Robust Reinforcement Learning for Shifting Dynamics During Deployment

Samuel Stanton
New York University

Rasool Fakoor
Amazon Web Services

Jonas Mueller
Amazon Web Services

Andrew Gordon Wilson
New York University

Alex Smola
Amazon Web Services

Abstract

While high-return policies can be learned on a wide range of systems through reinforcement learning, actual deployment of the resulting policies is often hindered by their sensitivity to future changes in the environment. Adversarial training has shown some promise in producing policies that retain better performance under environment shifts, but existing approaches only consider robustness to specific kinds of perturbations that must be specified a priori. As possible changes in future dynamics are typically unknown in practice, we instead seek a policy that is robust to a variety of realistic changes only encountered at test-time. Towards this goal, we propose a new adversarial variant of soft actor-critic, which produces policies on Mujoco continuous control tasks that are simultaneously more robust across various environment shifts, such as changes to friction and body mass.

1 Introduction

Control policies learned through reinforcement learning tend to be quite brittle. If the policy is deployed in a slightly altered version of the training environment, its performance significantly decreases (Fig. 1). Prior work has focused on defending against either a specific threat model, or against shifts in the agent's perception or actions (and not the underlying system dynamics) [7, 15, 18, 24]. Here evaluation is typically done under the same form of adversarial shifts that the agent faced during training. However, most shifts encountered in real deployments cannot be accurately specified during training. To develop more practical applicable methods, we seek to increase the robustness of an RL policy to a range of realistic shifts in the environment dynamics, all without requiring that the policy parameters can adapt during deployment or that these shifts can be simulated during training. We particularly study shifts that are likely to naturally occur rather than being specially designed to harm the agent.

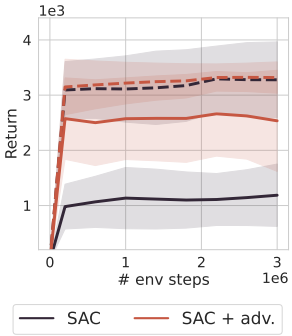
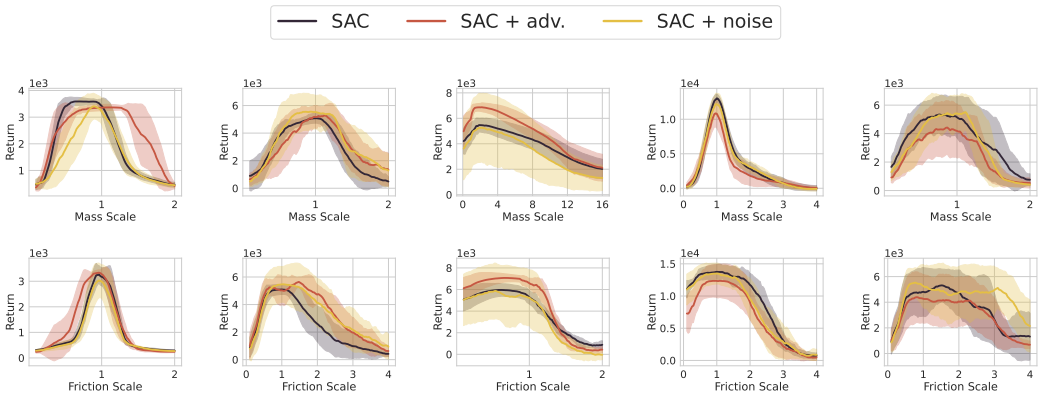


Figure 1: High return on the base task (Hopper, shown as dashed lines) does not guarantee high return under deployment shifts (reduced friction, solid lines). The shaded region denotes $\mu \pm \sigma$, estimated from 10 seeds.

This work makes the following contributions:

1. We propose an adversarial variant of soft actor-critic (SAC) [8] that improves robustness to realistic shifts in the environment encountered only during deployment.
2. We empirically study various ways to parameterize and constrain the adversary.
3. We discover that linear parametric adversaries can outperform gradient-based white-box adversaries. This contrasts with popular adversarial training methods for supervised learning.

The 2021 NeurIPS Workshop on Safe and Robust Control of Uncertain Systems



(a) Hopper (b) Walker2d (c) Ant (d) HalfCheetah (e) Humanoid

Figure 2: Robustness evaluation of SAC agents. **Top row:** agent return when the body masses from the base task are scaled by some constant. **Bottom row:** agent return when the friction between the agent and the environment is scaled by some constant. Depicted are returns of the learned policy during deployment in the changed environment, with distance from 1 along horizontal axis corresponding to larger shifts. The shaded region denotes $\mu \pm \sigma$, estimated from 10 seeds. Adversarial training can significantly expand the range of environments where the agent can be expected to perform well, at little cost to performance in the base task.

2 Related Work

Ensembling and domain randomization: one related approach to increasing agent robustness attempts to train the agent to be optimal in a broader range of environments by either maintaining an explicit set of hypotheses for the environment dynamics in the form of a model ensemble [16, 23], or with expertly-selected data augmentation procedures [14, 19]. Both approaches rely on task-specific knowledge of the environment parameters, which may not always be available.

Smoothing: the use of Gaussian noise to smooth the critic or the policy has been proposed in several works, including Pattanaik et al. [13], Kumar et al. [10], and Yarats et al. [22].

Adversarial RL: Originally proposed by Morimoto and Doya [12], adversarial reinforcement learning has been explored in a number of recent works [7, 13, 15, 17]. Our work is closely related to Zhang et al. [24], who also propose a defense against adversarial attacks on the agent observation model. We consider attacks to both the observation model and the transition dynamics, and in our empirical evaluation we focus on robustness to realistic (naturally-occuring) shifts in the environment, rather than robustness to a specific malicious actor. While much existing work on robust/adversarial ML/RL assumes the threat model can be correctly specified a priori (i.e. possible forms of environment shift are known during training), this is not the case for most real-world applications [6].

3 Background

We consider infinite-horizon, discrete-time Markov decision processes (MDPs) $(\mathcal{S}, \mathcal{A}, \rho, f, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\rho(\mathbf{s})$ the initial state distribution, f the dynamics function, $r(\mathbf{s}, \mathbf{a})$ the reward function, and γ the discount factor. A transition distribution $p(\mathbf{s}'|\mathbf{s}, \mathbf{a}) = f(\mathbf{s}, \mathbf{a}) + \omega(\mathbf{s}, \mathbf{a})$, is induced by f and a noise process ω (e.g. isotropic Gaussian noise). The expected return of a stochastic policy $\pi(\mathbf{a}|\mathbf{s})$ from a state-action pair $(\mathbf{s}, \mathbf{a})$ is defined as

$$Q^{\pi}(\mathbf{s}, \mathbf{a}) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \Big| \mathbf{s}_0 = \mathbf{s}, \mathbf{a}_0 = \mathbf{a}\right]. \tag{1}$$

We focus on continuous state-action domains, using SAC for policy optimization. Q_θ^π and $Q_{\theta'}^\tau$ denote the critic and target networks used to estimate returns.

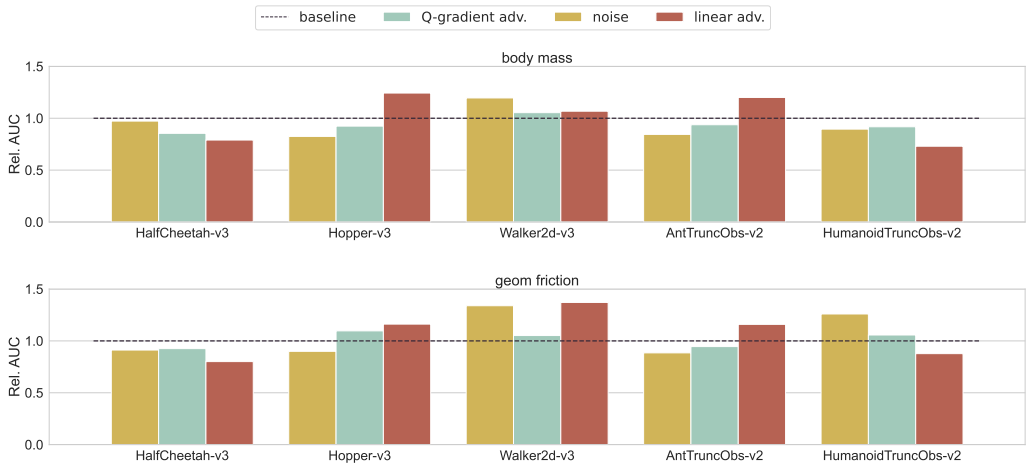


Figure 3: Ablating different types of adversarial defenses. Simple linear adversaries offer the most consistent improvement across tasks and shifts, outperforming both white-box gradient attacks on the critic and zero-mean spherical Gaussian noise (i.e. smoothing).

Dynamics shift: allows an adversary (or nature) to alter the transition distribution $p(\mathbf{s}'|\mathbf{s},\mathbf{a})$ (e.g. a change in surface friction).

Max-min formulation: we can frame robustness to a set of possible shifts as a zero-sum maximin game between the policy and an adversary, δ, generally given by: $\max_{\pi\in\Pi}\min_{\delta\in\Delta} Q^{\pi}$. Here we consider adversaries $\delta : \mathcal{S}\times\mathcal{A}\times\Omega\rightarrow\mathcal{S}$ parameterized by $\omega\in\Omega$ and denoted as $\delta_{\omega}(\mathbf{s},\mathbf{a})$ (for example, a linear transformation). We adopt the following specific formulation:

$$\max_{\pi\in\Pi} \mathop{\mathbb{E}}_{\substack{(\mathbf{s},\mathbf{a})\sim d^{\pi}(\cdot),\\ \mathbf{s}'\sim p(\cdot|\mathbf{s},\mathbf{a})}} \min_{||\delta(\mathbf{s},\mathbf{a})||_p<\varepsilon} \mathop{\mathbb{E}}_{\mathbf{a}'\sim\pi(\cdot|\mathbf{s}'+\delta(\mathbf{s},\mathbf{a}))} \Big[Q^{\pi}(\mathbf{s}'+\delta(\mathbf{s},\mathbf{a}),\mathbf{a}')\Big], \tag{2}$$

where $d^{\pi}(\mathbf{s},\mathbf{a})$ is the stationary distribution over state-action pairs induced by π. As an unconstrained adversary could hinder learning anything at all, we restrict the output of the adversary to an ℓ_p-norm ball of radius ε. A benefit of the minimax formulation is that the agent does not require all possible threats and shifts to be explicitly enumerated. If the minimax problem is solved to optimality, then the worst-case performance of π is guaranteed under all possible shifts that can be realized by δ.

4 Adversarial Soft Actor Critic

In this section we discuss the implementation and evaluation of adversarial SAC [8] agents. Our method is built on top of SAC and is shown as follows:

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{r},\mathbf{s}')\sim\mathcal{D}}\left[\Big(Q_\theta(\mathbf{s},\mathbf{a}) - r - \gamma\,\mathbb{E}_{\mathbf{a}'\sim\pi_\phi(\cdot|\mathbf{s}'+\delta_\omega(\mathbf{s},\mathbf{a}))}\Big[Q_{\hat{\theta}}(\mathbf{s}'+\delta_\omega(\mathbf{s},\mathbf{a}),\mathbf{a}')\Big]\Big)^2\right] \tag{3}$$

$$\mathcal{L}_\pi(\phi) = \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi_\phi(\cdot|\mathbf{s})}\Big[-Q_\theta(\mathbf{s},\mathbf{a}) + \alpha\log\pi_\phi(\mathbf{a}|\mathbf{s})\Big] \tag{4}$$

$$\mathcal{L}_\delta(\omega) = \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D},\mathbf{a}'\sim\pi_\phi(\cdot|\mathbf{s}'+\delta_\omega(\mathbf{s},\mathbf{a}))}\Big[Q_\theta(\mathbf{s}'+\delta_\omega(\mathbf{s},\mathbf{a}),\mathbf{a}')\Big] \tag{5}$$

Since obtaining unbiased estimates of Q^{π} is inefficient we instead rely on bootstrapped estimates Q^{π}_{θ} using networks parameterized by $\theta\in\Theta$. To avoid cycling between the policy and the adversary, we used a small population of adversaries ($n=2$), and randomly sampled an adversary from the population at the beginning of each training episode, a modification inspired by Vinitsky et al. [21]. We found that the simple combination of linear adversaries with an ℓ_2-norm constraint worked well for the tasks we considered. We restrict δ_ω to the ε-ball by projecting unconstrained adversarial perturbations back to the feasible region. For all tasks we set $\varepsilon=\exp(-5)$, see Appendix B for an ablation. We find that the effect of adversarial training on sample efficiency is minimal when ε is properly tuned (appendix Fig. S2).

4.1 Does adversarial training improve robustness?

Baseline agents: we compare our adversarially trained agents against two baselines. First we compare against standard SAC [8] agents with no modifications. We also assess the effectiveness of our adversarial δ by replacing it with zero-mean spherical Gaussian noise. We expend the same amount of compute to independently tune the hyperparameters of all agents considered in this work, and we require that the same hyperparameters be used for all tasks (see Appendix D for details).

Base tasks: we evaluate our agents on the standard benchmark suite of OpenAI Gym Mujoco continuous control tasks [3, 20]. See Appendix E for details.

Evaluation shifts: we consider changes to the friction coefficients between the agent and the environment, and changes to the mass of the agent body. The change in friction corresponds to a scenario where contacts are gradually worn smooth, and the change in mass corresponds to a scenario where the force of gravity changes (e.g. when deployed extra-terrestrially or in marine environments).

Fig. 2 shows the expected return of the final policy for each method as a function of the environment shift parameters. We find that our adversarial training procedure improves robustness on Hopper, Walker2d, and Ant for all shifts considered, in the sense that the policy realizes higher return over a broader range of evaluation tasks than the baseline. Adversarial training also appears to lower the variance in return achieved on the *base task* (i.e. when there is no shift during test-time). These improvements come at essentially no cost to sample efficiency for learning the base task (see Appendix B, Fig. S2). The results on HalfCheetah and Humanoid are less satisfactory. The challenge lies in finding a choice of ε that works well for all tasks and shifts. One solution is to select the largest ε that does not lead to a nontrivial drop in base task performance (which is observable during training), in which case our adversarial SAC formulation simply produces a similar policy as standard SAC for HalfCheetah and Humanoid, while offering significantly greater robustness in the other environments.

4.2 Are parametric adversaries necessary?

Although it is common in adversarial RL literature to parameterize the adversary like a policy [4, 7, 15], in supervised learning "white-box" gradient-based attacks are commonly used [11]. Indeed, if we reexamine Eq. (2), we see that under an ℓ_2 constraint, as $\varepsilon \to 0$,

$$\delta^*(\mathbf{s}, \mathbf{a}) = -\nabla_{\mathbf{s}} Q^\pi|_{\mathbf{s},\mathbf{a}}.$$

In Fig. 3 we compare the result of using adversarial shifts along $-\nabla_{\mathbf{s}} Q_\theta^\pi$ compared to the result of our linear adversaries. We again consider Gaussian shifts as an alternative baseline choice of δ. For ease of comparison, we adopt the *area under the robustness curve* (like those visualized in Fig. 2) as the performance metric. We normalize with the AUC of an unmodified SAC agent, so values greater than 1 indicate improved robustness. See Appendix A for a more detailed discussion of the metric. Surprisingly, linear adversaries perform the best across the full range of tasks and shifts. We conjecture that since Q_θ^π is a very biased estimate of Q^π, the gradient may indicate directions where the Q estimates are unreliable, rather than directions that are truly detrimental to the policy. This finding highlights the way in which adversarial attacks are fundamentally more difficult to construct when unbiased estimates of the gradient of the actual objective are not available.

5 Discussion

In this work we have identified several realistic, benign environment shifts that policies must be able to accommodate before RL agents can be deployed in the wild. It is not practical to enumerate every possible shift and construct a specific, corresponding defense, so we have focused on how to construct a general adversary that improves robustness to a range of potential shifts. Our initial results suggest that it is indeed possible to make policies more robust with little penalty to sample efficiency or final return in the base task. There are a couple directions for future work that could further improve our results. Making Q_θ^π more difficult for the adversary to exploit is one direction, and could be realized by introducing regularization terms as in Zhang et al. [24], or by using an ensemble of critic networks to estimate the Q-values. Another interesting direction would be a more dynamic constraint on the adversary than a simple ℓ_2 ball [2, 5].

References

[1] Amos, B., Stanton, S., Yarats, D., and Wilson, A. G. (2021). On the model-based stochastic value gradient for continuous reinforcement learning. In *Learning for Dynamics and Control*, pages 6–20. PMLR.

[2] Balaji, Y., Goldstein, T., and Hoffman, J. (2019). Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*.

[3] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

[4] Curi, S., Bogunovic, I., and Krause, A. (2021). Combining pessimism with optimism for robust and efficient model-based deep reinforcement learning. *arXiv preprint arXiv:2103.10369*.

[5] Erdemir, E., Bickford, J., Melis, L., and Aydore, S. (2021). Adversarial robustness with non-uniform perturbations. *arXiv preprint arXiv:2102.12002*.

[6] Gilmer, J., Adams, R. P., Goodfellow, I., Andersen, D., and Dahl, G. E. (2018). Motivating the rules of the game for adversarial example research. *arXiv preprint arXiv:1807.06732*.

[7] Gleave, A., Dennis, M., Wild, C., Kant, N., Levine, S., and Russell, S. (2019). Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*.

[8] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv:1801.01290*.

[9] Janner, M., Fu, J., Zhang, M., and Levine, S. (2019). When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*.

[10] Kumar, A., Levine, A., and Feizi, S. (2021). Policy smoothing for provably robust reinforcement learning. *arXiv preprint arXiv:2106.11420*.

[11] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

[12] Morimoto, J. and Doya, K. (2005). Robust reinforcement learning. *Neural computation*, 17(2):335–359.

[13] Pattanaik, A., Tang, Z., Liu, S., Bommannan, G., and Chowdhary, G. (2017). Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*.

[14] Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2018). Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE.

[15] Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. PMLR.

[16] Rajeswaran, A., Ghotra, S., Ravindran, B., and Levine, S. (2016). Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*.

[17] Ren, Y., Duan, J., Li, S. E., Guan, Y., and Sun, Q. (2020). Improving generalization of reinforcement learning with minimax distributional soft actor-critic. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

[18] Tessler, C., Efroni, Y., and Mannor, S. (2019). Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. PMLR.

[19] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE.

[20] Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE.

[21] Vinitsky, E., Du, Y., Parvate, K., Jang, K., Abbeel, P., and Bayen, A. (2020). Robust reinforcement learning using adversarial populations. *arXiv preprint arXiv:2008.01825*.

[22] Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. (2021). Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*.

[23] Yu, W., Tan, J., Liu, C. K., and Turk, G. (2017). Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*.

[24] Zhang, H., Chen, H., Boning, D., and Hsieh, C.-J. (2021). Robust reinforcement learning on state observations with learned optimal adversary. *arXiv preprint arXiv:2101.08452*.

Appendix

A Metrics

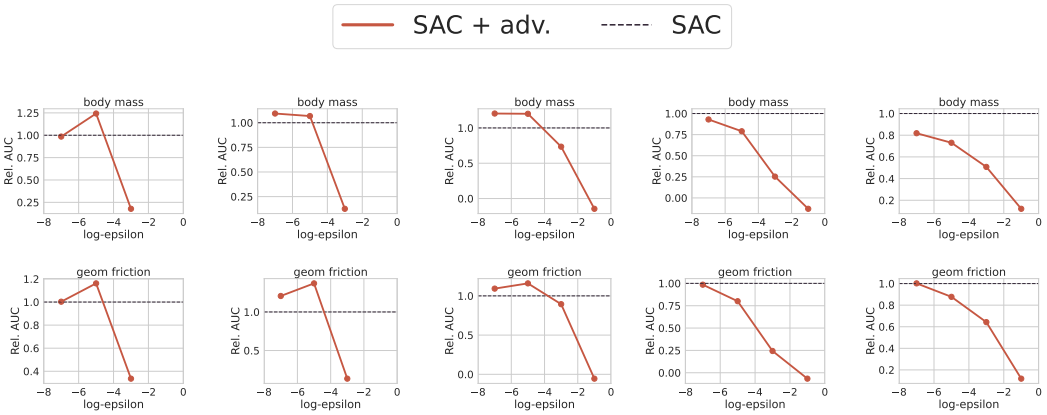
Area under the robustness curve: one challenge when comparing methods across a range of deployment environments is reducing their performance to a single number that is easy to compare. In this work we consider environments that are parameterized by continuous variables such as mass and friction. As a result, we can think of the expected return of a policy π as a function of the environment parameters $G^\pi(\mathbf{z})$. Given some measure $\mu(\mathbf{z})$, we can consider the hypervolume

$$h_\pi(Z) := \int_{z \in Z} G^\pi(\mathbf{z}) \mu(\mathbf{z}). \tag{6}$$

When Z is univariate and μ is the Lebesgue measure, $h_\pi(Z)$ is simply the area under $G^\pi(\mathbf{z})$ evaluated from $\inf(Z)$ to $\sup(Z)$. In our experiments we only vary one parameter at a time (e.g. mass or friction), so we can easily estimate $h_\pi(Z)$ with a discrete set of sample points using the trapezoid rule.

This hypervolume metric is much more robust to varying choices of Z than, for example, $\inf_{\mathbf{z} \in Z} G^\pi(\mathbf{z})$. If Z is taken to be large, containing environments where all methods fail, all methods will receive the same low score. However, regions of Z where the return is small simply contribute very little volume to the integral, so the value of h_π stops changing after Z is large enough.

B Ablations



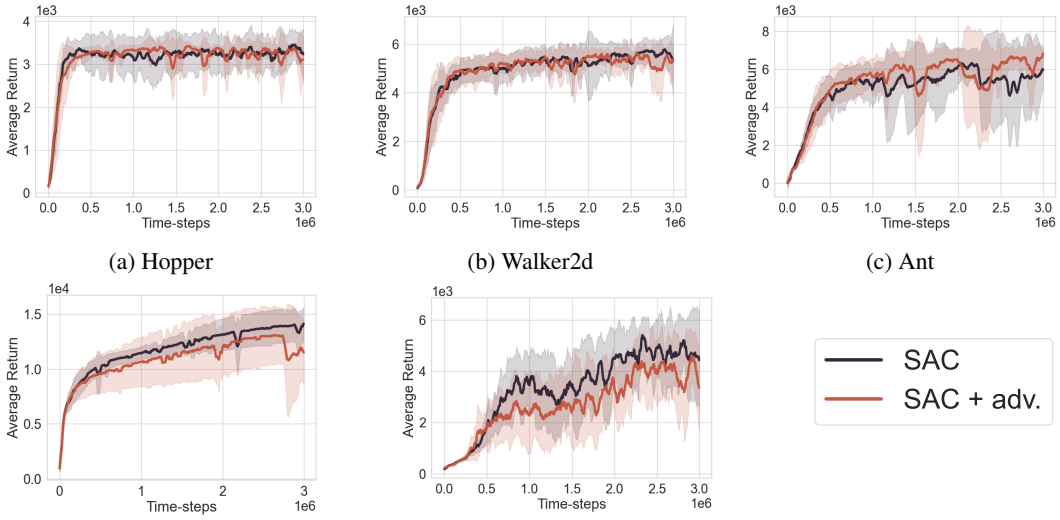
(a) Hopper (b) Walker2d (c) Ant (d) HalfCheetah (e) Humanoid

Figure S1: epsilon ablation. **Top:** ablation results for the body mass dynamics shift. **Bottom:** ablation results for the friction dynamics shift. We use the AUC metric defined in Eq. 6 to quantify robustness. For ease of interpretation, the results are normalized by the AUC of the baseline SAC returns. Taking $\varepsilon = e^{-5}$ works well for a range of tasks and shifts, but HalfCheetah and Humanoid do not show any improvement to robustness.

The choice of ε: In Fig. S1 we show the results of our ablation experiment on ε. Taking $\varepsilon = e^{-5}$ worked well for Hopper, Walker2d, and Ant.

C Does adversarial training decrease sample efficiency?

Fig. 2 demonstrates that adversarial training can improve the robustness of the final policy. Fig. S2 shows that the increase in robustness comes at almost no cost to sample efficiency.



(a) Hopper (b) Walker2d (c) Ant

(d) HalfCheetah (e) Humanoid

Figure S2: Training curves on the base tasks. After X time-steps (for varying X), we evaluate the current policy based on how much average return it generates when deployed in the base environment (with no shift). This reveals the cost of robustness if the environment does not experience any shift at test-time. In some tasks, adversarial training slightly decreases the return of the final policy. With the exception of Half-Cheetah and Humanoid, there is no appreciable change in sample-efficiency. The shaded region denotes $\mu \pm \sigma$, estimated from 10 seeds.

D Agent Details

We use the same standard SAC hyperparameters for all experiments [8]. We did not use the automatic entropy tuning heuristic. The states were normalized with running estimates of the mean and variance during training. We used the same MLP architecture for the policy and critic networks.

Hyper-parameters	value
# Env. steps	3M
# Burn-in steps	10K
Buffer capacity	1M
Learning rate	1e-3
MLP hidden size	256
MLP # layers	2
Batch size	100
Discount factor (γ)	0.99
Entropy coef (α)	0.2
Target network update rate (τ)	5e-3
Random Seeds	$\{0, \ldots, 9\}$

Unless stated otherwise (e.g. in the ablation), the log-radius of the adversary is $\log \varepsilon = -5$. Although we tuned the noise scale independently for the experiment in Fig 3, we found that $\log \sigma = -5$ also seemed to work the best across tasks.

E Task Details

We consider the following standard Gym Mujoco benchmark control tasks [1, 9]:

- HalfCheetah-v3
- Hopper-v3
- Walker2d-v3

- TruncatedAnt-v2: a variant of Ant that truncates the external forces (which are always 0) from the state observation.
- TruncatedHumanoid-v3: a variant of Humanoid that truncates the external forces (which are always 0) from the state observation.

F Dynamics Shifts

We implement our dynamics shifts along the lines of Pinto et al. [15].[1]

- Friction: We modify the `env.model.geom_friction` attribute of the Gym environment by multiplying all values by a scalar $c > 0$.
- Mass: We modify the `env.model.body_mass` attribute of the Gym environment by multiplying all values by a scalar $c > 0$.

[1] https://github.com/lerrel/rllab-adv/blob/13abc58f2d743ccdde12f9635e2dde36c77ca56c/adversarial/scripts/test_friction_robustness.py