Proximal Policy Optimization with Adaptive Threshold
for Symmetric Relative Density Ratio

Taisuke Kobayashi*

Division of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan

Abstract

Deep reinforcement learning (DRL) is one of the promising approaches for introducing robots into complicated environments. The recent remarkable progress of DRL stands on regularization of policy, which allows the policy to improve stably and efficiently. A popular method, so-called proximal policy optimization (PPO), and its variants constrain density ratio of the latest and baseline policies when the density ratio exceeds a given threshold. This threshold can be designed relatively intuitively, and in fact its recommended value range has been suggested. However, the density ratio is asymmetric for its center, and the possible error scale from its center, which should be close to the threshold, would depend on how the baseline policy is given. In order to maximize the values of regularization of policy, this paper proposes a new PPO derived using relative Pearson (RPE) divergence, therefore so-called PPO-RPE, to design the threshold adaptively. In PPO-RPE, the relative density ratio, which can be formed with symmetry, replaces the raw density ratio. Thanks to this symmetry, its error scale from center can easily be estimated, hence, the threshold can be adapted for the estimated error scale. From three simple benchmark simulations, the importance of algorithm-dependent threshold design is revealed. By simulating additional four locomotion tasks, it is verified that the proposed method statistically contributes to task accomplishment by appropriately restricting the policy updates.

Keywords:
Reinforcement learning, Deep learning, Policy regularization

1. Introduction

Reinforcement learning (RL) (Sutton and Barto, 2018) is one of the promising methodologies for resolving complicated robot control problems. Remarkable developments have received a lot of attention, especially by grace of the combination with deep neural networks (DNNs) (LeCun et al., 2015) to approximate value and stochastic policy functions, so-called deep RL (DRL) (Silver et al., 2016). The basic learning laws for RL have been established with two directions in a relatively early stage (Sutton and Barto, 2018). One is value-function-based methods, which can find global solutions for the problems with discrete action space. Another is policy-gradient-based methods, which can handle continuous action space although obtain local solutions. This paper focuses on the policy-gradient-based methods for applications such as continuous joint control of robots. Note that these basic learning laws are taken over by DRL.

In the basic research of them, various learning algorithms are being proposed in order to alleviate the numerical instability of the learning process behavior due to nonlinear regression of DNNs; e.g. experience replay (Lin, 1992; Schaul et al., 2015); target network (Mnih et al., 2015; Kobayashi and Ilboudo, 2021); entropy maximization of policy (Haarnoja et al., 2018; Shi et al., 2019); parallelization of learning with asynchronous agents (Mnih et al., 2016); and utilization of learned model (i.e. model-based RL) (Chua et al., 2018; Clavera et al., 2020). These methods, alone or in combination, have stabilized and accelerated the learning of the given tasks by DRL, and are steadily approaching the stage of practical use.

Among them, research on policy regularization (e.g. (Haarnoja et al., 2018; Shi et al., 2019)) has recently been systematized theoretically as policy-regularized RL (Geist et al., 2019), and its importance can be seen to be high. The motivation for the policy regularization comes from the fact that the policy improvement is an indirect optimization through the value function, and it is prone to learning instability; in other words, the variance of learning performance due to random seeds is large. By constraining the update amount and direction of policy, the policy regularization makes the policy smoothly improve and acquire the near-optimal one.

This study further focuses on proximal policy optimization (PPO) (Schulman et al., 2017) as one of the policy regularization methods. The main concept of PPO is proximal update by softly constraining the latest policy to the...
baseline (basically old) one. Although PPO has two versions, one if for soft constraint of Kullback-Leibler (KL) divergence and another is for clipping of density ratio, the latter version is major due to its intuitive and practical implementation while outperforming the first version. In this paper, therefore, PPO refers to the clipping version.

To improve PPO, several methods have been proposed by combining it with other techniques. For example, [Hamalainen et al., 2020] optimized the covariance of policy based on CMA-ES to improve the exploration efficiency. In [Imagawa et al., 2019], an optimistic bonus according to the uncertainty of return is added to facilitate the exploration. A single demonstration has been provided in [Libardi et al., 2021] to accomplish the tasks where rewards are sparsely given. All of these methods utilize PPO to stabilize learning, and then improve learning efficiency by using other methods in combination.

On the other hand, the way to regularize the policy in PPO is recently revisited. In PPO-RB [Wang et al., 2020], the density ratio over the clipping threshold is explicitly reverted to the clipping range for more certain regularization. PPOS [Zhu and Rosendo, 2021] relaxes the regularization in PPO-RB by weakening it according to the degree of exceedance from threshold. In the earlier work of this paper [Kobayashi, 2021a], a new regularization method has been proposed based on f-divergence to revise the ambiguous regularization caused by clipping.

While PPO has obtained many successes, open issues related to the clipping-based regularization can be raised.

1. PPO has no capability to make the latest policy softly constrain to the baseline one [Wang et al., 2020; Zhu and Rosendo, 2021].
2. Although the symmetric threshold for clipping is given, the density ratio is in asymmetric domain [Kobayashi, 2021a].
3. A recommended threshold range is provided, but it is not for arbitrary learning algorithms.

As for the third problem, a generalized advantage estimation [Schulman et al., 2015] has been utilized for the update law in the original PPO, but the clipping regularization can be acceptable to other learning algorithms. Although threshold-based regularization achieves the task-invariant tuning, it is easy to imagine that the desirable threshold varies depending on the update speed of policy, which further depends on the learning algorithm combined with the regularization. The optimization or adaptive design of the threshold for the learning algorithm to be employed has not been proposed yet. This should be done by considering the error scale from the center of the density ratio (i.e., 1), but in practice, its asymmetry interferes with the estimation of this error scale.

Hence, this paper proposes a new PPO variant to resolve the above issues by theoretically considering a new regularization problem of relative Pearson (RPE) divergence [Yamada et al., 2013; Sugiyama et al., 2013]. Since RPE is one of the divergence metrics between two probabilities, its minimization can softly constrain the latest policy to the baseline one. To inherit the threshold-based regularization like PPO, the strength of regularization in PPO-RPE is reshaped to the corresponding threshold mathematically. On the other hand, the relative density ratio is introduced instead of the raw density ratio, and by adjusting its relativity parameter, it can be in symmetric domain. Thanks to this symmetry, in addition to the earlier work of this paper [Kobayashi, 2021a], the error scale of the relative density ratio can easily be estimated, and it can be utilized as the adaptive threshold. Hence, PPO-RPE would be a versatile method that provides mathematically appropriate policy regularization and has a mechanism that automatically adjusts its strength (i.e., threshold) without any consideration of the task and learning algorithm.

As investigation of PPO-RPE, three simple benchmark tasks provided in Pybullet [Coumans and Bai, 2016] are simulated with two types of learning algorithms. In their results, the importance of algorithm-dependent threshold design is revealed. In addition, four locomotion tasks are simulated using PPO and PPO-RPE. The statistical tests after learning show that only PPO-RPE enables to achieve high scores even in unstable tasks by restricting the policy updates more appropriately than PPO. Therefore, it can be concluded that PPO-RPE does not require the task- and algorithm-dependent tuning, and brings easy application to a variety of practical problems.

2. Preliminaries

2.1. Policy-regularized reinforcement learning

Reinforcement learning (RL) [Sutton and Barto, 2018] optimizes an agent’s policy $\pi$ to maximize the sum of rewards $r^*$ so-called a return defined at time step $t$ as $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ with $\gamma \in [0, 1]$ discount factor. This problem can be solved in Markov decision process (MDP) with the tuple $\langle S, A, R, p_0, p_e \rangle$. Here, $S$ and $A$ denote the state and action spaces, respectively, and $R$ is the reward set. $p_0$ and $p_e$ represent the initial state probability and the state transition probability.

Specifically, MDP assumes the following process. At the time step $t$, the state $s_t \in S$ is first sampled from environment with either of probabilities, $s_0 \sim p_0(s_0)$ as the initial random state or $s_t \sim p_e(s_t \mid s_{t-1}, a_{t-1})$ as the state transition. According to $s_t$, an agent decides the action $a_t \in A$, using the learnable policy $a_t \sim \pi(a_t \mid s_t)$. $a_t$ acts on the environment and stochastically updates the state to the next one $p_e(s_{t+1} \mid s_t, a_t)$. At that time, the agent obtains a reward $r_t \in R$ from the environment: $r_t = r(s_t, a_t)$. By repeating this process with the experienced data $(s_t, a_t, s_{t+1}, r_t)$, the agent estimates the expected $R_t$ as a value function. In on-policy learning like PPO [Schulman et al., 2017], the policy-dependent action value function $Q^\pi(s_t, a_t) = E[R_t \mid s_t, a_t]$ and/or the policy-dependent
action value function $V^\pi(s_t) = E_{a_t \sim \pi}[Q^\pi(s_t, a_t)]$ can be learned through Bellman equation as the following minimization problem:

$$
\mathcal{L}(V^\pi) = \frac{1}{2} \{r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)\}^2 \rightarrow \min
$$

(1)

This paper utilizes $V^\pi$, although $Q^\pi$ can be learned by replacing the above $V^\pi$ with $Q^\pi$. Note that, with DNNs, the target network is often employed to output $V^\pi(s_{t+1})$ for numerical stability.

The optimization of $\pi$ is formulated as the following minimization problem.

$$
\mathcal{L}(\pi) = -E_{a_t \sim \pi}[Q^\pi(s_t, a_t)] - E_{a_t \sim \pi}[A(s_t, a_t)] \rightarrow \min
$$

(2)

where $A(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$ denotes the advantage function, and actually it corresponds to the temporal difference (TD) error. Note that $A$ also depends on $\pi$, but it is omitted to simplify the notation.

With a policy regularization term $\Omega$ (Geist et al., 2019), the original minimization target is reformulated as follows:

$$
\mathcal{L}^\dagger(\pi) = -E_{a_t \sim \pi}[A(s_t, a_t)] + E_{a_t \sim \pi}[\Omega(s_t, a_t)]
$$

$$
= -E_{a_t \sim b}[\pi(a_t | s_t) / b(a_t | s_t)] \{A(s_t, a_t) - \Omega(s_t, a_t)\}
$$

$$
= -E_{a_t \sim b}[\rho(s_t, a_t) A(s_t, a_t)]
$$

$$
= -E_{a_t \sim b}[\rho^\dagger(s_t, a_t) A(s_t, a_t)]
$$

(3)

(4)

where a baseline policy $b(a_t | s_t)$ (e.g. the old version of $\pi$ or the one outputted from slowly updated target network (Mnih et al., 2015)) is introduced and a density ratio $\rho = \pi/b$ is derived in $\mathcal{L}^\dagger$. This allows the agent to reuse past empirical data for training as experience replay (Lin, 1992; Schaul et al., 2015), where the actions have not been sampled from the current policy. $A^\dagger(s_t, a_t)$ and $\rho^\dagger(s_t, a_t)$ are the surrogate versions defined as $A^\dagger(s_t, a_t) = A(s_t, a_t) - \Omega(s_t, a_t)$ and $\rho^\dagger(s_t, a_t) = \rho(s_t, a_t)(1 - \Omega(s_t, a_t)/A(s_t, a_t))$, respectively.

By applying a policy-gradient method to minimize $\mathcal{L}^\dagger(\pi)$ with Monte Carlo approximation, which eliminates the expectation operation, the policy parameterized by a parameters set $\theta$ (e.g. weights and biases in DNNs) can be optimized.

$$
\nabla_\theta \mathcal{L}^\dagger(\pi) \simeq -\nabla_\theta \{\rho(s_t, a_t) A^\dagger(s_t, a_t)\}
$$

$$
= -\rho(s_t, a_t) A^\dagger(s_t, a_t) \nabla_\theta \ln \pi(a_t | s_t)
$$

$$
\theta \leftarrow \theta - \alpha \text{SGD}(\nabla_\theta \mathcal{L}^\dagger(\pi))
$$

(5)

(6)

where $A^\dagger = A^\dagger + \pi \nabla_\pi A^\dagger = A^\dagger + P \nabla_\pi P A^\dagger$. If $A^\dagger$ has no direct computational graph with $\pi$, $A^\dagger = A^\dagger$. In general DRL, $\theta$ is updated using one of the stochastic gradient descent (SGD) optimizers like (dono et al., 2020).

### 2.2. Proximal policy optimization

In the clipping version of proximal policy optimization (PPO) (Schulman et al., 2017) (and its variant named PPO-RB (Wang et al., 2020)), eq. (4) is given as the following condition $\rho^\dagger = \rho^\dagger_{\text{clip}}$, instead of the explicit $\Omega$, with a threshold parameter $\epsilon > 0$. Note that, to simplify the rest of the description, the arguments (e.g. $(s_t, a_t)$) are omitted in most cases.

$$
\rho^\dagger_{\text{PPO}} = \begin{cases} 
-\eta \rho + (1 + \eta)(1 + \sigma \epsilon) \quad &\sigma (\rho - 1) \geq \epsilon \\
\rho &\text{otherwise}
\end{cases}
$$

$$
\mathcal{L}^\dagger_{\text{PPO}}(\pi) = -E_{a_t \sim \pi}[\rho^\dagger_{\text{PPO}}(s_t, a_t) A(s_t, a_t)]
$$

(7)

(8)

where $\sigma$ denotes the sign of $A$. $\eta \geq 0$ is for rollback to the baseline policy, which has been developed for PPO-RB, and only if $\eta = 0$, it reverts to the original PPO.

In this condition, $\Omega_{\text{PPO}}$ for the specific regularization in PPO-RB can be analytically derived (see the appendix). However, its heuristic design does not corresponds to divergence between two probabilities like KL divergence.

### 2.3. Relative Pearson divergence

To inherit the abilities of PPO (eqs. (7) and (8)) and overcome its issues, the following conditions are considered.

1. When $\rho = 1$, the term inside the expectation of the loss function should be $\rho A$.
2. Only when $\rho$ is on the threshold, the gradient must be zero and there is only one local maximum.
3. For the explicit constraint between $\pi$ and $b$, one of the $f$-divergences, which is the generalized divergence (Liese and Vajda, 2006), is desired to be utilized.
4. Symmetrical shape of $\rho^\dagger$ is desired to symmetrically clip it and to adjust the threshold $\epsilon$ easily according to the error scale of $\rho^\dagger$ from its center.

KL divergence, which is utilized in the prior version of PPO (Schulman et al., 2017), cannot satisfy all of them (especially, the symmetry). Thus, the alternative divergence among one of the $f$-divergences that satisfies all the above is desired to be selected even heuristically. This study finds that relative Pearson (RPE) divergence (Yamada et al., 2013; Sugiyama et al., 2013) has the potential to satisfy them.

Specifically, the standard Pearson (PE) divergence between two policies, $\pi$ and $b$, is given at first.

$$
\text{PE}(\pi, b) = E_{a_t \sim b} \left[ \frac{1}{2} (\rho(a) - 1)^2 \right]
$$

(9)

Due to $\int b(a | s) \rho(s, a) da = 1$ and $\int \pi(a | s) da = 1$, the mean of $\rho$ is equal to 1. That is, this PE divergence means the expected squared error of $\rho$ from its mean. This is non-negative and vanishes if and only if $\pi = b$.
Instead of $b$, RPE divergence introduces the relative density function with a mixture ratio $\beta \in [0, 1]$ as follows:

$$\pi_\beta = \beta \pi + (1 - \beta) b$$

(10)

In addition, the relative density ratio $\rho_\beta$ is defined using $\pi_\beta$.

$$\rho_\beta = \pi_\beta = \frac{\pi}{\beta \rho + 1 - \beta}$$

(11)

where the term on the right side can be derived by multiplying it with $1 = b^{-1}/b^{-1}$. As important properties, $\rho_\beta$ is finite within $[0, 1/\beta]$, and its mean is still 1. Therefore, if $\beta = 0.5$, $\rho_\beta$ obtains the symmetry, as shown in Fig. 1.

By replacing $\pi$ in eq. (9) to $\rho_\beta$, RPE divergence is defined as follows:

$$\text{PE}_\beta(\pi, b) = \mathbb{E}_{\pi \sim \pi_\beta} \left[ \frac{1}{2} (\rho_\beta(a) - 1)^2 \right]$$

$$= \mathbb{E}_{\pi \sim \pi_\beta} \left[ \frac{1}{2} \left( \frac{\rho(a)}{\beta \rho(a) + (1 - \beta)} - 1 \right)^2 \right]$$

(12)

3. Proposed method

3.1. Proximal policy optimization with relative Pearson divergence: PPO-RPE

This paper proposes a proximal policy optimization with relative Pearson divergence, so-called PPO-RPE. The explicit f-divergence minimization can softly constrain the latest policy $\pi$ to the baseline one $b$. In particular, by employing RPE divergence, the gain of this regularization $C$ can be adaptively tuned as PPO-like (but adaptive, not fixed) threshold, as introduced in the next section. The main process of PPO-RPE except the adaptive threshold for determining the gain $C$ is summarized in Alg. 1.

3.1.1. Minimization target

To make PPO-RPE be a sub-class of policy-regularized RL described in eq. (3), the regularization term for PPO-RPE, $\Omega_{\text{RPE}}$ is derived. With $\pi_\beta$ in eq. (10) and $\rho_\beta$ in eq. (11), $\Omega_{\text{RPE}}$ is given as follows:

$$\Omega_{\text{RPE}} = C \left( \frac{\rho_\beta - 1}{\rho_\beta} \right)^2 = C \left( 1 - \beta + \beta \rho \right) (\rho_\beta - 1)^2$$

(13)

where $C$ again denotes the gain of this regularization. The expectation w.r.t $\pi$ of this regularization is equivalent to eq. (14) amplified by the fixed $C$ as shown in below.

$$\mathbb{E}_{\pi \sim \pi} [\Omega_{\text{RPE}}] = \int \pi C (\rho_\beta - 1)^2 \frac{\beta}{\pi} \frac{da}{\pi}$$

$$= \int \pi_\beta C (\rho_\beta - 1)^2 da$$

$$= \text{CPE}_\beta(\pi, b)$$

(14)

Note that if $C$ depends on $\rho_\beta$ (or $\pi$), this derivation is not exact but sufficiently relevant.

By substituting eq. (13) for eq. (3), the following regularized loss function is derived for PPO-RPE.

$$\mathcal{L}_{\text{RPE}}(\pi) = -\mathbb{E}_{a \sim b} \left[ \rho A - C (1 - \beta + \beta \rho) (\rho_\beta - 1)^2 \right]$$

$$= -\mathbb{E}_{a \sim b} \left[ \rho A_{\text{RPE}} \right]$$

(15)

where $\Omega_{\text{RPE}}$ is included in $A_{\text{RPE}}$, and it can be represented as $\rho_{\text{RPE}} A$ with $\rho_{\text{RPE}} = \rho A_{\text{RPE}}/A$.

3.1.2. Policy gradient

Here, the policy gradient for eq. (15) is analytically derived. According to eq. (5), $\overline{A}_{\text{RPE}} = A_{\text{RPE}} + \rho \nabla_\rho A_{\text{RPE}}$ is computed analytically as follows (its derivation is in the appendix (Appendix .2)).

$$\overline{A}_{\text{RPE}} = A \left[ 1 - \frac{C}{A} (\rho_\beta - 1) \left\{ \beta (\rho_\beta - 1) + 2 (1 - \beta) \frac{\rho_\beta}{\rho} \right\} \right]$$

(16)
Algorithm 1 PPO-RPE: the value function $Q$ and/or $V$ will be trained in parallel according to Bellman equation.

```
1: Given environment as $p_e(s_{t+1} \mid s_t, a_t)$ and $r(s_t, a_t)$
2: Set $\beta \in [0, 1]$ (0.5 is the default value for symmetry)
3: Set $\gamma \in [0, 1]$ (0.99 is the default value)
4: Initialize the policy $\pi$ with $\theta$ and the baseline policy $b$
5: Initialize optimizer as SGD with learning rate $\alpha$
6: while True do
7:     $s_t \sim p_e(s_t \mid s_{t-1}, a_{t-1})$
8:     $a_t \sim b(a_t \mid s_t)$
9:     $s_{t+1} \sim p_e(s_{t+1} \mid s_t, a_t)$
10:    $r_t = r(s_t, a_t)$
11:   $\rho = \frac{1 - \beta + \beta \rho}{\pi(a_t \mid s_t)}$
12:   $\rho_\beta = \rho$  
13:   Get $C$ from Alg. 2, given $A$ and $\rho_\beta$ as inputs
14:   $\tilde{A}^{\text{PPO-RPE}} = A - C(\rho_\beta - 1) \left\{ \beta(\rho_\beta - 1) + 2(1 - \beta) \frac{\rho_\beta}{\rho} \right\}$
15:   $g = -\tilde{A}^{\text{PPO-RPE}} \ln \pi(a_t \mid s_t)$
16:   $\theta \leftarrow \theta - \alpha \text{SGD}(g)$
17:   Update $b$ according to $\pi$ with $\theta$
18:   $t \leftarrow t + 1$
19: end while
```

This is substituted for eq. [5], and used for updating the parameters set $\theta$ in eq. [6].

The second term in the square bracket is consistent with the regularization, and vanishes if and only if $\rho = \rho_\beta = 1$ (i.e. $\pi = b$) regardless of $C$ and $\beta$. Namely, with this policy gradient, $\pi$ has the potential to converge on $\pi^*$, which maximizes $A$ (more specifically, the action-value function $Q$), while being regularized to $b$ during the policy updates.

3.2. Adaptive threshold for designing gain of regularization

Actually, the gain for regularization, $C$, is hard to be tuned since the ratio between $A$ for the original purpose and $\Omega^{\text{PPO-RPE}}$ for the regularization is task- and algorithm-dependent. In contrast to $C$, PPO utilizes the threshold with $\epsilon$ for tuning the strength of regularization. The threshold allows for more intuitive design, and in fact, the original paper [Schulman et al. 2017] shows the recommended range of $\epsilon$ for any task.

Inspired by this, $C$ is converted to the PPO-like threshold, which releases from the task dependency. In addition, by applying $\epsilon$ to the relative density ratio $\rho_\beta$, not the raw density ratio $\rho$, the symmetric threshold works in the same symmetric domain. Such symmetry enables PPO-RPE to easily estimate the algorithm-dependent error scale of $\rho_\beta$ from its center $t$, which would be utilized for the adaptive design of $\epsilon$. The pseudo code for the adaptive threshold is described in Alg. 2.

Algorithm 2 Adaptive threshold design for PPO-RPE

```
1: Set $\beta \in [0, 1]$ (0.5 is the default value)
2: Set $\lambda \in (0, 1)$ (0.999 is the recommended value)
3: Set $\kappa \in (0, 1)$ (0.5 is the recommended value)
4: Set $\Delta \in (0, 0.5)$ (0.1 is the recommended value)
5: Initialize $\Delta_{\max} = 0$ (i.e. no prior information)
6: Initialize $\Delta = 1$ (i.e. theoretical maximum value)
7: while True do
8:     Compute $C$ from Alg. 2
9:     Get $A$ and $\rho_\beta$ from Alg. 1
10:    $\epsilon = \kappa \max(\Delta, 1 - \Delta)$
11:    $C = \frac{\beta \sigma^2 + 2 \epsilon (1 - \beta (1 + \epsilon))}{|A|}$
12:    Yield $C$ to Alg. 1
13: end while
```

3.2.1. Conversion from gain to threshold

We focus on the fact that the density ratio $\rho$ is asymmetric around 1, and not consistent with the symmetric threshold in PPO, causing unbalanced regularization. Instead, PPO-RPE can serve the threshold for the relative metric around 1, and not consistent with the symmetric threshold in PPO, causing unbalanced regularization. In particular, PPO-RPE can serve the threshold for the relative metric around 1, and not consistent with the symmetric threshold in PPO, causing unbalanced regularization.

Only at $\rho_\beta$ (and $\rho'$), in order to softly constrain the latest policy $\pi$ to the baseline $b$, $\tilde{A}^{\text{PPO-RPE}}$ is desired to be zero, which makes $L^{\text{PPO-RPE}}$ be convex. Using this condition, $C$ can be derived from eq. [16] as follows:

$$C = \frac{|A|}{\beta \sigma^2 + 2 \epsilon (1 - \beta (1 + \epsilon))}$$

where $A/\sigma = |A|$ is utilized. Indeed, $C$ designed above is regarded as the function of $A$, and therefore, yields the adaptability to various tasks with different scales of reward function.
By substituting eq. (19) for eq. (16), PPO-RPE achieves the following policy gradient term with $\epsilon$ instead of $C$.

$$\tilde{A}^{\text{RPE}} = A - \frac{|A|(\rho^* - 1) \left\{ \beta(\rho^* - 1) - 2(1 - \beta) \frac{\rho^*}{\rho} \right\} }{\beta\sigma^2 - 2\epsilon(1 - \beta(1 + \sigma\epsilon))} \quad (20)$$

Using this, the parameters set $\theta$ is updated according to SGD with the policy gradient $-\rho\tilde{A}^{\text{RPE}}\nabla_{\theta} \ln \pi(a_t | s_t)$.

Here, all the examples of the negative loss functions (i.e. $\rho A^1$ or $\rho^1 A$) for the proposed and conventional methods are illustrated in Fig. 2. As can be seen in this figure, PPO-RPE has the asymmetric threshold in $\rho$ domain, which is symmetric in $\rho_{\beta=0.5}$ domain.

### 3.2.2. Adaptive threshold using estimated error scale

In eq. (20), we can see that the threshold-based gain for regularization can adapt to differences in reward scales for different tasks. However, the mean of divergence between $\pi$ and $b$ depends on the learning algorithm. For example, when $b$ is given by the target network, when experience replay (Lin 1992) is not used, or even when the learning rate $\alpha$ is changed, the mean would vary. The algorithm-dependent design of threshold is therefore required, but as can be imagined, hand-tuning is burdensome since the recent DRL methods are constructed like by patchworks of various modules and there are many combinations. In addition, it is difficult to completely eliminate task dependency only by the threshold-based design. From the earlier work (Kobayashi 2021a), this paper additionally contributes to design the adaptive threshold $\epsilon$.

Specifically, thanks to the symmetry of $\rho^*_b$ (with $\beta = 0.5$) and its consistency with the threshold, its maximum error scale from its center (i.e. one) is revealed as one: $|0 - 1| = |1/0.5 - 1| = 1$. With this fact, $\epsilon$ can be interpreted as defining a threshold value for the algorithm-independent (i.e. absolute) maximum error scale, although the algorithm-dependent (i.e. relative) maximum error scale may vary from it. It is therefore expected that if $\epsilon$ is given for the relative maximum error scale, it would release from the algorithm dependency.

To estimate the relative maximum error scale in the recent past, $\Delta$, the following heuristics is implemented with three hyperparameters: $\lambda \in (0,1)$ for smoothing updates; $\kappa \in (0,1)$ for the relative threshold; and $\Delta \in (0,0.5)$ and $\bar{\Delta} = 1 - \Delta$ for numerical stability:

$$\Delta_{\max} \leftarrow \max(\lambda\Delta_{\max}, |\rho^*_b - 1|) \quad (21)$$
$$\Delta \leftarrow \lambda\Delta + (1 - \lambda)\Delta_{\max} \quad (22)$$
$$\epsilon = \kappa\max(\min(\Delta, \bar{\Delta}), \Delta) \quad (23)$$

Here, $\Delta_{\max}$ stores the recent maximum value, then it is smoothly transferred into $\Delta$. Since max operator switches the output value discretely, this two-step update provides smooth estimation of $\Delta$ and suppresses regularization fluctuations. By giving $\epsilon$ in this way, the adaptive threshold in PPO-RPE is obtained against the relative maximum error scale that depends on the learning algorithm, easing algorithm-specific tuning.

As a remark, although the new hyperparameters have been introduced here, they can be given regardless of other implementations: for $\lambda$, 0.999, which is also used in the most of SGDs (Ziyin et al. 2020; Kobayashi 2021b; Ilboudo et al. 2020) for relatively slow updates, would be the appropriate value; $\kappa$ can be fixed to place the local maximum of the regularization onto half of the relative maximum error scale, i.e. 0.5; and $\Delta$ can be designed based on the minimum recommended value of $\epsilon$ in PPO, i.e. 0.1.

### 4. Simulations

#### 4.1. Implementations

In this paper, the proposed method with its hyperparameters is implemented in almost the same way as in (Kobayashi and Ilboudo 2021). The details are introduced below. Note that all the hyperparameters commonly used are listed in Table 1.

First, the policy $\pi$ is parameterized by student-t distribution (Kobayashi 2019) with neural networks. $d_a$-dimensional student-t distribution has three model parameters: $\mu \in \mathbb{R}^{d_a}$; $\sigma \in \mathbb{R}_+^{d_a}$; and $\nu \in \mathbb{R}_+$. They are given from...
are with the same hyperparameter as \cite{Kobayashi2021}. The buffer size of the experience replay is reduced due to memory limitation. According to them, the proposed method will appropriately modify the threshold \( \epsilon \) adaptively.

In this experiment, seven benchmark tasks are simulated on Pybullet dynamical engine \cite{Coumans2016} wrapped by OpenAI Gym \cite{Brockman2016}. They can be divided into three simple and four complex tasks. That is, the first three tasks (i.e. InvertedPendulum, Swingup, and DoublePendulum) are the simple ones with only one-dimensional action space, although they are nonlinear systems. The remaining tasks (i.e. Hopper, Walker2D, HalfCheetah, and Ant) are the complex locomotion control tasks with multivariate action space and more than 10-dimensional state space. Totally, 20 trials for the simple tasks and 10 trials for the complex tasks with each condition are conducted with different random seeds. Note that the robot’s power in Walker2D was insufficient to walk, hence, it is amplified by a factor of 3.75 (i.e. 0.4 to 1.5).

4.3. Results for simple tasks

To verify the value of the adaptive threshold, the first three tasks are conducted with the following conditions.

1. \{e,r\} PPO \cite{Schulman2017}: \( \eta = 0 \) as original PPO with no rollout
2. \{e,r\} PPO-RB \cite{Wang2020}: \( \eta = 0.3 \) as the recommended value
3. \{e,r\} PPOS \cite{Zhu2021}: \( \eta = 0.3 \) as the recommended value
4. \{e,r\} PPO-RPE \cite{Kobayashi2021}: the fixed threshold
5. \{e,r\} PPO-RPE-A (propositional): the adaptive threshold

where \{e,r\} denote the use of the adaptive eligibility traces or the experience replay. For the fixed threshold, \( \epsilon = 0.1 \) is set as one of the recommended value and the same value as the earlier work \cite{Kobayashi2021}, which is with the adaptive eligibility traces. For the adaptive threshold, all the recommended values in Alg 2 are utilized. Note that the other PPO variants introduced in the introduction \cite{Hämäläinen2020, Imagawa2019, Libardi2021} were omitted because they are not for improvements of regularization and are difficult to compare fairly with the proposed method.

First of all, all the learning curves are illustrated in Fig. 3. With the adaptive eligibility traces, InvertedPendulum and Swingup could be completely accomplished by all the conditions since the threshold \( \epsilon \) was already tuned in the earlier work \cite{Kobayashi2021}. However, only

![Table 1: Common hyperparameters for the simulations](image)

| Symbol       | Meaning                          | Value               |
|--------------|----------------------------------|---------------------|
| \( N \)      | Number of neurons                | 100                 |
| \( L \)      | Number of layers                 | 5                   |
| \( \gamma \) | Discount factor                  | 0.99                |
| \( \alpha \) | Learning rate                    | 0.001               |
| \( (\tau, \nu) \) | For t-soft update                 | \((0.5, 1.0)\)     |
| \( \beta_{DE} \) | Gain for entropy regularization | \(3 \times 10^{-4}\) |
| \( \beta_{TD} \) | Gain for TD regularization       | \((0.3, 1.0)\)     |
| \( (N_c, N_d, N_b) \) | For experience replay            | \((10, 10, 10)\)   |
| \( (\lambda_{max}, \lambda_{min}, \kappa) \) | For adaptive eligibility traces  | \((0.5, 0.95, 10)\) |

\( L \) layers

- Fully connected layer
- Layer normalization
- Activation (Swish)
- Mapping

![Network architecture in actor for policy or critic for value function](image)

Figure 3: Network architecture in actor for policy or critic for value function: state is inputted to \( L \) series-connected modules; each module contains a fully connected layer, a layer normalization \cite{Ba2016}, and a swish activation function \cite{Elfwing2018}; an additional fully connected layer uses for shaping the features given by \( L \) modules to the output.

The networks with \( d_x \)-dimensional state space inputs. The value function \( V \) is approximated by the corresponding networks with the same architecture as that of \( \pi \).

The networks contains \( L = 5 \) fully connected layers with \( N = 100 \) neurons and pairs of layer normalization \cite{Ba2016} and Swish activation function \cite{Elfwing2018} for nonlinearity (also see Fig. 3). These implementations are built on PyTorch \cite{Paszke2017}. As an optimizer of the networks, a robust SGD, i.e., LaProp \cite{Ziyin2020} with t-momentum \cite{Ilboudo2020} and d-AmsGrad \cite{Kobayashi2021}, is employed with their default parameters except the learning rate.

To learn the networks stably, a target network with t-soft update \cite{Kobayashi2021} is employed for stable and efficient improvement. Here, the baseline policy \( b \) is regarded as the output from the target network for \( \pi \), that is, \( b \) is a kind of the old policy. In addition, the policy entropy regularization based on SAC \cite{Haarnoja2018} with a regularization weight \( \beta_{DE} \); and the TD regularization \cite{Parisi2019} with a regularization weight \( \beta_{TD} \) are combined.

For acceleration of learning, either or both of adaptive eligibility traces \cite{Kobayashi2021} and the experience replay are employed. Although the adaptive eligibility traces
PPO-RPE(-A) could accomplish DoublePendulum sometimes, and in particular, only the adaptive threshold enabled to reach the score around 5000, namely, it could succeed in accomplishing DoublePendulum over 50%. This is because online learning using the eligibility traces cannot maintain sufficient sample efficiency in failure-prone task such as DoublePendulum, but PPO-RPE limits the exploration range of the policy by regularizing the policy well, making it easier to find the optimal solution.

With the experience replay, PPO-RB, PPOS, and PPO-RPE failed to acquire all the tasks. Since their performances generally decreased with stronger regularization methods, these results are probably due to too strong regularization of $\pi$ to $b$ for the replayed data. As pointed out in (Wang et al., 2020; Zhu and Rosendo, 2021), PPO has no capability to softly constrain $\pi$ to $b$ even with the too small threshold, and therefore, it yielded the near-optimal policy in the tasks except Swingup. In contrast to them, only PPO-RPE-A stably learned all the tasks. This is probably due to the relaxation of regularization (larger $\epsilon$ than 0.1).

To confirm the adaptability of the proposed method, Fig. 5 additionally shows the average thresholds in $\{\epsilon, r\}$ PPO-RPE-A. Basically, we can see that the adaptive eligibility traces and the experience replay required the small and large thresholds (i.e. around $\kappa \Delta$ and $\kappa \bar{\Delta}$), respectively. Note that clamping by $\Delta$ and $\bar{\Delta}$ worked in a limited number of scenes, such as second half of learning in InvertedPendulum. In addition, most of them did not converge to completely constant values, indicating its adaptive behavior according to the learning progress. The benefits of such an adaptive behavior need to be investigated.

### 4.4. Results for complex tasks

For further investigation of the performance of the proposed method, the following experiment is also conducted on the remaining complex locomotion tasks under the following conditions.

1. **PPO01 (Schulman et al., 2017):** $\epsilon = 0.1$
2. **PPO03 (Schulman et al., 2017):** $\epsilon = 0.3$
3. **PPO-RPE (Kobayashi, 2021a):** the fixed threshold $\epsilon = 0.3$
4. **PPO-RPE-A (proposal):** the adaptive threshold $\epsilon$

| ID | Name | State space $d_s$ | Action space $d_a$ | Episode $E$ |
|----|------|------------------|-------------------|-------------|
| InvertedPendulumBulletEnv-v0 | InvertedPendulum | 5 | 1 | 200 |
| InvertedPendulumSwingupBulletEnv-v0 | Swingup | 5 | 1 | 200 |
| InvertedDoublePendulumBulletEnv-v0 | DoublePendulum | 9 | 1 | 2000 |
| HopperBulletEnv-v0 | Hopper | 15 | 3 | 2000 |
| Walker2DBulletEnv-v0 | Walker2D | 22 | 6 | 2000 |
| HalfCheetahBulletEnv-v0 | HalfCheetah | 26 | 6 | 2000 |
| AntBulletEnv-v0 | Ant | 28 | 8 | 2000 |

where all the conditions are with both of the adaptive eligibility traces and the experience replay. Due to the bad performance of PPO-RB in the previous results, it is omitted here. To evaluate the value of the adaptive behavior in the learning progress, as indicated in the previous results, PPO-RPE (and PPO) with $\epsilon \approx 0.3$, which is one of the recommended values closest to the approximate mean of the adaptive threshold (see Fig. 6), are also compared.

All the learning curves are illustrated in Fig. 7. In addition, after learning the policy for each task with each condition, the agent performed the task using the learned policy 100 times to compute the median of the scores. This test results are summarized in Fig. 8.

The wide confidence intervals of the learning curves and the test scores suggest that Hopper and Walker2D are very unstable tasks, where the agent often encounters falling over and resetting the episode. In such an environment, the policy should be updated carefully, and in fact, only the proposed method (i.e PPO-RPE-A) accomplished the task multiple times in 10 trials. Note that PPO-RPE without the adaptive threshold could not obtain the same results because the appropriate regularization strength is not served.

On the other hand, in HalfCheetah, PPO with both thresholds outperformed the proposed method. This is probably because this task is relatively stable, easily falling into a local solution (i.e. an upright state); as a result, sufficient exploration is required to get out of such a local solution. As pointed out in (Wang et al., 2020; Zhu and Rosendo, 2021), PPO does not softly constrain the latest policy to the baseline, and therefore, it may hold the exploration capability enough. Nevertheless, it is clear that given the regularization strength adaptively, the task accomplishment rate by PPO-RPE-A is better than that by PPO-RPE. Although Ant also seems to be a stable task, it only requires local exploration to improve the performance, hence, the proposed method obtained the maximum performance.

Finally, to reveal the gap between the latest and baseline policies, numerically-approximated Pearson divergence between the latest and baseline policies is depicted in Fig. 9. Clearly, PPO fails to softly constrain the latest policy to the baseline, and is inadequate as a policy regularization method, although it does not impair the exploration capability. In contrast, PPO-RPE succeeded...
This paper proposed PPO-RPE, a variant of PPO integrated with RPE divergence regularization, to clearly constrain the policy to its baseline in a symmetric manner. The threshold-based gain was derived for PPO-RPE, as the standard PPO does, and makes this regularization tuning task-invariant. In addition to the earlier work (Kobayashi, 2021a), this paper further contributes to design the adaptive threshold based on the symmetric property in the relative density ratio for RPE divergence. The conventional threshold can be interpreted that it is set based on the theoretical error scale of the relative density ratio from its center, but the actual error scale is algorithm-dependent. According to this fact, the adaptive threshold was designed to consider the algorithm-dependent error scale, which is estimated from experience heuristically. This design can cancel the algorithm dependency of the regularization tuning. In the numerical simulations with two types of learning algorithms, only the proposed method could accomplish all the tasks using both the learning algorithms, although the standard PPO was also robust to the change of learning algorithm due to the lack of the capability to softly constrain the latest policy to the baseline; by relaxation of the threshold, PPO-RPE-A achieved the best performance in all the tasks.

5. Conclusion and future work

This paper proposed PPO-RPE, a variant of PPO integrated with RPE divergence regularization, to clearly constrain the policy to its baseline in a symmetric manner. The threshold-based gain was derived for PPO-RPE, as the standard PPO does, and makes this regularization tuning task-invariant. In addition to the earlier work (Kobayashi, 2021a), this paper further contributes to design the adaptive threshold based on the symmetric property in the relative density ratio for RPE divergence. The conventional threshold can be interpreted that it is set based on the theoretical error scale of the relative density ratio from its center, but the actual error scale is algorithm-dependent. According to this fact, the adaptive threshold was designed to consider the algorithm-dependent error scale, which is estimated from experience heuristically. This design can cancel the algorithm dependency of the regularization tuning. In the numerical simulations with two types of learning algorithms, only the proposed method could accomplish all the tasks using both the learning algorithms, although the standard PPO was also robust to the change of learning algorithm due to the lack of the capability to softly constrain the latest policy to the baseline; by relaxation of the threshold, PPO-RPE-A achieved the best performance in all the tasks.
Figure 5: Adaptive thresholds with two learning algorithms: the corresponding shaded areas show the 95 % confidence intervals; with the eligibility traces, the value of ε was smoothly decreased to the near minimum because the latest and baseline policies (π and b) are not far apart enough; with the experience replay, the value of ε was almost kept in the initial one (i.e. the maximum) since b in the buffer is often old and π tends to be far from it; however, in both cases, most of ε did not converge to a constant, and fluctuated adaptively according to the learning progress.

Figure 6: Trajectories of adaptive thresholds in the respective tasks: in stable HalfCheetah and Ant, their thresholds converged on almost the same values (i.e. ∼ 0.33); in contrast, Hopper and Walker2D had oscillatory threshold due to their instability, which reduced the number of iterations for the policy updates.

Performance while holding the regularization capability. In addition, the proposed method can be integrated with the other PPO variants (Hämäläinen et al., 2020; Imagawa et al., 2019; Libardi et al., 2021). Hence, by investigating the integration with them, we can expect to obtain better learning performance in the near future.

Appendix .1. Regularization term in PPO and PPO-RB

With eqs. (4), (5), and (6), the regularization term for PPO, which has not been explicitly defined in the previous work (Schulman et al., 2017; Wang et al., 2020) yet, is derived. The changes in ρPPO compared to the original ρ should be transferred into the surrogate advantage function APPO, which is the sum of the original A and the negative regularization term for PPO −ΩPPO. When no clipping or rollback is done, ΩPPO is clearly derived as zero. In contrast, when clipping or rollback, the following relationship is derived.

\[
\rho_{PPO} A = \rho_A \{−η + ρ^{−1}(1 + η)(1 + σε)\} = \rho_A \{1 − (1 + η) + ρ^{−1}(1 + η)(1 + σε)\} = \rho_A \{1 − (1 + η)(1 − ρ^{−1}(1 + σε))\} = \rho A_{PPO} = ρ(A − Ω_{PPO}) \tag{.1}
\]

Therefore, the regularization term in PPO, ΩPPO, is given as follows:

\[
Ω_{PPO} = \begin{cases} 
A(1 + η)(1 − ρ^{−1}(1 + σε)) & σ(ρ − 1) ≥ ε \\
0 & \text{otherwise} \end{cases} \tag{.2}
\]

As can be seen in this regularization term, PPO regularizes the policy adaptively according to the advantage function A. Although this seems to be a kind of hinge loss function, which is zero if and only if ρ = 1 + σε or no clipping and rollback, its mathematical meaning is not intuitive, even by converting it to its expectation E_b[ρA_{PPO}] as the regularization target.

Appendix .2. Derivation of eq. (10)

With ρβ = ρ/(1 − β + βρ), the partial differentiation of ∂ρβ/∂ρ is given as follows:

\[
\frac{∂ρ_β}{∂ρ} = \frac{1 − β + βρ − βρ}{(1 − β + βρ)^2} = \frac{1 − β}{(1 − β + βρ)^2} \tag{.3}
\]
(b) Walker2D (w/ stronger power)
(c) HalfCheetah
(d) Ant

Figure 7: Learning curves of four locomotion tasks: although (a) Hopper and (b) Walker2D tasks are with instability, which caused the larger confidence intervals, the proposed method, PPO-RPE-A, obtained the higher/same results than/as the others; in (c) HalfCheetah, only PPO succeeded in acquiring locomotion stably, and PPO-RPE failed probably due to the lack of the capability to explore a global solution; in contrast, (d) Ant probably requires only local exploration to find a global solution, and therefore, PPO-RPE-A, which can update the policy more carefully, eventually outperformed the others.

Figure 8: Test scores after training: in the unstable Hopper and Walker2D tasks, only the proposed method, PPO-RPE-A, succeeded in acquiring locomotion multiple times in 10 trials; PPO is suitable for learning HalfCheetah, which is stable and requires the high exploration capability, although the proposed method could adjust its regularization strength, yielding the multiple successes; in Ant task, PPO-RPE-A outperformed the others probably because it is stable and local exploration is enough to find a global solution.

Using this, eq. (16) can be derived in the following steps.

\[
\begin{aligned}
\tilde{A}_{RPE} &= A-C\beta(\rho_3 - 1)^2 - 2C(1 - \beta + \beta \rho)(\rho_3 - 1) \frac{\partial \rho_3}{\partial \beta} \\
&= A - C\beta(\rho_3 - 1)^2 - C(\rho_3 - 1) \frac{2(1 - \beta + \beta \rho)(1 - \beta)}{(1 - \beta + \beta \rho)^2} \\
&= A - C\beta(\rho_3 - 1)^2 - C(\rho_3 - 1) \frac{2(1 - \beta)}{1 - \beta + \beta \rho} \\
&= A - C(\rho_3 - 1) \left\{ \beta(\rho_3 - 1) + \frac{2(1 - \beta)}{1 - \beta + \beta \rho} \right\} \\
&= A \left[ 1 - \frac{C}{A}(\rho_3 - 1) \left\{ \beta(\rho_3 - 1) + \frac{2(1 - \beta)}{1 - \beta + \beta \rho} \right\} \right] \\
&= A \left[ 1 - \frac{C}{A}(\rho_3 - 1) \left\{ \beta(\rho_3 - 1) + 2(1 - \beta) \frac{\rho_3}{\rho} \right\} \right]
\end{aligned}
\]

where \( 1/(1 - \beta + \beta \rho) = \rho_3/\rho \) is utilized for the last line of the equation transformation.

Acknowledgments
This work was supported by JSPS KAKENHI, Grant-in-Aid for Scientific Research (B), Grant Number JP20H04265.

References
Ba, J.L., Kiros, J.R., Hinton, G.E., 2016. Layer normalization. arXiv preprint arXiv:1607.06450 .
Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W., 2016. Openai gym. arXiv preprint arXiv:1606.01540 .
Chua, K., Calandra, R., McAllister, R., Levine, S., 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models, in: Advances in Neural Information Processing Systems, pp. 4754–4765.
Clavera, I., Fu, Y., Abbeel, P., 2020. Model-augmented actor-critic: Backpropagating through paths, in: International Conference on Learning Representations.
Coumans, E., Bai, Y., 2016. Pybullet, a python module for physics simulation for games, robotics and machine learning. GitHub repository .
Ellwing, S., Uchibe, E., Doya, K., 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Networks 107, 3–11.
Farsang, M., Szelegtes, L., 2021. Decaying clipping range in proximal policy optimization. arXiv preprint arXiv:2102.10456 .
Geist, M., Scherrer, B., Pietquin, O., 2019. A theory of regularized markov decision processes. arXiv preprint arXiv:1901.11275 .
Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290 .
Hämäläinen, P., Babadi, A., Ma, X., Lehtinen, J., 2020. Ppo-cma: Proximal policy optimization with covariance matrix adaptation, in: IEEE International Workshop on Machine Learning for Signal Processing, IEEE, pp. 1–6.
Ilboudo, W.E.L., Kobayashi, T., Sugimoto, K., 2020. Robust stochastic gradient descent with student-t distribution based first-order momentum. IEEE Transactions on Neural Networks and Learning Systems , 1–14.
Imagawa, T., Hiraoka, T., Tsuruoka, Y., 2019. Optimistic proximal policy optimization, arXiv preprint arXiv:1906.11075 .
Kobayashi, T., 2019. Student-t policy in reinforcement learning to acquire global optimum of robot control. Applied Intelligence 49, 4335–4347.
Figure 9: Trajectories of numerically-approximated Pearson divergence between the latest and baseline policies: PPO obviously failed to softly constrain the latest policy to the baseline; both PPO-RPE with and without the adaptability of threshold served the appropriate regularization of the policy updates.

Kobayashi, T., 2021a. Proximal policy optimization with relative policy divergence, in: IEEE International Conference on Robotics and Automation, IEEE, pp. 8416–8421.
Kobayashi, T., 2021b. Towards deep robust learning with optimizer applicable to non-stationary problems, in: IEEE/SICE International Symposium on System Integration, IEEE, pp. 190–194.
Kobayashi, T., 2022. Adaptive and multiple time-scale eligibility traces for online deep reinforcement learning. Robotics and Autonomous Systems, 104019.
Kobayashi, T., Ilboudo, W.E.L., 2021. t-soft update of target network for deep reinforcement learning. Neural Networks 136, 63–71.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning, nature 521, 436.
Libardi, G., De Fabritiis, G., Dittert, S., 2021. Guided exploration with proximal policy optimization using a single demonstration, in: International Conference on Machine Learning, PMLR. pp. 6611–6620.
Liese, F., Vajda, I., 2006. On divergences and informations in statistics and information theory. IEEE Transactions on Information Theory 52, 4394–4412.
Lin, L.J., 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning 8, 293–321.
Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K., 2016. Asynchronous methods for deep reinforcement learning, in: International conference on machine learning, pp. 1928–1937.
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., 2015. Human-level control through deep reinforcement learning. nature 518, 529–533.
Parisi, S., Tangkaratt, V., Peters, J., Khan, M.E., 2019. Tel-regularized actor-critic methods. Machine Learning, 1–35.
Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A., 2017. Automatic differentiation in pytorch, in: Advances in Neural Information Processing Systems Workshop.
Schaul, T., Quan, J., Antonoglou, I., Silver, D., 2015. Prioritized experience replay. arXiv preprint arXiv:1511.05952.
Schulman, J., Moritz, P., Levine, S., Jordan, M., Abbeel, P., 2015. High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438.
Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.
Shi, W., Song, S., Wu, C., 2019. Soft policy gradient method for maximum entropy deep reinforcement learning, in: International Joint Conference on Artificial Intelligence, pp. 3425–3431.
Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al., 2016. Mastering the game of go with deep neural networks and tree search. nature 529, 484.
Sugiyama, M., Liu, S., Du Plessis, M.C., Yamanaka, M., Yamada, M., Suzuki, T., Kanamori, T., 2013. Direct divergence approximation between probability distributions and its applications in machine learning. Journal of Computing Science and Engineering 7, 99–111.
Sutton, R.S., Barto, A.G., 2018. Reinforcement learning: An introduction. MIT press.
Wang, Y., He, H., Tan, X., 2020. Truly proximal policy optimization, in: Uncertainty in Artificial Intelligence, PMLR. pp. 113–122.
Yamada, M., Suzuki, T., Kanamori, T., Hachiya, H., Sugiyama, M., 2013. Relative density-ratio estimation for robust distribution comparison. Neural computation 25, 1324–1370.
Zhu, W., Rosendo, A., 2021. A functional clipping approach for policy optimization algorithms. IEEE Access 9, 96056–96063.
Ziyin, L., Wang, Z.T., Ueda, M., 2020. Laprop: a better way to combine momentum with adaptive gradient. arXiv preprint arXiv:2002.04839.