Appendix: Behavior-Guided Reinforcement Learning

A. Additional Experiments

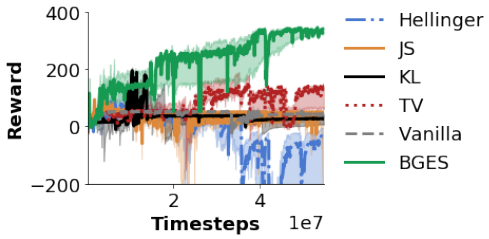


Figure 6: **Escaping Local Maxima** A comparison of BGES with those using different distances on Policy Embeddings.

A.1. Escaping Local Maxima.

In Fig. 6 we compare our methods with methods using regularizers based on other distances or divergences (specifically, Hellinger, Jensen-Shannon (JS), KL and Total Variation (TV) distances), as well as vanilla ES (i.e., with no distance regularizer). Experiments were performed on a **Swimmer** environment from OpenAI Gym (Brockman et al., 2016), where the number of samples of the ES optimizer was drastically reduced. BGES is the only one that manages to obtain good policies which also proves that the benefits come here not just from introducing the regularizer, but from its particular form.

A.2. Imitation Learning

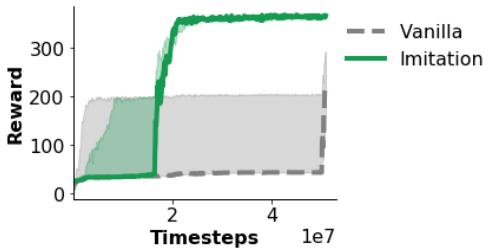


Figure 7: **Imitation Learning.**

As discussed in Section 5.4, we can also utilize the BGES algorithm for imitation learning, by setting $\beta < 0$, and using an expert's trajectories for the Policy Embedding. For this experiment we use the reward-to-go BEM (Section 5). In Fig. 7, we show that this approach significantly outperforms vanilla ES on the Swimmer task. Although conceptually simple, we believe this could be a powerful approach with potential extensions, for example in designing safer algorithms.

B. Further Experimental Details

B.1. BGPG

Here we reproduce a full version of Algorithm 3:

Algorithm 5 Behvaior-Guided Policy Gradient with **On-Policy** Embeddings

Input: Initialize stochastic policy π_0 parametrized by θ_0, $\beta < 0, \eta > 0$, $M, L \in \mathbb{N}$
for $t = 1, \ldots, T$ **do**

1. Run π_{t-1} in the environment to get advantage values $A^{\pi_{t-1}}(s,a)$ and trajectories $\{\tau_i^{(t)}\}_{i=1}^M$
2. Update policy and test functions via several alternating gradient steps over the objective:

$$F(\theta) = \mathop{\mathbb{E}}_{\tau_1,\tau_2 \sim \mathbb{P}_{\pi_{t-1}} \bigotimes \mathbb{P}_{\pi_\theta}} \Big[\sum_{i=1}^{H} A^{\pi_{t-1}}(s_i,a_i) \frac{\pi_\theta(a_i|s_i)}{\pi_{t-1}(a_i|s_i)} + \beta\lambda_1(\Phi(\tau_1)) - \beta\lambda_2(\Phi(\tau_2)) + \beta\gamma \exp\left(\frac{\lambda_1(\Phi(\tau_1)) - \lambda_2(\Phi(\tau_2)) - C(\Phi(\tau_1)), \Phi(\tau_2))}{\gamma} \right) \Big]$$

Where $\tau_1 = s_0, a_0, r_0, \cdots, s_H, a_H, r_H$. Let $\theta_{t-1}^{(0)} = \theta_{t-1}$.
for $\ell = 1, \cdots, L$ **do**

a. Approximate $\mathbb{P}_{\pi_{t-1}} \bigotimes \mathbb{P}_{\pi_\theta}$ via $\frac{1}{M}\{\tau_i^{(t)}\}_{i=1}^M \bigotimes \frac{1}{M}\{\tau_i^{\theta}\}_{i=1}^M := \hat{P}_{\pi_t,\pi_\theta}$ where $\tau_i^\theta \overset{\text{i.i.d}}{\sim} \mathbb{P}_{\pi_\theta}$
b. Take SGA step $\theta_{t-1}^{(\ell)} = \theta_{t-1}^{(\ell-1)} + \eta\hat{\nabla}_\theta \hat{F}(\theta_{t-1}^{(\ell-1)})$ using samples from $\hat{P}_{\pi_{t-1},\pi_\theta}$.
c. Use samples from $\hat{P}_{\pi_{t-1},\pi_\theta}$ and Algorithm 1 to update λ_1, λ_2.

Set $\theta_t = \theta_{t-1}^{(M)}$.

Algorithm 6 Behvaior-Guided Policy Gradient with **Off-Policy** Embeddings

Input: Initialize stochastic policy π_0 parametrized by θ_0, $\beta < 0, \eta > 0$, $M, L \in \mathbb{N}$, state probing distribution $\mathbb{P}_{\mathcal{S}_0}$.
for $t = 1, \ldots, T$ **do**

1. Run π_{t-1} in the environment to get advantage values $A^{\pi_{t-1}}(s,a)$. 2. Update policy and test functions via several alternating gradient steps over the objective:

$$F(\theta) = \mathop{\mathbb{E}}_{\tau_1 \sim \mathbb{P}_{\pi_{t-1}}} \Big[\sum_{i=1}^{H} A^{\pi_{t-1}}(s_i,a_i) \frac{\pi_\theta(a_i|s_i)}{\pi_{t-1}(a_i|s_i)} \Big] + \mathbb{E}_{s_1,s_2 \sim \mathbb{P}_{\mathcal{S}_\theta} \bigotimes \mathbb{P}_{\mathcal{S}_{t-1}}} \Big[\beta\lambda_1(s_1, \pi_\theta(s_1)) - \beta\lambda_2(s_2, \pi_{t-1}(s_2)) + \beta\gamma \exp\left(\frac{\lambda_1(s_1, \pi_\theta(s_1)) - \lambda_2(s_2, \pi_{t-1}(s_2)) - C((s_1, \pi_\theta(s_1)), (s_2, \pi_{t-1}(s_2)))}{\gamma} \right) \Big]$$

Where $\tau_1 = s_0, a_0, r_0, \cdots, s_H, a_H, r_H$. Let $\theta_{t-1}^{(0)} = \theta_{t-1}$.
for $\ell = 1, \cdots, L$ **do**

a. Approximate the expectation $\mathbb{E}_{s_1,s_2 \sim \mathbb{P}_{\mathcal{S}_\theta} \bigotimes \mathbb{P}_{\mathcal{S}_{t-1}}}$ via $2M$ samples.
b. Take SGA step $\theta_{t-1}^{(\ell)} = \theta_{t-1}^{(\ell-1)} + \eta\hat{\nabla}_\theta \hat{F}(\theta_{t-1}^{(\ell-1)})$ using samples from a. and trajectories from current π_θ.
c. Use samples from $a.$ and Algorithm 1 to update λ_1, λ_2.

Set $\theta_t = \theta_{t-1}^{(M)}$.

A Lower-variance Gradient Estimator via Off-Policy embeddings: As explained in Section 5.2, the BGPG considers an objective which involves two parts: the conventional surrogate loss function for policy optimization (Schulman et al., 2017), and a loss function that involves the Behavior Test Functions. Though we could apply vanilla reinforced gradients on

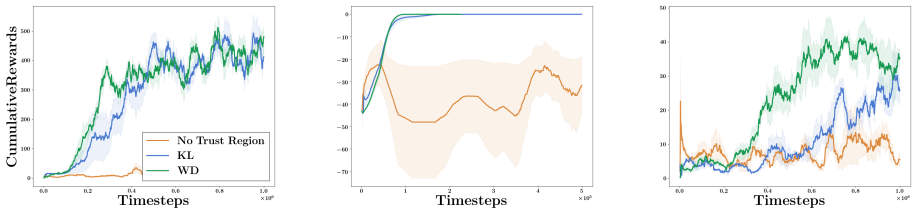
both parts, it is straightforward to notice that the second part can be optimized with reparameterized gradients (Kingma & Welling, 2013), which arguably have lower variance compared to the reinforced gradients. In particular, we note that under random feature approximation (5), as well as the action-concatenation embedding, the Wasserstein distance loss $\widehat{\text{WD}}_\gamma(P^\Phi_{\pi_\theta}, P^\Phi_b)$ is a differentiable function of θ. To see this more clearly, notice that under a Gaussian policy $a \sim \mathcal{N}(\mu_\theta(s), \sigma_\theta(s)^2)$ the actions $a = \mu_\theta(s) + \sigma_\theta(s) \cdot \epsilon$ are reparametrizable for ϵ being standard Gaussian noises. We can directly apply the reparametrization trick to this second objective to obtain a gradient estimator with potentially much lower variance. In our experiments, we applied this lower-variance gradient estimator. In Algorithm 6 we allow the state probing distribution to evolve with the iteration index of the algorithm t.

Trust Region Policy Optimization: Though the original TRPO (Schulman et al., 2015) construct the trust region based on KL-divergence, we propose to construct the trust region with WD. For convenience, we adopt a dual formulation of the trust region method and aim to optimize the augmented objective $\mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] - \beta \text{WD}_\gamma(\mathbb{P}^\Phi_{\pi'}, \mathbb{P}^\Phi_{\pi_\theta})$. We apply the concatenation-of-actions embedding and random feature maps to calculate the trust region. We identify several important hyperparameters: the RKHS (for the test function) is produced by RBF kernel $k(x, y) = \exp(\|x - y\|_2^2/\sigma^2)$ with $\sigma = 0.1$; the number of random features is $D = 100$; recall the embedding is $\Phi(\tau) = [a_1, a_2 ... a_H]$ where H is the horizon of the trajectory, here we take 10 actions per state and embed them together, this is equivalent to reducing the variance of the gradient estimator by increasing the sample size; the regularized entropy coefficient in the WD definition as $\gamma = 0.1$; the trust region trade-off constant $\beta \in \{0.1, 1, 10\}$. The alternate gradient descent is carried out with $T = 100$ alternating steps and test function coefficients $\mathbf{p} \in \mathbb{R}^D$ are updated with learning rate $\alpha_\mathbf{p} = 0.01$.

The baseline algorithms are: No trust region, and trust region with KL-divergence. The KL-divergence is identified by a maximum KL-divergence threshold per update, which we set to $\epsilon = 0.01$.

Across all algorithms, we adopt the open source implementation (Dhariwal et al., 2017). Hyper-parameters such as number of time steps per update as well as implementation techniques such as state normalization are default in the original code base.

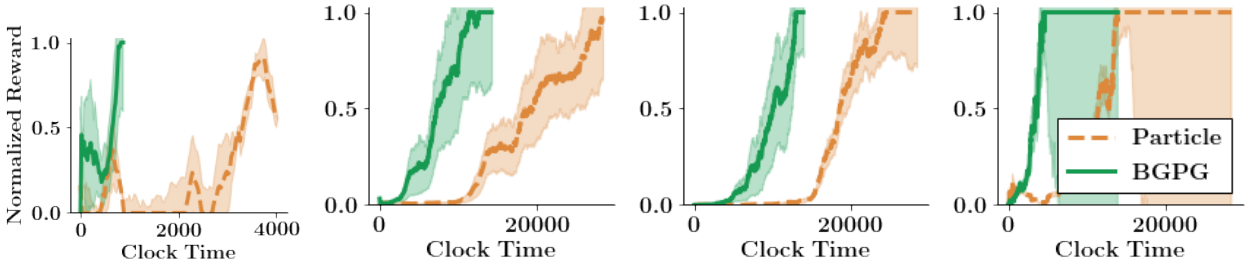
The additional experiment results can be found in Figure 8 where we show comparison on additional continuous control benchmarks: Tasks with DM are from DeepMind Contol Suites (Tassa et al., 2018). We see that the trust region constructed from the WD consistently outperforms other baselines (importantly, trust region methods are always better than the baseline without trust region, this confirms that trust region methods are critical in stabilizing the updates).



(a) **Reacher** (b) **MountainCar** (c) **Acrobot**

Figure 8: Additional Experiment on TRPO. We compare No Trust Region with two alternative trust region constructions: KL-divergence and Wassertein distance (ours).

Wasserstein AO vs. Particle Approximation: To calculate the regularized Wasserstein distance, we propose a gradient descent method that iteratively updates the test function. The alternting optimization (AO) scheme consists of updating both the test function and the distribution parameters such that the regularized Wasserstein distance of the trainable distribution against the reference distribution is minimized. Alternatively, we can also adopt a particle approximation method to calculate the Wasserstein distance and update the distribution parameters using an approximate gradient descent method (Zhang et al., 2018). We see the benefit in clock time in Fig 9.



(a) **Pendulum** (b) **Hopper: Stand** (c) **Hopper: Hop** (d) **Walker: Stand**

Figure 9: The clock-time comparison (in sec) of BGPG (alternating optimization) with particle approximation.

One major advantage of BGPG against particle approximation is its ease of parallelization. In particular, when using the concatenation-of-actions embedding, the aggregate Wasserstein distance can be decomposeed into an average of a set of Wasserstein distances over states. To calculate this aggregated gradient, BGPG can easily leverage the matrix multiplication; on the other hand, particle approximation requires that the dual optimal variables of each subproblem be computed, which is not straightforward to parallelize.

We test both methods in the context of trust region policy search, in which we explicitly calculate the Wasserstein distance of consecutive policies and enforce the constraints using a line search as in (Schulman et al., 2015). Both methods require the trust region trade-off parameter $\beta \in \{0.1, 1, 10\}$. We adopt the particle method in (Zhang et al., 2018) where for each state there are $M = 16$ particles. The gradients are derived based a RKHS where we adaptively adjust the coefficient of the RBF kernel based on the mean distance between particles. For the AO, we find that it suffices to carry out $T \in \{1, 5, 10\}$ gradient descents to approximate the regularized Wasserstein distance.

B.2. BGES

Here we reproduce a detailed version of Algorithm 4:

Algorithm 7 Behavior-Guided Evolution Strategies with **On-Policy** Embeddings

Input: learning rate η, noise standard deviation σ, iterations T, BEM Φ, β
Initialize: Initial policy π_0 parametrized by θ_0, Behavioral Test Functions λ_1, λ_2. Evaluate policy π_0 to return trajectory τ_0 and subsequently use the BEM to produce an initial $\hat{\mathbb{P}}^{\Phi}_{\pi_0}$.
for $t = 1, \ldots, T-1$ **do**

1. Sample $\epsilon_1, \cdots, \epsilon_n$ independently from $\mathcal{N}(0, I)$.
2. Evaluate policies $\{\pi_t^k\}_{k=1}^n$ parameterized by $\{\theta_t + \sigma\epsilon_k\}_{k=1}^n$ to return rewards R_k and trajectories τ_k for all k.
3. Use BEM to map trajectories τ_k to produce empirical $\hat{\mathbb{P}}^{\Phi}_{\pi_t^k}$ for all k.
4. Update λ_1 and λ_2 using Algorithm 1, where $\mu = \frac{1}{n} \cup_{k=1}^n \hat{\mathbb{P}}^{\Phi}_{\pi_{t-1}^k}$ and $\nu = \frac{1}{n} \cup_{k=1}^n \hat{\mathbb{P}}^{\Phi}_{\pi_t^k}$ are the uniform distribution over the set of from 3 for $t-1$ and t.
5. Approximate $\widehat{\mathrm{WD}}\gamma(\mathbb{P}^{\Phi}_{\pi_t^k}, \mathbb{P}^{\Phi}_{\pi_t})$ plugging in λ_1, λ_2 into Eq. 5 for each perturbed policy π_k
6. Update Policy: $\theta_{t+1} = \theta_t + \eta \nabla_{ES} F$, where:

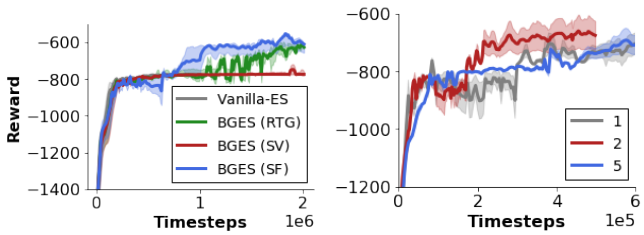
$$\nabla_{ES} F = \frac{1}{\sigma} \sum_{k=1}^{n} [(1-\beta)(R_k - R_t) + \beta \widehat{\mathrm{WD}}_{\gamma}(\mathbb{P}^{\Phi}_{\pi_t^k}, \mathbb{P}^{\Phi}_{\pi_t})]\epsilon_k$$

Efficient Exploration: To demonstrate the effectiveness of our method in exploring deceptive environments, we constructed two new environments using the MuJoCo simulator. For the point environment, we have a 6 dimensional state and 2 dimensional action, with the reward at each timestep calculated as the distance between the agent and the goal. We use a horizon of 50 which is sufficient to reach the goal. The quadruped environment is based on Ant from the Open AI Gym (Brockman et al., 2016), and has a similar reward structure to the point environment but a much larger state space (113) and action space (8). For the quadruped, we use a horizon length of 400.

To leverage the trivially parallelizable nature of ES algorithms, we use the ray library, and distribute the rollouts across 72 workers using AWS. Since we are sampling from an isotropic Gaussian, we are able to pass only the seed to the workers, as in (Salimans et al., 2017). However we do need to return trajectory information to the master worker.

For both the point and quadruped agents, we use random features with dimensionality $m = 1000$, and 100 warm-start updates for the WD at each iteration. For point, we use the final state embedding, learning rate $\eta = 0.1$ and $\sigma = 0.01$. For the quadruped, we use the reward-to-go embedding, as we found this was needed to learn locomotion, as well as a learning rate of $\eta = 0.02$ and $\sigma = 0.02$. The hyper-parameters were the same for all ES algorithms. When computing the WD, we used the previous 2 policies, θ_{t-1} and θ_{t-2}.

Our approach includes several new hyperparameters, such as the kernel for the Behavioral Test Functions and the choice of BEM. For our experiments we did not perform any hyperparameter optimization. We only considered the rbf kernel, and only varied the BEM for BGES. For BGES, we demonstrated several different BEMs, and we show an ablation study for the point agent in Fig. 12 where we see that both the reward-to-go (RTG) and Final State (SF) worked, but the vector of all states (SV) did not (for 5 seeds). We leave learned BEMs as exciting future work.



(a) **Embeddings** (b) **Previous Policies**

Figure 10: A sensitivity analysis investigating a) the impact of the embedding and b) the number of previous policies $\theta_{t-i}, i \in 1, 2, 5$

For embeddings, we compare the reward-to-go (RTG), concatenation of states (SV) and final state (SF). In both the RTG and SF case the agent learns to navigate past the wall (> -800). For the number of previous policies, we use the SF embedding, and using 2 appears to work best, but both 1 and 5 do learn the correct behavior.

Escaping Local Maxima: We also demonstrated that our method leads to faster training even in more standard settings, where exploration is not that crucial, but the optimization can be trapped in local maxima. To show it, we compared baseline ES algorithm for ES optimization from (Salimans et al., 2017) with its enhancements, where regularizers using different metrics on the space of probabilistic distributions corresponding to policy embeddings were used, as in the previous paragraph. We noticed that adding Wasserstein regularizers drastically improved optimization, whereas regularizers based on other distances/divergencies, namely: Hellinger, Jensen-Shannon, KL and TV did not have any impact. We considered Swimmer task from OpenAI Gym and to make it challenging, reduced the number of perturbations per iteration to 80. In that setting our method was the only one that was not trapped in local maxima and managed to learn effective policies.

B.3. Imitation Learning:

For the Imitation Learning experiment we used the reward-to-go embedding, with learning rate $\eta = 0.1$ and $\sigma = 0.01$. We use one oracle policy, which achieves > 360 on the environment. The only information provided to the algorithm is the embedded trajectory, used to compute the WD. This has exciting future applications since no additional information about the oracle is required in order to significantly improve learning.

B.4. Repulsion Learning and Attraction learning

Here we reproduce a full version of Algorithm 2:

Algorithm 8 Behvaior-Guided Repulsion (and Attraction) Learning with **On-Policy** Embeddings

Input: $\beta, \eta > 0$, $M \in \mathbb{N}$

Initialize: Initial stochastic policies $\pi_0^{\mathbf{a}}, \pi_0^{\mathbf{b}}$, parametrized by $\theta_0^{\mathbf{a}}, \theta_0^{\mathbf{b}}$ respectively, Behavioral Test Functions $\lambda_1^{\mathbf{a}}, \lambda_2^{\mathbf{b}}$

for $t = 1, \dots, T$ **do**

1. Collect M trajectories $\{\tau_i^{\mathbf{a}}\}_{i=1}^M$ from $\mathbb{P}_{\pi_{t-1}^{\mathbf{a}}}$ and M trajectories $\{\tau_i^{\mathbf{b}}\}_{i=1}^M$ from $\mathbb{P}_{\pi_{t-1}^{\mathbf{b}}}$. Approximate $\mathbb{P}_{\pi_{t-1}^{\mathbf{a}}} \bigotimes \mathbb{P}_{\pi_{t-1}^{\mathbf{b}}}$ via $\frac{1}{M}\{\tau_i^{\mathbf{a}}\}_{i=1}^M \bigotimes \frac{1}{M}\{\tau_i^{\mathbf{b}}\}_{i=1}^M := \hat{P}_{\pi_{t-1}^{\mathbf{a}}, \pi_{t-1}^{\mathbf{b}}}$
2. Form two distinct surrogate rewards for joint trajectories of agents $\mathbf{a}$ and $\mathbf{b}$:

$$\tilde{R}_{\mathbf{a}}(\tau_1, \tau_2) = \mathcal{R}(\tau_1) + \beta\lambda_1^{\mathbf{a}}(\Phi(\tau_1)) + \beta\gamma \exp\left(\frac{\lambda_1^{\mathbf{a}}(\Phi(\tau_1)) - \lambda_2^{\mathbf{b}}(\Phi(\tau_2)) - C(\Phi(\tau_1)), \Phi(\tau_2))}{\gamma}\right)$$
$$\tilde{R}_{\mathbf{b}}(\tau_1, \tau_2) = \mathcal{R}(\tau_2) - \beta\lambda_2^{\mathbf{b}}(\Phi(\tau_2)) + \beta\gamma \exp\left(\frac{\lambda_1^{\mathbf{a}}(\Phi(\tau_1)) - \lambda_2^{\mathbf{b}}(\Phi(\tau_2)) - C(\Phi(\tau_1)), \Phi(\tau_2))}{\gamma}\right)$$

3. For $\mathbf{c} \in \{\mathbf{a}, \mathbf{b}\}$ use the Reinforce estimator to take gradient steps:

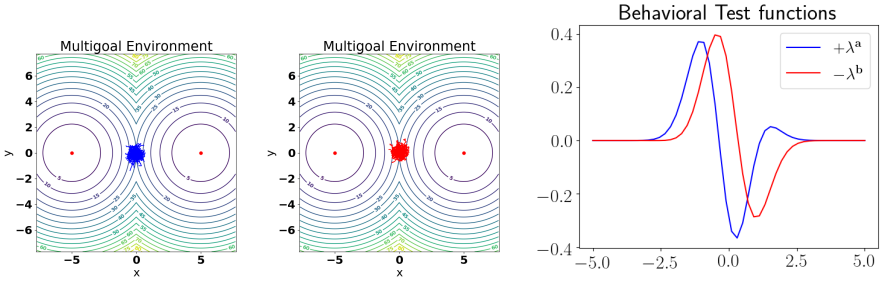
$$\theta_t^{\mathbf{c}} = \theta_{t-1}^{\mathbf{c}} + \eta \mathop{\mathbb{E}}_{\tau^{\mathbf{a}}, \tau^{\mathbf{b}} \sim \hat{P}_{\pi_{t-1}^{\mathbf{a}}, \pi_{t-1}^{\mathbf{b}}}} \left[\tilde{\mathcal{R}}_{\mathbf{c}}(\tau^{\mathbf{a}}, \tau^{\mathbf{b}}) \left(\sum_{i=0}^{H-1} \nabla_{\theta_{t-1}^{\mathbf{c}}} \log\left(\pi_{t-1}^{\mathbf{c}}(a_i^{\mathbf{c}}|s_i^{\mathbf{c}})\right)\right)\right]$$

Where $\tau^{\mathbf{a}} = s_0^{\mathbf{a}}, a_0^{\mathbf{a}}, r_0^{\mathbf{a}}, \cdots, s_H^{\mathbf{a}}, a_H^{\mathbf{a}}, r_H^{\mathbf{a}}$ and $\tau^{\mathbf{b}} = s_0^{\mathbf{b}}, a_0^{\mathbf{b}}, r_0^{\mathbf{b}}, \cdots, s_H^{\mathbf{b}}, a_H^{\mathbf{b}}, r_H^{\mathbf{b}}$.

5. Use samples from $\hat{P}_{\pi_{t-1}^{\mathbf{a}}, \pi_{t-1}^{\mathbf{b}}}$ and Algorithm 1 to update the Behavioral Test Functions $\lambda_1^{\mathbf{a}}, \lambda_2^{\mathbf{b}}$.

Algorithm 8 is the de version of the repulsion algorithm from Section 5.2. The algorithm maintains two policies $\pi^{\mathbf{a}}$ and $\pi^{\mathbf{b}}$. Each policy is optimized by taking a policy gradient step (using the Reinforce gradient estimator) in the direction optimizing surrogate rewards $\tilde{\mathcal{R}}_{\mathbf{a}}$ and $\tilde{\mathcal{R}}_{\mathbf{b}}$ that combines the signal from the task's reward function $\mathcal{R}$ and the repulsion (or attraction) score encoded by the behavioral test functions $\lambda^{\mathbf{a}}$ and $\lambda^{\mathbf{b}}$.

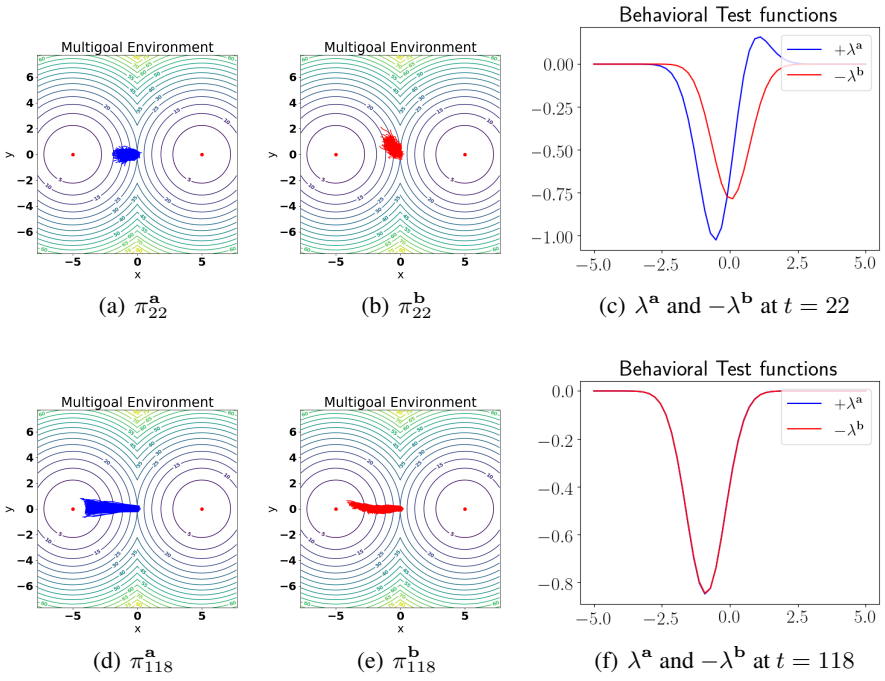
We conducted experiments testing Algorithm 2 on a simple Mujoco environment consisting of a particle that moves on the plane and whose objective is to learn a policy that allows it to reach one of two goals. Each policy outputs a velocity vector and stochasticity is achieved by adding Gaussian noise to the mean velocity encoded by a neural network with two size 5 hidden layers and ReLu activations. If an agent performs action a at state s, it moves to state $a + s$. The reward of an agent after performing action a at state s equals $-\|a\|^2 * 30 - \min(d(s, \text{Goal}_1), d(s, \text{Goal}_2))^2$ where $d(x, y)$ denotes the distance between x and y in $\mathbb{R}^2$. The initial state is chosen by sampling a Gaussian distribution with mean $\binom{0}{0}$ and diagonal variance 0.1. In each iteration step we sample 100 trajectories. In the following pictures we plot the policies' behavior by plotting 100 trajectories of each. The embedding $\Phi : \Gamma \to \mathbb{R}$ maps trajectories τ to their mean displacement in the $x-$axis. We use the squared absolute value difference as the cost function. When $\beta < 0$ we favour attraction and the agent are encouraged to learn a similar policy to solve the same task.



(a) $\pi_0^{\mathbf{a}}$ (b) $\pi_0^{\mathbf{b}}$ (c) $\lambda^{\mathbf{a}}$ and $-\lambda^{\mathbf{b}}$ at $t = 0$

Figure 11: Initial state of policies $\pi^{\mathbf{a}}, \pi^{\mathbf{b}}$ and Behavioral Test functions $\lambda^{\mathbf{a}}, \lambda^{\mathbf{b}}$ in the Multigoal environment.

There are two optimal policies, moving the particle to the left goal or moving it to the right goal. We now plot how the policies' behavior and evolves throughout optimization and how the Behavioral Test Functions guide the optimization by favouring the two policies to be close by or far apart.



(a) $\pi^{\mathbf{a}}_{22}$ (b) $\pi^{\mathbf{b}}_{22}$ (c) $\lambda^{\mathbf{a}}$ and $-\lambda^{\mathbf{b}}$ at $t = 22$

(d) $\pi^{\mathbf{a}}_{118}$ (e) $\pi^{\mathbf{b}}_{118}$ (f) $\lambda^{\mathbf{a}}$ and $-\lambda^{\mathbf{b}}$ at $t = 118$

Figure 12: Evolution of the policies and Behavioral Test Functions throughout optimization.

Let $\mathcal{X}$ and $\mathcal{Y}$ be the domains of two measures μ, and ν. Recall that in case $\gamma = 0$, $\mathcal{X} = \mathcal{Y}$, and $C(x, x) = 0$ for all $x \in \mathcal{X}$, then $\lambda^*_\mu(x) = \lambda^*_\nu(x) = \lambda^*(x)$ for all $x \in \mathcal{X}$. In the case of regularized Wasserstein distances with $\gamma > 0$, this relationship may not hold true even if the cost satisfies the same diagonal assumption. For example when the regularizing measure is the product measure, and μ, ν have disjoint supports, since the soft constraint $\gamma \exp\left(\frac{\lambda_\mu(\mathbf{x}) - \lambda_\nu(\mathbf{y}) - C(\mathbf{x},\mathbf{y})}{\gamma}\right)$ is enforced in expectation over the product measure there may exist optimal solutions $\lambda^*_\mu, \lambda^*_\nu$ that do not satisfy $\lambda^*_\mu = \lambda^*_\nu$.

C. Theoretical results

We start by exploring some properties of the Wasserstein distance and its interaction with some simple classes of embeddings. The first lemma we show has the intention to show conditions under which two policies can be shown to be equal provided the Wasserstein distance between its trajectory embeddings is zero. This result implies that our framework is capable of capturing equality of policies when the embedding space equals the space of trajectories.

Lemma C.1. *Let $\mathcal{S}$ and $\mathcal{A}$ be finite sets, the* MDP *be episodic (i.e. of finite horizon H), and $\Phi(\tau) = \sum_{t=0}^{H} e_{s_t,a_t}$ with $e_{s,a} \in \mathbb{R}^{|\mathcal{S}|+|\mathcal{A}|}$ the indicator vector for the state action pair (s, a). Let $C(\mathbf{v}, \mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|_p^p$ for $p \geq 1$. If $\gamma = 0$ and $\mathrm{WD}_\gamma(\mathbb{P}^\Phi_\pi, \mathbb{P}^\Phi_{\pi'}) = 0$ then $\pi = \pi'$.*

Proof. If $\mathrm{WD}_\gamma(\mathbb{P}^\Phi_\pi, \mathbb{P}^\Phi_{\pi'}) = 0$, there exists a coupling Π between $\mathbb{P}^\Phi_\pi$ and $\mathbb{P}^\Phi_{\pi'}$ such that:

$$\mathbb{E}_{u,v\sim\Pi}\left[\|u - v\|_p^p\right] = 0$$

Consequently:

$$\mathbb{E}_{u,v\sim\Pi}\left[\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} |u_{s,a} - v_{s,a}|^p\right] = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathbb{E}_{u,v\sim\Pi}\left[|u_{s,a} - v_{s,a}|^p\right] = 0$$

Therefore for all $(s, a) \in \mathcal{S} \times \mathcal{A}$:

$$\left|\mathbb{E}_{u\sim\mathbb{P}_{\pi}^{\Phi}}\left[u_{s,a}\right] - \mathbb{E}_{v\sim\mathbb{P}_{\pi'}^{\Phi}}\left[v_{s,a}\right]\right|^p \leq \mathbb{E}_{u,v\sim\Pi}\left[|u_{s,a} - v_{s,a}|^p\right] = 0$$

Where $u_{s,a}$ and $v_{s,a}$ denote the (s, a) entries of u and v respectively. Notice that for all $(s, a) \in \mathcal{S} \times \mathcal{A}$:

$$\mathbb{P}_{\pi}^{\Phi}(s, a) = \mathbb{P}_{\pi'}^{\Phi}(s, a) \tag{11}$$

Since for all $s \in \mathcal{S}$ and $p \geq 1$:

$$\left|\sum_{a\in\mathcal{A}} u_{s,a} - v_{s,a}\right|^p \leq \sum_{a\in\mathcal{A}} |u_{s,a} - v_{s,a}|^p$$

Therefore for all $s \in \mathcal{S}$:

$$\left|\mathbb{E}_{u\sim\mathbb{P}_{\pi}^{\Phi}}\left[\sum_{a\in\mathcal{A}} u_{s,a}\right] - \mathbb{E}_{v\sim\mathbb{P}_{\pi'}^{\Phi}}\left[\sum_{a\in\mathcal{A}} v_{s,a}\right]\right|^p \leq \mathbb{E}_{u,v\sim\Pi}\left[\sum_{a\in\mathcal{A}} |u_{s,a} - v_{s,a}|^p\right] = 0$$

Consequently $\mathbb{P}_{\pi}^{\Phi}(s) = \mathbb{P}_{\pi'}^{\Phi}(s)$ for all $s \in \mathcal{S}$. By Bayes rule, this plus equation 11 yields:

$$\mathbb{P}_{\pi}^{\Phi}(a|s) = \mathbb{P}_{\pi'}^{\Phi}(a|s)$$

And therefore: $\pi = \pi'$. □

These results can be extended in the following ways:

1. In the case of a continuous state space, it is possible to define embeddings using Kernel density estimators. Under the appropriate smoothness conditions on the visitation frequencies, picking an adequate bandwidth and using the appropriate norm to compare different embeddings it is possible to derive similar results to those in Lemma C.1 for continuous state spaces.
2. For embeddings such as Φ_5 in Section 3.1 or $\Phi(\tau) = \sum_{t=0}^{H} e_{s_t,a_t}$, when $\gamma = 0$, if $\mathrm{WD}_{\gamma}(\mathbb{P}_{\pi}^{\Phi}, \mathbb{P}_{\pi'}^{\Phi}) \leq \epsilon$ then $|V(\pi) - V(\pi')| \leq \epsilon R$ for $R = \max_{\tau\in\Gamma} \mathcal{R}(\tau)$ thus implying that a small Wasserstein distance between π and π's PPEs implies a small difference in their value functions.

C.1. Random features stochastic gradients

Let ϕ_κ and ϕ_ℓ be two feature maps over $\mathcal{X}$ and $\mathcal{Y}$ and corresponding to kernels κ and ℓ respectively. For this and the following sections we will make use of the following expression:

$$\begin{aligned} G(\mathbf{p}^\mu, \mathbf{p}^\nu) = \beta \int_{\mathcal{X}} \left(\mathbf{p}^\mu\right)^\top \phi_\kappa(\mathbf{x}) d\mu(\mathbf{x}, \theta) - \beta \int_{\mathcal{Y}} \left(\mathbf{p}^\nu\right)^\top \phi_\ell(\mathbf{y}) d\nu(\mathbf{y}) + \\ \gamma\beta \int_{\mathcal{X}\times\mathcal{Y}} \exp\left(\frac{\left(\mathbf{p}^\mu\right)^\top \phi_\kappa(\mathbf{x}) - \left(\mathbf{p}^\nu\right)^\top \phi_\ell(\mathbf{y}) - C(\mathbf{x}, \mathbf{y})}{\gamma}\right) d\mu(\mathbf{x}) d\nu(\mathbf{y}) \end{aligned} \tag{12}$$

We now show how to compute gradients with respect to the random feature maps:

Lemma C.2. *The gradient $\nabla_{\binom{\mathbf{p}^\mu}{\mathbf{p}^\nu}} G(\mathbf{p}^\mu, \mathbf{p}^\nu)$ of the objective function from Equation 12 with respect to the parameters $\binom{\mathbf{p}^\mu}{\mathbf{p}^\nu}$ satisfies:*

$$\nabla_{\binom{\mathbf{p}^\mu}{\mathbf{p}^\nu}} G(\mathbf{p}^\mu, \mathbf{p}^\nu) = \beta \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mu\otimes\nu}\left[\left(1 - \exp\left(\frac{\left(\mathbf{p}^\mu\right)^\top \phi_\kappa(\mathbf{x}) - \left(\mathbf{p}^\nu\right)^\top \phi_\ell - C(\mathbf{x}, \mathbf{y})}{\gamma}\right)\right)\binom{\phi_\kappa(\mathbf{x})}{-\phi_\ell(\mathbf{y})}\right]$$

Proof. A simple use of the chain rule, taking the gradients inside the expectation, and the fact that $\mathbf{p}^\mu$ and $\mathbf{p}^\nu$ are vectors yields the desired result. □

The main consequence of this formulation is the stochastic gradients we use in Algorithm 1.

C.2. Behavior Guided Policy Gradient and Wasserstein trust region

For a policy π, we denote as: V^π, Q^π and $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ the: value function, Q-function and advantage function.

The chief goal of this section is to prove Theorem 5.1. We restate the section's definitions here for the reader's convenience: To ease the discussion we make the following assumptions:

- Finite horizon T.
- Undiscounted MDP.
- States are time indexed. In other words, states visited at time t can't be visited at any other time.
- $\mathcal{S}$ and $\mathcal{A}$ are finite sets.

The third assumption is solely to avoid having to define a time indexed Value function. It can be completely avoided. We chose not to do this in the spirit of notational simplicity. These assumptions can be relaxed, most notably we can show similar results for the discounted and infinite horizon case. We chose to present the finite horizon proof because of the nature of our experimental results.

Let $\Phi = \mathrm{id}$ be the identity embedding so that $\mathcal{E} = \Gamma$. In this case $\mathbb{P}_\pi^\Phi$ denotes the distribution of trajectories corresponding to policy π. We define the value function $V^\pi : \mathcal{S} \to \mathbb{R}$ as

$$V^\pi(s_t = s) = \mathbb{E}_{\tau \sim \mathbb{P}_\pi^{\mathrm{id}}} \left[\sum_{\ell=t}^{T} R(s_{\ell+1}, a_\ell, s_\ell) | s_t = s \right]$$

The Q-function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as:

$$Q^\pi(s_t, a_t = a) = \mathbb{E}_{\tau \sim \mathbb{P}_\pi^{\mathrm{id}}} \left[\sum_{\ell=t}^{T} R(s_{\ell+1}, a_\ell, s_\ell) \right]$$

Similarly, the advantage function is defined as:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$

We denote by $V(\pi) = \mathbb{E}_{\tau \sim \mathbb{P}_\pi^{\mathrm{id}}} \left[\sum_{t=0}^{T} R(s_{t+1}, a_t, s_t) \right]$ the expected reward of policy π and define the visitation frequency as:

$$\rho_\pi(s) = \mathbb{E}_{\tau \sim \mathbb{P}_\pi^{\mathrm{id}}} \left[\sum_{t=0}^{T} \mathbf{1}(s_t = s) \right]$$

The first observation in this section is the following lemma:

Lemma C.3. *two distinct policies π and $\tilde{\pi}$ can be related via the following equation :*

$$V(\tilde{\pi}) = V(\pi) + \sum_{s \in \mathcal{S}} \left(\rho_{\tilde{\pi}}(s) \left(\sum_{a \in \mathcal{A}} \tilde{\pi}(a|s) A^\pi(s,a) \right) \right)$$

Proof. Notice that $A^\pi(s,a) = \mathbb{E}_{s' \sim P(s'|a,s)} \left[R(s',a,s) + V^\pi(s') - V^\pi(s) \right]$. Therefore:

$$\begin{aligned}
\mathbb{E}_{\tau \sim \mathbb{P}_{\tilde{\pi}}^{\mathrm{id}}} \left[\sum_{t=0}^{T} A_\pi(s_t, a_t) \right] &= \mathbb{E}_{\tau \sim \mathbb{P}_{\tilde{\pi}}^{\mathrm{id}}} \left[\sum_{t=0}^{T} R(s_{t+1}, a_t, s_t) + V^\pi(s_{t+1}) - V^\pi(s_t) \right] \\
&= \mathbb{E}_{\tau \sim \mathbb{P}_{\tilde{\pi}}^{\mathrm{id}}} \left[\sum_{t=0}^{T} R(s_{t+1}, a_t, s_t) \right] - \mathbb{E}_{s_0} \left[V^\pi(s_0) \right] \\
&= -V(\pi) + V(\tilde{\pi})
\end{aligned}$$

The result follows. □

See (Sutton et al., 1998) for an alternative proof. We also consider the following linear approximation to V around policy π (see: (Kakade & Langford, 2002)):

$$L(\tilde{\pi}) = V(\pi) + \sum_{s \in \mathcal{S}} \left(\rho_{\pi}(s) \left(\sum_{a \in \mathcal{A}} \tilde{\pi}(a|s) A^{\pi}(s,a) \right) \right)$$

Where the only difference is that $\rho_{\tilde{\pi}}$ was substituted by ρ_{π}. Consider the following embedding $\Phi^s : \Gamma \to \mathbb{R}^{|\mathcal{S}|}$ defined by $(\Phi(\tau))_s = \sum_{t=0}^{T} \mathbf{1}(s_t = s)$, and related cost function defined as: $C(\mathbf{v}, \mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|_1$.

Lemma C.4. *The Wasserstein distance* $\mathrm{WD}_0(\mathbb{P}_{\tilde{\pi}}^{\Phi^s}, \mathbb{P}_{\pi}^{\Phi^s})$ *is related to visit frequencies since:*

$$\mathrm{WD}_0(\mathbb{P}_{\tilde{\pi}}^{\Phi^s}, \mathbb{P}_{\pi}^{\Phi^s}) \geq \sum_{s \in \mathcal{S}} |\rho_{\pi}(s) - \rho_{\tilde{\pi}}(s)|$$

Proof. Let Π be the optimal coupling between $\mathbb{P}_{\tilde{\pi}}^{\Phi^s}$ and $\mathbb{P}_{\pi}^{\Phi^s}$. Then:

$$\begin{aligned} \mathrm{WD}_0(\mathbb{P}_{\tilde{\pi}}^{\Phi^s}, \mathbb{P}_{\pi}^{\Phi^s}) &= \mathbb{E}_{u,v \sim \Pi} \left[\|u - v\|_1 \right] \\ &= \sum_{s \in \mathcal{S}} \mathbb{E}_{u,v \sim \Pi} \left[|u_s - v_s| \right] \end{aligned}$$

Where u_s and v_s denote the $s \in \mathcal{S}$ indexed entry of the u and v vectors respectively. Notice that for all $s \in \mathcal{S}$ the following is true:

$$\left| \underbrace{\mathbb{E}_{u \sim \mathbb{P}_{\pi}^{\Phi^s}} \left[u_s \right]}_{\rho_{\pi}(s)} - \underbrace{\mathbb{E}_{v \sim \mathbb{P}_{\pi}^{\Phi^s}} \left[v_s \right]}_{\rho_{\pi'}(s)} \right| \leq \mathbb{E}_{u,v \sim \Pi} \left[|u_s - v_s| \right]$$

The result follows.

□

These observations enable us to prove an analogue of Theorem 1 from (Schulman et al., 2015), namely:

Theorem C.5. *If* $\mathrm{WD}_0(\mathbb{P}_{\tilde{\pi}}^{\Phi^s}, \mathbb{P}_{\pi}^{\Phi^s}) \leq \delta$ *and* $\epsilon = \max_{s,a} |A^{\pi}(s,a)|$*, then* $V(\tilde{\pi}) \geq L(\tilde{\theta}) - \delta\epsilon$*.*

As in (Schulman et al., 2015), Theorem 5.1 implies a policy improvement guarantee for BGPG from Section 5.4.

Proof. Notice that:

$$V(\tilde{\pi}) - L(\tilde{\pi}) = \sum_{s \in \mathcal{S}} \left((\rho_{\tilde{\pi}}(s) - \rho_{\pi}(s)) \left(\sum_{a \in \mathcal{A}} \tilde{\pi}(a|s) A^{\pi}(s,a) \right) \right)$$

Therefore by Holder inequality:

$$|V(\tilde{\pi}) - L(\tilde{\pi})| \leq \underbrace{\left(\sum_{s \in \mathcal{S}} |\rho_{\pi}(s) - \rho_{\tilde{\pi}}(s)| \right)}_{\leq \mathrm{WD}_0(\mathbb{P}_{\tilde{\pi}}^{\Phi^s}, \mathbb{P}_{\pi}^{\Phi^s}) \leq \delta} \underbrace{\left(\sup_{s \in \mathcal{S}} \left| \sum_{a \in \mathcal{A}} \tilde{\pi}(a|s) A^{\pi}(s,a) \right| \right)}_{\leq \epsilon}$$

The result follows. □

We can leverage the results of Theorem C.5 to show wasserstein trust regions methods with embedding Φ^s give a monotonically improving sequence of policies. The proof can be concluded by following the logic of Section 3 in (Schulman et al., 2015).

C.3. Off policy embeddings and their properties.

It is easy to see that if the cost function equals the l_2 norm between state-policy pairs and if $\mathrm{WD}_0(\mathbb{P}_{\pi}^{\Phi_{\mathcal{S}}}, \mathbb{P}_{\pi'}^{\Phi_{\mathcal{S}}}) = 0$ then $\mathbb{E}_{\mathbb{P}_{\mathcal{S}}}\left[1(\pi(S) \neq \pi'(S))\right] = 0$. If $\mathbb{P}_{\mathcal{S}}$ has mass only in relevant areas of the state space, a value of zero implies the two policies behave similarly where it matters. In the case when the user may care only about the action of a policy within a set of states of interest, this notion applies.

When the sampling distribution can be identified with the stationary distribution over states of the current policy, we can recover trust region-type of results for BGPG.