Dimension-Wise Importance Sampling Weight Clipping for Sample-Efficient Reinforcement Learning

Seungyul Han [1] **Youngchul Sung** [1]

Abstract

In importance sampling (IS)-based reinforcement learning algorithms such as Proximal Policy Optimization (PPO), IS weights are typically clipped to avoid large variance in learning. However, policy update from clipped statistics induces large bias in tasks with high action dimensions, and bias from clipping makes it difficult to reuse old samples with large IS weights. In this paper, we consider PPO, a representative on-policy algorithm, and propose its improvement by dimension-wise IS weight clipping which separately clips the IS weight of each action dimension to avoid large bias and adaptively controls the IS weight to bound policy update from the current policy. This new technique enables efficient learning for high action-dimensional tasks and reusing of old samples like in off-policy learning to increase the sample efficiency. Numerical results show that the proposed new algorithm outperforms PPO and other RL algorithms in various Open AI Gym tasks.

1. Introduction

Many deep reinforcement learning (RL) algorithms aim to maximize the discounted return for a given environment by a parameterized policy (Mnih et al., 2013; 2015; Lillicrap et al., 2015; Schulman et al., 2015a). For learning continuous action tasks, which is the focus of this paper, many algorithms are based on policy gradient (PG), which updates the policy parameter to the direction of gradient decent of an objective function (Sutton et al., 2000). PG methods can be classified into on-policy PG and off-policy PG. On-policy PG updates the target policy from the samples generated by the target policy only (Schulman et al., 2015a; Peters et al., 2005), whereas off-policy PG updates the target policy from samples generated by the behavior policy which can be different from the target policy. Off-policy PG methods has the advantage that they can reuse old sample batches and set the behavior policy to better explore state and action spaces, and thus many state-of-the-art algorithms are based on off-policy PG. Current off-policy PG methods can be divided into several types. Value-based PG methods update the policies to maximize the Q-value statistics estimated by TD errors (Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2017) and IS-based PG methods calibrate the statistics differences between the target policy and the behavior policy (Degris et al., 2012; Wang et al., 2016; Levine & Koltun, 2013). Due to the advantages of off-policy learning, reusing off-policy data in on-policy learning is recently considered (Gu et al., 2016; 2017; Nachum et al., 2017; Han & Sung, 2017).

In this paper, we consider PPO (Schulman et al., 2017), which is a widely-considered stable and fast-converging on-policy algorithm, and propose its improvement by adopting several new features. PPO uses a clipped objective function to bound policy update from the current policy for stable learning. However, due to its structure, the gradient of clipped samples completely vanishes and this causes sample inefficiency in high action-dimensional tasks. In order to solve this problem, we propose *dimension-wise importance sampling weight clipping (DISC)*, which clips the IS weight of each action dimension instead of clipping the overall likelihood ratio as in PPO. With DISC, the gradient of samples does not completely vanish for most samples since there exist unclipped dimensions, and this increases the sample efficiency. In order to further enhance the sample efficiency, we propose reusing old samples generated at the previous iterations like in off-policy learning. PPO uses Generalized Advantage Estimator (GAE) (Schulman et al., 2015b) to estimate the advantage of the on-policy n-step trajectory, but GAE has difficulty in estimating from the old trajectories generated at the previous iterations. To circumvent this difficulty, we consider V-trace (Espeholt et al., 2018) that estimates target values from off-policy trajectories and combine GAE with V-trace to estimate the advantage in DISC. We evaluate our algorithm on various OpenAI GYM tasks (Brockman et al., 2016) and numerical results show that DISC outperforms the PPO baseline and other state-of-

[1]School of Electrical Engineering, KAIST, Daejeon, South Korea. Correspondence to: Youngchul Sung <ycsung@kaist.ac.kr>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

the-art continuous RL algorithms.

Notations: $\mathbb{E}[\cdot]$ denotes expectation, $\text{sgn}(\cdot)$ is the sign function of $(\cdot)$, and $x \sim P$ means that random sample x is generated from distribution P. $\hat{X}$ denotes the empirical estimation of random variable X. $\mathcal{N}(\mathbf{u}, \mathbf{\Sigma})$ represents Gaussian distribution with mean vector $\mathbf{u}$ and covariance matrix $\mathbf{\Sigma}$.

2. Related Works

IS-based PG: Calibration based on IS weight has been considered in various off-policy PG methods for unbiased objective function estimation since the target policy and the behavior policy are different in off-policy RL (Meuleau et al.; Shelton, 2001; Peshkin & Shelton, 2002; Jie & Abbeel, 2010). Large IS weights cause large variances and hence truncation of IS weights is proposed to prevent large IS weights (Wawrzyński, 2009). However, truncation induces large bias. The off-policy actor-critic algorithm (Degris et al., 2012) reduces variance by applying per-decision IS (not the whole trajectory) in actor-critic. The actor-critic with experience replay (ACER) algorithm further applies bias correction in addition to truncated IS of actor-critic to use off-policy data with low variance (Wang et al., 2016).

Value estimation: One can estimate the Q-function with 1-step TD error which can be used for both on-policy and off-policy. In addition, there exist many n-step value estimation methods to improve accuracy. For on-policy PG, TD(λ) (Sutton, 1988), Least-Squares TD(λ) (Bradtke & Barto, 1996), GAE (Schulman et al., 2015b) are commonly used. For off-policy PG, a n-step trajectory is generated from the behavior policy, so the path statistic should be calibrated to evaluate the target policy. Thus, IS weight is applied in off-policy evaluation: (Precup, 2000) suggested using per-decision IS weight, and Retrace (λ) (Munos et al., 2016) and V-trace (Espeholt et al., 2018) evaluate the policy by using the truncated trace for low-variance target value estimation.

Extension of on-policy PG using off-policy data: In on-policy learning, the target policy and the behavior policy are the same and hence samples generated by the target policy can be used only at the current iteration (Schulman et al., 2015a). Recently, in order to improve the sample efficiency of on-policy learning, using off-policy data to update on-policy gradient is proposed (Gu et al., 2016; 2017). In particular, Trust-PCL (Nachum et al., 2017) applies path consistency learning to use off-policy data while maintaining the stability of trust region policy optimization.

Extension of PPO: There exist several improvements of PPO, e.g., distributed optimization for enhancing the performance of large scale RL (Heess et al., 2017), reusing old sample batches with small IS weights (Han & Sung, 2017), and reducing variance by using action-dependent stein control variate (Liu et al., 2017).

3. Set Up and Background

We assume a Makrov decision process (MDP) environment $(\mathcal{S}, \mathcal{A}, P, \gamma, R)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the transition probability, γ is the discount factor, and R is the reward function. The agent has policy $\pi : \mathcal{S} \to \mathcal{A}$ that selects action a_t for given state s_t, and interacts with the environment to generate an episode $(s_t, a_t, R(s_t, a_t))_{t=0}^{T-1}$ of length T. The agent optimizes its policy to maximize the discounted return of the episode $\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)$, and a parameterized policy π_θ is assumed in this paper.

The basic idea of PG is as follows. The objective function of discounted return is defined as a function of the policy parameter θ: $J(\theta) = \mathbb{E}_{\tau_0 \sim \pi_\theta}[\sum_{t=0}^{T-1} \gamma^t r_t]$, where $r_t = R(s_t, a_t)$, and $\tau_t = (s_t, a_t, \cdots, s_{T-1}, a_{T-1})$ is the sample trajectory from time t. The policy gradient theorem (Sutton et al., 2000) gives the gradient of $J(\theta)$ with respect to θ as

$$\nabla_\theta J(\theta) = \mathbb{E}_{s_t \sim \eta_{\pi_\theta}, a_t \sim \pi_\theta}[A_{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t|s_t)],$$

where η_{π_θ} is the discounted state visitation frequency, and $A_{\pi_\theta}(s_t, a_t) = Q_{\pi_\theta}(s_t, a_t) - V_{\pi_\theta}(s_t)$ is the advantage function with the state-action value function $Q_{\pi_\theta}(s_t, a_t) = \mathbb{E}_{\tau_t \sim \pi_\theta}[\sum_{t=i}^{T-1} \gamma^{t-i} r_t]$ and the state value function $V_{\pi_\theta}(s_t) = \mathbb{E}_{a_t, \tau_{t+1} \sim \pi_\theta}[\sum_{t=i}^{T-1} \gamma^{t-i} r_t]$. Then, the policy parameter θ can be optimized by stochastic gradient descent to the direction $\nabla_\theta J(\theta)$ to maximize the return. In the case that the behavior policy μ that generates samples is different from the target policy π_θ, the objective function can be calibrated by using importance sampling weight as (Degris et al., 2012)

$$\nabla_\theta J_{\text{off}}(\theta) = \mathbb{E}_{s_t \sim \eta_{\pi_\theta}, a_t \sim \mu}[\rho_t A_{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t|s_t)],$$

where $\rho_t = \frac{\pi_\theta(a_t|s_t)}{\mu(a_t|s_t)}$ is the IS weight between π_θ and μ.

PPO is based on the minorization and maximization (MM) technique for the original objective function $J(\theta)$ and is a widely-considered on-policy PG method that updates the policy π_θ multiple times by sampling mini-batches from the current sample batch. In the beginning of the i-th iteration, the current sample batch $B_i = \{(s_{i,0}, a_{i,0}, r_{i,0}), \cdots, (s_{i,N-1}, a_{i,N-1}, r_{i,N-1})\}$ of length N is generated by π_{θ_i}. Then, it updates the target policy π_θ by sampling mini-batches from B_i for several epochs. Since the target policy π_θ for policy update is different from the policy π_{θ_i} that generated B_i, PPO calibrates the statistics difference between the current target policy π_θ and the policy π_{θ_i} that generated B_i based on the IS weight $\rho_t = \frac{\pi_\theta(a_t|s_t)}{\mu(a_t|s_t)}$ with $\mu = \pi_{\theta_i}$. Furthermore, PPO clips the IS weight to bound the amount of policy update for stable

learning (Schulman et al., 2015a; Wang et al., 2016). Thus, the objective function of PPO is given by

$$\hat{J}_{PPO}(\theta) = \frac{1}{M}\sum_{m=0}^{M-1} \min\{\rho_m \hat{A}_m, \text{clip}_\epsilon(\rho_m)\hat{A}_m\}, \quad (1)$$
$$= \frac{1}{M}\sum_{m=0}^{M-1} \min\{\kappa_m \rho_m, \kappa_m \text{clip}_\epsilon(\rho_m)\}\kappa_m \hat{A}_m,$$

where $\rho_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_i}(a_t|s_t)}$, $\hat{A}_t$ is an estimate of $A_{\pi_{\theta_i}}(s_t, a_t)$, $\text{clip}_\epsilon(\cdot) := \text{clip}(\cdot, 1-\epsilon, 1+\epsilon)$ is the clip function, $\kappa_m = \text{sgn}(\hat{A}_m)$, and the M samples of each mini-batch are randomly sampled from B_i. For empirical estimation of $A_{\pi_{\theta_i}}$, PPO uses GAE (Schulman et al., 2015b). GAE estimates the advantage as $\hat{A}_t = \sum_{l=t}^{T-1}(\gamma\lambda)^{l-t}\delta_l$, where $\delta_t = r_t + \gamma V_{w_i}(s_{t+1}) - V_{w_i}(s_t)$ is the temporal difference (TD) residual, V_w is the parameterized value function that approximates V_{π_θ}, and w_i is the value parameter at the i-th iteration. The value parameter w is updated to minimize the mean square error (MSE) between V_w and the target value $\hat{V}_t = \hat{A}_t + V_{w_i}(s_t)$, given by $\hat{J}_V(w) = \frac{1}{M}(V_w(s_m) - \hat{V}_m)^2$. GAE can be computed recursively as $\hat{A}_t = \delta_t + (\gamma\lambda)\hat{A}_{t+1}$ in a similar way to that TD(λ) backup performs value estimation (Sutton & Barto, 1998).

4. Dimension-Wise Clipping

Note that in the PPO objective function (1), ρ_t is a function of the optimization variable θ but $\hat{A}_t$ is fixed for the given behavior policy π_{θ_i} which generated the current sample batch B_i. Hence, on average, maximization of the cost (1) with respect to θ increases ρ_t if $\hat{A}_t > 0$, and decreases ρ_t if $\hat{A}_t < 0$. For stable update, PPO limits the amount of the policy update by clipping the objective function unlike TRPO restricts the policy update by using a KL divergence constraint. This clipping prevents ρ_t from becoming too large or too small, and results in stable update for many environments. The clipped objective function of PPO $\hat{J}_t = \min\{\rho_t\hat{A}_t, \text{clip}_\epsilon(\rho_t)\hat{A}_t)\}$ reduces to a constant $\hat{J}_t = (1+\epsilon)\hat{A}_t$ when $\hat{A}_t > 0$ and $\rho_t > 1+\epsilon$, and to a constant $\hat{J}_t = (1-\epsilon)\hat{A}_t$ when $\hat{A}_t < 0$ and $\rho_t < 1-\epsilon$. Therefore, the gradient vanishes for such samples. This vanishment of gradient can cause a serious problem especially in high action-dimensional tasks. In order to assess the amount of gradient vanishment, we consider the average shifted deviation of the IS weight from one $\rho'_t := |1-\rho_t| + 1$[1] and the corresponding amount of gradient vanishment, which are shown in Fig. 1. Here, the average ρ'_t at the i-th iteration is defined as the average of IS weight deviation ρ'_t of each mini-batch averaged again over all drawn mini-batches at the last epoch and the the amount of gradient vanishment is defined as the ratio of the samples with clipped IS weights to the total samples at the last epoch. It is seen that for the low action-dimensional tasks such as Hopper and HalfCheetah whose action dimensions are 3 and 6, respectively, average ρ'_t is below the clipping factor of PPO (the black line) and the amount of gradient vanishment is not significant ($15 \sim 20\%$). However, for the high action-dimensional tasks such as Humanoid whose action dimension is 17, average ρ'_t exceeds the clipping line (the reason will be explained shortly) and the corresponding amount of gradient vanishment is too large (above 60%). Hence, clipping the loss function directly and the corresponding zero gradient samples in PPO reduce sample efficiency for high action-dimensional tasks.

[1] To measure how much IS weight is far from 1, we consider $\rho'_t := |1-\rho_t| + 1$.

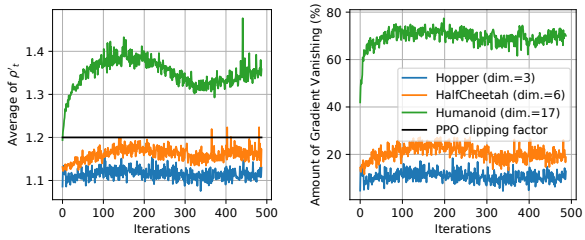


Figure 1: (left) ρ'_t and (right) the amount of gradient vanishment

4.1. Dimension-Wise Clipping

For learning continuous action tasks, many RL algorithms use independent Gaussian distribution $\pi_\theta(\cdot|s_t) = \mathcal{N}(\mathbf{u}, \sigma^2\mathbf{I})$ as the stochastic policy, where $\mu = (u_0, u_1, \cdots, u_{D-1})$ is the mean vector, D is the action dimension, and $\mathbf{I}$ is an identity matrix. Then, the policy can be factorized into the action dimensions as $\pi_\theta(a_t|s_t) = \prod_{d=0}^{D-1}\pi_{\theta,d}(a_{t,d}|s_t)$, where $\pi_{\theta,d}(\cdot|s_t) \sim \mathcal{N}(\mu_d, \sigma^2)$ and $a_{t,d}$ is the d-th element of a_t. Suppose that the amount of policy distribution change by update is similar for each dimension. Then, the variation of π_θ grows exponentially as D increases. This yields large IS weights for high action-dimensional tasks, as seen in the case of Humanoid in Fig. 1. In order to prevent large IS weights in high action-dimensional tasks while maintaining the advantage of IS weight clipping, we propose a new objective function for policy optimization, which clips the IS weight of each dimension separately. That is, using the fact that the IS weight can be factorized as

$$\rho_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_i}(a_t|s_t)} = \prod_{d=0}^{D-1}\frac{\pi_{\theta,d}(a_{t,d}|s_t)}{\pi_{\theta_i,d}(a_{t,d}|s_t)} =: \prod_{d=0}^{D-1}\rho_{t,d},$$

we clip the IS weight $\rho_{t,d}$ of each dimension as $\text{clip}_\epsilon(\rho_{t,d})$. In addition, in order to prevent the IS weight from being too far from 1, we include an additional loss for IS weight

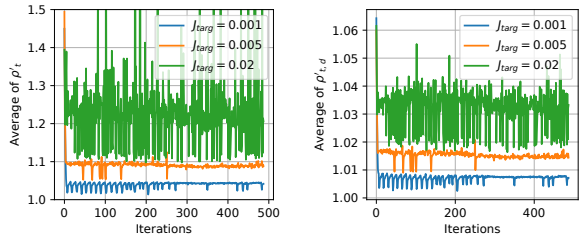


Figure 2: (left) average ρ'_t and (right) average $\rho'_{t,d}$ (averaged over all samples and all dimensions) of DISC for Humanoid

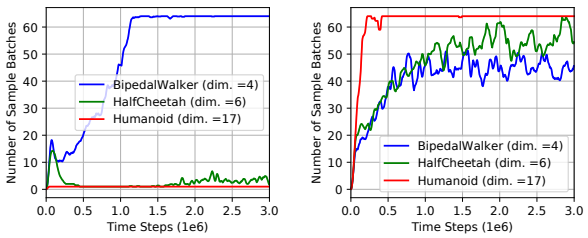


Figure 3: The number of reused sample batches: (left) PPO-AMBER (Han & Sung, 2017) and (right) DISC

control: $J_{IS} = \frac{1}{2M}\sum_{m=0}^{M-1}(\log(\rho_m))^2$. Thus, our objective function for DISC is given by

$$\hat{J}_{DISC}(\theta) = \frac{1}{M}\sum_{m=0}^{M-1}\left[\prod_{d=0}^{D-1}\min\{\kappa_m\rho_{m,d}, \kappa_m \text{clip}_\epsilon(\rho_{m,d})\}\right]\kappa_m\hat{A}_m - \alpha_{IS}J_{IS}, \tag{2}$$

where $\kappa_m = \text{sgn}(\hat{A}_m)$, and the weighting factor α_{IS} is adaptively determined in a similar way of adapting the weighting factor for the KL divergence in PPO (Schulman et al., 2017):

$$\begin{cases} \alpha_{IS} \leftarrow \alpha_{IS}/2 & \text{if } J_{IS} < J_{targ}/1.5 \\ \alpha_{IS} \leftarrow \alpha_{IS} \times 2 & \text{if } J_{IS} > J_{targ} \times 1.5 \end{cases} \tag{3}$$

DISC solves the gradient vanishment problem of PPO by dimension-wise clipping. Even though $\min\{\kappa_m\rho_{m,d}, \kappa_m\text{clip}_\epsilon(\rho_{m,d})\}$ becomes constant from clipping for some d, there exist other dimensions that are not clipped for most samples. Thus, we can update the policy by the gradient from the unclipped objective function unless $\rho_{t,d}$ are clipped for all d. Fig. 2 shows average ρ'_t and average $\rho'_{t,d} := |1-\rho_{t,d}|+1$ of DISC for $J_{targ} = 0.0001,\ 0.0002,\ 0.0005$ for the Humanoid task, and the corresponding amount of gradient vanishment is almost zero. Note that average overall ρ'_t of DISC is much smaller as compared to the Humanoid curve in Fig. 1(left) especially for small $J_{targ} < 0.005$. Furthermore, average $\rho'_{t,d}$ for each dimension is much less than ρ'_t, as expected. Hence, dimension-wise IS clipping efficiently reduces gradient vanishment as well as increases sample efficiency.

4.2. Reusing Old Sample Batches

In order to further enhance sample efficiency, we reuse old sample batches. The experience replay buffer $\mathbf{R}$ stores old sample batches $B_{i-1}, \cdots, B_{i-L+1}$ in addition to the current sample batch B_i, where L is the maximum number of old sample batches. If $L = 1$, then the algorithm is on-policy, and otherwise ($L > 1$), it is off-policy. We sample mini-batches multiple times in the whole buffer $\mathbf{R}$. However, if the IS weights of samples in the old batches deviate too much from 1, then the weights will be clipped and it is sample inefficient. Thus, we follow (Han & Sung, 2017), which considers reusing old sample batches only if the IS weights of samples in the old batches does not deviate too much from 1. That is, if for a given old batch

$$\frac{1}{N}\sum_{t=0}^{N-1}\rho'_t < 1+\epsilon_b, \tag{4}$$

where ϵ_b is a parameter determining batch inclusion and N is the number os samples in the batch, then the old batch is reused. However, in our DISC, we slightly modify the inclusion criterion and reuse old sample batches satisfying the condition

$$\frac{1}{ND}\sum_{t=0}^{N-1}\sum_{d=0}^{D-1}\rho'_{t,d} < 1+\epsilon_b. \tag{5}$$

Here, we reuse old sample batches only for computation of the first term in the right-hand side (RHS) of (2) not for the second term J_{IS}. J_{IS} is computed by using the on-policy batch B_i to implement policy update near the current policy.

Fig. 3 shows the number of reused sample batches of PPO-AMBER (Han & Sung, 2017) and DISC for BidepalWalker, HalfCheetah, and Humanoid whose action dimensions are 4, 6, and 17, respectively. PPO-AMBER selects more sample batches based on (4) for lower action-dimensional tasks to avoid too much clipping from reusing samples, and it is ineffective for high action-dimensional tasks such as Humanoid, as seen in Fig. 3. Thus, it cannot use off-policy sample batches for high action-dimensional tasks. On the other hand, DISC can exploit more old sample batches for higher action-dimensional tasks since it determines old batch inclusion based on $\rho'_{t,d}$ which is not too big even in Humanoid. Old batch inclusion of DISC matches the purpose of experience replay of using more samples for better exploration in high action-dimensional tasks.

4.3. GAE with V-trace

PPO computes the GAE $\hat{A}_t$ and the corresponding target value $\hat{V}_t$ to evaluate the current policy by using the on-policy

sample batch B_i at the beginning of iteration once. Thus, at the policy evaluation step, the evaluated policy is the same as the behavior policy, and PPO does not need IS calibration. However, if we use old sample batches generated from the previous policies $\pi_{\theta_{i-1}}, \pi_{\theta_{i-2}} \cdots$ which are different from the current behavior policy π_{θ_i}, we need to apply IS calibration even for evaluating the current policy from old samples for low variance and safe learning, like in Retrace(λ) (Munos et al., 2016) or V-trace (Espeholt et al., 2018). In particular, V-trace evaluates the multi-step target value from an off-policy trajectory as

$$\hat{V}_t = V_{w_i}(s_t) + \sum_{l=t}^{T-1} \left(\prod_{i=t}^{l} c_i \right) \gamma^{l-t} \delta_l, \tag{6}$$

where $c_t = \lambda \min(1.0, \rho_t)$ is the truncated IS weight (Espeholt et al., 2018). It is proven that the V-trace operator is a contraction mapping (Espeholt et al., 2018). Note that V-trace becomes the GAE target value in the on-policy case since $\rho_t = 1$. Thus, in order to estimate the advantages of the previous trajectories $B_{i-1}, B_{i-2}, \cdots$ in DISC, we combine GAE with V-trace (GAE-V), given by

$$\hat{A}_t = \sum_{l=t}^{T-1} \left(\prod_{i=t+1}^{l} \min(1.0, \rho_i) \right) (\gamma\lambda)^{l-t} \delta_l, \tag{7}$$

and its target value $\hat{V}_t = \min(1.0, \rho_t)\hat{A}_t + V_{w_i}(s_t)$ is just the V-trace target value. Combining the new features, the final DISC algorithm is summarized in Algorithm 1.

Algorithm 1 DISC

```
Initialize parameters α_IS ← 1, θ, w,
for each iteration do
  θ_i ← θ, w_i ← w
  Sample on-policy trajectory B_i of size N from π_{θ_i}
  Store B_i in replay buffer R
  Compute GAE-V target Â_t, V̂_t by using all off-policy
  trajectories B_i, ··· , B_{i−L+1} in R
  for each epoch do
    for each gradient step do
      Sample mini-batch of size M from the sample
      batches satisfying (5) in R
      Compute the DISC objective function Ĵ_DISC(θ)
      Compute the value loss Ĵ_V(w)
      θ ← θ + β∇_θ Ĵ_DISC(θ)
      w ← w + β∇_w Ĵ_V(w)
    end for
  end for
  Update α_IS as (3)
end for
```

5. Experiments

In this section, we provide numerical results and ablation study on our DISC algorithm. The source code for DISC is available at http://github.com/seungyulhan/disc/.

5.1. Simulation Setup

Detailed description of the hyper-parameters of PPO, PPO-AMBER and DISC is provided in Table A.1 in Appendix. We basically followed the hyper-parameter setup of PPO in the Open AI baseline (Dhariwal et al., 2017). In PPO, the learning rate β anneals from 0.0003 to 0, but we set a minimum learning rate as 0.0001 to prevent convergence to local optimum. We set the mini-batch size M = (the number of sample batches/gradient steps per epoch), where the number of sample batches is computed based on (4) for PPO-AMBER and (5) for DISC at each iteration. Note that we unify the hyper-parameter setup for PPO, PPO-AMBER and DISC, and only consider the single actor case for all continuous tasks for fair comparison. The hyper-parameters of all other algorithms is explained in Section A in Appendix.

We used deterministic evaluation based on 10 episodes generated by the corresponding deterministic policy (i.e., the mean network $\mathbf{u}$ of the Gaussian target policy π_{θ_i}) at the time of evaluation, i.e., at the end of each iteration. We averaged the performance over 5 seeds and used the moving average window of 10 iterations for figures. The solid line and shaded part in each episode reward figure represent the average episode reward sum and its standard deviation for all seeds, respectively. In the max average return tables, the bold value is the best max average return among the compared algorithms and $\pm$ means its standard deviation.

5.2. Comparison with the PPO Baseline

First, we compared DISC with two PPO baselines: PPO whose loss is described as (1) (Schulman et al., 2017) and PPO-AMBER (Han & Sung, 2017) which simply uses old sample batches as (4) and uses GAE for advantage estimation. For PPO-AMBER, we consider two cases with different batch inclusion factors: $\epsilon_b = 0.1,\ 0.2$. We evaluated the three algorithms on Mujoco simulator (Todorov et al., 2012) and Box2d engine (Catto, 2011) in OpenAI GYM continuous control tasks (Brockman et al., 2016) which are challenging environments with continuous action spaces, as described in Fig. 4. The dimensions of state and action spaces of each task are given in Table A.2 in Appendix.

Fig. 5 shows the average episodic reward sum of DISC and the two baseline algorithms, and Table 1 shows the corresponding maximum average return. Note that DISC significantly outperforms PPO and PPO-AMBER for all continuous action tasks. Note that PPO works in low-dimensional

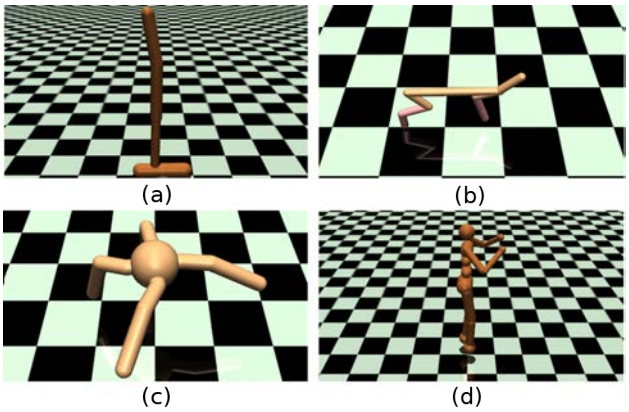


Figure 4: Examples of Open AI GYM continuous control tasks: (a) Hopper-v1 (b) HalfCheetah-v1 (c) Ant-v1 (d) Humanoid-v1

tasks such as Hopper and BipedalWalker whose action dimensions are 3 and 4, respectively, but it does not learn for high action-dimensional tasks such as Ant and Humanoid whose action dimensions are 8 and 17, respectively. This is because the gradient of most samples vanishes for these high action-dimensional tasks, as explained in Section 4.1. Furthermore, this gradient vanishment problem makes it difficult to reuse old samples and hence using experience replay for PPO-AMBER does not work in high action-dimensional tasks. On the other hand, DISC effectively solves the gradient vanishment problem, as described in Section 4.2. Hence, the performance gain of DISC over PPO is significant in these high dimensional difficult tasks. PPO with large batch or distributed system also works well in Humanoid task (Heess et al., 2017), but it requires hyperparameter adjustment for each environment. Here, we did not adjust the hyperparameters of PPO, PPO-AMBER and DISC for each task.

5.3. Ablation Study

In this section, we provide in-depth ablation study on the important components and several hyper-parameters of DISC: GAE-V, the clipping factor ϵ, the IS target factor J_{targ} in (3), and the batch inclusion constant ϵ_b in (5) based on Mujoco tasks. With this ablation study, we investigate the impact of the key elements of DISC and the sensitivity of the associated hyper-parameters. Fig.6 shows the ablation study result on the Humanoid-v1 environment, and the ablation study results on other environments (Ant-v1, HalfCheetah-v1, Hopper-v1, and Walker2d-v1) are given in Appendix B. For the ablation study of each hyper-parameter, all other hyper-parameters are fixed as the values in Table A.1 in Appendix. The best value for each hyper-parameter is represented in orange color in the figures and is the hyper-parameter value in Table A.1 in Appendix.

GAE-V vs. GAE: In DISC, we combine GAE with V-trace to incorporate the statistics difference of the previous sample trajectories. Fig. 6a shows the Humanoid performance of DISC with GAE and DISC with GAE-V. It is seen that the impact of GAE-V over GAE for DISC is noticeable. For PPO, GAE-V reduces to GAE because it uses only the current trajectory ($\rho_t = 1$). However, DISC uses old sample batches in the replay buffer and their statistics are different from that of the current policy. Therefore, GAE-V for advantage estimation is beneficial for the current policy evaluation from the old trajectories.

Clipping Factor ϵ: The clipping parameter ϵ controls clipping of the IS weight of each dimension $\rho_{t,d}$. If $\rho_{t,d} > 1+\epsilon$ and $\hat{A}_t > 0$ (or $\rho_{t,d} < 1-\epsilon$ and $\hat{A}_t < 0$), then the gradient of dimension d becomes zero. Fig. 6b shows the performance of DISC for several clipping factors: $\epsilon = 0.1,\ 0.2,\ 0.4,\ 0.8$. When the clipping factor is too low, the number of clipped samples is too large, and this slows down learning. On the other hand, when the clipping factor is too large, it will update the policy too much with large IS weights and this causes unstable learning. Thus, DISC with a large clipping factor converges fast in the initial stage of learning, but degrades at the final performance. Although the results for other tasks are not shown, simulations show that the best clipping factor for DISC is around 0.4, as seen in Fig. 6b. Note that the performance of DISC is sensitive when the clipping factor is too small. This means that reducing the clipped samples is important to enhance the performance.

IS Loss Target Parameter J_{targ}: In Fig. 2, we see that the average IS weight deviation from one for each dimension $\rho'_{t,d}$ decreases as the IS target parameter J_{targ} decreases. Note that J_{IS} has a similar role as the KL divergence loss $J_{KL} = D_{KL}(\pi_{\theta_i}||\pi_\theta)$ in some PPO variants, which bounds policy update from the current policy point. In order to see the difference of J_{IS} and J_{KL}, we ran DISC with J_{IS} and DISC with J_{KL} for several values of J_{targ}, and the result is shown in Fig. 6c. It is seen that J_{IS} is much more effective than J_{KL} for Humanoid, which has high action dimensions. Although it is not shown here, we observe that J_{KL} works reasonably well for low action-dimensional environments such as HalfCheetah, or Hopper environment, as seen in Fig. B.3 in Appendix, whereas J_{IS} works well in all environments.

Fig. 7a shows the number of sample batches included for DISC with J_{IS} based on (5) for several values of J_{targ} with a fixed ϵ_b. As J_{targ} increases, the number of included sample batches decreases and the performance degrades, as seen in Fig 6c. Large J_{targ} means large policy update step and hence the speed of learning is faster at the beginning of learning. However, it causes larger IS weights and it reduces the number of sample batches satisfying (5). As result, it

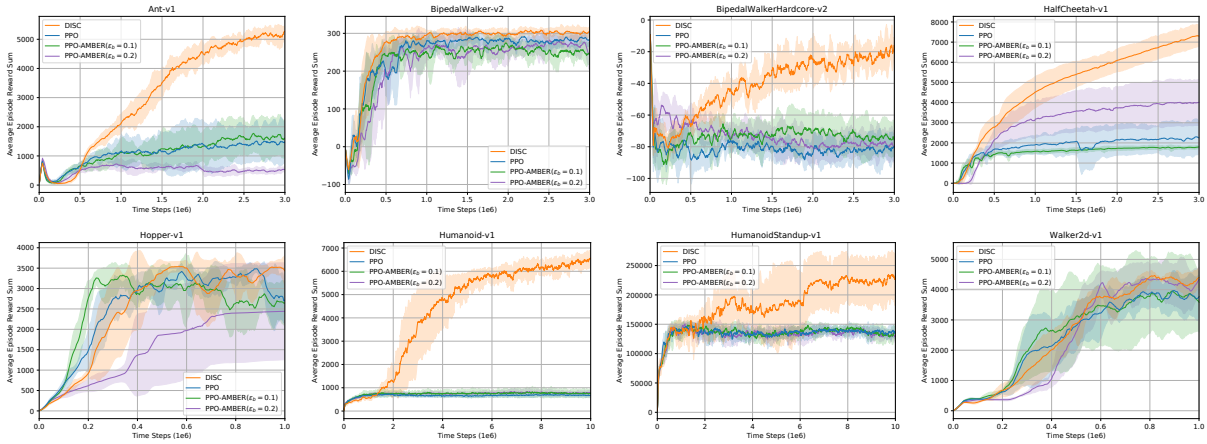


Figure 5: Performance comparison on Open AI GYM continuous control tasks

	DISC	PPO	PPO-AMBER ($\epsilon_b = 0.1$)	PPO-AMBER ($\epsilon_b = 0.2$)
Ant-v1	$\mathbf{5469.04 \pm 283.62}$	1628.96	1935.02	831.09
BipedalWalker-v2	$\mathbf{314.29 \pm 5.16}$	**302.90**	297.24	294.20
BipedalWalkerHardcore-v2	$\mathbf{8.89 \pm 25.81}$	-62.11	-54.67	-45.90
HalfCheetah-v1	$\mathbf{7413.89 \pm 635.30}$	2342.75	1814.84	4073.58
Hopper-v1	$\mathbf{3570.40 \pm 227.53}$	**3571.22**	**3498.32**	2445.57
Huamnoid-v1	$\mathbf{6705.12 \pm 313.44}$	821.30	917.44	901.33
HumanoidStandup-v1	$\mathbf{246435.89 \pm 40387.30}$	154048.51	153212.20	154163.24
Walker2d-v1	$\mathbf{4769.96 \pm 142.25}$	4202.48	4168.16	**4654.61**

Table 1: Max average return of DISC and baseline algorithms

reduces the sample efficiency of DISC and degrades the performance. Hence, small J_{targ} is preferred for DISC.

Batch Inclusion Parameter ϵ_b: In order to reduce bias for DISC, we only include the old sample batches whose average $\rho'_{t,d}$ is less than $1 + \epsilon_b$, as seen in (5). Here, ϵ_b is chosen to be smaller than the clipping factor ϵ.

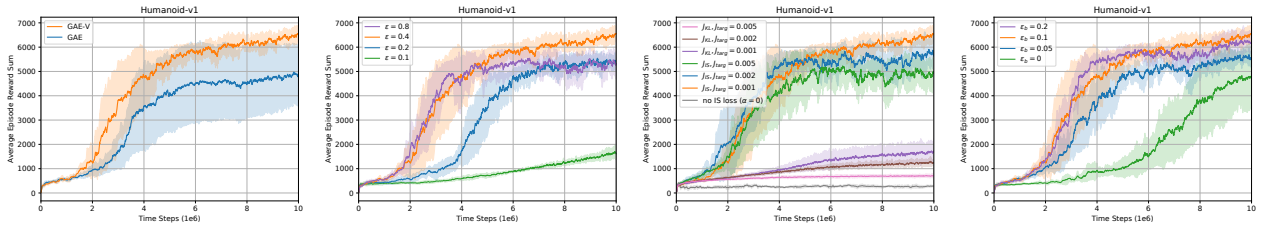
Fig. 7b shows that the average of $\rho'_{t,d}$ of the latest 4 sample batches: B_i, B_{i-1}, B_{i-2}, B_{i-3} for each iteration i for $\epsilon_b = 0.1$, $\epsilon = 0.4$, and $J_{targ} = 0.001$. As expected, the average $\rho'_{t,d}$ is larger for older sample batches since the statistics of older samples deviate more from the current sample static.

Fig. 6d shows the performance of DISC for several values of the batch inclusion parameter: $\epsilon_b = 0,\ 0.05,\ 0.1,\ 0.2$, where $\epsilon_b = 0$ means that we use only the current sample batch B_i for policy update. It is seen that small $\epsilon_b < 0.1$ degrades the performance because DISC does not reuse old sample batches much and this reduces sample efficiency. Especially, DISC with $\epsilon_b = 0$ learns much slowly and this shows the importance of reusing old sample batches. On the other hand, DISC with too large ϵ_b also degrades the performance because it uses too old sample batches that induce bias from clipping. So, there exists a soft spot for ϵ_b. We observe that $\epsilon_b = 0.1$ works well for most tasks.

Consistency : From the above ablation studies, we suggest the best hyper-parameter setup for DISC in Table A.1 in Appendix. Fortunately, the best parameter setup for DISC does not depend much on the environment. DISC has the consistent roughly best hyper-parameter setup for most environments, as seen in Fig. B.4 in Appendix, while some of the recent state-of-the-art RL algorithms require a different hyper-parameter setup for each environment.

5.4. Comparison to Other Continuous RL Algorithms

In this subsection, we provide the performance comparison between DISC and other state-of-the-art on-policy and off-policy RL algorithms. We compare DISC with the following state-of-the-art RL algorithms: Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2015) which applies deep Q-learning to the deterministic policy gradient, Trust Region Policy Optimization (TRPO) (Schulman et al., 2015a) which uses a KL divergence constraint to stably update the policy in the trust region, Actor Critic using Kronecker-factored Trust Region (ACKTR) (Wu et al., 2017) which

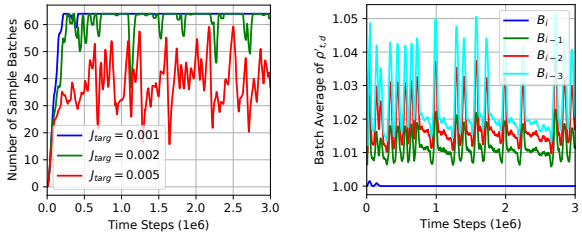


(a) GAE-V vs. GAE (b) Clipping factor ϵ (c) IS target parameter J_{targ} (d) Batch inclusion parameter ϵ_b

Figure 6: Ablation study results on Humanoid-v1

	DISC	DDPG	TRPO	ACKTR	Trust-PCL	SQL	TD3	SAC
Ant	**5469.04**	−6.87	1562.98	3015.22	**5482.45**	2802.18	**5508.08**	**5671.21**
HalfCheetah	7413.89	4020.33	2394.03	3678.57	5597.58	6673.42	11244.30	**14817.63**
Hopper	**3570.40**	729.23	2662.36	3004.15	3073.03	2432.42	2942.88	3322.59
Huamnoid	**6705.12**	857.98	1420.34	4814.80	138.46	5010.72	63.33	**6883.53**
HumanoidS	**246435.89**	142220.05	147258.61	109655.30	79492.38	138996.84	58693.87	139513.04
Walker2d	**4769.96**	810.93	2468.22	2350.81	2226.43	2592.78	**4633.84**	3884.05

Table 2: Max average return of DISC and other state-of-art continuous RL algorithms



(a) The num. of sample batches (b) The average of $\rho'_{t,d}$

Figure 7: (a) the number of used sample batches for different J_{targ} and (b) the average of $\rho'_{t,d}$ for samples in the previous 4 sample batches

optimizes both actor and critic by using Kronecker-factored curvature to approximate natural gradient, Trust region with Path Consistency Learning (Trust-PCL) (Nachum et al., 2017) which uses off-policy data in trust region methods by using path consistency learning, Soft Q-Learning (SQL) (Haarnoja et al., 2017) which optimizes energy-based policy by using stein variational gradient descent, Twin Delayed Deep Deterministic Policy Gradient (TD3) (Fujimoto et al., 2018) which reduces Q-function approximation error based on clipped double Q-learning for continuous control, and Soft Actor Critic (SAC) (Haarnoja et al., 2018) which applies energy-based soft policy iteration and evaluation to off-policy actor critic methods.

TRPO and ACKTR are on-policy RL like PPO, and other algorithms are off-policy RL. For DDPG, TRPO, and ACKTR, we implemented the algorithms by following the OpenAI baselines (Dhariwal et al., 2017). For other algorithms, we use the author-provided implementations in Github. Table 2 shows the maximum average return of all the considered algorithms on Mujoco tasks[2], and the results show that DISC has the top-level performance in five tasks out of the six considered tasks. Especially, DISC achieves the highest performance in HumanoidStandup with 17 action dimensions so that other state-of-the-art RL algorithms cannot catch up. Indeed, it is seen that DISC is a stable and good algorithm for high action-dimensional tasks. Detailed learning curves are given in Fig. C.1 in Appendix.

6. Conclusion

In this paper, we have proposed a new continuous action control algorithm, DISC, by applying dimension-wise IS weight clipping together with GAE-V for advantage estimation. The proposed DISC algorithm separately clips the IS weight of each action dimension to solve gradient vanishment problem, and adaptively controls the IS weight to bound policy update from the current policy for stable learning. The proposed method enables efficient learning for high action-dimensional tasks and reusing of old samples like in off-policy learning to increase the sample efficiency. Numerical results show that the proposed new algorithm outperforms PPO and other RL algorithms in various Open AI Gym tasks, and the performance gain by DISC is significant for high action-dimensional tasks, which have been considered difficult tasks for conventional learning. Further works may include the bias/variance analysis of DISC, proof of the convergence of the DISC objective function, and analysis of the IS weight loss J_{IS}.

[2]We skipped Box2D simulation since the parameter setup is not given for some RL algorithms. HumaniodS is the abbreviation of the HumanoidStandup environment.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning(NRF-2017R1E1A1A03070788).

References

Bradtke, S. J. and Barto, A. G. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1-3):33–57, 1996.

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

Catto, E. Box2d: A 2d physics engine for games, 2011.

Degris, T., White, M., and Sutton, R. S. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

Gu, S. S., Lillicrap, T., Turner, R. E., Ghahramani, Z., Schölkopf, B., and Levine, S. Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 3846–3855, 2017.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

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

Han, S. and Sung, Y. Amber: Adaptive multi-batch experience replay for continuous action control. *arXiv preprint arXiv:1710.04423*, 2017.

Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, A., Riedmiller, M., et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Jie, T. and Abbeel, P. On a connection between importance sampling and the likelihood ratio policy gradient. In *Advances in Neural Information Processing Systems*, pp. 1000–1008, 2010.

Levine, S. and Koltun, V. Guided policy search. In *International Conference on Machine Learning*, pp. 1–9, 2013.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Liu, H., Feng, Y., Mao, Y., Zhou, D., Peng, J., and Liu, Q. Sample-efficient policy optimization with stein control variate. *arXiv preprint arXiv:1710.11198*, 2017.

Meuleau, N., Peshkin, L., Kaelbling, L. P., and Kim, K.-E. Off-policy policy search.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

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

Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 1054–1062, 2016.

Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. Trust-pcl: An off-policy trust region method for continuous control. *arXiv preprint arXiv:1707.01891*, 2017.

Peshkin, L. and Shelton, C. R. Learning from scarce experience. *arXiv preprint cs/0204043*, 2002.

Peters, J., Vijayakumar, S., and Schaal, S. Natural actor-critic. In *European Conference on Machine Learning*, pp. 280–291. Springer, 2005.

Precup, D. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, pp. 80, 2000.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1889–1897, 2015a.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shelton, C. R. Importance sampling for reinforcement learning with multiple objectives. 2001.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. The MIT Press, Cambridge, MA, 1998.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1063, 2000.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 5026–5033. IEEE, 2012.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016.

Wawrzyński, P. Real-time reinforcement learning by sequential actor–critics and experience replay. *Neural Networks*, 22(10):1484–1497, 2009.

Wu, Y., Mansimov, E., Grosse, R. B., Liao, S., and Ba, J. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in neural information processing systems*, pp. 5279–5288, 2017.