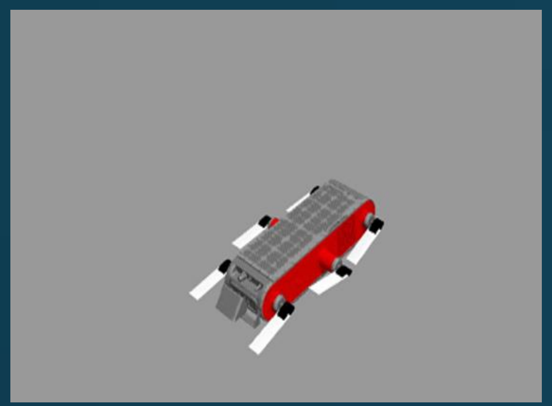
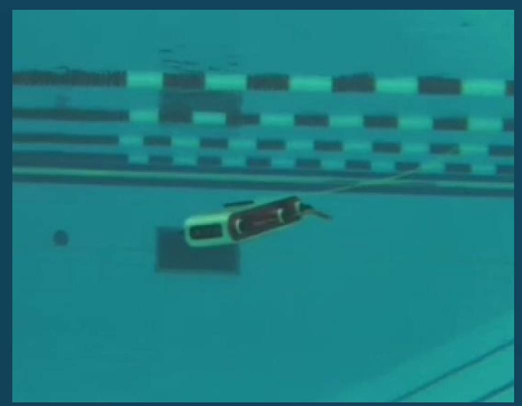
Learning Domain Randomization Distributions for Transfer of Locomotion Policies

Melissa Mozifian[1,2], Juan Camilo Gamboa Higuera[1], David Meger[1,2] and Gregory Dudek[1,2]
[1:] School of Computer Science, McGill University, Montreal, Canada
[2:] Mila – Quebec Artificial Intelligence Institute, Montreal, Canada

Problem

Learning policies with ***low-fidelity*** parametric simulation. Unknown dynamics of target system


The AQUA robot simulator


The AQUA robot

What's the best that we can do with ***little or no data*** from the real system on the desired control task?

Domain Randomization

Let $p(z)$ be the distribution of simulation parameters (context) z. Domain randomization [Tobin et al, 2017] [Peng et al, 2018] aims to find the policy π that maximizes performance over different choices for z

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{p(z)}[J(\pi, z)]$$
$$J(\pi, z) = \mathbb{E}_{p(\tau|\pi,z)}[R(\tau)]$$

where $R(\tau)$ are the cumulative rewards over trajectory τ.

Performance on the target system (e.g. real robot) depends on

- How the policy adapts to different z
- The choice of $p(z)$

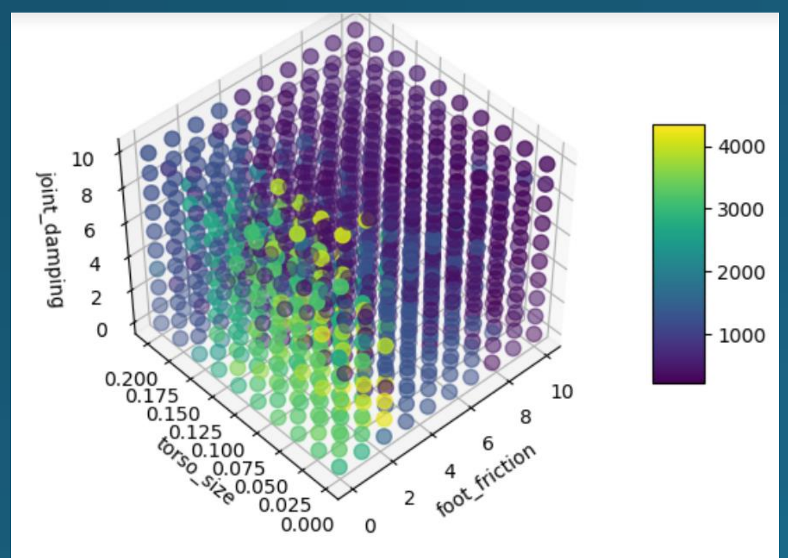
Choosing the training distribution

To ensure task success, we'd like $p(\tau)$ to be as diverse as possible

- The trajectories induced by $p(z)$ should capture some of the target system dynamics
- But our policy models usually have ***finite capacity***

Making $p(z)$ have high variance does not help

- The set of values for z where the task is solvable may be small
- But we don't know it a priori



Performance of best policy for hopper with varying z

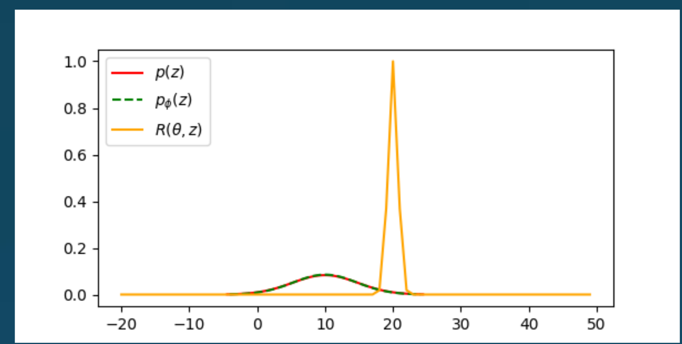
Our approach

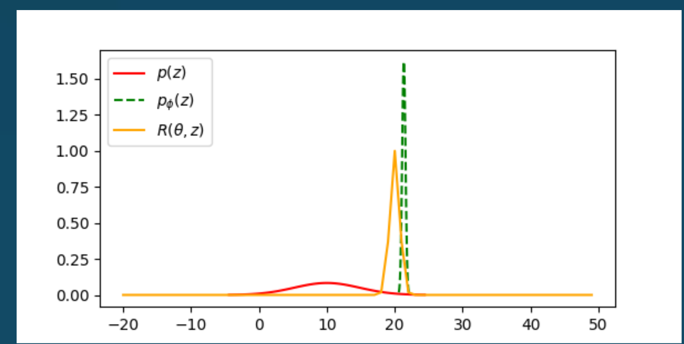
Pick wide prior $p(z)$, where uniform DR may fail.

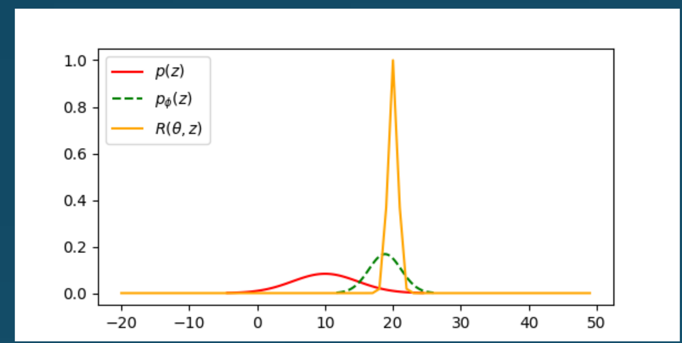
Use parametric training distribution $p_\phi(z)$, train policy conditioned on simulation parameters $\pi_\theta(a|s,z)$. Alternate between:

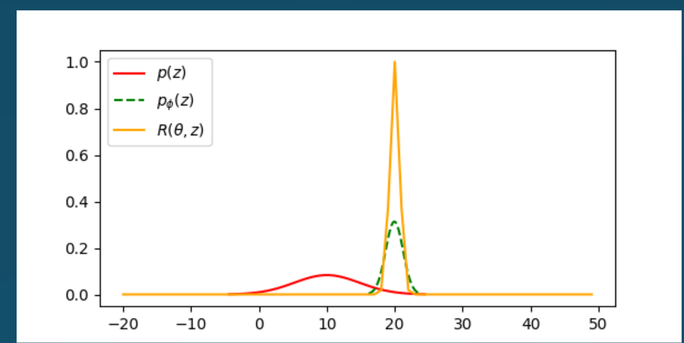
- Optimizing θ over the training distribution $p_\phi(z)$
- Optimize ϕ to focus on environments where the current policy performs well
- Use prior to keep from collapsing to easy environments

$$\arg\max_{\theta} \mathbb{E}_{p_\phi(z)}[J(\pi_\theta, z)]$$
$$\arg\max_{\phi} \mathbb{E}_{p_\phi(z)}[J(\pi_\theta, z)\log p_\phi(z)] - \alpha D_{KL}\left(p(z)||p_\phi(z)\right)$$


Before optimization


After optimization $\alpha = 0$

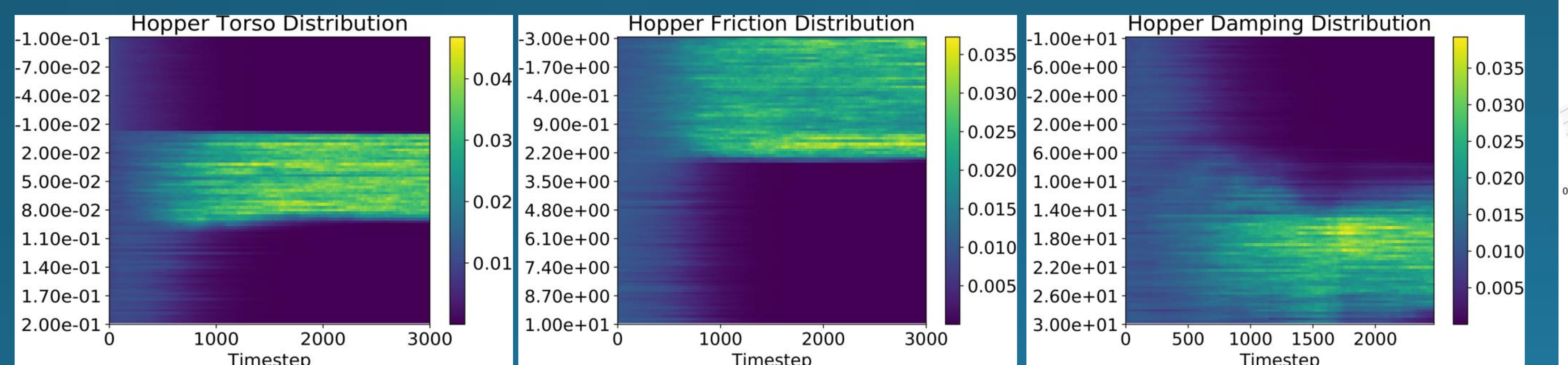

After optimization $\alpha = 10^{-4}$


After optimization $\alpha = 10^{-5}$

Experiments

- Experiments on *Hopper* and *Half-Cheetah* benchmarks [Brockman et al, 2016]
- Modelled $p_\phi(z)$ with discrete distribution, $p(z)$ with uniform prior
- Compared against uniform DR and using EpOpt-PPO [Rajeswaran , 2016] as policy optimizer

Example results on Hopper:

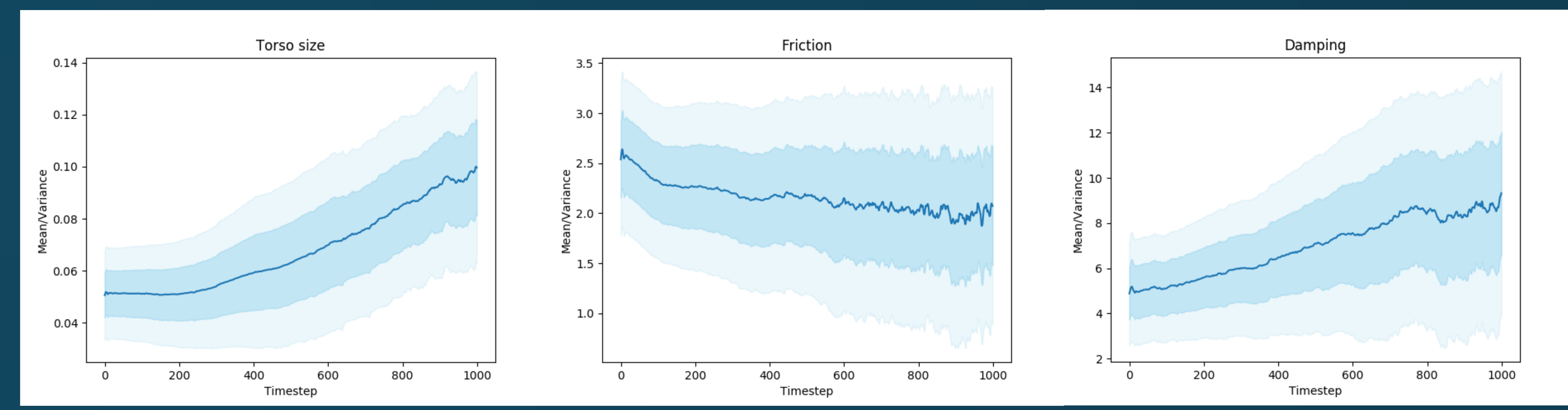


Learned distributions approximate the ranges found with exhaustive grid evaluations

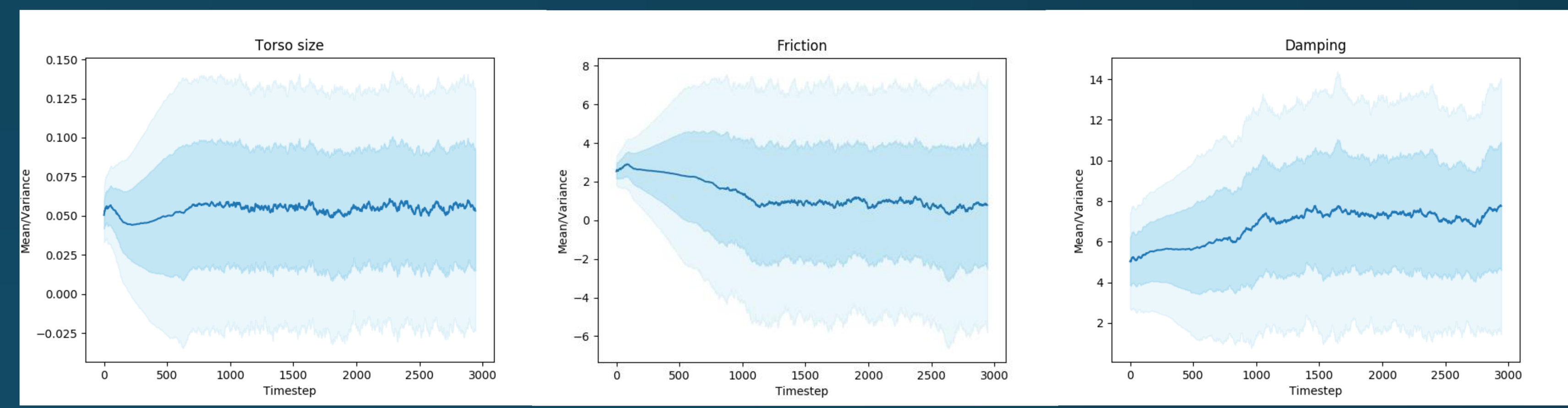
Multi-context Distribution Learning

Using rewards seeking objective:

- Multi-context distributions learned for Hopper and Half-Cheetah

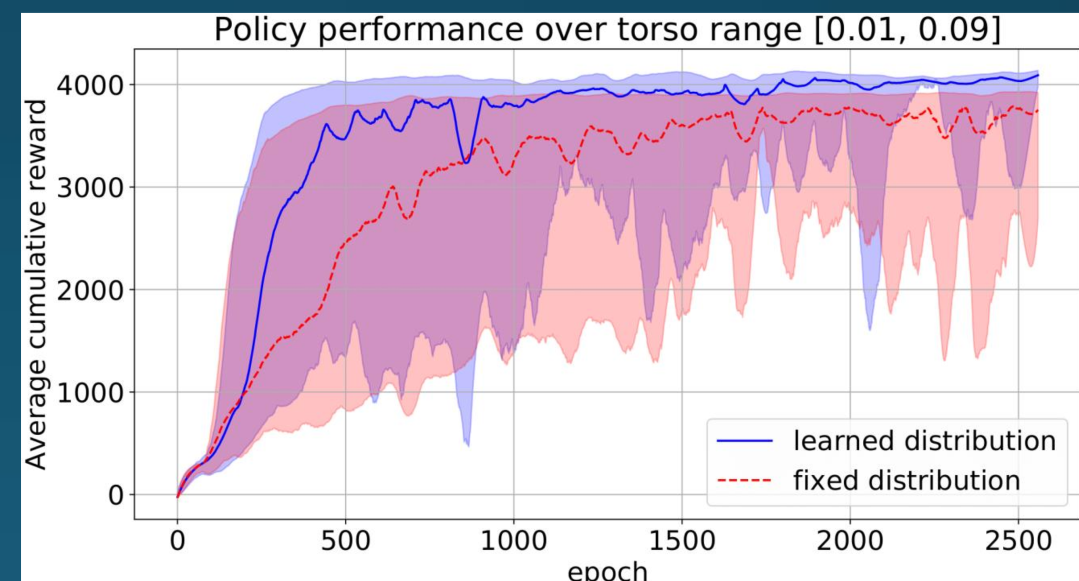


Half-Cheetah Learned Context Distribution



Hopper Learned Context Distribution

Optimizing for worst-case contexts

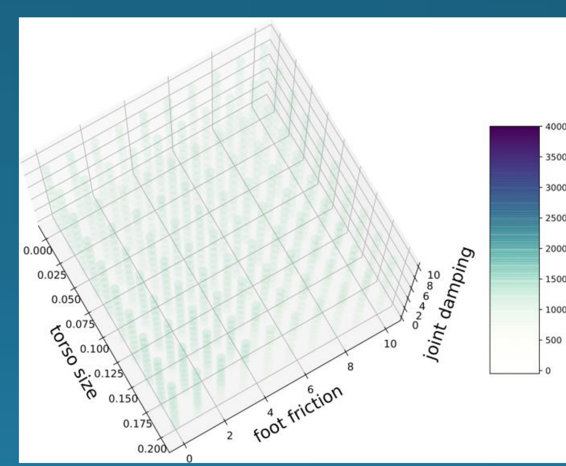


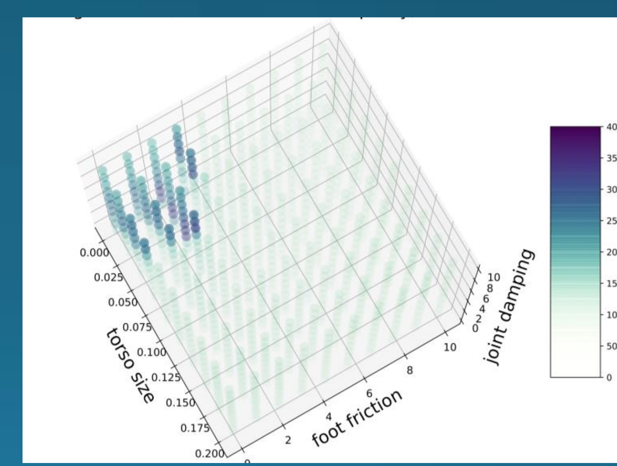
Using the CVaR objective as in EpOpt [Rajeswaran , 2016]

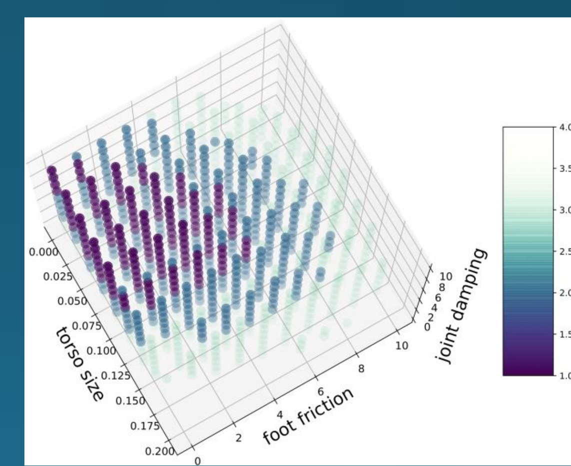
- Learned DR converges faster than uniform DR
- Better asymptotic performance

Validating the learned distribution

- Performance of policies trained on Hopper over the grid with:


Uniform domain randomization


Learning the parameters of a Gaussian DR distribution


Confidence regions of the learned distribution for 1: 68%, 2: 95%, and 3: 99% confidence