ABC: ADVANTAGE-BASED CONTROL VARIATES FOR REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent progress in reinforcement learning with verifiable rewards (RLVR) has highlighted the effectiveness of simple critic-free policy-gradient methods such as Group Relative Policy Optimization (GRPO). In contrast, actor-critic methods rely on learned value functions whose approximation error can introduce bias through commonly used advantage estimators such as temporal-difference error. Motivated by this observation, we revisit trajectory-level control variates through an advantage–value formulation, which we call Advantage-Based Control Variates (ABC). This formulation reveals that the covariance structure is closely related to the return decomposition used in Direct Advantage Estimation (DAE). Finally, we combine ABC with DAE into a single actor-critic algorithm and evaluate it in an offline-to-online RLVR setting, where the critic is first trained on previously collected trajectories and adapted during online learning. On mathematical reasoning tasks, ABC achieves performance competitive with GRPO using substantially fewer online optimization steps.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become a powerful approach for improving large language models (LLMs) beyond supervised fine-tuning, particularly in domains such as mathematical reasoning, code generation, and formal theorem proving, where rewards can be computed automatically (Lambert et al., 2024; Shao et al., 2024; Guo et al., 2025).

Actor-critic methods (Konda & Tsitsiklis, 1999; Mnih et al., 2016) such as Proximal Policy Optimization (PPO) estimate the advantage function using a learned value function (Schulman et al., 2015; 2017) to reduce policy optimization variance, and remain a strong baseline for deep RL applications (Rudin et al., 2022; Radosavovic et al., 2024). In LLM post-training, however, maintaining a separate critic can be computationally expensive. As such, modern RLVR methods often adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which removes the learned critic and estimates advantages by comparing groups of on-policy rollouts. GRPO has achieved strong results despite discarding the learned critic, raising the possibility that learning value functions in this case can be difficult and the bias from the approximation of the advantages can offset their variance-reduction benefits. However, GRPO's group-based estimator requires multiple fresh rollouts per prompt, provides trajectory-level feedback, and no learning signal when rewards within a group are identical (Yu et al., 2026). These limitations motivate a method that allows fine-grained token-level feedback while ensuring that the bias stays controlled.

Motivated by these observations, we study **A**dvantage-**B**ased **C**ontrol variate (ABC), an advantage-value formulation of trajectory-level control variates (Cheng et al., 2020) that naturally combines learned advantage and value functions for variance reduction. We show that its covariance is closely related to Direct Advantage Estimation (DAE) (Pan et al., 2022), a method that directly estimates the advantage and value functions, and develop a simple actor-critic method that combines both ABC and DAE. We evaluate our approach in an offline-to-online setting, where the critic is first trained on an offline dataset to approximate advantage and value functions for the target policy, and the actor and critic are subsequently updated using online sampled trajectories. We show that, despite

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

its simplicity, this combination achieves performance competitive with Dr. GRPO (Liu et al., 2025), an extension of the original GRPO. To summarize, our contributions are:

1. We present ABC, an advantage-value formulation of trajectory-level control variates that incorporates learned advantage and value functions into policy-gradient estimation while preserving unbiasedness.
2. We theoretically characterize its variance and show that the true advantage and value functions minimize its covariance. In deterministic environments, we bound the excess variance in terms of the return prediction error from the estimated advantage and value functions, motivating DAE as the critic objective.
3. We evaluate the combination of ABC and DAE in an offline-to-online RLVR setup, and show that it achieves competitive performance with Dr. GRPO using as little as 100 policy gradient updates.

2 BACKGROUND

We consider a discounted Markov decision process (MDP) defined by $(\mathcal{S}, \mathcal{A}, r, p, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r(s,a)$ is the reward function, $p(s'|s,a)$ is the transition probability, and $\gamma \in [0,1)$ is the discount factor. We consider the discounted setting here for theoretical considerations, but the results carry over to other settings (e.g., finite horizon) as well. The policy is denoted by $\pi(a|s)$, the value functions for a given policy π are denoted by $V^\pi(s) = \mathbb{E}_\pi\left[\sum_{t=0}^\infty \gamma^t r(s_t,a_t) \mid s_0{=}s\right]$, $Q^\pi(s,a) = \mathbb{E}_\pi\left[\sum_{t=0}^\infty \gamma^t r(s_t,a_t) \mid s_0{=}s, a_0{=}a\right]$, and $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. The objective of RL is to optimize $J(\pi) = \mathbb{E}[V^\pi(s_0)]$.

RLVR In RLVR, states correspond to the prompts and previously generated tokens, while the actions correspond to the generated tokens. The rewards are typically rule-based, for example indicating whether a mathematical answer is correct, and often without discounting (i.e., $\gamma = 1$). The environment dynamics are typically deterministic conditional on the selected action, since appending a token uniquely determines the next state. Finally, we note that RLVR is typically finite-horizon and undiscounted ($\gamma = 1$), and we use the infinite-horizon discounted formulation only for the theoretical presentation.

Policy Optimization One popular method for optimizing parametrized policies is by estimating the gradient of the expected return directly. The REINFORCE algorithm does this by,

$$\nabla J(\theta) = \mathbb{E}_{\pi_\theta}\left[G \sum_{t=0}^{\infty} \nabla \log \pi_\theta(a_t|s_t)\right], \tag{1}$$

where $G = \sum_{t=0}^\infty \gamma^t r(s_t,a_t)$ (Williams, 1992). Equivalently, by the policy gradient theorem, we have $\nabla J(\theta) = \mathbb{E}_{(s,a)\sim d^{\pi_\theta}}\left[G \nabla \log \pi_\theta(a|s)\right] = \mathbb{E}_{(s,a)\sim d^{\pi_\theta}}\left[A^\pi(s,a) \nabla \log \pi_\theta(a|s)\right]$, where d^{π_θ} is the (unnormalized) occupancy measure induced by π_θ (Sutton et al., 1999). We focus on the on-policy case and omit d^{π_θ} in the following unless otherwise stated.

This approach forms the basis of modern policy optimization algorithms such as PPO or GRPO, where we optimize the clipping objective given by:

$$\mathcal{L}(\theta) = \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\min\left(\rho_\theta(s,a)\hat{A}(s,a), \text{clip}\left(\rho_\theta(s,a), 1-\epsilon, 1+\epsilon\right)\hat{A}(s,a)\right)\right], \tag{2}$$

where ϵ is the clipping parameter, $\pi_{\theta_{\text{old}}}$ is the data collecting policy, $\rho_\theta(s,a) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}$ is the importance ratio and $\hat{A}(s,a)$ is an estimate of the advantage function. The importance ratio permits multiple optimization steps on data collected by $\pi_{\theta_{\text{old}}}$, while clipping limits the changes to the policy.

In practice, the advantage function is unknown, and has to be estimated from sampled trajectories. PPO achieves this using a combination of TD(λ) and TD-error (Schulman et al., 2015), while GRPO achieves this by sampling n rollouts from the same initial state (prompt) via

$$\hat{A}_i = G_i - \hat{\mu}, \qquad \hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} G_i, \tag{3}$$

where G_i denote the return of rollout i. Here, $\hat{A}_i$ is a trajectory-level advantage estimate: every state-action pair along rollout i is assigned the same advantage. Note that here, unlike the original GRPO formulation, we do not standardize group-relative advantages, as standardization can introduce undesirable bias (Liu et al., 2025).

Control Variate for Policy Gradient The original REINFORCE algorithm is known to have high variance. As such, it is common for practitioners to incorporate the value function into the policy gradient estimator via $\nabla J(\theta) = \mathbb{E}\left[(G - \hat{V}(s))\nabla \log \pi_\theta(a|s)\right]$, where the value function $\hat{V} \approx V^\pi$ acts as a control variate to reduce the variance (Greensmith et al., 2004). This does not bias the gradient as $\mathbb{E}\left[\hat{V}(s)\nabla \log \pi_\theta(a|s)\right] = 0$ for arbitrary $\hat{V}$, and the choice of $\hat{V}$ only affects the variance of the estimator. While it is common in practice to choose $\hat{V} \approx V^\pi$ as the baseline (Mnih et al., 2016), it should be noted that this choice is neither optimal (Greensmith et al., 2004) nor guaranteed to reduce variance even when $\hat{V} \equiv V^\pi$ (see Appendix A for a counterexample).

It is possible to extend this idea beyond just state-dependent control variates. TrajCV (Cheng et al., 2020) considers trajectory-level control variates, where instead of just using the value of the first state, TrajCV incorporates estimated state-action values $\hat{Q}(s_t, a_t)$ along the trajectory by noticing that $\mathbb{E}[(\hat{Q}(s_t, a_t) - \mathbb{E}[\hat{Q}(s_t, a_t)|s_t])\nabla \log \pi_\theta(a_0|s_0)] = 0$ $(t > 0)$. This can further explain away part of the variance caused by stochastic actions from the trajectory.

Direct Advantage Estimation (DAE) DAE is a method that directly parametrizes and estimates the advantage function from sampled trajectories (Pan et al., 2022; Pan & Schölkopf, 2024). For deterministic environments, DAE minimizes the following constrained least square objective:

$$\mathcal{L}(\hat{A}, \hat{V}) = \mathbb{E}_\mu \left[\left(\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) - \hat{A}(s_t, a_t) \right) - \hat{V}(s_0) \right)^2 \right], \tag{4}$$

$$\text{subject to} \sum_{a \in \mathcal{A}} \pi(a|s)\hat{A}(s, a) = 0 \quad \forall s \in \mathcal{S}, \tag{5}$$

where μ is the behavior policy. Under mild assumptions on the coverages of μ and π, minimizing this constrained objective gives $(A^\pi, V^\pi) = \operatorname{argmin} \mathcal{L}(\hat{A}, \hat{V})$. DAE is particularly attractive for RLVR as it can learn from off-policy data by simply minimizing a least squares without relying on techniques such as importance sampling, which can cause high variance for long trajectories. Finally, we note that we do not use bootstrapping and simply optimize the objective as is.

3 Advantage-Based Control Variate for Policy Gradient

In this section, we present the ABC estimator, which extends value-based control variate by considering advantages from future state-actions, and show how it allows us to combine estimated advantage and value functions without biasing policy gradients. We then study its variance structure and show that minimum variance is achieved when the estimates of the advantage and value functions are exact. Finally, we show that the variance is closely related to the return prediction error from the advantage and value estimates, and draw a connection to DAE.

We first provide some intuitions for the ABC estimator. For state-based control variates, using the value function can be seen as removing the variance of the weighting (i.e., $\mathrm{Var}(G - V^\pi(s_0)) \leq \mathrm{Var}(G)$) that is explained by the initial state. A sampled trajectory, however, contains more information than just the initial state (e.g., intermediate actions). Pan & Schölkopf (2024) showed that the return can be decomposed into a sum of advantages along the trajectory, and the advantages capture the variance caused by the intermediate actions. ABC incorporates this insight into policy gradients to reduce their variance.

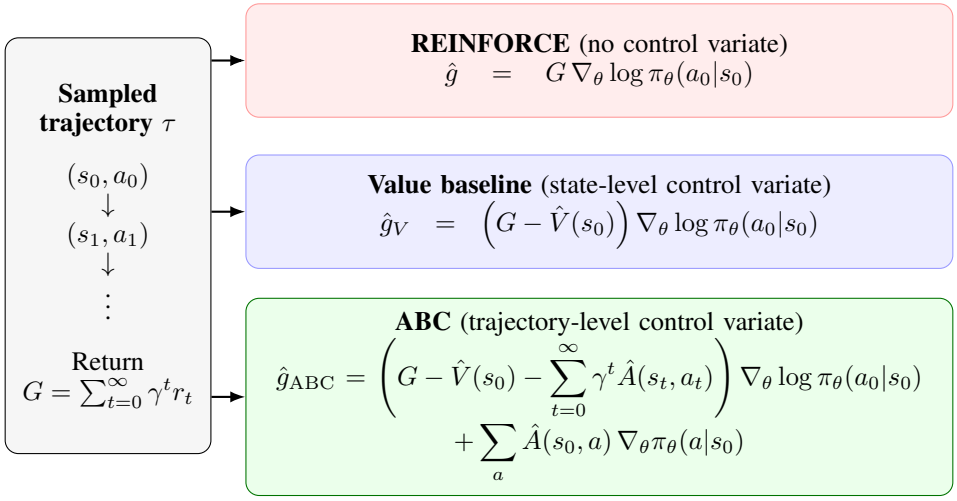


Figure 1: Comparison of unbiased policy-gradient estimators. REINFORCE uses the sampled return directly, while a value baseline removes state-dependent variation. ABC additionally uses estimated advantages as a trajectory-level control variate and adds an analytic correction term to preserve unbiasedness.

3.1 Advantage-Based Control Variate

We begin with the centering property of the advantage function:

$$\sum_a \pi(a|s) A^\pi(s,a) = \sum_a \pi(a|s) Q^\pi(s,a) - V^\pi(s) = 0. \tag{6}$$

This centering property has an interesting implication on the policy gradient, namely:

Lemma 1. *If $\hat{A}$ satisfies the centering property with respect to π_θ, then we have, for $t' \neq t$,*

$$\mathbb{E}_{\pi_\theta}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t)\right] = 0, \tag{7}$$

where the expectation is with respect to the policy π_θ.

See Appendix B for a proof. This shows that if an advantage estimate is centered, then it does not bias the gradient from different time-steps. Intuitively, this is because the advantages only capture the effects from the corresponding actions. Now, we can combine this with an estimated value function to arrive at the following theorem.

Theorem 2 (Advantage-Based Control Variate for Policy Gradient)**.** *Let $\hat{V} : \mathcal{S} \to \mathbb{R}$, and $\hat{A} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ satisfy the centering property. Define the following gradient estimator:*

$$\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V}) = \left(G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)\right) \nabla \log \pi_\theta(a_0|s_0) + \sum_a \hat{A}(s_0, a) \nabla \pi_\theta(a|s_0), \tag{8}$$

where G is the discounted return along $(s_0, a_0, s_1, a_1, \dots)$. First of all, $\hat{g}_{\mathrm{ABC}}$ is an unbiased estimator of the policy gradient. Secondly, define $\Delta V = \hat{V} - V^\pi$, $\Delta A = \hat{A} - A^\pi$,

$$D = -\left(\Delta V(s_0) + \sum_{t=0}^{\infty} \gamma^t \Delta A(s_t, a_t)\right) \nabla \log \pi_\theta(a_0|s_0) + \sum_a \Delta A(s_0, a) \nabla \pi_\theta(a|s_0), \tag{9}$$

and the covariance matrix $\Sigma(\hat{A}, \hat{V}) = \mathrm{Cov}(\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V}))$, we have

$$\Sigma(\hat{A}, \hat{V}) = \Sigma(A^\pi, V^\pi) + \mathbb{E}\left[DD^T\right]. \tag{10}$$

Consequently, $(\hat{A}, \hat{V}) = (A^\pi, V^\pi)$ minimizes the covariance of $\hat{g}_{\mathrm{ABC}}$ (Loewner order).

See Appendix B for a proof. We note that the value-baseline estimator is recovered as a special case of ABC by setting $\hat{A} \equiv 0$. Finally, Theorem 2 shows that using the exact functions (A^π, V^π) yields covariance no worse than any value-baseline, including REINFORCE without a baseline.

Relationship to Rao-Blackwellization Readers familiar with the skill-luck decomposition (Pan & Schölkopf, 2024) may notice that the weighting of the score function $G - \sum_{t=0}^{\infty} \gamma^t A^\pi(s_t, a_t) - V^\pi(s_0)$ is precisely the cumulative effect from the environment stochasticity, $\sum_{t=0}^{\infty} \gamma^t B^\pi(s_t, a_t, s_{t+1})$, where $B^\pi(s_t, a_t, s_{t+1}) = r_t + \gamma V^\pi(s_{t+1}) - Q^\pi(s_t, a_t)$ captures how the stochastic transition from (s_t, a_t) to s_{t+1} affects the return. As such, the estimator can be seen as separating environment noise (first term) from action effects (second term, integrated over the action space). Now, if we consider deterministic environments, where $B^\pi \equiv 0$, such that the following equality holds $G = V^\pi(s_0) + \sum_{t=0}^{\infty} \gamma^t A^\pi(s_t, a_t)$, then the first term (score function) of the ABC estimator vanishes and the estimator reduces to

$$\hat{g}_{\mathrm{ABC}}(A^{\pi_\theta}, V^{\pi_\theta}) = \sum_a A^{\pi_\theta}(s_0, a) \nabla \pi_\theta(a|s_0) = \mathbb{E}\left[A^{\pi_\theta}(s_0, a) \nabla \log \pi_\theta(a|s_0) \mid s_0\right],$$

which is precisely the policy gradient theorem conditioned on the state (Sutton et al., 1999). As such, the ABC estimator (given A^π and V^π) can also be understood as Rao-Blackwellizing the policy gradient estimator $A^{\pi_\theta}(s, a) \nabla \log \pi_{\pi_\theta}(a|s)$ (Blackwell, 1947).

Finally, we note that ABC is algebraically equivalent to TrajCV (Cheng et al., 2020) under the reparameterization $\hat{Q}(s, a) = \hat{A}(s, a) + \hat{V}(s)$, where the centering constraint implies $\mathbb{E}_{a \sim \pi(\cdot|s)}[\hat{Q}(s, a)] = \hat{V}(s)$. However, our advantage-value formulation makes the multi-step return prediction from $(\hat{A}, \hat{V})$ explicit, motivating the connection to DAE developed next.

3.2 Learning Control Variates

In practice, we often only have estimates of (A^π, V^π), and it is not immediately clear how much we suffer from the approximations. Here, we show that this can be bounded by the error we make in predicting the return G.

Corollary 3. *For deterministic MDPs, if $(\hat{A}, \hat{V})$ satisfies $|G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)| < \epsilon$, then*

$$\Sigma(\hat{A}, \hat{V}) - \Sigma(A^\pi, V^\pi) \leq \epsilon^2 F,$$

where $F = \mathbb{E}[\nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T]$ is the Fisher information matrix, and $\leq$ follows the Loewner order.

See Appendix B for a proof. This suggests that, to reduce the variance of policy gradients, we should try to minimize the error $|G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)|$. Notably, the error appearing in this bound is similar to the quantity minimized by the DAE objective (cf. Equation 4). This motivates our practical actor-critic algorithm, which iteratively updates the actor with the ABC gradient estimate and the critic with DAE. In the next section, we empirically verify the performance of this combination.

4 Experiments

In this section, we examine the effectiveness of combining the ABC estimator and DAE (Pan et al., 2022; Pan & Schölkopf, 2024) in RLVR. We consider an offline-to-online actor-critic setting, where an offline dataset of rollouts is available. This allows the critic LLM to learn useful representations and outputs before online training of the actor, which is particularly useful as value and advantage prediction differ substantially from next-token prediction. Figure 2 summarizes the overall training pipeline. Next, we detail the experimental design.

Model Class We consider the Qwen3 family of models, which includes models of various sizes that share the same tokenizer and hence the same action space. This allows us to train asymmetric actor-critic pairs and control for the sizes of the critic independently. For the following experiments, both actors and critics are fine-tuned from base models using LoRA (Hu et al., 2021). We test our method on actors of sizes {0.6/1.7/4}B, while fixing the size of the critic to 8B.

Math Dataset We use the DeepMath-103k dataset (He et al., 2026), a large dataset of challenging math question-answer pairs spanning a wide range of topics. This dataset is used for both offline dataset generation and for online training. For evaluation, we consider the standard benchmark

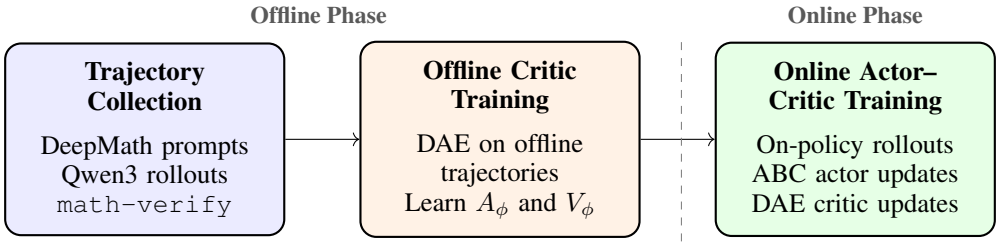


Figure 2: Overview of the offline-to-online training pipeline. We first collect trajectories with verifiable rewards and use DAE to pretrain the critic. During online training, trajectories sampled from the current actor are used both for ABC policy-gradient updates and continued DAE critic training.

Algorithm 1 Online Actor-Critic Training with ABC and DAE

Require: Problem dataset $\mathcal{D}$, Actor π_θ, Critic (A_ϕ, V_ϕ), Batch size N, Num. critic update K

while not converged **do**
  Sample a batch of prompts $\{x_i\}_{i=1,\dots,N} \sim \mathcal{D}$
  Sample trajectories $\{\tau_i\}_{i=1,\dots,N}$ for each prompt using π_θ
  Compute (A_ϕ, V_ϕ) for all states(-actions) in $\{\tau_i\}_{i=1,\dots,N}$ (centered by π_θ)
  Compute the ABC policy-gradient estimator ($|\tau|$ denotes the trajectory length)

$$\hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{|\tau_i|-1} \hat{g}_{\mathrm{ABC}}(A_\phi, V_\phi)(s_t, a_t)$$

  Update the actor parameters θ with gradient $\hat{g}$
  for $k = 1, \dots, K$ **do**
    Sample a minibatch $\mathcal{B} \sim \{\tau_i\}_{i=1,\dots,N}$
    Optimize the DAE critic loss $\mathcal{L}_{\mathrm{DAE}}(\phi; \mathcal{B})$ with respect to policy π_θ
  end for
end while

datasets MATH500 (Lightman et al., 2023), AMC23 (He, 2024), AIME25 (Zhang & Math-AI Team, 2025), and AIME26 (Zhang & Math-AI Team, 2026). We use `math-verify` for answer extraction and verification[1]. Finally, we do not use additional templates for prompting, following the approach by Liu et al. (2025).

Offline Critic Training The offline dataset consists of trajectories generated by the Qwen3-{0.6/1.7/4/8/14}B base models. For each question in DeepMath-103k dataset, we collect 16 rollouts from each model. This results in a dataset of 103k (questions) $\times$ 16 (rollouts) $\times$ 5 (base models) $\approx$ 8.2 million trajectories. We then use the DAE objective (Equation 4) to fine-tune the critic model. For fine-tuning, we use LoRA (Hu et al., 2021) for intermediate layers, and add separate value and advantage heads on top of the base model to parametrize the advantage and value functions, similar in spirit to the dueling architecture (Wang et al., 2016).

Online Actor-Critic Training Algorithm 1 summarizes the online actor-critic training. In contrast to many modern policy optimization methods, we strive to keep the algorithm simple and *do not* use clipping, importance sampling (actor update uses fully on-policy data) or additional regularizations (e.g., KL or entropy regularization). At each iteration, the actor is updated with exactly one gradient step, avoiding the need for off-policy corrections, while the critic is updated for K steps. We train the actor for exactly 100 gradient steps (8,192 trajectories per step, $\sim 0.8M$ trajectories in total).

It should be noted that there is a lag between the critic and the actor: the critic learns from trajectories collected before the actor update, but the target policy is, in fact, the updated actor. This is mitigated by the off-policy nature of DAE, which can estimate the value and advantage functions of a target policy using data collected by a different behavior policy without additional corrections.

[1] `https://github.com/huggingface/math-verify`

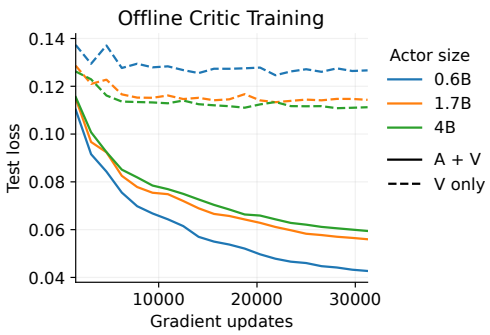


Figure 3: Offline critic training loss.

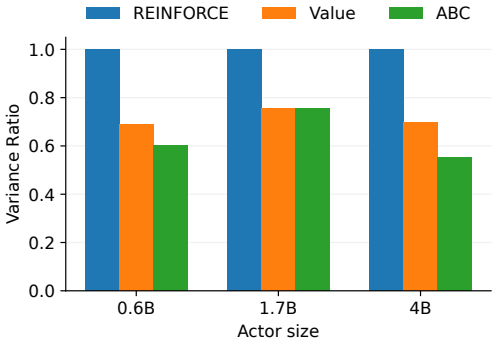


Figure 4: Variance ratio between estimators. REINFORCE $= 1$.

4.1 Results

Offline Advantage & Value Learning We first consider the offline training phase. To evaluate the quality of the estimated advantage and value functions, we split the offline dataset into training and validation sets (10000 trajectories for validation). We note that this phase is simply supervised learning with a mean squared error (MSE) objective (Equation 4). To further assess the quality of the learned advantage, we compare the MSE with or without the advantages. Ideally, adding the advantages should allow the model to better approximate the return and reduce the MSE. Figure 3 shows the offline training loss curves. Including the advantage term substantially reduces held-out MSE, indicating that the learned advantages capture the part of the return that is not explained by the value function alone. Interestingly, we see the gap between the MSE (with or without the advantages) decreases with actor size. This is likely due to larger actors having lower entropy policies (note that the gap reflects how much the variance can be explained by the actions, and the more deterministic the policy is, the less the variance can be explained by the actions), and we see a similar effect over the course of online training (see Figure 10 and Appendix C for additional discussion).

Variance Reduction We now compare the variance of the policy gradients between different methods. We estimate the first gradient step variance by computing per-sample policy gradient over 4096 sample trajectories with respect to each method (REINFORCE, Value baseline, and ABC), and consider the sum of the variances across parameters (i.e., trace of the estimated covariance). Figure 4 shows the ratio of the sum of variances across estimators. Across model sizes, we see that using value baselines already reduces the variance by approximately 30%. We also see that, aside from the 1.7B model, adding advantages (ABC) further reduces the variance compared to the value baseline. For the 1.7B model, we see a few outlier states with extreme critic prediction error, which likely explains why the variance is not reduced (see Appendix C for additional discussion).

Policy Optimization Performance To demonstrate the effectiveness of our method, we compare with Dr. GRPO (Liu et al., 2025), an extension of the original GRPO (Shao et al., 2024) that fixes the bias from advantage standardization and length normalization. The Dr. GRPO baseline is trained with 1 epoch over the training set, corresponding to approximately 6000 gradient steps, or 1.6M generated trajectories (see Appendix C for hyperparameters). Table 1 shows that our method (ABC) achieves performance competitive with the Dr. GRPO baseline on the evaluation datasets. Figure 5 shows the learning curves of ABC, and we find that ABC achieves similar training performance to Dr. GRPO using only about 50 gradients (each actor update corresponds to exactly one policy gradient). In addition, we observe similar increases in completion length under ABC despite using an unbiased policy-gradient estimator, suggesting that this phenomenon is not solely due to biases specific to GRPO (Liu et al., 2025). Finally, we find that the variance reduction is crucial, as REINFORCE is not able to achieve similar performance as Dr. GRPO (see Figure 9).

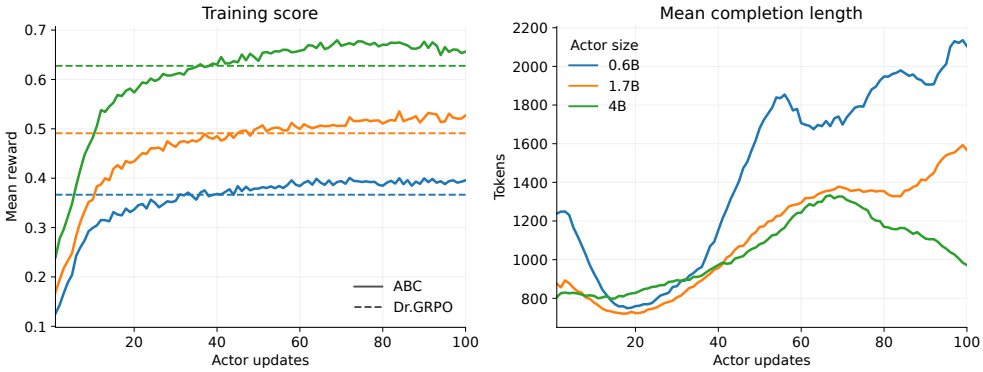


Figure 5: ABC training curves.

Model size	Method	MATH500	AMC23	AIME25	AIME26
0.6B	Dr. GRPO	47.4 / 80.4	25.9 / 67.5	**1.4** / 13.3	**0.8** / **10.0**
	REINFORCE	29.5 / 72.2	15.3 / 65.0	0.5 / 6.7	0.1 / 3.3
	ABC	**53.2** / **83.6**	**29.3** / **75.0**	0.6 / **16.7**	0.7 / **10.0**
1.7B	Dr. GRPO	67.1 / 90.0	39.5 / **85.0**	3.5 / 20.0	**3.5** / 16.7
	REINFORCE	62.3 / 89.6	36.7 / 82.5	2.4 / 13.3	2.4 / 16.7
	ABC	**68.8** / **92.2**	**40.7** / 82.5	**4.8** / **23.3**	3.1 / **20.0**
4B	Dr. GRPO	**80.3** / **95.0**	**59.1** / **90.0**	10.6 / **43.3**	10.7 / **43.3**
	REINFORCE	75.5 / 94.6	52.3 / 87.5	8.0 / 40.0	7.6 / 33.3
	ABC	78.5 / 92.8	**59.1** / 82.5	**13.1** / 40.0	**10.9** / 30.0

Table 1: Final-model evaluation scores in percent, reported as (mean/pass)@32. Mean is the average reward over 32 sampled completions per problem; pass is the percentage of problems with at least one correct completion.

Comparison with Biased Estimator As DAE learns an approximation of the advantage function, one may wonder if we can use this directly in the policy gradient via $\hat{A}(s,a)\nabla \log \pi_\theta(a|s)$. Unlike the unbiased ABC estimator, the performance of this approach depends heavily on the critic, as the actor update depends on the environment reward only through the learned critic. Figure 6 compares the training score between ABC and this biased approach. We see that at initialization, the biased approach actually outperforms ABC, which suggests that the offline training is, in fact, effective in learning the advantage function. However, as learning proceeds, the biased approach slows down while ABC steadily improves. This suggests that critic bias can limit performance even when the initial advantage estimates are informative.

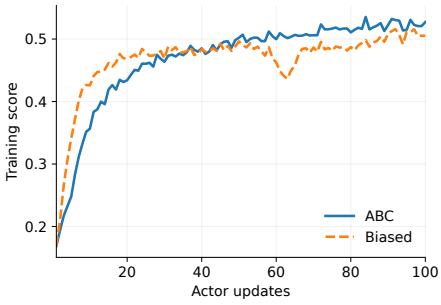


Figure 6: Comparison with biased gradients.

5 Related Work

RLVR RLVR has recently emerged as an effective approach for improving the reasoning capabilities of LLMs, particularly for domains where rewards can be verified automatically. A prominent family of methods builds on Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which removes the learned critic used in PPO and instead estimates advantages by comparing multiple rollouts generated from the same prompt. GRPO has been used extensively in large-scale reasoning models such as DeepSeek-R1 (Guo et al., 2025), and has motivated a number of subsequent methods aimed at improving its stability and efficiency (Yu et al., 2026; Liu et al., 2025; Piché et al., 2025;

Yue et al., 2025; Li et al., 2025). In contrast to critic-free approaches, our method revisits the use of learned advantages, while avoiding the need to estimate them solely from online MC rollouts.

Advantage Estimation and Variance Reduction Reducing the variance of policy-gradient estimators has been studied extensively, beginning with state-dependent baselines and actor-critic methods (Williams, 1992; Sutton et al., 1999), and later through methods such as Generalized Advantage Estimation (GAE) (Schulman et al., 2015). More general control-variate constructions have also been proposed. For example, Q-Prop uses an off-policy critic as a control variate for policy gradients (Gu et al., 2016), while Liu et al. (2017) propose action-dependent control variates based on Stein's identity. Trajectory-level information has also been used for variance reduction; for example, Mesnard et al. (2020) introduce future-conditional value functions that exploit future information while preserving unbiasedness. TrajCV (Cheng et al., 2020) similarly studied trajectory-level control variates using estimated Q-values, and ABC can be viewed as an advantage-value reparameterization, that naturally connects to DAE. Direct Advantage Estimation (DAE) has been shown to improve policy optimization (Pan et al., 2022; Pan & Schölkopf, 2024; 2026a) and reduce the variance of policy evaluation (Pan & Schölkopf, 2026b). In contrast to prior works on DAE, we study how estimates of the advantage function can be used as control variates to reduce policy-gradient variance while preserving an unbiased gradient estimator, which is particularly relevant in light of the strong empirical performance of critic-free RLVR methods.

On-policy Distillation Recent on-policy distillation (OPD) methods also exploit dense teacher feedback on tokens visited by the current policy (Agarwal et al., 2024). In full-vocabulary OPD, objectives such as reverse KL take the form $\sum_a w(s,a)\nabla\pi_\theta(a|s)$, which closely resembles the analytic advantage term in ABC, $\sum_a \hat{A}(s,a)\nabla_\theta\pi_\theta(a|s)$. The two methods differ mainly in the goal of the objectives: OPD uses teacher log-probabilities to move the student toward the teacher policy, whereas ABC uses estimated advantages as a control variate and combines the analytic term with the residual term to preserve an unbiased gradient of the original RL objective.

6 Discussion

Recent RLVR methods have shown that critic-free approaches can perform strongly despite potentially higher variance, raising the possibility that bias introduced through approximate advantage estimation from the critic can be more harmful than the variance they are intended to reduce. The ABC estimator provides a natural way to integrate estimated advantages and values from DAE into the policy gradient estimator without introducing bias. Furthermore, our theoretical analysis demonstrates that the variance of the ABC estimator is closely related to the objective function of DAE, which motivates a natural combination of both algorithms.

Empirically, our offline-to-online RL results suggest that this combination can make critic-based optimization competitive with modern critic-free RLVR methods. This may be especially attractive when large amounts of rollout data already exist, since critic learning can partially be moved to the offline phase rather than starting from scratch during online RL. Furthermore, critic training can in principle reuse off-policy data and proceed independently of rollout collection. Combined with the fact that ABC does not rely on group-based advantage estimates, this decoupling may make the approach more amenable to asynchronous RL settings (Piché et al., 2025; Noukhovitch et al., 2025).

Limitations First, our empirical evaluation focuses on mathematical problems, and it remains to be seen how well ABC transfers to other RL domains and to settings with stochastic environment dynamics. Second, the quality of the control variate depends on the learned critic. Although approximation errors do not bias the ABC estimator, poor estimates may provide little or even negative variance reduction. Third, our current experiments focus primarily on optimization performance rather than end-to-end computational efficiency. Training and maintaining a critic incurs additional memory and computation costs, and a complete systems-level comparison with critic-free methods is an important direction for future work. Finally, understanding how the distribution of the offline data affects the critic is another important direction for future study.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks: generate synthetic data sets (the offline training dataset), formulate mathematical claims, assist in the writing of proofs, design or provide feedback on research methodology or experiments, support qualitative and thematic data analysis. We have not used generative AI tools for the following tasks: help develop theoretical models or conceptual frameworks, provide critical ingredients for proving mathematical claims, implement methods (core algorithms were implemented by the authors), propose or refine hypotheses, clean and reformat dataset, interpret results, assist with translation. Additionally, we used generative AI tools for the following tasks: create or modify scientific figures or images, create or edit software code, draft parts of a research paper, edit a research paper to improve readability. We have reviewed all AI-assisted work, including but not limited to: proof-reading the generated text, importing citations manually, verifying generated scripts. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

David Blackwell. Conditional Expectation and Unbiased Sequential Estimation. *The Annals of Mathematical Statistics*, 18(1):105 – 110, 1947. doi: 10.1214/aoms/1177730497. URL https://doi.org/10.1214/aoms/1177730497.

Ching-An Cheng, Xinyan Yan, and Byron Boots. Trajectory-wise control variates for variance reduction in policy gradient methods. In *Conference on Robot Learning*, pp. 1379–1394. PMLR, 2020.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Zhiwei He. AMC23 dataset, 2024. URL https://huggingface.co/datasets/zwhe99/amc23.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 138306–138322, 2026.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. Repo: Replay-enhanced policy optimization. *arXiv preprint arXiv:2506.09340*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent control variates for policy optimization via stein's identity. *arXiv preprint arXiv:1710.11198*, 2017.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. *arXiv preprint arXiv:2011.09464*, 2020.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Michael Noukhovitch, Shengyi Huang, Sophie Xhonneux, Arian Hosseini, Rishabh Agarwal, and Aaron Courville. Asynchronous rlhf: Faster and more efficient off-policy rl for language models. In *International Conference on Learning Representations*, volume 2025, pp. 4003–4029, 2025.

Hsiao-Ru Pan and Bernhard Schölkopf. Skill or luck? return decomposition via advantage functions. In *The Twelfth International Conference on Learning Representations*, 2024.

Hsiao-Ru Pan and Bernhard Schölkopf. Direct advantage estimation for scalable and sample-efficient deep reinforcement learning. *arXiv preprint arXiv:2606.20411*, 2026a.

Hsiao-Ru Pan and Bernhard Schölkopf. On the variance of temporal difference learning and its reduction using control variates. *arXiv preprint arXiv:2606.20357*, 2026b.

Hsiao-Ru Pan, Nico Gürtler, Alexander Neitz, and Bernhard Schölkopf. Direct advantage estimation. *Advances in Neural Information Processing Systems*, 35:11869–11880, 2022.

Alexandre Piché, Ehsan Kamalloo, Rafael Pardinas, Xiaoyin Chen, and Dzmitry Bahdanau. Pipelinerl: Faster on-policy reinforcement learning for long sequence generation. *arXiv preprint arXiv:2509.19128*, 2025.

Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 9(89):eadi9579, 2024.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on robot learning*, pp. 91–100. PMLR, 2022.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformers Reinforcement Learning, 2020. URL `https://github.com/huggingface/trl`.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025, 2025. URL `https://huggingface.co/datasets/math-ai/aime25`.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2026, 2026. URL `https://huggingface.co/datasets/math-ai/aime26`.

A Baseline Counterexample

Here, we give a concrete example showing that using V^π as the baseline function may increase the variance of policy gradients.

Consider a two-armed bandit problem with action space ($\mathcal{A} = \{0, 1\}$) and rewards $r(0) = 0$, $r(1) = 1$. We parameterize the policy as

$$\pi_\theta(a) = \begin{cases} \frac{1}{1+e^\theta}, & a = 0, \\ \frac{e^\theta}{1+e^\theta}, & a = 1. \end{cases} \tag{11}$$

The score function is then

$$\nabla_\theta \log \pi_\theta(a) = a - \pi_\theta(1), \tag{12}$$

and the exact value function is

$$V^{\pi_\theta} = \pi_\theta(1). \tag{13}$$

Denote $\pi_\theta(1)$ by p, then the REINFORCE estimator is equal to

$$g = (a - p)r(a), \tag{14}$$

and

$$\mathbb{E}[g] = p(1 - p), \quad \mathrm{Var}(g) = p(1 - p)^3. \tag{15}$$

Now, the baseline estimator with exact value function is equal to

$$g_b = (a - p)\,(r(a) - p) = (a - p)^2, \tag{16}$$

and

$$\mathrm{Var}(g_b) = p(1 - p)(1 - 2p)^2. \tag{17}$$

As such, when $(1 - 2p)^2 > (1 - p)^2$ (e.g., $p > \frac{2}{3}$), we have $\mathrm{Var}(g_b) > \mathrm{Var}(g)$.

B PROOFS

Lemma 1. *If $\hat{A}$ satisfies the centering property with respect to π_θ, then we have, for $t' \neq t$,*

$$\mathbb{E}_{\pi_\theta}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t)\right] = 0, \tag{7}$$

where the expectation is with respect to the policy π_θ.

Proof. If $t' > t$, then

$$\begin{aligned}\mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t)\right] &= \mathbb{E}\left[\mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t) \mid s_t, a_t\right]\right] \\ &= \mathbb{E}\left[\mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \mid s_t, a_t\right] \nabla \log \pi_\theta(a_t|s_t)\right] = 0,\end{aligned}$$

where the last equation follows from the centering property. If $t' < t$, then

$$\begin{aligned}\mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t)\right] &= \mathbb{E}\left[\mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \nabla \log \pi_\theta(a_t|s_t) \mid s_{t'}, a_{t'}\right]\right] \\ &= \mathbb{E}\left[\hat{A}\left(s_{t'}, a_{t'}\right) \mathbb{E}\left[\nabla \log \pi_\theta(a_t|s_t) \mid s_{t'}, a_{t'}\right]\right] = 0,\end{aligned}$$

□

Theorem 2 (Advantage-Based Control Variate for Policy Gradient)**.** *Let $\hat{V} : \mathcal{S} \to \mathbb{R}$, and $\hat{A} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ satisfy the centering property. Define the following gradient estimator:*

$$\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V}) = \left(G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)\right) \nabla \log \pi_\theta(a_0|s_0) + \sum_a \hat{A}(s_0, a) \nabla \pi_\theta(a|s_0), \tag{8}$$

where G is the discounted return along $(s_0, a_0, s_1, a_1, \dots)$. First of all, $\hat{g}_{\mathrm{ABC}}$ is an unbiased estimator of the policy gradient. Secondly, define $\Delta V = \hat{V} - V^\pi$, $\Delta A = \hat{A} - A^\pi$,

$$D = -\left(\Delta V(s_0) + \sum_{t=0}^{\infty} \gamma^t \Delta A(s_t, a_t)\right) \nabla \log \pi_\theta(a_0|s_0) + \sum_a \Delta A(s_0, a) \nabla \pi_\theta(a|s_0), \tag{9}$$

and the covariance matrix $\Sigma(\hat{A}, \hat{V}) = \mathrm{Cov}(\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V}))$, we have

$$\Sigma(\hat{A}, \hat{V}) = \Sigma(A^\pi, V^\pi) + \mathbb{E}\left[DD^T\right]. \tag{10}$$

Consequently, $(\hat{A}, \hat{V}) = (A^\pi, V^\pi)$ minimizes the covariance of $\hat{g}_{\mathrm{ABC}}$ (Loewner order).

Proof. We begin by proving that the estimator is unbiased.

$$\mathbb{E}[\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V})] = \mathbb{E}[G \nabla \log \pi_\theta(a_0|s_0)] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{A}(s_t, a_t) \nabla \log \pi_\theta(a_0|s_0)] \tag{18}$$

$$- \mathbb{E}[\hat{V}(s_0) \nabla \log \pi_\theta(a_0|s_0)] + \mathbb{E}[\sum_a \hat{A}(s_0, a) \nabla \pi_\theta(a|s_0)] \tag{19}$$

By Lemma 1, $\mathbb{E}[\hat{A}(s_t, a_t) \nabla \log \pi_\theta(a_0|s_0)] = 0$ for $t > 0$. Next, use the log derivative trick, we have

$$\mathbb{E}[\sum_a \hat{A}(s_0, a) \nabla \pi_\theta(a|s_0)] = \mathbb{E}[\hat{A}(s_0, a_0) \nabla \log \pi_\theta(a_0|s_0)]$$

Consequently,

$$\mathbb{E}[\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V})] = \mathbb{E}[G \nabla \log \pi_\theta(a_0|s_0)]. \tag{20}$$

This concludes the first part of the proof. Next, we consider the covariance of the estimator. To simplify the notation, we first denote $\hat{g}^\pi_{\mathrm{ABC}} = \hat{g}_{\mathrm{ABC}}(A^\pi, V^\pi)$. It follows that

$$\hat{g}_{\mathrm{ABC}}(\hat{A}, \hat{V}) = \hat{g}^\pi_{\mathrm{ABC}} + D. \tag{21}$$

The covariance follows

$$\Sigma(\hat{A}, \hat{V}) = \Sigma(A^\pi, V^\pi) + \mathbb{E}[DD^T] + \mathrm{Cov}(\hat{g}^\pi_{\mathrm{ABC}}, D) + \mathrm{Cov}(D, \hat{g}^\pi_{\mathrm{ABC}}), \tag{22}$$

where $\mathrm{Cov}(D) = \mathbb{E}[DD^T]$ as $\mathbb{E}[D] = 0$ (unbiasedness). Consequently, we only need to show that

$$\mathrm{Cov}(\hat{g}^\pi_{\mathrm{ABC}}, D) + \mathrm{Cov}(D, \hat{g}^\pi_{\mathrm{ABC}}) = 0 \tag{23}$$

We first decompose the return of a trajectory into:

$$G = V^\pi(s_0) + \sum_{t=0}^{\infty} \gamma^t \left(A^\pi(s_t, a_t) + B^\pi(s_t, a_t, s_{t+1}) \right), \tag{24}$$

where $B^\pi(s_t, a_t, s_{t+1}) = r_t + \gamma V^\pi(s_{t+1}) - Q^\pi(s_t, a_t)$ (Pan & Schölkopf, 2024). Note that B^π satisfies $\mathbb{E}[B^\pi(s_t, a_t, s_{t+1})|s_t, a_t] = 0$. We can now rewrite the estimator:

$$\hat{g}^\pi_{\mathrm{ABC}} = \left(\sum_{t=0}^{\infty} \gamma^t B^\pi(s_t, a_t, s_{t+1}) \right) \nabla \log \pi_\theta(a_0|s_0) + \sum_a A^\pi(s_0, a) \nabla \pi_\theta(a|s_0). \tag{25}$$

Note that the second term $\sum_a A^\pi(s_0, a) \nabla \pi_\theta(a|s_0)$ can be viewed as a function of s_0. As such, $\mathbb{E}[(\sum_a A^\pi(s_0, a) \nabla \pi_\theta(a|s_0)) D^T] = 0$.

For the B^π terms, we show that

$$\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] = 0. \tag{26}$$

If $t' \leq t$,

$$\begin{aligned}
&\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] \\
=&\mathbb{E}[\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T | s_0, a_0, \ldots, s_t, a_t]] \\
=&\mathbb{E}[\mathbb{E}[B^\pi(s_t, a_t, s_{t+1})|s_t, a_t] \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] \\
=&0
\end{aligned}$$

If $t' > t$,

$$\begin{aligned}
&\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] \\
=&\mathbb{E}[\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \Delta A(s_{t'}, a_{t'}) \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T | s_0, a_0, \ldots, s_{t'}]] \\
=&\mathbb{E}[B^\pi(s_t, a_t, s_{t+1}) \mathbb{E}[\Delta A(s_{t'}, a_{t'})|s_{t'}] \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] \\
=&0
\end{aligned}$$

The terms involving $\Delta V(s_0) \nabla \log \pi_\theta(a_0|s_0)$ and $\Delta A(s_0, a) \nabla \pi_\theta(a|s_0)$ similarly vanish by the same argument. Consequently, $\mathrm{Cov}(\hat{g}^\pi_{\mathrm{ABC}}, D) = 0$, and

$$\mathrm{Cov}(\hat{g}^\pi_{\mathrm{ABC}}, D) + \mathrm{Cov}(D, \hat{g}^\pi_{\mathrm{ABC}}) = 0. \tag{27}$$

□

Remark The covariance optimality result applies to individual gradient contributions. In practice (e.g., Algorithm 1), we sum the gradients along the trajectories. This introduces additional covariances between the gradients across time-steps, so optimality of the individual gradients does not, in general, imply the optimality of their sum. Unbiasedness, however, is preserved.

Corollary 3. *For deterministic MDPs, if* $(\hat{A}, \hat{V})$ *satisfies* $|G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)| < \epsilon$*, then*

$$\Sigma(\hat{A}, \hat{V}) - \Sigma(A^\pi, V^\pi) \leq \epsilon^2 F,$$

where $F = \mathbb{E}[\nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T]$ *is the Fisher information matrix, and* $\leq$ *follows the Loewner order.*

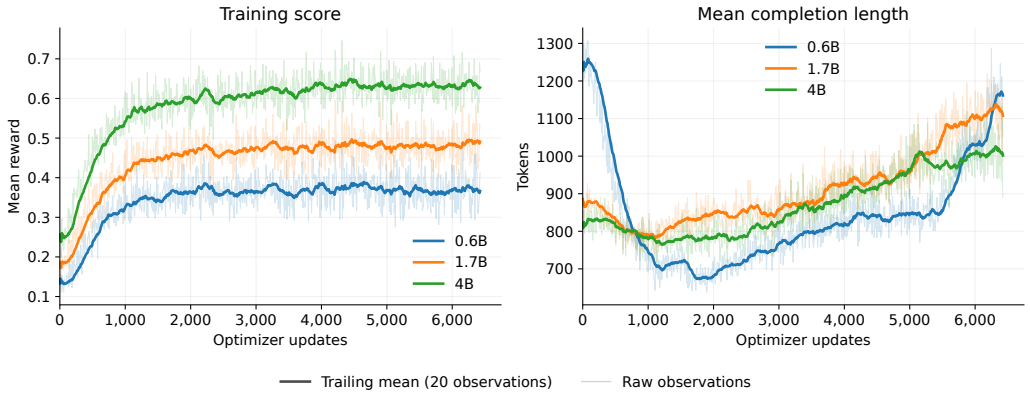


Figure 7: Dr. GRPO training curves.

Proof. From Theorem 2, we have

$$\Sigma(\hat{A}, \hat{V}) = \Sigma(A^\pi, V^\pi) + \mathbb{E}\left[DD^T\right]. \tag{28}$$

We first denote $\delta = G - \sum_{t=0}^{\infty} \gamma^t \hat{A}(s_t, a_t) - \hat{V}(s_0)$. D can then be rewritten into

$$D = \delta \nabla \log \pi_\theta(a_0|s_0) - \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0]. \tag{29}$$

We have

$$\begin{aligned} DD^T &= \delta^2 \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T && (30)\\ &- \delta \nabla \log \pi_\theta(a_0|s_0) \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0]^T && (31)\\ &- \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0] \delta \nabla \log \pi_\theta(a_0|s_0)^T && (32)\\ &+ \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0] \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0]^T. && (33) \end{aligned}$$

By law of total expectation, we have

$$\begin{aligned} \mathbb{E}[DD^T] &= \mathbb{E}[\delta^2 \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] && (34)\\ &- \mathbb{E}[\mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0] \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0]^T]. && (35) \end{aligned}$$

Since $\mathbb{E}[\mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0] \mathbb{E}[\delta \nabla \log \pi_\theta(a_0|s_0)|s_0]^T]$ is positive semi-definite, we have

$$\mathbb{E}[DD^T] \leq \mathbb{E}[\delta^2 \nabla \log \pi_\theta(a_0|s_0) \nabla \log \pi_\theta(a_0|s_0)^T] \leq \epsilon^2 F \tag{36}$$

□

C Experiment Details

C.1 Offline Dataset

We generate the offline dataset using models of different sizes to have a wider coverage of trajectories. The dataset is generated once and used for all subsequent experiments. We use vLLM (Kwon et al., 2023) for efficient trajectory generation.

C.2 Additional Results

GRPO Figure 7 shows the Dr. GRPO training curves. Despite the length bias correction of Dr. GRPO, we still see steady increase of completion lengths over the course of training. This is, in fact, consistent with our method, where we also see increases in completion length despite being unbiased.

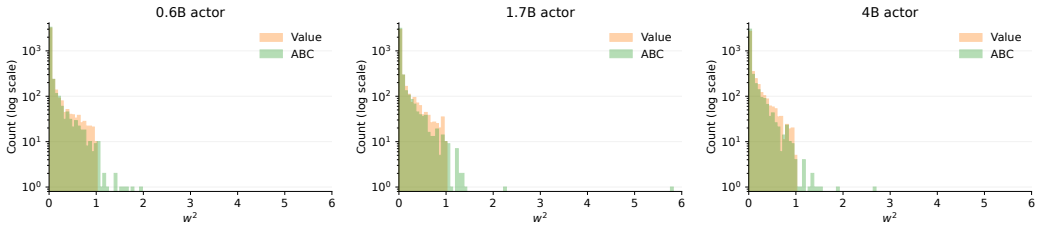


Figure 8: Residual (w) squared distribution.

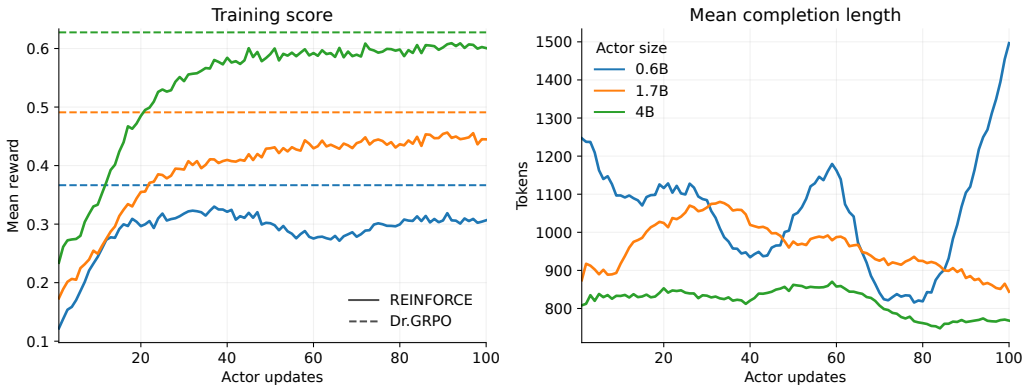


Figure 9: REINFORCE training curves.

Gradient Variance Figure 4 showed that the 1.7B model does not benefit from the advantages. Recall that the policy gradient is weighted by either $(G - \hat{V}(s_0))$ (value baseline) or $(G - \hat{V}(s_0) - \sum_{t=0}^{\infty} \hat{A}(s_t, a_t))$ (ABC). We denote this weighting by w. Corollary 3 suggests that variance is closely related to this residual error. Figure 8 shows that, in general, ABC's w distributions tend to be closer to 0. However, for the 1.7B model, we see an outlier with $w^2 > 5$, which may explain why the ABC gradient variance is higher.

REINFORCE Figure 9 summarizes the learning curves for REINFORCE.

Entropy and Critic Loss In Section 4, we observed the pattern that larger actors tend to have a smaller gap between loss with and without advantages. Figure 10 shows the critic loss with respect to different actor sizes. For the 4B model, we see that the gap almost vanishes towards the end of training. This can likely be explained by the gradual decrease of the entropy. Recall that the advantage function explains away part of the variance of the return caused by the actions. As such, if actions are near deterministic, then the advantages will become less significant.

C.3 Hyperparameters

Dr. GRPO baseline We use the implementation by von Werra et al. (2020), which includes various corrections that are not present in the original implementation (e.g., training-inference mismatch). Table 2 summarizes the hyperparameters used for the GRPO baselines in Section 4. The hyperparameters largely follow the ones used by Liu et al. (2025), except that we increase the completion length to 4096, increase the number of generations per prompt from 8 to 16, and use LoRA for fine-tuning. We have also tried tuning the learning rates but found that larger learning rates often lead to performance collapsing.

Offline DAE Critic Training Table 3 summarizes the hyperparameters used for the offline training. We found that using a much smaller learning rate for the advantage head significantly stabilizes learning. In our early experiments, we tried using the same learning rate across all parameters and found that the loss became unstable. This is likely due to the fact that the DAE objective involves a

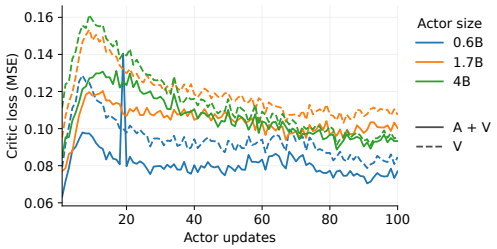


Figure 10: Critic loss during online training.

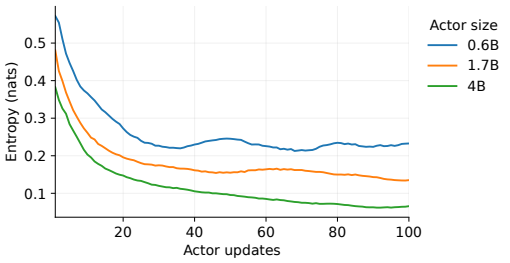


Figure 11: Policy entropy during online training.

Hyperparameter	Value
Optimization and objective	
Objective	Dr. GRPO (Liu et al., 2025)
Optimizer	AdamW
Learning rate	1×10^{-6} (constant)
Adam $(\beta_1, \beta_2, \epsilon)$	$(0.9, 0.95, 10^{-8})$
Weight decay	0
Maximum gradient norm	1.0
Policy-ratio clipping ϵ	0.2
KL penalty coefficient	0
Entropy bonus coefficient	0
Policy importance-sampling granularity	Token
vLLM importance-sampling correction	Sequence mask, upper threshold 3.0
Batching and rollout sampling	
Microbatch size	16 completions
Gradient accumulation steps	16
Completions per prompt	16
Prompt groups per rollout batch	128
Global rollout batch size	2048 completions
Optimizer updates per rollout batch	8
Training passes per rollout batch	1
Maximum completion length	4096 tokens
Parameter-efficient training	
LoRA rank / α	128 / 256
LoRA target layers	all hidden linear layers

Table 2: Hyperparameters for the GRPO baselines.

sum over the advantages across time-steps, which makes it more sensitive to the output advantages. In addition, we initialize the advantage and value heads to zero, which also stabilizes early training.

Online Actor-Critic Training Table 4 summarizes the hyperparameters for the online RL training. In general, the actor configuration follows the ones used in the Dr. GRPO baseline (e.g., LoRA configuration, Adam β). The critic hyperparameters largely follow the ones used in the offline training phase (cf. Table 3), except we lowered the learning rate. For critic training, we only pass through the online-collected trajectories once. As such, the number of gradient steps per iteration (K in Algorithm 1) is $\frac{8192}{256} = 32$.

C.4 Implementation Details

Reducing memory cost Training with DAE requires enforcing the centering constraint, which is done in practice by parametrizing the function in the following way:

$$\hat{A}_\theta(s, a) = f_\theta(s, a) - \sum_a \pi(a|s) f_\theta(s, a).$$

Hyperparameter	Value
Optimization and objective	
Optimizer	AdamW
Batch size	256 trajectories
Learning rate (advantage head)	2.5×10^{-6}
Learning rate (others)	2.5×10^{-5}
Learning rate schedule	Linear warmup (5%) followed by linear annealing
Adam $(\beta_1, \beta_2, \epsilon)$	$(0.9, 0.99, 10^{-8})$
Weight decay	0
Parameter-efficient training	
LoRA rank / α (rsLoRA)	128 / 32
LoRA target layers	all hidden linear layers

Table 3: Hyperparameters for the offline critic training.

Hyperparameter	Value
Shared Hyperparameters	
Optimizer	AdamW
Learning rate schedule	constant
Weight decay	0
Rollout Batch size	8192 trajectories
Actor Hyperparameters	
Learning rate (actor)	1×10^{-5}
Adam $(\beta_1, \beta_2, \epsilon)$	$(0.9, 0.95, 10^{-8})$
LoRA rank / α	128 / 256
LoRA target layers	all hidden linear layers
Critic Hyperparameters	
Learning rate (advantage head)	1×10^{-6}
Learning rate (others)	1×10^{-5}
Adam $(\beta_1, \beta_2, \epsilon)$	$(0.9, 0.99, 10^{-8})$
Mini Batch size	256 trajectories

Table 4: Hyperparameters for the online actor-critic training.

This ensures that the centering constraint is always satisfied. However, naively implementing this can be expensive as it requires storing two tensors ($\pi(a|s)$, $f_\theta(s,a)$) of shape (T, V), where T is the sequence length and V is the vocabulary size (typically in the order of $\sim$100000). Note that the output layer is linear with respect to the output of the base model (i.e., $f_\theta(s,a) = h_\theta(s)W$, where W is the output linear layer for the advantage head with dimension (C, V)). Since C ($\sim$1000) is usually much smaller than V, we can reorder the operation by:

$$\begin{aligned}\sum_{a'} \pi(a'|s) f_\theta(s,a') &= \sum_{a'} \sum_{c} \pi(a'|s) h_\theta(s)_c W_{ca'} \\ &= \sum_{c} \left(\sum_{a'} W_{ca'} \pi(a'|s) \right) h_\theta(s)_c.\end{aligned}$$

This avoids constructing the full tensor $f_\theta(s,a')$, and reduces its memory cost from (T, V) to (T, C). For future work, it might be valuable to try FlashAttention style fusion of the softmax-value product to further reduce memory footprint (Dao et al., 2022).

Centering in the ABC estimator Recall that the ABC estimator requires computing the following term:

$$\sum_a \hat{A}(s,a) \nabla \pi_\theta(a|s).$$

Note that in practice, we center the advantages via:

$$\hat{A}(s,a) = f(s,a) - \sum_{a'} \pi(a'|s) f(s,a'),$$

where f is the underlying approximator (linear advantage head output). This means that $\hat{A}(s,a)$ and $f(s,a)$ differ only by a function of s (i.e., $\sum_a \pi(a|s)f(s,a)$). As we only care about the gradient for the policy in this case, we can ignore the centering because:

$$\sum_a \hat{A}(s,a)\nabla\pi_\theta(a|s) = \sum_a f(s,a)\nabla\pi_\theta(a|s) - \sum_a \left(\sum_{a'} \pi(a'|s)f(s,a')\right)\nabla\pi_\theta(a|s) \tag{37}$$

$$= \sum_a f(s,a)\nabla\pi_\theta(a|s) - \left(\sum_{a'} \pi(a'|s)f(s,a')\right)\left(\sum_a \nabla\pi_\theta(a|s)\right) \tag{38}$$

$$= \sum_a f(s,a)\nabla\pi_\theta(a|s) \tag{39}$$