Appendix

A Overview

This supplementary document provides additional technical details, hyperparameteres used in implementations and experiments and more quantitative results to the main paper.

In Section B, some essential notations and preliminaries will be presented, then in Section C, we will provide detailed derivations to the approximate solving of our policy optimization problem with mimicry constraints. Specifically, the solving of duality and a closed-form solution will be demonstrated. Then in Section D, the hyperparameters used in the implementations of all the baselines and MCPO will be presented, while some specifications to the tested environments and tasks will also be illustrated. Finally in Section E, some numerical results in the comparative and ablation experiments will be listed, and some analysis on the mimicry constraint value under different experiment settings will be provided.

B Preliminaries

This section reviews some fundamental conceptions presented in our method. Before further introduction, we first provide the key notations used in this paper.

Notations. For modeling the action decision process in our context, a standard Markov decision process (MDP) [Sutton and Barto, 1998] $(\mathcal{S}, \mathcal{A}, r, \mathcal{T}, \mu, \gamma)$ is considered, where $\mathcal{S}$ and $\mathcal{A}$ denotes the space of feasible states and actions respectively, $r(s,a) \to \mathbb{R}$ is the reward function, $\mathcal{T}(s'|s,a)$ and $\mu(s)$ represent the transition probability and initial state distribution and $\gamma \in (0,1)$ is the discount factor. A stochastic policy $\pi(a|s) : \mathcal{S} \times \mathcal{A} \to [0,1]$ maps state into action distribution. A trajectory ζ is given by the sequence of state-action pairs $\{(s_0,a_0),(s_1,a_1),...\}$.

B.1 Local Policy Search for Reinforcement Learning

Local policy search [Kakade, 2002] is a category of RL methods that iteratively update a θ-parameterized policy π_θ by maximizing the expected advantage over a local neighborhood of the most recent iterate π_{θ_k}

$$\begin{aligned} \theta_{k+1} = \arg\max_{\theta} \ & \mathbb{E}_{\pi_{\theta_k}} \left[A_{\pi_{\theta_k}}(s,a) \right] \\ \text{s.t. } & \mathbb{D}_{\mathrm{KL}} \left(\pi_{\theta_k} \| \pi_{\theta_{k+1}} \right) \leq \delta, \end{aligned} \tag{1}$$

where δ determines the step size per update, A_π is advantage action value and $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$. V_π and Q_π are state and state-action value functions in RL.

Recent popular RL approaches including TRPO [Schulman *et al.*, 2015] and PPO [Schulman *et al.*, 2017] are both built upon this paradigm. In this paper, we also consider the optimization problem in (1) and develop our Mimicry Constraint Policy Optimization (MCPO) algorithm based on it.

B.2 Occupancy Measure

Occupancy measure [Puterman, 1994; Syed *et al.*, 2008] defined below characterizes the distribution of the state-action pairs within the exploration trajectories when policy π is executed, which will be useful in the following analysis.

Definition 1 (Occupancy Measure)**.** *Given a stationary policy π, let $\rho_\pi(s) : \mathcal{S} \to \mathbb{R}$ and $\rho_\pi(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denote the density of the state distribution and the joint distribution for state and action under the policy π, namely,*

$$\begin{aligned} \rho_\pi(s) &\triangleq \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi) \\ \rho_\pi(s,a) &\triangleq \rho_\pi(s)\pi(a|s). \end{aligned} \tag{2}$$

Then we name $\rho_\pi(s,a)$ as occupancy measure of policy π.

B.3 Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) [Borgwardt *et al.*, 2006; Sriperumbudur *et al.*, 2008; Gretton *et al.*, 2012] is a non-parametric relevant criterion of the discrepancy between distributions. Formally, let $\mathcal{X}$, $\mathcal{H}$ to be a feature space and an universal Reproducing Kernel Hilbert Space (RKHS) [Steinwart, 2001] respectively, and $\phi(x) : \mathcal{X} \to \mathcal{H}$. For two distributions p and q, MMD is an instance of the integral probability metric [Müller, 1997] defined as

$$\mathrm{MMD}[\mathcal{H}, p, q] \triangleq \left\| \int_{\mathcal{X}} \phi(x) dp(x) - \int_{\mathcal{X}} \phi(y) dq(y) \right\|_{\mathcal{H}}. \tag{3}$$

Given two set of samples $X = \{x_i\}_{i=1}^M$ and $Y = \{y_j\}_{j=1}^N$ from p and q respectively, an empirical estimation of MMD is obtained by

$$\begin{aligned}\mathrm{MMD}^2[\mathcal{H}, X, Y] =& \left\| \frac{1}{M}\sum_{i=1}^{M}\phi(x_i) - \frac{1}{N}\sum_{i=1}^{N}\phi(y_i) \right\|^2 \\ =& \frac{1}{M^2}\sum_{i=1}^{M}\sum_{i'=1}^{M}k(x_i, x_{i'}) + \frac{1}{N^2}\sum_{j=1}^{N}\sum_{j'=1}^{N}k(y_j, y_{j'}) \\ & - \frac{2}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}k(x_i, y_j),\end{aligned} \tag{4}$$

where $k(\cdot, x)$ is the corresponding reproducing kernel of feature map $\phi(x)$. Therefore, the discrepancy between two distributions can be estimated by computing the distance between the means of their samples mapped into a RKHS. Compared to other parametric distance metric like *Kullback-Leiber* (KL) divergence, MMD can avoid the bias introduced by density estimation when only samples are available.

C Solving the Optimization Problem with Mimicry Constraint

In general, considering the approximated optimization problem in our main paper, it can be seen as a constrained optimization problem with linear and quadratic constraints as shown below

$$\begin{aligned} p^* = \min_x \ & g^T x \\ \text{s.t. } & b^T x \le c \\ & \frac{1}{2}x^T H x \le \delta, \end{aligned} \tag{5}$$

where $x = (\theta - \theta_k)$. Due to the positive definiteness of the Hussein matrix H, this optimization problem would be convex. According to Slater's condition, once there exists a strictly feasible solution x, the strong duality holds and the primer problem can be solved by solving its dual problem

$$p^* = \min_x \max_{\substack{\lambda\ge 0\\ \nu\ge 0}} g^T x + \lambda(\frac{1}{2}x^T H x - \delta) + \nu(b^T x - c) \tag{6}$$

$$= \max_{\substack{\lambda\ge 0\\ \nu\ge 0}} \min_x (g + \nu b)^T x + \lambda(\frac{1}{2}x^T H x) - \nu c - \lambda\delta \tag{7}$$

$$\Rightarrow x^* = -\frac{1}{\lambda^*}H^{-1}(g + \nu^* b) \tag{8}$$

$$p^* = \max_{\substack{\lambda\ge 0\\ \nu\ge 0}} -\frac{1}{2\lambda}(g^T H^{-1} g + 2\nu b^T H^{-1} g + \nu^2 b^T H^{-1} b) - \nu c - \lambda\delta \tag{9}$$

$$\Rightarrow \nu^* = \max\left\{ -\frac{\lambda^* c + b^T H^{-1} g}{b^T H^{-1} b}, 0 \right\} \tag{10}$$

$$\Rightarrow \lambda^* = \begin{cases} \max\left\{ \sqrt{\frac{g^T H^{-1} g}{2\delta}}, 0 \right\} & \nu^* = 0, \\ \max\left\{ \sqrt{\frac{g^T H^{-1} g - (b^T H^{-1} g)/(b^T H^{-1} b)}{2\delta - (c^2)/(b^T H^{-1} b)}}, 0 \right\} & \nu^* > 0 \end{cases}. \tag{11}$$

We use the strong duality here from Eqn. (6) to Eqn. (7), then the optimal x^* could be obtained by calculating its stationary point on Lagrange function as Eqn. (8). Substituting x^* and then we can get Eqn. (9), the optimal ν^*, λ^* could be obtained by discussion on the value of ν^*. Once ν^* and λ^* is determined, we can obtain the approximated optimal θ_{k+1}

$$\theta_{k+1} = \theta_k - \frac{1}{\lambda^*}H^{-1}(g + \nu^* b). \tag{12}$$

When the original problem is infeasible, the update direction cannot be directly obtained from the original constrained optimization problem. Instead, we optimize an MMD-IL sub-problem (13) as a recovery mechanism, which aims to decrease the constraint function value as much as possible.

$$\theta^* = \arg\min_\theta \mathrm{MMD}[\mathcal{H}, \rho_{\pi_\theta}, \rho_E]. \tag{13}$$

Once there exists one feasible point, we can go back to solve original optimization problem.

For determining whether the problem is feasible or not, we calculate the nearest point on the constraint plain

$$\begin{aligned} \min_{x} \ & \frac{1}{2} x^T H x \\ \text{s.t.} \ & b^T x = c, \end{aligned} \tag{14}$$

whose optimal solution can be obtained by Lagrange method as $x^* = \frac{cH^{-1}b}{b^T H^{-1} b}$. Thus if the approximated KL discrepancy $x^{*T} H x^* = \frac{c^2}{b^T H^{-1} b} \leq 2\delta$, there could be feasible solution, otherwise, if $\frac{c^2}{b^T H^{-1} b} > 2\delta$ and at current θ_k the problem is feasible, the linear constraint would be satisfied wherever the quadratic constraint is satisfied; otherwise the problem is infeasible and recovery target Eqn. (13) should be used instead.

Figure 1 illustrates how our proposed method improves the original policy optimization with the mimicry constraint defined by the expert demonstrations. The mimicry constraint indicates an area with high return that can be used as an exploration reference. By integrating it into the policy optimization procedure, we can prevent the update direction from pointing to sub-optimal area, which encourages a better on-policy exploration efficiency.

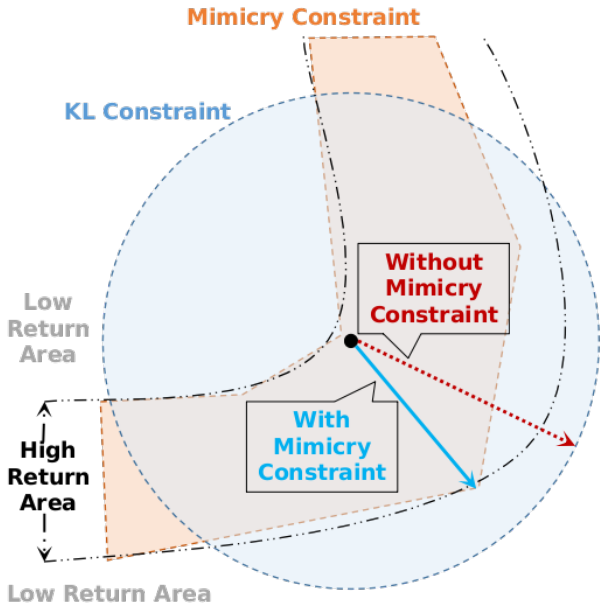

Figure 1: An overview to our proposed Mimicry Constraint Policy Optimization (MCPO) algorithm.

D Hyperparameters

Table 1 lists the parameters for Pre-training [Silver *et al.*, 2016], POfD [Kang *et al.*, 2018], PPO [Schulman *et al.*, 2017], MMD-IL and proposed MCPO used in the comparative evaluation. Table 2 lists the specifications about the benchmark environments, the number of trajectories in demonstrations and mimicry constraint tolerances for each environment.

Table 1: Hyperparameters for Evaluated Algorithms

Parameter	Value
Shared	
Optimizer	Adam [Kingma and Ba, 2015]
Learning rate	$1e^{-4}$
GAE	0.95
L2 penalty	$1e^{-3}$
Discount (γ)	0.99
Architecture of policy, value and discriminator networks	(300, 400)
Nonlinearity	Tanh
Batch size	64 (*MountainCar*), 1024 (*others*)
Pre-training	
Pre-training epoches	50
POfD	
Weight for GAN term	0.1 (0.01 for *HalfCheetah* and *Walker2d*)
Weight for entropy term	0.0
PPO	
Weight of clipping epsilon	0.2
MMD-IL	
N/A	N/A
MCPO (Ours)	
KL tolerance δ	$5e^{-3}$
Damping	$1e^{-2}$

Table 2: Hyperparameters for Evaluated Environments

Environment	$\mathcal{S}$	$\mathcal{A}$	Max-Step	Demonstrations Size	Tolerance d	Annealing factor ϵ
MountainCar-v0	$\mathbb{R}^4$	$\{0, 1\}$	200	1 traj	10^{-3}	5×10^{-3}
Pendulum-v2	$\mathbb{R}^4$	$\mathbb{R}^1$	1000	1 traj	10^{-3}	10^{-2}
DoublePendulum-v2	$\mathbb{R}^{11}$	$\mathbb{R}^1$	1000	1 traj	10^{-3}	5×10^{-2}
HalfCheetah-v2	$\mathbb{R}^{17}$	$\mathbb{R}^6$	1000	1 traj	10^{-3}	10^{-3}
Hopper-v2	$\mathbb{R}^{11}$	$\mathbb{R}^3$	1000	1 traj	10^{-4}	10^{-3}
Walker2d-v2	$\mathbb{R}^{17}$	$\mathbb{R}^6$	1000	1 traj	10^{-2}	10^{-3}
Ant-v2	$\mathbb{R}^{111}$	$\mathbb{R}^8$	1000	1 traj	10^{-5}	10^{-3}

E Empirical Results

We evaluate MCPO against several baselines on seven physics-based control benchmarks [Duan *et al.*, 2016], ranging from low-dimensional classic control to challenging high-dimensional continuous robotic control tasks. All experiments are evaluated using the exact reward functions, defined by OpenAI Gym [Brockman *et al.*, 2016]. We aim to first investigate the effectiveness of our method compared with other counterparts and then performing an ablations study on determining the impact on the mimicry constraint of our method.

Table 3: Comparison results. All results are measured in the original exact reward.

Environment	Sparsification	Demo	Pre-training	POfD	MCPO (Ours)
MountainCar-v0	SPARSE1	81.25	82.73±6.91	49.24±39.70	**83.55±2.56**
Pendulum-v2	SPARSE3	553.00	950.28±164.89	**1000.00±00.00**	**1000.00±00.00**
DoublePendulum-v2	SPARSE3	1488.28	8605.39±984.68	4191.07±2778.00	**9190.54±432.88**
HalfCheetah-v2	SPARSE2	2109.80	2696.95±334.77	3508.84±259.71	**3541.23±135.72**
Hopper-v2	SPARSE2	969.71	637.96±120.09	32.71±12.75	**1897.69±903.92**
Walker2d-v2	SPARSE2	1843.75	2552.48±567.80	-6.83±6.32	**3533.36±501.68**
Ant-v2	SPARSE2	1942.05	-5477.28±2441.13	-70.04±28.37	**2602.55±230.17**

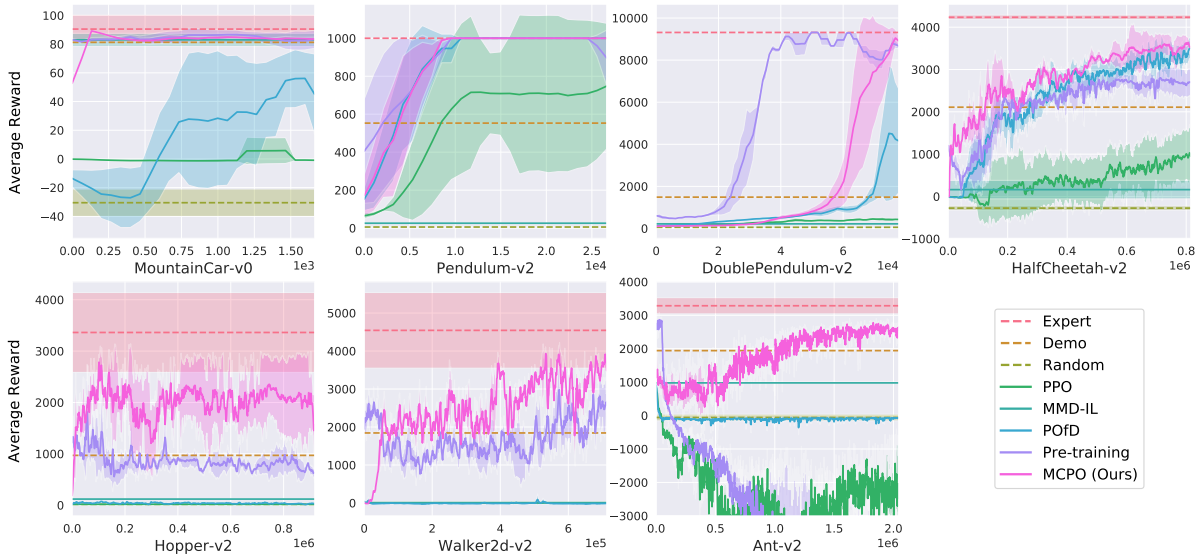

Figure 2: Learning curves of our method versus baselines under challenging robotic control benchmark. For each experiment, **a step represents one interaction with the environment**.

E.1 Settings

To simulate the sparse reward conditions using existing control tasks in Gym, we first propose several reward sparsification methods with details as follows:

- **SPARSE1**: Only provide reward $+1$ when the agent reaches a specific terminal state, otherwise no reward will be provided.
- **SPARSE2**: A reward of $+1$ will be provided when the agent has already moved towards a certain direction for some distance.
- **SPARSE3**: When the last pole is higher than a given height, a reward of $+1$ will be provided. This is only for the *Pendulum* and *DoublePendulum* tasks.

For the demonstrations, we train expert policies for each tested tasks with PPO [Schulman *et al.*, 2017] based on the exact reward (shown as **Expert**), and select a meanwhile learned policy, record only a single trajectory as the imperfect demonstrations (shown as **Demo**). Finally, the baselines we carry out include:

Pre-training [Silver *et al.*, 2016]: This method first performs policy pre-training with provided demonstrations. We use behavior cloning [Schaal, 1997] as the pre-training strategy for 50 iterations with one imperfect trajectory demonstrations. Then PPO [Schulman *et al.*, 2017] with sparse reward is adopted for the rest of policy learning.

POfD [Kang *et al.*, 2018]: In POfD, the training of GAN is paralleled with the policy optimization, which is different from the pre-training baseline. We adopt the same neural structure to the discriminator as the original POfD. As the action space of all the considered tasks is continuous, POfD with discrete policy does not present in our implementation.

PPO & MMD Imitation (MMD-IL): Although PPO and MMD Imitation does not belong to the solutions to RLfD, here we still implement them to verify the sparsification of reward and the quality of demonstrations. Specifically, the PPO baseline will run with sparse reward, while MMD Imitation is directly optimizing MMD between agent exploration rollouts and expert demonstrations with the imperfect demonstrations.

For a fair comparison, the policies of all the methods and tasks are parameterized by the same neural network architecture with two hidden layers (300 and 400 units each) and Tanh activation functions. For the sake of efficiency, all the algorithms are evaluated within the fixed amount of environment steps for each task. Due to the space limitation, we defer more experimental details to the supplementary material.

(a) (b)

Figure 3: Learning curves over 5 trails on *HalfCheetah* task. (a): ablation study about the different tolerance factor d, (b): sensitivity of choosing fixed or annealing strategy of tolerance.

E.2 Comparative Experiment

In comparative evaluations, we conduct challenging control tasks ranging from low state and action dimension (*MountainCar*) to complex ones (*Pendulum*, *DoublePendulum*, *HalfCheetah*, *Hopper* and *Walker2d*) with one single imperfect demonstrations (see the **Demo** curve). The corresponding learning curve are demonstrated in Figure 2, while the averaged cumulative rewards are recorded in Table 3. For each single task, we run each algorithm over five times with different random seeds and the solid curves in Figure 2 correspond to the mean reward and the shaded region represents the variance over the five trials. The numerical results in Table 3 are averaged with 50 trials under the best policy over the five times obtained at the end of training.

The results overall read that our method achieves comparable performances with the baselines on relatively simple tasks (such as *MountainCar*) and outperforms them on difficult tasks. In particular, during policy optimization, our method can converge faster than other RLfD counterparts as well as obtains better final performances. And by comparing with the strong baseline of pre-training, we can see that although convergence efficiency of proposed MCPO during the early phase of training may not have significant advantages, but as it continues, the performance of MCPO can be improved persistently like *DoublePendulum*(+585.15) and *HalfCheetah*(+844.28), while pre-training struggles on achieving higher return, which demonstrates that MCPO could benefit more from the exploration guidance by mimicry constraint during the whole policy optimization procedure than by only imitating at the beginning.

On the other hand, we also find that our algorithm exhibits a more stable and robust behavior. By comparing with POfD, since MCPO does not rely on either complex training strategies or auxiliary model, it can be more stable to different tasks and environment specifications. Moreover, leveraging the demonstrations as a mimicry constraint can also help for a more robust optimization procedure than the penalty mechanism in POfD, especially in difficult *Hopper*(+1864.98), *Walker2d*(+3540.19) and *Ant*(+2672.59) tasks.

From the results of PPO and MMD-Imitation, the experiment settings of reward sparsification and imperfect demonstrations can be verified. As it illustrates, under sparse environmental feedback, pure PPO is failed to find an optimal policy on most of the tested tasks, which also indicates the importance of exploration. Furthermore, with few imperfect demonstrations, MMD-Imitation also cannot learn a satisfying policy; thus imitation only accounts for a small part of the advantages of MCPO, and the exploration improvement via mimicry constraint is the major one.

E.3 Ablation Experiment

The results presented in the previous section suggest that our proposed method can outperform other RLfD approaches on several challenging tasks. Now we will further investigate the impact of the core mimicry constraint in our method. More specifically, we are interested in the tolerance factor of d. We will compare the performance of MCPO with different fixed d, and analyze the sensibility of choice different tolerance strategy (fixed or annealing).

Different tolerance. We design four groups of parameters for the ablation experiments on the tolerance choosing in *HalfCheetah-v2* task, where the annealing mechanism is disabled by setting ϵ fixed at zero, and choose d from $\{10^0, 10^{-1}, 10^{-3}, 10^{-6}\}$. The learning curves are plotted in Figure 3(a). As the results demonstrate, when given relatively large tolerance, the exploration reference from demonstrations will not work as the constraint almost does not affect policy optimization. In contrast, a too small tolerance will hurt the final performance when the demonstrations are imperfect. Therefore, hand-crafting the tolerance for mimicry constraint can be difficult, and an automatic adjustment with the annealing mechanism should be adopted.

Fixed vs. Annealing tolerance. In the previous experiment, we mention the importance of annealing of tolerance. Now we explore the advantages of annealing mechanism quantitatively In *HalfCheetah-v2* task, we choose a fixed $d = 10^{-3}$ and select the annealing factor ϵ from $\{0, 2 \times 10^{-3}, 10^{-3}, 10^{-6}\}$. Corresponding learning curves are shown in Figure 3(b). We can see that the performances of MCPO with an annealing tolerance are overall better than with a fixed one (simply by setting ϵ as zero). Moreover, when the annealing factor ϵ is set properly, the performance of MCPO is not sensitive to the minor changes of ϵ as the results of different factors are almost at the same level. This further demonstrates the robustness of our proposed method.

E.4 Additional Results for Comparative and Ablation Experiments

Here we provide additional results of expert (ideal), random policy, PPO [Schulman *et al.*, 2017] and MMD-IL on the benchmark environments in Table 4. Table 5 lists the numerical results of ablation experiments on *HalfCheetah-v2* task. All results are tested with 50 runs under the best policy over the five times.

Table 4: Additional comparison results on expert, random policy, PPO and MMD-IL

Environment	Expert	Random	PPO	MMD-IL
MountainCar-v0	90.49±9.87	-30.50±9.61	-0.74±0.40	82.99±4.57
Pendulum-v2	1000.00±0.00	5.61±3.28	727.96±360.14	25.71±1.03
DoublePendulum-v2	9314.57±0.41	54.61±16.84	456.97±111.45	218.43±13.72
HalfCheetah-v2	4234.40±55.13	-274.96±55.86	978.84±665.61	161.74±219.85
Hopper-v2	3362.54±778.92	15.83±11.10	17.09±13.54	118.66±0.38
Walker2d-v2	4543.67±997.34	1.49±5.60	1.54±5.75	8.88±6.07
Ant-v2	3284.22±243.78	-58.46±102.09	-2332.95±2193.85	967.83±0.87

Table 5: Ablation results on HalfCheetah-v2 task.

$d = 10^{0}, \epsilon = 0$	-586.52 ± 987.00
$d = 10^{-1}, \epsilon = 0$	-53.96 ± 712.29
$d = 10^{-3}, \epsilon = 0$	2947.76 ± 453.19
$d = 10^{-6}, \epsilon = 0$	-11.30 ± 0.82
$d = 10^{-3}, \epsilon = 2 \times 10^{-3}$	2943.08 ± 99.60
$d = 10^{-3}, \epsilon = 10^{-3}$	3286.78 ± 376.80
$d = 10^{-3}, \epsilon = 10^{-6}$	3471.43 ± 208.78

E.5 Constraint Value in Policy Optimization

Here we provide the value of mimicry constraint during the policy optimization of all the two ablation experiments in Figure 4. As we can see, MCPO can almost guarantee the satisfactory of constraint during the policy optimization as we use exact dual solving. Moreover, when the mimicry constraint tolerance d is relatively small, the recovery mechanism will dominate the optimization procedure at the beginning, and the constraint value will decrease very quickly, as is shown in Figure 4(b).

In the rightmost plot, the purple curve increases at last as the training continues due to the relatively large tolerance annealing factor ϵ, which demonstrates the effect of annealing mechanism in MCPO. We can also see that the final performances do not have a strong correlation to the absolute value of mimicry constraint especially during the late phase of policy optimization, as the agent has already been well-guided on the exploration.

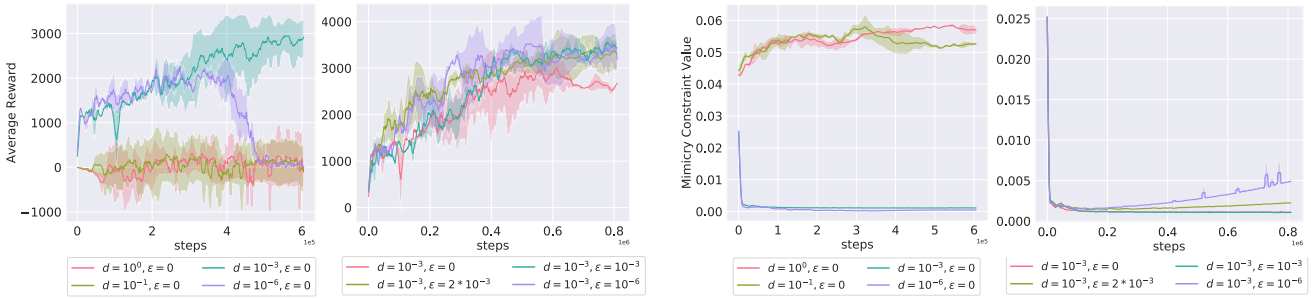

Figure 4: Learning curves over 5 trails on *HalfCheetah* task. (a): the averaged reward in two ablation experiments, (b): the corresponding mimicry constraint value during the training in two ablation experiments.

References

[Borgwardt *et al.*, 2006] Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 2006.

[Brockman *et al.*, 2016] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[Duan *et al.*, 2016] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning (ICML)*, 2016.

[Gretton *et al.*, 2012] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research (JMLR)*, 2012.

[Kakade, 2002] Sham M Kakade. A natural policy gradient. In *Advances in neural information processing systems (NIPS)*, 2002.

[Kang *et al.*, 2018] Bingyi Kang, Zequn Jie, and Jiashi Feng. Policy optimization with demonstrations. In *International Conference on Machine Learning (ICML)*, 2018.

[Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International conference on Learning Representation (ICLR)*, 2015.

[Müller, 1997] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 1997.

[Puterman, 1994] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[Schaal, 1997] Stefan Schaal. Learning from demonstration. In *Advances in neural information processing systems (NIPS)*, 1997.

[Schulman *et al.*, 2015] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, 2015.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[Sriperumbudur *et al.*, 2008] Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Gert Lanckriet, and Bernhard Schölkopf. Injective hilbert space embeddings of probability measures. In *Annual Conference on Learning Theory (COLT)*, 2008.

[Steinwart, 2001] Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of machine learning research (JMLR)*, 2001.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT Press, 1998.

[Syed *et al.*, 2008] Umar Syed, Michael Bowling, and Robert E Schapire. Apprenticeship learning using linear programming. In *International Conference on Machine Learning (ICML)*, 2008.