et al., 2020; Wang et al., 2021; Zhan et al., 2022). The dynamics models in MBRL focus on constructing a predictive model to represent the forward transition dynamics of the system. Whereas, TDM is formulated as a reconstruction model with T-symmetry preserving embedded ODE latent dynamics, which aims at explaining and extracting the fundamental dynamics of the system. As a result, TDM can be substantially more well-behaved and robust when learning from small datasets.

4. T-Symmetry Regularized Offline RL

In this section, we discuss how to incorporate the properties of TDM to construct a sample-efficient offline RL algorithm, which we call T-Symmetry regularized offline RL (TSRL).

T-symmetry regularized representation. Representation learning has been shown to be an effective approach to enhancing sample efficiency and generalization in many online and offline RL studies (Zhang et al., 2020; Srinivas et al., 2020; Agarwal et al., 2021; Yang & Nachum, 2021; Uehara et al., 2021). A notable property of TDM is that the learned latent state-action representations from the encoder $(z_s,z_a) = \phi(s,a)$ are compatible with both the latent forward and reverse ODE dynamics f and g. This leads to well-regularized and T-symmetry preserving representations that can potentially generalize better on OOD areas under small dataset settings. We can simply use the latent state-action representation (z_s,z_a) extracted by the encoder $\phi(s,a)$ of TDM in the value function learning, which gives the following policy evaluation objective:

$$\begin{aligned}Q = \arg\min_{Q} \mathbb{E}_{(s,a,s')\sim\mathcal{D}}\Big[&\big(r(s,a) + \gamma\hat{Q}(\phi(s',\pi(\cdot|s'))) \\ &- Q(\phi(s,a))\big)^2\Big]\end{aligned} \tag{9}$$

T-symmetry regularized policy constraints. Existing offline RL methods primarily penalize the divergence between the learned policy π and the behavioral data in the original action space, which ignores the underlying manifold structure of actions in the latent space (Zhou et al., 2021) and the system dynamics properties. In TSRL, we instead consider an alternative regularization scheme, which restricts the deviation on latent actions and the T-symmetry consistency of policy-induced samples, corresponding to the following policy optimization objective:

$$\begin{aligned}&\max_{\pi} \mathbb{E}_{(s,a)\sim\mathcal{D}}[Q(\phi(s,\pi(\cdot|s)))] \\ &\text{s.t. } \|z_{a^\pi} - z_a\|_2^2 \le \eta \quad \textit{(Latent action deviation)} \\ &\quad\ \ell_{T\text{-}sym}(\phi(s,\pi(\cdot|s))) = 0 \quad \textit{(T-sym consistency)}\end{aligned} \tag{10}$$

where latent actions z_a and z_{a^π} are obtained from $\phi(s,a)$ and $\phi(s,\pi(\cdot|s))$ respectively. The second term restricts the latent action z_{a^π} of policy π from deviating too much from the latent action z_a in data. The third term regularizes the T-symmetry consistency of policy-induced samples $(s,\pi(\cdot|s))$, which is evaluated based on Eq. (7) and the learned TDM. λ_1 and λ_2 are weight parameters, which only need to be roughly adjusted to ensure both the regularization terms are in a similar scale as the first term. We also introduce a normalization term α on the value function for training stability similar to TD3+BC (Fujimoto & Gu, 2021), which is computed based on a training batch B of samples as $\alpha_0/[\sum_{(s,a)\in B} Q(\phi(s,\pi(\cdot|s)))]$. We set $\alpha_0 = 2.5$ in all of our experiments without tuning.

$$\begin{aligned}\underset{\pi}{\operatorname{argmax}}\ & \mathbb{E}_{(s,a)\sim\mathcal{D}}\big[\alpha Q(\phi(s,\pi(\cdot|s))) \\ & - \lambda_1\|z_{a^\pi} - z_a\|_2^2 - \lambda_2 \ell_{T\text{-}sym}(\phi(s,\pi(\cdot|s)))\big]\end{aligned} \tag{11}$$

where latent actions z_a and z_{a^π} are obtained from $\phi(s,a)$ and $\phi(s,\pi(\cdot|s))$ respectively. The second term restricts the latent action z_{a^π} of policy π from deviating too much from the latent action z_a in data. The third term regularizes the T-symmetry consistency of policy-induced samples $(s,\pi(\cdot|s))$, which is evaluated based on Eq. (7) and the learned TDM. λ_1 and λ_2 are weight parameters, which only need to be roughly adjusted to ensure both the regularization terms are in a similar scale as the first term. We also introduce a normalization term α on the value function for training stability similar to TD3+BC (Fujimoto & Gu, 2021), which is computed based on a training batch B of samples as $\alpha_0/[\sum_{(s,a)\in B} Q(\phi(s,\pi(\cdot|s)))]$. We set $\alpha_0 = 2.5$ in all of our experiments without tuning.

T-symmetry consistent latent space data augmentation. It has been shown in previous studies (Sinha et al., 2022; Weissenbacher et al., 2022; Lyu et al., 2022) that data augmentation can potentially improve the function approximation of the Q-networks by smoothing out the learned state-action space, hence often lead to more robust policy and better data efficiency. However, existing data augmentation methods in offline RL studies either blindly add random perturbations to states (Sinha et al., 2022) or utilize costly non-linear symmetry transformations, such as Koopman theory (Weissenbacher et al., 2022). With TDM, we can provide a very simple yet principled data augmentation scheme based on the T-symmetry property.

Assuming we add a small perturbation ϵ to a latent state z_s, i.e., $(z_s, z_a) \mapsto (z_s+\epsilon, z_a)$, then the corresponding perturbation ϵ' on the next latent state $z_{s'}$ according to the latent forward dynamics $\dot{z}_s = f(z_s, z_a)$ satisfies: $z_{s'} + \epsilon' = z_s + \epsilon + f(z_s+\epsilon, z_a)$. On the other hand, by the T-symmetry construction in TDM, we can recover back the current perturbed latent state based on the latent reverse dynamics $-\dot{z}_s = g(z_{s'}, z_a)$ as: $z_s + \epsilon'' = z_{s'} + \epsilon' + g(z_{s'} + \epsilon', z_a)$. Clearly, we should have $\epsilon = \epsilon''$, which suggests the following condition:

$$\epsilon'' - \epsilon = f(z_s+\epsilon, z_a) + g(z_s+\epsilon+f(z_s+\epsilon, z_a), z_a) = 0 \tag{12}$$

This is exactly equivalent to requiring the instance-wise T-symmetry consistency loss (Eq. (7)) $\ell_{T\text{-}sym}(z_s+\epsilon, z_a) = 0$. Hence we can use T-symmetry consistency loss $\ell_{T\text{-}sym}(\cdot)$ as a reliability measure to filter out unreliable augmented samples $(z_s+\epsilon, z_a)$ that are inconsistent with the T-symmetry property of the learned latent dynamics in TDM. In our implementation, we only keep augmented samples that satisfy $\ell_{T\text{-}sym}(z_s+\epsilon, z_a) \le h$, where we consider a non-parametric treatment for threshold h, by setting it as the τ-quantile value of all $\ell_{T\text{-}sym}(\phi(s,a))$ values of (s,a) in $\mathcal{D}$ (we choose $\tau = 50\%$ or 70% in our experiments). This ensures that the augmented samples at least maintain the similar level of T-symmetry agreement explained by TDM as the data samples in $\mathcal{D}$.

5. Experiments

We evaluate TSRL on the D4RL MuJoCo-v2 and Adroit-v1 benchmark datasets (Fu et al., 2020) against behavior cloning (BC) as well as state-of-the-art (SOTA) offline RL methods, including model-free methods TD3+BC (Fujimoto & Gu, 2021), CQL (Kumar et al., 2020), and IQL (Kostrikov et al., 2021b), and model-based method MOPO (Yu et al., 2020). We report the final normalized performance of each algorithm after training 1M steps.

Performance on small datasets. We compare the performance of TSRL and the baseline methods on both the full D4RL datasets and their reduced-size datasets with only 5k$\sim$10k samples, which are constructed by randomly sampling a given fraction of trajectories in the full datasets[1]. These reduced-size datasets are only about 1/20$\sim$1/200 of their original size. Compared with the performances on the full datasets, the baseline offline RL methods suffer from a noticeable performance drop under these extremely small datasets, mainly due to their over-reliance on the size and coverage of training data. By contrast, TSRL achieves substantially better performance in all small dataset tasks, indicating superior sample efficiency. Moreover, although MOPO also learns a dynamics model for offline policy learning, it performs badly when the dataset is small, revealing the importance of using a well-regularized model like TDM in the small-sample regime.

To further examine the impact of the training data size on algorithm performance, we also conduct experiments on three Walker2d datasets (medium, medium-expert, and expert) by varying the size of samples from 1M to 5k. The results are presented in Figure 3. It can be observed that most baseline offline RL algorithms experience a sharp performance drop when the datasets are reduced to 10k samples. Whereas, TSRL is still capable of preserving reasonable performance as the decrease of data size, even for extremely small datasets that contain only 5k samples.

Investigation on learned representations. To investigate the quality of the latent representation learned in TDM, we compare the performance of different representation learning approaches on the 10k datasets in Figure 4. To solely evaluate the impact of the representation, we remove the latent space data augmentation component from TSRL ("TSRL-no-A") and replace the state-action encoder $\phi(s,a)$

[1] We didn't construct reduced-size datasets for Adroit-human tasks, as the full datasets are already very small.

Table 1: Average normalized score on D4RL MuJoCo and Adroit tasks with full and reduced-size datasets. Some of the full dataset performance scores are reported from the IQL (Kostrikov et al., 2021b) and MOPO (Yu et al., 2020) papers. Complete scores for Adroit-human and cloned tasks are included in Appendix C.

Task	Ratio	Size	BC	TD3+BC	MOPO	CQL	IQL	TSRL(ours)
Hopper-m	1	1M	52.9	59.3	28.0	58.5	66.3	**86.7±8.7**
	1/100	10k	29.7±11.7	40.1±18.6	5.5±2.3	43.1±24.6	46.7±6.5	**62.0±3.7**
Hopper-mr	1	400k	18.1	60.9	67.5	95.0	**94.7**	78.7±28.1
	1/40	10k	12.1±5.3	7.3±6.1	6.8±0.3	2.3±1.9	13.4±3.1	**21.8±8.2**
Hopper-me	1	2M	52.5	98.0	23.7	**105.4**	91.5	95.9±18.4
	1/200	10k	27.8±10.7	17.8±7.9	5.8±5.8	29.9±4.5	34.3±8.7	**50.9±8.6**
Hopper-e	1	1M	108.0	100.1	16.2±6.2	98.4	99.3	**110.0 ±3.3**
	1/100	10k	20.8±6.9	23.2±18.2	6.5±3.7	33.0±22.2	38.4±11.3	**82.7±21.9**
Halfcheetah-m	1	1M	42.6	**48.3**	42.3	44.0	47.4	**48.2 ±0.7**
	1/100	10k	26.4±7.3	16.4±10.2	-1.1±4.1	35.8±3.8	29.9±0.12	**38.4±3.1**
Halfcheetah-mr	1	200k	55.2	44.6	**53.1**	45.5	44.2	42.2±3.5
	1/20	10k	14.3±7.8	17.9±9.5	11.7±5.2	8.1±9.4	22.7±6.4	**28.1±3.5**
Halfcheetah-me	1	2M	55.2	90.7	63.3	91.6	86.7	**92.0±1.6**
	1/200	10k	19.1±9.4	15.4±10.7	-1.1±1.4	26.5±10.8	10.5±8.8	**39.9±21.1**
Halfcheetah-e	1	1M	92.2	82.1	1.4±2.2	**95.6**	88.9±1.2	**94.3±5.5**
	1/100	10k	1.10±2.4	1.72±3.3	-0.6±1.1	4.2±0.94	-2.0±0.4	**40.6±24.4**
Walker2d-m	1	1M	75.3	**83.7**	17.8	72.5	78.3	77.5 ±4.5
	1/100	10k	15.8±14.1	7.4±13.1	3.1±4.7	18.8±18.8	22.5±3.8	**49.7±10.6**
Walker2d-mr	1	300k	26.0	**81.8**	39.0	77.2	73.9	66.1±12.0
	1/30	10k	1.4±1.9	5.7±5.8	3.3±2.7	8.5±2.19	10.7±11.9	**26.0±11.3**
Walker2d-me	1	2M	107.5	**110.1**	44.6	108.8	109.6	**109.8±3.12**
	1/200	10k	21.7±8.2	7.9±9.1	0.6±2.7	19.1±14.4	26.5±8.6	**46.4±17.4**
Walker2d-e	1	1M	107.9	108.2	0.1±0.3	101.3	**109.7±0.1**	**110.2±0.3**
	1/100	10k	10.4±5.3	23.8±16.0	1.4±3.4	41.6±21.6	12.6±4.5	**102.2±11.3**
Adroit-human-total	1	5k	71.5	10.6	9.5	52.2	77.3	**80.9±21.1**
Adroit-cloned-total	1	500k	**60.1**	41.1	-1.2	41.6	40.8	**58.6±25.4**
	1/50	10k	29.5±37.8	0.2±0.1	-1.7±1.5	0.6±0.8	32.7±24.6	**44.9±25.7**

learned from other representation learning approaches, including the autoencoder ("AE-rep"), autoencoder with latent forward dynamics ("AE-fwd-rep") without the ODE structure and the T-symmetry regularization in TDM, and a recent popular self-supervised representation learning method SimSiam (Chen & He, 2021) ("SimSiam"). To further investigate the impact of enforcing the ODE property, we also consider a variant of TDM ("TDM-no-ODE") by removing the ODE structure in latent forward and reverse dynamics. More detailed experiment setups are presented in Appendix C.

The results demonstrate that TDM representation achieves the best performance in all small-dataset experiments. By comparing "TSRL-no-A" and "TDM-no-ODE" with "AE-rep" and "AE-fwd-rep", we can see that the bi-directional design and the T-symmetry regularization are crucial for performance improvement. Moreover, we find "TSRL-no-A" consistently achieves better performance and lower variance as compared to "TDM-no-ODE", further confirming the benefit of incorporating ODE structure in producing a well-behaved representation for downstream tasks under small datasets.

Evaluation on data augmentation. We evaluate the impact of the proposed T-symmetry consistent latent space data augmentation in Figure 5. The results show that our proposed data augmentation scheme can help speed up convergence and reduce variance. Compared with less principled data augmentation methods such as adding zero-mean Gaussian noises as in S4RL (Sinha et al., 2022), our method offers much better performance improvement. As shown in Figure 5, blindly adding random perturbations could suffer from performance degradation over the course of training, while TSRL with T-symmetry consistent data augmentation enjoys better training robustness.

Additional ablations. We also investigate the joint impact of the three design components in TSRL, including T-symmetry regularized representation and policy constraints,

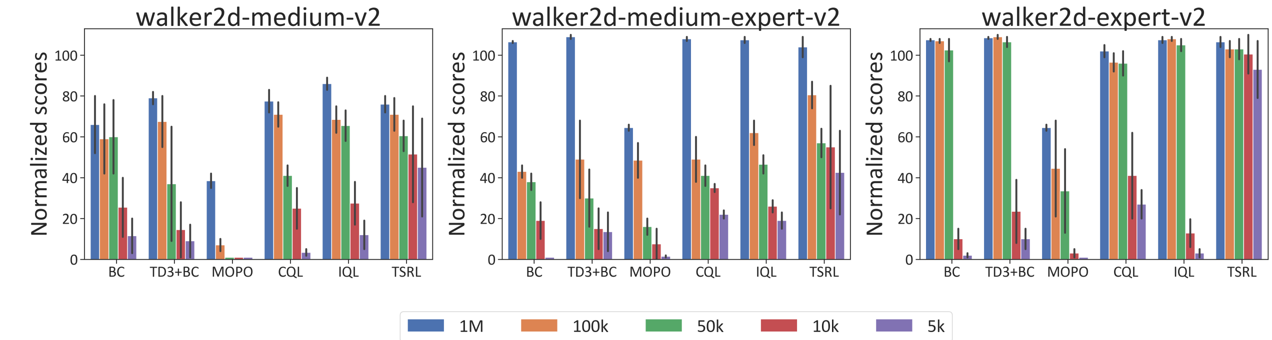

Figure 3: Performance of TSRL and baselines algorithms on datasets with different sample sizes

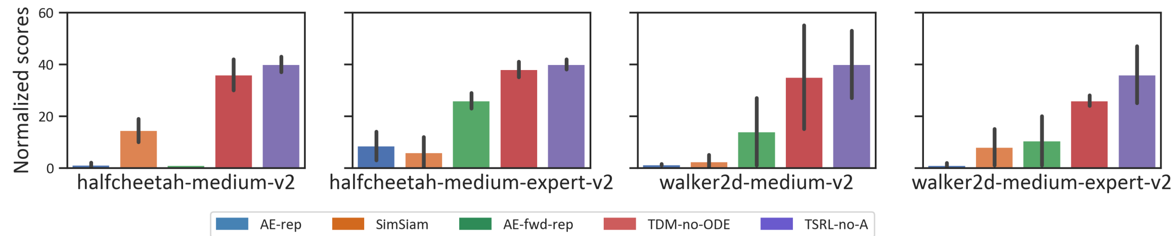

Figure 4: Impacts of different representation learning methods on 10k datasets.

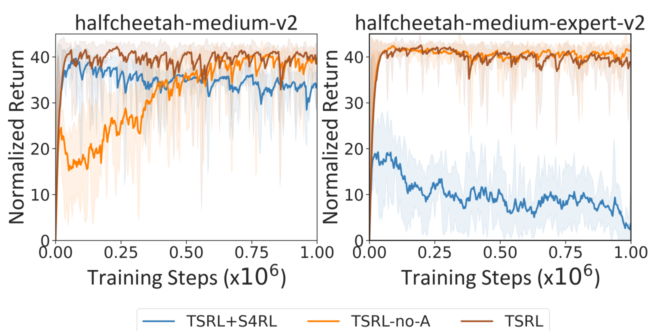

Figure 5: Impact of T-symmetry consistent latent space data augmentation on Halfcheetah medium and medium-expert 10k datasets. "TSRL+S4RL" denotes replacing the latent space data augmentation in TSRL with the zero-mean Gaussian noise ($N(0, 0.1\boldsymbol{I})$) as proposed in S4RL(Sinha et al., 2022).

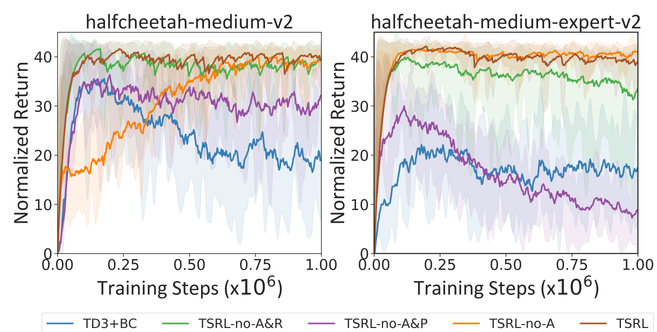

Figure 6: Ablation on TSRL on Halfcheetah medium and medium-expert 10k datasets. "no-R" denotes no T-symmetry regularized representation; "no-P" denotes no T-symmetry policy constraints, and use BC constraint similar to TD3+BC; "no-A" denotes no T-symmetry consistent latent space data augmentation.

as well as the T-symmetry consistent latent space data augmentation. We include TD3+BC for comparison, as it can be perceived as a vanilla version of TSRL without the previous three components. Moreover, the variant of TSRL with only the latent representation removed is not evaluated, as the latent space data augmentation also depends on the representation provided by TDM. Figure 6 presents the performance of all variants of TSRL on the 10k Halfcheetah medium and medium-expert datasets. As expected, it is observed that T-symmetry regularized representation and policy constraints jointly play a critical role in maintaining the performance of TSRL. As discussed previously, adding T-symmetry consistent latent space data augmentation also shows a positive impact on performance. Furthermore, it is observed that TD3+BC suffers from over-fitting on small datasets as its performance drops significantly with the increase of training steps, especially in the 10k Halfcheetah-medium dataset, while this phenomenon is not observed in TSRL. Nevertheless, the complete TSRL achieves the best performance in both tasks.

Generalization performance. To further verify the generalizability of TSRL, we construct a low-speed dataset from the Walker2d-medium dataset by filtering out all high x-velocity samples (x-velocity of the top$>$ $0.2\times$ max-x-velocity). This results in a smaller dataset (about 200k samples) with a large proportion of transition dynamics unobserved. We want to test if the agent can still generalize and learn well given only these low x-velocity data, with all high-speed samples removed. The experiment results are

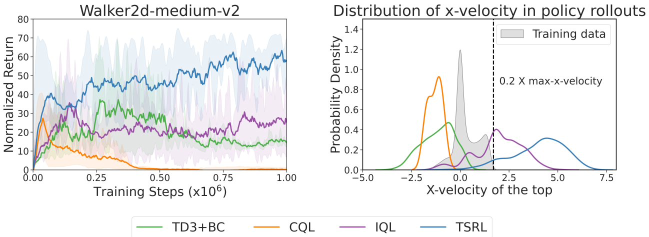

Figure 7: Comparison of TSRL and baselines trained on the Walk2d-medium dataset that removing all samples with x-velocity of the top$>0.2\times$ max-x-velocity recorded in the data. Left: learning curves. Right: x-velocity distribution of policy evaluation rollouts during the last 10k training steps. Additional generalization results on other filtered D4RL datasets can be found in Appendix C.

presented in Figure 7, for other results see Appendix C. It is observed that the baseline methods perform poorly when trained with only the low-speed dataset. This is primarily due to over-conservative data-related regularizations, which cause ineffective policy learning if the OOD region occupies the majority of state-action space. However, we observe that TSRL is still able to achieve good performance, due to the access to more fundamental dynamics information that remains invariant in both low- and high-speed data. This can be further verified if we inspect the policy rollout distribution (right figure of Figure 7) that the policy learned by TSRL indeed generalizes to high-speed behavior that is not present in the training data.

6. Related Work

Learning fundamental dynamics in physical systems. Learning conservation laws or invariant properties within a physical system is an active research area in physics (Anderson & Wendt, 1995; Grigorenko & Grigorenko, 2003; Brunton et al., 2016; Champion et al., 2019), climate science (Trenberth & Trenberth, 1992), and neuroscience (Izhikevich, 2007), etc. A classic approach is based on Koopman theory, which represents the nonlinear dynamics in terms of an infinitedimensional linear operator (Mezić, 2005). In practice, this is achieved by finding a coordinate transformation to produce a finite-dimensional representation in which the non-linear dynamics are approximately linear. However, it also suffers from computationally expensive coordinate transformations and is only able to approximate the system dynamics. Another approach is utilizing a sparse regression model with the fewest terms to describe the nonlinear system dynamics (Brunton et al., 2016; Champion et al., 2019). However, it assumes that the dynamical systems only have a few critical terms, which severely limits the model expressiveness and often requires prior knowledge of these critical terms. Based on expressive deep neural networks, a recently emerged research direction is to build ODE networks to learn conservation law in the dynamical system from data (Chen et al., 2018; Dupont et al., 2019; Liu et al., 2019; Huh et al., 2020). Our proposed TDM falls within this direction, which models both forward and reverse latent ODE dynamics with deep neural networks and incorporates additional regularization on T-symmetry.

Offline reinforcement learning. Offline RL addresses the challenge of deriving policies from fixed, pre-collected datasets without interaction with the environment. Under this offline learning paradigm, conventional off-policy RL approaches are prone to substantial value overestimation when there is a large deviation between the policy and data distributions. Existing offline RL methods address this issue by following several directions, such as constraining the learned policy to be "close" to the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019; Fujimoto & Gu, 2021; Wang et al., 2020), regularizing value function on OOD samples (Kumar et al., 2020; Kostrikov et al., 2021a; Xu et al., 2022b), enforcing strict in-sample learning (Brandfonbrener et al., 2021; Kostrikov et al., 2021b; Xu et al.; 2022a), and performing pessimistic policy learning with uncertainty-based reward or value penalties (Yu et al., 2020; Kidambi et al., 2020; Zhan et al., 2022; Bai et al., 2021; An et al., 2021). Most existing offline RL methods adopt the pessimism principle and avoid policy evaluation on OOD samples. Although this treatment helps to alleviate exploitation error accumulation, it can be over-conservative and causes severe performance degradation if the training dataset is small or has poor state-action space coverage (Li et al., 2022). TSRL tackles this issue by allowing dynamics explainable OOD samples for policy optimization, thus offering greatly improved small-sample performance.

7. Discussion and Conclusion

In this paper, we propose a physics-informed dynamics model TDM and a new offline RL algorithm TSRL, which exploit the fundamental symmetries in the system dynamics for sample-efficient offline policy learning. TDM embeds and enforces T-symmetry between a pair of latent forward and reverse ODE dynamics to learn fundamental dynamics patterns in data. The well-behaved representations and a new reliability measure for OOD samples based on T-symmetry from TDM can be readily used to construct the proposed TSRL algorithm, which achieves strong performance on small D4RL benchmark datasets and exhibits good generalization ability. There are also some limitations in our proposed approach. For example, in order to learn a well-behaved dynamics model, we introduced a set of dynamics and symmetry regularizations in TDM, which are beneficial to improve model generalization, but will lose some model expressiveness. However, we believe this can be a worthwhile trade-off between precision and generalization under small dataset settings, due to substantially improved model robustness.

References

Agarwal, R., Machado, M. C., Castro, P. S., and Bellemare, M. G. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint arXiv:2101.05265*, 2021.

An, G., Moon, S., Kim, J.-H., and Song, H. O. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Anderson, J. D. and Wendt, J. *Computational fluid dynamics*, volume 206. Springer, 1995.

Bai, C., Wang, L., Yang, Z., Deng, Z.-H., Garg, A., Liu, P., and Wang, Z. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

Bluman, G. W. and Kumei, S. *Symmetries and differential equations*, volume 81. Springer Science & Business Media, 2013.

Brandfonbrener, D., Whitney, W., Ranganath, R., and Bruna, J. Offline rl without off-policy evaluation. *Advances in Neural Information Processing Systems*, 34:4933–4946, 2021.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Champion, K., Lusch, B., Kutz, J. N., and Brunton, S. L. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

Chen, J. and Jiang, N. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pp. 1042–1051. PMLR, 2019.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Chen, X. and He, K. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural odes. *Advances in Neural Information Processing Systems*, 32, 2019.

Elliott, J. and Dawber, P. *Symmetry in Physics, Volume 1 and 2*. Macmillan, 1979.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596, 2018.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Grigorenko, I. and Grigorenko, E. Chaotic dynamics of the fractional lorenz system. *Physical review letters*, 91(3): 034101, 2003.

Huh, I., Yang, E., Hwang, S. J., and Shin, J. Time-reversal symmetric ode network. *Advances in Neural Information Processing Systems*, 33:19016–19027, 2020.

Izhikevich, E. M. *Dynamical systems in neuroscience*. MIT press, 2007.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, pp. 12519–12530, 2019.

Jiang, S., Pang, J., and Yu, Y. Offline imitation learning with a misspecified simulator. *Advances in neural information processing systems*, 33:8510–8520, 2020.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Kostrikov, I., Fergus, R., Tompson, J., and Nachum, O. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021a.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *Deep RL Workshop NeurIPS 2021*, 2021b.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pp. 11761–11771, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Lamb, J. S. and Roberts, J. A. Time-reversal symmetry in dynamical systems: a survey. *Physica D: Nonlinear Phenomena*, 112(1-2):1–39, 1998.

Le, H. M., Voloshin, C., and Yue, Y. Batch policy learning under constraints. *arXiv preprint arXiv:1903.08738*, 2019.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Li, J., Zhan, X., Xu, H., Zhu, X., Liu, J., and Zhang, Y.-Q. When data geometry meets deep function: Generalizing offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022.

Liu, X., Xiao, T., Si, S., Cao, Q., Kumar, S., and Hsieh, C.-J. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355*, 2019.

Lyu, J., Li, X., and Lu, Z. Double check your state before trusting it: Confidence-aware bidirectional offline model-based imagination. *arXiv preprint arXiv:2206.07989*, 2022.

Mezić, I. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41: 309–325, 2005.

Sinha, S., Mandlekar, A., and Garg, A. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pp. 907–917. PMLR, 2022.

Srinivas, A., Laskin, M., and Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Trenberth, K. E. and Trenberth, K. E. *Climate system modeling*. Cambridge University Press, 1992.

Uehara, M., Zhang, X., and Sun, W. Representation learning for online and offline rl in low-rank mdps. *arXiv preprint arXiv:2110.04652*, 2021.

Wang, J., Li, W., Jiang, H., Zhu, G., Li, S., and Zhang, C. Offline reinforcement learning with reverse model-based imagination. *Advances in Neural Information Processing Systems*, 34:29420–29432, 2021.

Wang, Z., Novikov, A., Zolna, K., Merel, J. S., Springenberg, J. T., Reed, S. E., Shahriari, B., Siegel, N., Gulcehre, C., Heess, N., et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33: 7768–7778, 2020.

Weissenbacher, M., Sinha, S., Garg, A., and Yoshinobu, K. Koopman q-learning: Offline reinforcement learning via symmetries of dynamics. In *International Conference on Machine Learning*, pp. 23645–23667. PMLR, 2022.

Wu, Y., Tucker, G., and Nachum, O. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Xu, H., Jiang, L., Li, J., Yang, Z., Wang, Z., Chan, V. W. K., and Zhan, X. Offline rl with no ood actions: In-sample learning via implicit value regularization. In *The Eleventh International Conference on Learning Representations*.

Xu, H., Li, J., Li, J., and Zhan, X. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022a.

Xu, H., Zhan, X., and Zhu, X. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022b.

Yang, C., Ma, X., Huang, W., Sun, F., Liu, H., Huang, J., and Gan, C. Imitation learning from observations by minimizing inverse dynamics disagreement. *Advances in neural information processing systems*, 32, 2019.

Yang, M. and Nachum, O. Representation matters: offline pretraining for sequential decision making. In *International Conference on Machine Learning*, pp. 11784–11794. PMLR, 2021.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Zhan, X., Xu, H., Zhang, Y., Zhu, X., Yin, H., and Zheng, Y. Deepthermal: Combustion optimization for thermal power generating units using offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Zhang, A., McAllister, R., Calandra, R., Gal, Y., and Levine, S. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

Zhou, W., Bajracharya, S., and Held, D. Plas: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pp. 1719–1735. PMLR, 2021.

A. Implementation Details

Implementation details for TDM. TSRL can be implemented based upon TD3 (Fujimoto et al., 2018), with the addition of the proposed T-symmetry regularized representation and policy constraints as in Eq. (9) and (11), as well as the T-symmetry consistent latent space data augmentation. In our experiments, we generate $K = 1$ augmented samples for each transition in the dataset, and filter based on the T-symmetry consistency loss. The pseudo-code of TSRL is summarized in Algorithm 1.

- **Network structure:** In all our D4RL experiments, we implement the encoder, decoders, latent forward and reverse dynamics as 4-layer feed-forward neural networks with ReLU activation, and optimized using Adam optimizer. For the state decoder $\psi(\cdot, \delta_s)$, we concatenate an extra indicator δ_s in the input to help the state decoder to decide the target output. More specifically, to decode $z_s \to s$, we concatenate $\delta_s = 0$ with z_s as input; and for $\dot{z}_s \to \dot{s}$, we concatenate $\delta_s = 1$ with $\dot{z}_s$.

- **Computing second derivative of $\phi(\cdot)$:** As TDM involves a pair of latent ODE forward and reverse dynamics models, whose training losses Eq. (4) and (6) involve regressing on $\frac{\partial \phi(s,a)}{\partial s}\dot{s}$ and $\frac{\partial \phi(s',a)}{\partial s'}(-\dot{s})$ as target values. This results in a gradient through a gradient of $\phi(\cdot)$. Computationally, we calculate the Jacobian matrix $\frac{\partial \phi(s,a)}{\partial s}$ using the `vmap()` function in Functorch[2] to ensure the second derivative of $\phi(\cdot)$ can be correctly backpropagated during stochastic gradient descent. Similar a treatment can also be implemented with other auto-differentiation frameworks like Jax[3] that support computing higher-order derivatives.

- **Pre-training the encoder and decoders:** As the final learning objective of TDM Eq. (8) involves several loss terms, we observe that in small datasets, loss terms such as the reconstruction loss (Eq. (3)) for the encoder and decoders converges much slower than other loss terms. When updating all the loss terms with the same number of training steps, some losses suffer from over-fitting while others are still not fully converged. For these cases, we pre-train the encoder and decoders with the reconstruction loss for a given number of training steps, and then use the complete learning objective of TDM (Eq. (4)) for the rest of the training. The numbers of pre-training/training epochs for the experiments in this paper are reported in Table 2.

  As reported in Table 2, we find that the number of pre-training epochs required for TDM to reach the best learning performance is associated with the specific task and the size of training data. For small datasets, TDM generally needs more training and pre-training epochs to avoid overfitting the latent dynamics and T-symmetry losses. For MuJoCo locomotion tasks, we recommend pre-training the encoder and decoders for 10% of the total training epoch. For the more complex adroit tasks, TDM requires more epochs to extract the ODE dynamics and T-symmetry property of the system dynamics. In this case, there is no pre-training necessary for the encoder and decoders.

Table 2: Training epochs of TDM for D4RL tasks with different dataset scales

	Locomotion Tasks			**Adroit Tasks**	
	5k&10k	50k & 100k	Full dataset	5k&10k	Full dataset
Training epoch	2000	1000	200	2000	200
Pre-train epoch	200	100	20	0	0

- **Enhancement on the T-symmetry regularization:** We observe that in some small datasets (mainly in the Halfcheetah environment), the training of the latent reverse dynamics model g might suffer from a certain level of degeneration. This is reflected as the $g(z_s + f(z_s, z_a), z_a)$ produces similar values as $-f(z_s, z_a)$, resulting in small T-symmetry consistency loss values (Eq. (7)), however, the discrepancy between $g(z_{s'}, z_a)$ and $-f(z_s, z_a)$ remains large. To solve this issue and further enforce the T-symmetry, we apply the following enhanced T-symmetry regularization when such a phenomenon is observed:

$$\ell_{Enhanced\text{-}T\text{-}sym}(z_s, z_a) = \|f(z_s, z_a) + g(z_s + f(z_s, z_a), z_a)\|_2^2 + \|f(z_s, z_a) + g(z_{s'}, z_a)\|_2^2 \tag{13}$$

  We find that applying the above enhanced T-symmetry loss can successfully resolve the degeneration issue of the latent reverse dynamics model and achieve good performance in the downstream offline RL tasks. However, we find in most small datasets, the original T-symmetry consistency loss is sufficient. We advise only to use the above enhanced T-symmetry

[2]https://pytorch.org/functorch/stable/functorch.html
[3]https://github.com/google/jax

Algorithm 1 T-Symmetry Regularized Offline RL (TSRL)

Require: Offline dataset $\mathcal{D}$, encoder ϕ, latent forward and reverse dynamics models f and g from TDM trained using objective Eq. (8).
Compute the T-symmetry consistency loss $\ell_{T\text{-}sym}(\phi(s,a))$ (Eq. (7)) for all samples in $\mathcal{D}$, and set their τ-quantile value as the augmentation threshold h.
Initialize the policy network π, critic networks Q and their target network.
for $t = 1, \cdots, M$ training steps **do**
    Sample a mini-batch B of samples $\{(s,a,r,s')\} \sim \mathcal{D}$ and compute their representations $\{(z_s, z_a, z_{s'})\}$.
    // T-symmetry consistent latent space data augmentation
    Generate K perturbed samples by adding perturbations $\epsilon \sim N(0, 0.01\sigma_{z_s})$ on latent states z_s of each sample in B, where σ_{z_s} is the std of latent states in data.
    Add augmented samples $(z_s+\epsilon, z_a, z_{s'}+\epsilon')$ to B if satisfies $\ell_{T\text{-}sym}(z_s+\epsilon, z_a) \le h$.
    // Critic training with T-symmetry regularized representation
    Update the value function Q based on the policy evaluation objective Eq. (9).
    // Policy training with T-symmetry regularized policy constraints
    Update the policy π based on the policy improvement objective Eq. (11).
    Soft update the target networks.
end for

consistency loss when large discrepancies between $\|f(z_s,z_a)+g(z_s+f(z_s,z_a),z_a)\|_2^2$ and $\|f(z_s,z_a)+g(z_{s'},z_a)\|_2^2$ are observed.

Implementation details for TSRL. TSRL can be implemented based upon TD3 (Fujimoto et al., 2018), with the addition of the proposed T-symmetry regularized representation and policy constraints as in Eq. (9) and (11), as well as the T-symmetry consistent latent space data augmentation. In our experiments, we generate $K=1$ augmented samples for each transition in the dataset, and filter based on the T-symmetry consistency loss. The pseudo-code of TSRL is summarized in Algorithm 1.

Hyperparameter details. The architecture parameters of TDM and TSRL, as well as the TSRL hyperparameters are summarized in Table 3. It should be noted that **we use the same set of TSRL hyperparameters for all D4RL-MuJoCo experiments in this paper without tuning**. It is expected that fine-tuning the hyperparameters could potentially produce much better results. But as off-policy evaluation in real-world scenarios can be rather difficult, which in most cases makes offline hyperparameter fine-tuning infeasible, thus we choose to report all results under the same set of hyperparameters. We find these hyperparameters already produce good performance on both small and large datasets of D4RL MuJoCo tasks.

B. Detailed Experiment Setups

Reduced-size dataset generation. To create reasonable reduced-size D4RL datasets for a fair comparison, we sub-sample the trajectories in the datasets rather than directly sampling the (s,a,s',r) transitions. For example, there are 2M (s,a,s',r) transitions in the "halfcheetah-medium-expert" dataset, we first split these records into 2,000 trajectories based on the done condition, then randomly draw 10 trajectories (10k transition points) to serve as the reduced-size datasets for model training.

Experiment setups for representation learning evaluation. To evaluate the representation quality and the impact of each design choice of TDM, we compare TDM representation with several baselines on the small dataset settings. We provide the detailed description of the representation learning baselines as follows:

- **"AE-rep" model:** We construct a vanilla auto-encoder without any further constraints during the learning process, which was trained by the reconstruction loss only. The network sizes of the encoder and decoders are the same as the ones used in TDM.
- **"AE-fwd-rep" model:** Similar to the "AE-rep" model but with a latent forward dynamics prediction model f, which is implemented as a 4-layer feed-forward neural network with ReLU activation, and optimized using Adam optimizer (same as TDM). The forward model was trained by minimizing the loss term $\|\dot{z}_s - f(\phi(s,a))\|_2^2$, where we directly regress $f(\phi(s,a))$ with the $\dot{z}_s$ derived from the latent states obtained from the encoder as $\dot{z}_s = z_{s'} - z_s$. Note that in this baseline, no ODE property nor T-symmetry regularization is included. Again we use the decoder to decode $\dot{z}_s \rightarrow \dot{s}$ as in TDM for the next state prediction.
- **"TDM-no-ODE" model:** Holds the same structure with TDM but trained with no ODE property. More specifically,

Table 3: Hyperparameter details for TDM and TSRL

	Hyperparameters	Value
TDM Architecture	Optimizer type	Adam
	Weight of ℓ_{T-sym} and ℓ_{ds} and ℓ_{rec}	1
	Weight of ℓ_{rvs} and ℓ_{fwd}	0.1
	Learning rate	3×10^{-4}
	State normalization	True
	Hidden units of forward and reverse model	512
	Hidden units of encoder	$512 \times 256 \times 128$
TSRL Architecture	Critic neural network layer width	512
	Actor neural network layer width	512
	State normalization	True
	Actor learning rate	3×10^{-4}
	Critic learning rate	3×10^{-4}
	Policy noise	0.2
	Policy noise clipping	0.5
	Policy update frequency	2
	Discount factor γ	0.99
	Number of iterations	10^6
	Target update rate	0.005
	λ_{L1}	1e-5
TSRL Hyperparameters	α	2.5
	τ	50% for Walker2d and Adroit tasks, 70% for HalfCheetah and Hopper2d
	λ_1	MuJoCo: 5 or 10 for full dataset, 100 or 200 for 10k dataset Adroit: 10,000 for both full and reduced datasets
	λ_2	1 for MuJoCo full & Adroit datasets 100 for MuJoCo 10k dataset

similar with "AE-fwd-rep", the latent forward and reverse dynamics model was trained by $\|\dot{z}_s - f(\phi(s,a))\|_2^2$ and $\|(-\dot{z}_s) - g(\phi(s',a))\|_2^2$, where $-\dot{z}_s$ is directly calculated from the encoded latent states, i.e., $\dot{z}_s = z_{s'} - z_s$. Note that in this baseline, the T-symmetry is also implicitly captured, since both the latent forward and reverse dynamics models are regressing the same $\dot{z}_s$ and its opposite value.

- **"SimSiam" model:** For the self-supervised representation learning baseline, we implement an auto-encoder structure with the optimization objective proposed in the SimSiam paper (Chen & He, 2021). For detailed model description and hyperparameters setting, please refer to Chen et al. (Chen & He, 2021).

Experiment setups for evaluating generalization performance. To evaluate TSRL's generalization capability beyond the offline datasets, we construct two low-speed datasets based on the original D4RL Walker2d medium and medium-expert datasets. In accordance with the Gym documentation, we selected the "x-coordinate velocity of the top" (8th dimension of the states) in the walker environment to perform data filtering. We remove all samples with the x-coordinate velocity of the top greater than $0.2\times$ max-x-velocity recorded in the data. This results in two smaller low-speed datasets (about 200k for the medium dataset and 250k for the medium-expert dataset). We train TDM and TSRL on these low-speed datasets and the results are reported in Figure 7 (main paper) and 8.

C. Additional Results

Complete results on D4RL Adroit tasks. The complete results of TSRL in Adroit human and cloned tasks with different dataset scales are presented in Table 4. As shown in the results, TSRL achieves much better performance in the pen tasks, both the full datasets and the reduced-size datasets.

Table 4: Complete results on D4RL Adroit tasks

Task	Ratio	Size	BC	TD3+BC	MOPO	CQL	IQL	TSRL
Pen-human	1	5k	34.4	8.4	9.7	37.5	71.5	**80.1±18.1**
Hammer-human	1	5k	1.5	2.0	0.2	**4.4**	1.4	0.2± 0.3
door-human	1	5k	0.5	0.5	-0.2	**9.9**	4.3	0.5± 0.3
Relocate-human	1	5k	0.0	-0.3	-0.2	**0.2**	0.1	0.1 ± 0.1
Pen-cloned	1	500k	56.9	41.5	-0.1	39.2	37.3	**64.9 ± 20.1**
	1/50	10k	37.4 ± 37.6	-0.1 ± 6.9	-0.1 ± 0.1	1.5 ± 4.8	35.6 ± 30.5	**41.6 ± 27.5**
Hammer-cloned	1	500k	0.8	0.8	0.2	**2.1**	**2.1**	1.7 ± 1.9
	1/50	10k	0.3 ± 0.4	0.2 ± 0.1	0.1 ± 0.1	0.2 ± 0.1	0.4 ± 0.2	**0.6 ± 0.3**
Door-cloned	1	500k	-0.1	-0.4	-0.1	0.4	**1.6**	-0.1 ± 0.6
	1/50	10k	-0.1 ± 0.1	-0.3 ± 0.1	-0.2 ± 0.1	-0.3 ± 0.1	**1.5 ± 0.8**	-0.1 ± 0.3
Relocate-cloned	1	500k	-0.1	-0.3	-0.3	**0.1**	-0.2	-0.2 ± 0.1
	1/50	10k	-0.2 ± 0.1	-0.3 ± 0.1	-0.3 ± 0.1	-0.3± 0.1	**-0.1 ± 0.5**	-0.2 ± 0.1

Table 5: Results on D4RL Antmaze-umaze tasks with full and reduced-size datasets

Task	Ratio	Size	BC	TD3+BC	CQL	IQL	TSRL(ours)
Antmaze-u	1	1M	54.6	78.6	84.8	**85.5**	81.4 ± 19.2
	1/100	10k	44.7 ±42.1	0.7 ± 1.2	0.1 ± 0.0	65.1 ± 19.4	**76.1±15.6**
Antmaze-u-d	1	1M	45.6	71.4	43.4	66.7	**76.5 ± 29.7**
	1/100	10k	24.1±22.2	16.27 ± 16.4	0.5 ± 0.1	34.6 ± 18.5	**52.2 ±22.1**

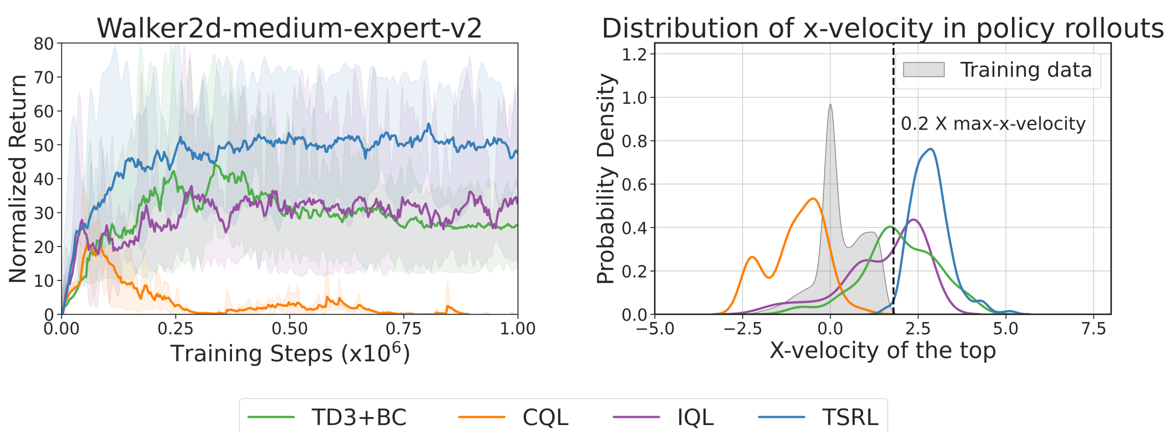

Figure 8: Comparison of TSRL and baselines trained on the Walker2d-medium-expert dataset that removing all samples with x-velocity of the top$>0.2\times$ max-x-velocity recorded in the data. Left: learning curves. Right: x-velocity distribution of policy evaluation rollouts during the last 10k training steps.

Additional results on Antmaze-umaze tasks. We also conduct experiments on the D4RL Antmaze-umaze tasks with full and reduced-size 10k datasets. The results are presented in Table 5. We use the same hyperparameters as in the D4RL MuJoCo tasks. Again, we find that TSRL achieves comparable performance as other baselines on the full datasets, but is substantially better under small datasets.

Additional results on the generalization performance. Due to the limited space of the main paper, we provide the additional experiment results on the low-speed Walker2d-medium-expert dataset in Figure 8. Similar to the results on the low-speed Walker2d-medium dataset (Figure 7 in the main paper), we find that the policy learned by TSRL effectively generalizes to novel high-speed behaviors that are not present in the offline data but can achieve high returns.

In both low-speed Walker2d medium and medium-expert datasets, a large proportion of transitions in the high-speed regime are removed. Under this setting, we find that all existing offline RL baselines fail to achieve reasonable performance, due to their over-conservative regularization on the offline dataset. CQL performs especially poorly in both tasks, perhaps due to over-conservative value function learning that impedes the policy to acquire some necessary control strategy to finish the task. TD3+BC performs poorly in the low-speed medium dataset, probably because this dataset has a narrower data distribution than the medium-expert dataset, and the latter could still contain some samples with reasonable speeds after filtering with x-velocity$> 0.2\times$max-x-velocity in the dataset. IQL exhibits some level of generalization capability but is still much weaker as compared to TSRL. By comparison, we observe that the policy rollout distribution generated by TSRL policies can substantially deviate from the training data distribution while also achieving very good performance.

Ablation on the level of ODE and T-symmetry regularization in TDM. As discussed in the conclusion section of the main paper, TDM adds extra ODE dynamics and symmetry regularizations, which are beneficial to improve model generalization, but will lose some model expressiveness if the regularization is too strong. In this section, we conduct an ablation study on the impact of the regularization strength of the ODE property and T-symmetry satisfaction. Specifically, we vary the loss weights of ℓ_{fwd}, ℓ_{rvs} and ℓ_{T-sym} in the TDM learning objective (Eq. (8)), and train a loosely regularized and a strongly regularized TDM model on the 10k datasets (see Table 6). The loosely regularized model has the maximum reconstruction expressivity but may not produce a well-behaved representation due to weak regularization. Whereas the strongly regularized model sacrifices the expressivity for regularized behaviors. We further evaluate their performance with TSRL, with the results reported in Table 7. The experiment results demonstrated that an overly expressive model could not help the RL algorithm to derive a well-behaved policy with limited data due to potential overfitting and inconsistency with the T-symmetry property. On the other hand, an overly regularized model may also hurt performance. This is consistent with our previous insight that a trade-off exists between model expressiveness and T-symmetry agreement. A proper balance between these two behaviors can be necessary for small-sample learning.

Table 6: TDM with different regularization strengths

Different versions of TDM	ℓ_{rec}	ℓ_{ds}	ℓ_{fwd}	ℓ_{rvs}	ℓ_{T-sym}	λ_{L1}
Loosely regularized	1	1	0.01	0.01	0.01	1e-5
Paper	1	1	0.1	0.1	0.1	1e-5
Strongly regularized	1	1	1	1	1	1e-5

Table 7: Performance of TSRL with different TDM models on 10k datasets

Task	**TDM (loosely regularized)**	**TDM (paper)**	**TDM (strongly regularized)**
Hopper-m	50.7±13.6	**62.0**±**3.7**	43.6±14.3
Hopper-m-r	15.4±9.7	**21.8**±**8.2**	15.6±9.8
Hopper-m-e	49.7±17.1	**50.9**±**8.6**	30.9±20.5
Halfcheetah-m	**39.1**±**3.6**	38.4±3.1	36.6±30.0
Halfcheetah-m-r	**28.3**±**6.9**	28.1±3.5	22.9±8.4
Halfcheetah-m-e	36.2±5.4	**39.9**±**21.1**	31.0 ± 3.4
Walker2d-m	43.2±27.3	**49.7**±**10.6**	35.6±26.2
Walker2d-m-r	20.2±18.1	**26.0**±**11.3**	21.7±6.1
Walker2d-m-e	25.9±20.7	**46.4**±**17.4**	29.4±24.7

Learning curves for TSRL. The learning curves for reduced-size D4RL-MuJoCo datasets with 10k samples are showed in Figure 9. The policies are evaluated with 5 episodes over 3 random seeds.

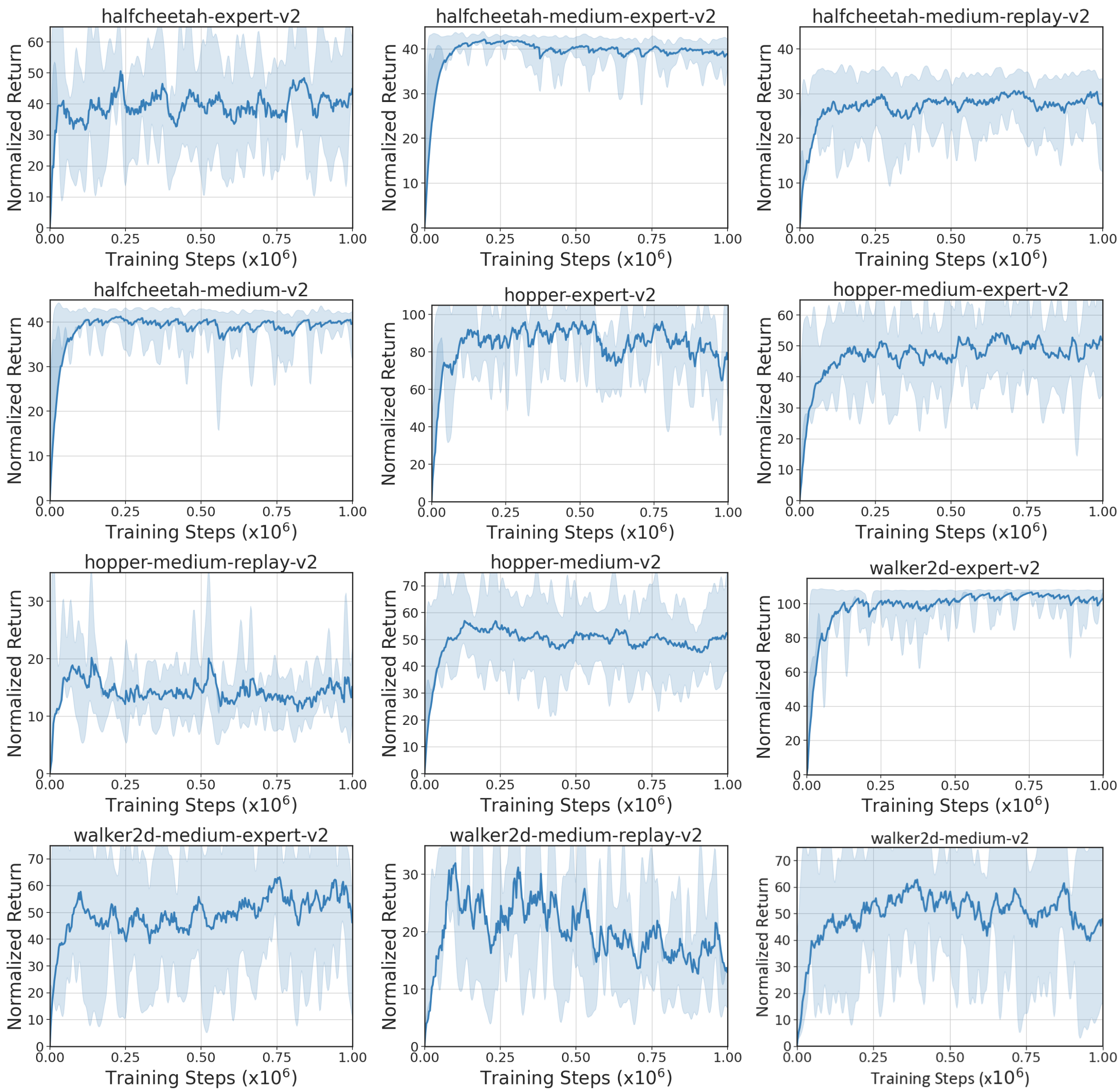

Figure 9: Learning curves for reduced-size D4RL MuJoCo datasets. Error bars indicate min and max over 5 seeds.