Distributional Multivariate Policy Evaluation and Exploration with the Bellman GAN - Supplementary Material

Dror Freirich [1] **Tzahi Shimkin** [1] **Ron Meir** [1] **Aviv Tamar** [2]

A. The Wasserstein-1 Distance

Let $\mathcal{X}$ be a Polish space with a complete metric d, and let $\mathfrak{B}(\mathcal{X}) \subset 2^{\mathcal{X}}$ denote the σ-algebra of Borel subsets. Denote by $\mathcal{P}(\mathcal{X})$ the set of probability measures on $(\mathcal{X}, \mathfrak{B}(\mathcal{X}))$. For $\mu_1, \mu_2 \in \mathcal{P}(\mathcal{X})$, $\Pi(\mu_1, \mu_2)$ is the set of joint distributions whose marginal distributions correspond to μ_1, μ_2. Let $p \in [1, \infty)$. The Wasserstein-p distance w.r.t. the metric d is defined by

$$W_p(\mu_1, \mu_2) = (\inf_{\gamma \in \Pi(\mu_1, \mu_2)} \mathbb{E}_{(X,Y)\sim\gamma} d^p(X, Y))^{\frac{1}{p}}. \tag{1}$$

An important special case is the *Earth Mover's distance,* also commonly called the Kantorovich–Rubinstein distance, or simply, Wasserstein-1 (Villani, 2008), where $\mathcal{X} = \mathbb{R}^n$ and

$$W_1(\mu_1, \mu_2) = \inf_{\gamma \in \Pi(\mu_1, \mu_2)} \mathbb{E}_{(X,Y)\sim\gamma} \|X - Y\|. \tag{2}$$

The Wasserstein-1 distance has the following duality property (Villani, 2008). For any $\mu_1, \mu_2 \in \mathcal{P}(\mathcal{X})$ with $\int_{\mathcal{X}} d(x_0, x) d\mu_i < \infty$, $i = 1, 2$ (here x_0 is an arbitrary point), the W_1 distance has the following dual integral probability metric (IPM; Müller (1997)) form

$$W_1(\mu_1, \mu_2) = \sup_{f \in 1-\mathrm{Lip}} \left\{ \int_{\mathcal{X}} f(x) d\mu_1 - \int_{\mathcal{X}} f(x) d\mu_2 \right\}, \tag{3}$$

where $1-\mathrm{Lip}$ is the class of Lipschitz functions $f : \mathcal{X} \to \mathbb{R}$, with a best admissible Lipschitz constant smaller or equal to 1.

B. 2-Face Climber Testbench

Consider the following problem: A climber is about to conquer the summit of a mountain. There are two possible ways to reach the top. The South Face is mild and easy to climb, while the north face is steep and much harder to climb, but the track is shorter, and reaching the top bears a greater reward.

The climb starts at base-camp ($s_0 = 0$), where the climber chooses the face to climb by taking an action $a_0 \in \{North, South\}$. When simulation starts, 2 random bit strings are chosen (a string for each face, 1 digit for every possible state). By $seq(s|face)$ we denote the bit chosen for state s on each face .'Climbing' a face is made by taking an action $a_t \in \{0, 1\}$ and comparing to the digit of current state. We can write transition rule for $s_t \neq 0$ as

$$\begin{cases} \begin{pmatrix} s_{t+1} \\ face_{t+1} \end{pmatrix} = \begin{pmatrix} s_t + 1 \\ face_t \end{pmatrix}, & a_t = seq(s_t|face_t) \\ \begin{pmatrix} s_{t+1} \\ face_{t+1} \end{pmatrix} = \begin{pmatrix} (s_t - fall) \vee 0 \\ face_t \end{pmatrix}, & a_t \neq seq(s_t|face_t) \end{cases} \tag{4}$$

where $face_t \in \{North, South\}$, $fall \sim unif\{0, \ldots, slope(face_t)\}$. For $s_t = 0$ (Camp) we have

$$\begin{cases} \begin{pmatrix} s_{t+1} \\ face_{t+1} \end{pmatrix} = \begin{pmatrix} 1 \\ South \end{pmatrix}, & a_t = 1 \\ \begin{pmatrix} s_{t+1} \\ face_{t+1} \end{pmatrix} = \begin{pmatrix} 1 \\ North \end{pmatrix}, & a_t = 0 \end{cases}. \tag{5}$$

Simulation is terminated when reaching the top, i.e.

$$s_{t+1} = s_{terminal}(face_t). \tag{6}$$

B.1. Rewards

We have a negative reward (cost) for every climb step (regardless of action),

$$r(s_t, a_t, s_{t+1}, face_{t+1}) = C_{face} < 0,$$

for $s_{t+1} \neq s_{terminal}(face_t)$ where 'climbing' the North face typically costs a little more than the South.

[1] The Viterbi Faculty of Electrical Engineering, Technion - Israel Institute of Technology [2] Berkeley AI Research Lab, UC Berkeley. Correspondence to: Dror Freirich <drorfrc@gmail.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

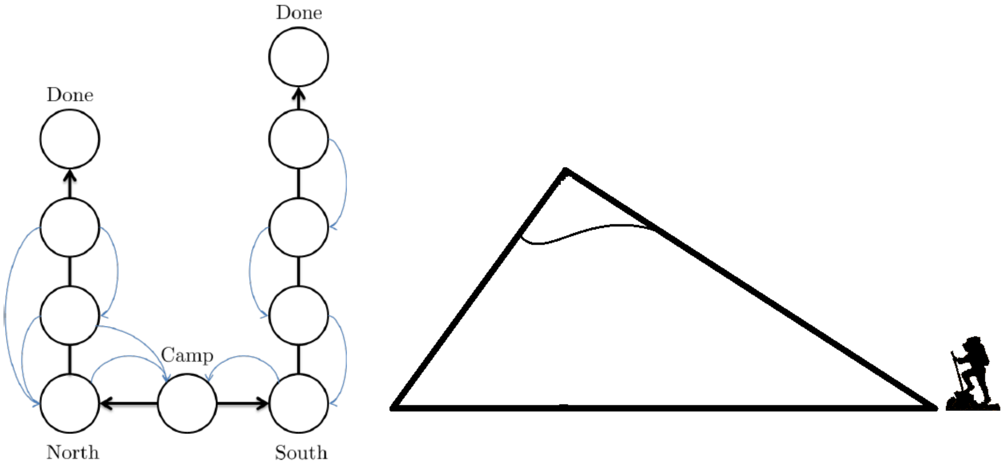


Figure 1. 2-Face Climber. The South Face is easy to climb. The North face is harder to climb, but the route is shorter. Choosing the right action progresses the climber towards the summit (bold edges), while the other action causes her to slip with some probability (light edges).

We also have a reward for reaching the top ($s_{t+1} = s_{terminal}(face_t)$),

$$r(s_t, a_t, s_{t+1}, face_{t+1}) = R_{face} \gg 0.$$

B.2. Parametric Setup

We set $s_{terminal}(North) = 10$, $s_{terminal}(South) = 20$, $slope(North) = 4$, $slope(South) = 1$, $C_{North} = -0.02$, $C_{South} = -0.01$, $R_{North} = 10$, $R_{South} = 1$.

B.3. Results

Figure 2 shows results of W-1ME exploration using different values of η in (21a), where we used reward-systematic exploration (Sec. 6.1). For policy improvement in Algorithm 2 we used DQN, where we refer to the full algorithm as W-1ME+DQN. $\eta = 0$ is for DQN, where we applied an ϵ-greedy exploration ($\epsilon = 0.05$). Average and median are over 100 independent seeds, where we run 1000 episodes on each experiment. Shaded areas represents the $\pm\sigma$ (std) range for reward averages, and min/max range for medians. We also present the state-space visit counts for the different η's. Here we can see that indeed, higher exploration η's increment the visit rate to the North face states, resulting in higher average returns.

C. LQR Testbench

We consider the LQR problem

$$s_{t+1} = As_t + Ba_t, \tag{7}$$

where $s_t \in \mathbb{R}^{n_s}$ and $a_t \in \mathbb{R}^{n_a}$. A, B are the system matrices with appropriate dimensions and S_0 is Normally distributed. We consider a Gaussian reward $r(s,a) \sim \mathcal{N}(-s^T Qs - a^T Ra, 1)$. Q, R are positive-definite matrices.

C.1. Parametric Setup

In our experiments, we set $n_s = 64$, $n_a = 2$. The matrices A, B, Q, R were randomly set for each independent experiment (seed).

C.2. Results

Figure 3 shows the median of the averaged cumulative reward (at each iteration) on a fixed set of 20 random seeds. Shaded areas represent the interquartile range. We used $\eta = 10^{-7}$ for both exploration methods. Here we can see that W-1ME exploration outperforms both VIME and plain TRPO.

D. Reward-Systematic Exploration

Here we evaluate the contribution of using reward systematic exploration (described in Section 6.1).
Figure 4 presents the results obtained on Cartpole Swingup sparse task using scalar setting, where we predict the value distribution, and using reward systematic exploration, where we learn the transition model together with the value distribution. The Figure shows that exploring using learned state transition results in improved median performance.

E. Implementation

We use the following architecture for both generator and discriminator (Figure 5(a)). DNN_0 and DNN_1 are constructed by a sequence of fully connected linear layers fol-

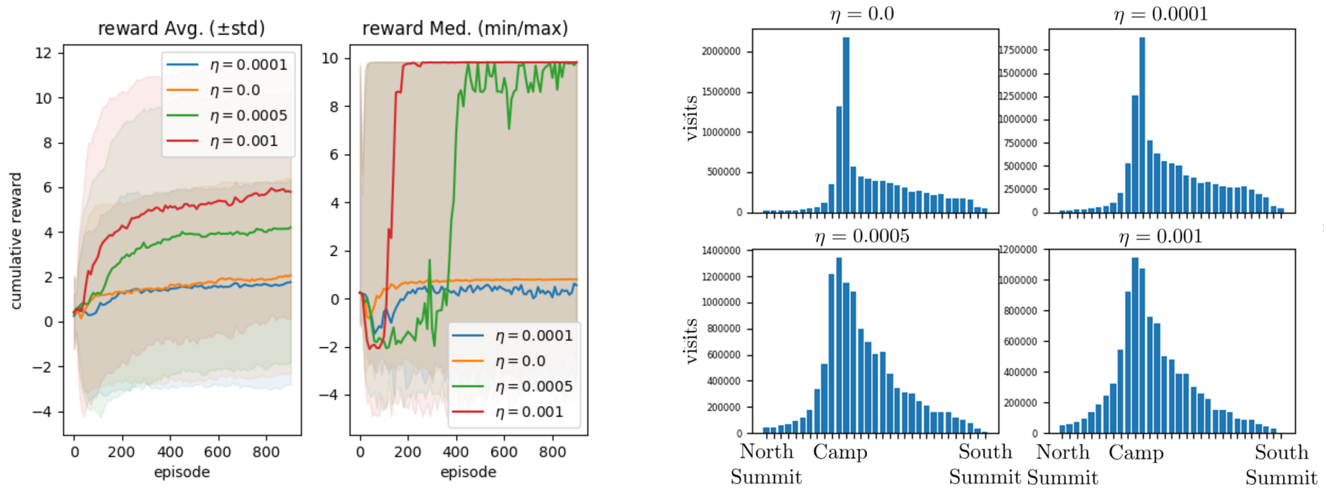
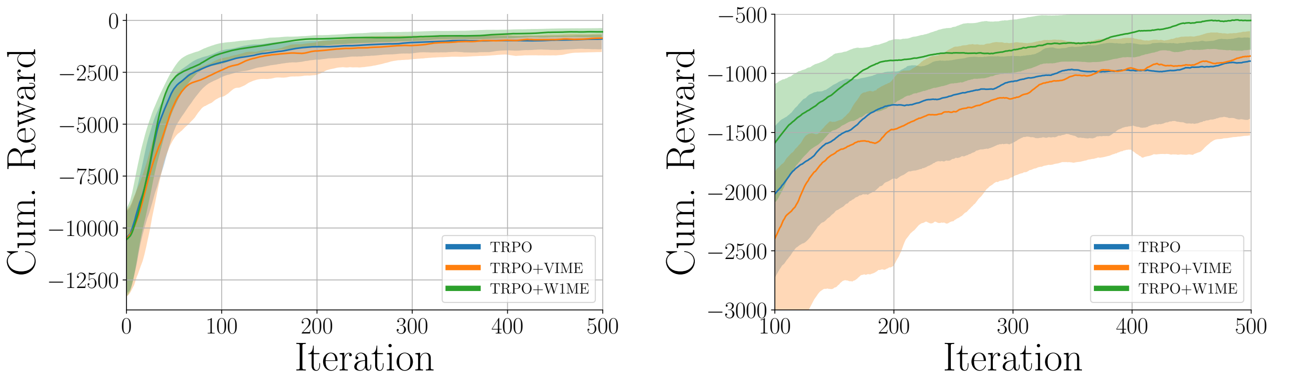


Figure 2. W-1ME exploration on 2Face climber. **Left:** Improvement in average and median return using W-1ME+DQN with different η parameters. **Right:** Histograms present the number of visits to each state. Observe that higher η's incremented the visit rate to the North face states, resulting in higher average returns. This shows the utility of our exploration method.



(a)

(b)

Figure 3. W-1ME vs. VIME and TRPO on LQR testbench. (a) The full learning session, (b) The final stages of learning.

lowed by Leaky ReLU activation. The generator's input is a Normal-distributed noise, and output dimension is the same as the return vector. Discriminator output is 1-dimensional. We optimize (14a) using the Adam optimizer.

We implemented DQN using the Double DQN algorithm (Van Hasselt et al., 2016), based on an action and a target network. Networks are implemented using the same architecture (Figure 5(b)). DNN is constructed by a sequence of fully connected linear layers followed by ReLU activation. We train DQN using the Adam optimizer with the Huber loss (Mnih et al., 2015).

In Adam optimizers (Kingma & Ba, 2014) we used hyperparameters $\beta_1 = 0.9, \beta_2 = 0.999$.

In the policy evaluation scenario (Multi-reward maze) we used DQN as part of generator structure (4.1), where in the exploration (Climber) scenario it was used for policy improvements in Algorithm 2.

In the continuous control domains we used TRPO (Schulman et al., 2015) with Normally distributed policy. Policy mean is state-dependent and represented by a DNN. We used the Rllab framework (Duan et al., 2016) to run experiments.

In the SwimmerGather scenario, the agent should collect "apples" while avoiding "bombs", gaining positive or negative reward, respectively. We used the reward signal $\tilde{r} = \begin{bmatrix} r_{\text{apples}} \\ r_{\text{bombs}} \end{bmatrix} \in \mathbb{R}^2$ (where $r_{apples} = 1$ when apple is collected and 0 otherwise, and similarly for bombs) as input to BellmanGAN. Pay attention that policy improvement is still considered w.r.t the scalar reward signal.

Test-specific parameters are stated below.

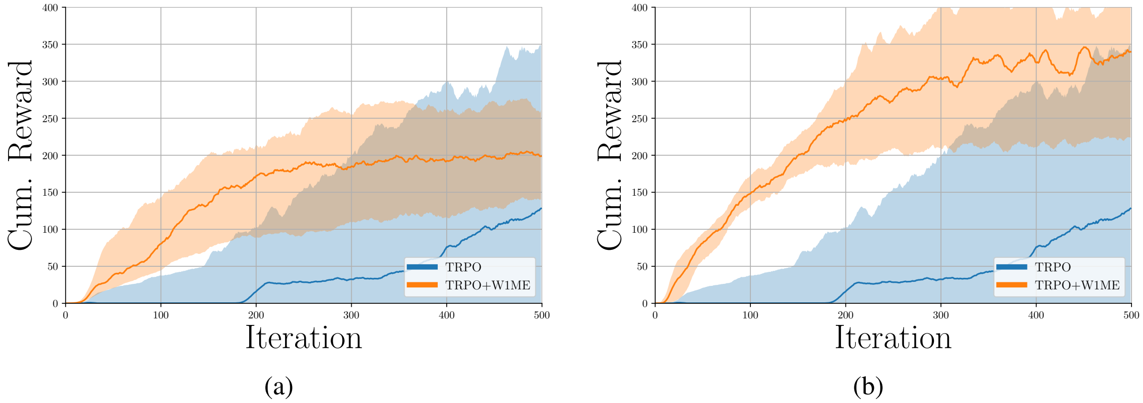


Figure 4. Evaluation of reward-systematic exploration (Section 6.1) on Cartpole Swingup sparse task. (a) W-1ME using scalar settings (without 6.1). (b) W-1ME exploration where 6.1 is applied. We can see that under both settings W-1ME outperforms plain TRPO, where reward-systematic exploration improves median performance. This shows the contribution of Section 6.1 to performance.

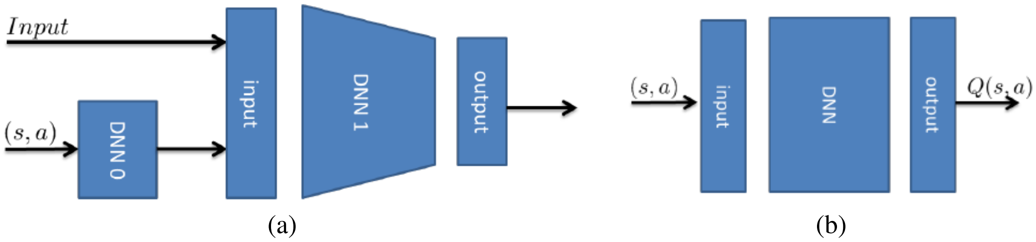


Figure 5. NN architecture
(a) discriminator and generator, and (b) DQN

E.1. Multi-Reward Maze

- DDQN:
  - Layers size: DNN [16,16,16], output [1].
- Generator:
  - Layers size: DNN0 [8,8,8] , DNN1 [128,128], output [8].
  - Activation function: Leaky ReLU, Output activation: Linear.
  - Input noise dim: 8.
- Discriminator:
  - Layers size: DNN0 [8,8,8] , DNN1 [256,128], output [1].
  - Activation function: Leaky ReLU, Output activation: Linear.
- Train parameters: $\lambda = 0.1$, $\gamma = 0.95$, minibatch size 64, learning rate 0.001.

E.2. 2-Face Climber

- DDQN:
  - Layers size: DNN [16,16,16], output [1].
- Generator:
  - Layers size: DNN0 [4,4,4] , DNN1 [128,64], output [2].
  - Activation function: Leaky ReLU, Output activation: Linear.
  - Input noise dim: 2.
- Discriminator
  - Layers size: DNN0 [4,4,4] , DNN1 [256,256,16], output [1].
  - Activation function: Leaky ReLU, Output activation: Linear.
- Train parameters: $\lambda = 0.1$, $\gamma = 0.9$, minibatch size 64, learning rate 0.0001.
- Exploration parameters: $N_{explore} = 4$, $T = 32$.

E.3. Continuous Control Tasks

- Normally distributed policy, with state-dependent μ represented by a DNN with one hidden layer of size 32 (LQR and CartpoleSwingup), or two hidden layers of sizes [63,32] (SwimmerGather), with tanh activation.
- Horizon: $T = 200$ for LQR, $T = 500$ for sparse tasks.

- Generator:
  - Layers size: DNN1 [32].
  - Output size: [65] for LQR, [5] for CartpoleSwingup, [35] for SwimmerGather.
  - Activation function: Leaky ReLU, Output activation: Linear.
  - Input noise dim: [65] for LQR, [5] for CartpoleSwingup, [35] for SwimmerGather.
- Discriminator:
  - Layers size: DNN1 [64], output [1].
  - Activation function: Leaky ReLU, Output activation: Linear.
- Train parameters: $\lambda = 0.1$, $\gamma = 0.99$, minibatch size 10 (500 minibatches per iteration), learning rate 0.0001.
- Exploration parameters: $N_{explore} = 10$.

References

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continous control. *ICML*, 2016.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., Graves, A., Riedmiller, M., Fidjeland, A., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Müller, A. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29 (2):429–443, 1997.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, pp. 5. Phoenix, AZ, 2016.

Villani, C. *Optimal transport: old and new*. Springer Science & Business Media, 2008.