BELLMAN RESIDUAL MINIMIZATION FOR SAMPLE-EFFICIENT DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work on improving the sample efficiency of deep reinforcement learning has largely focused on network architecture design, normalization schemes, and optimizer choices. The loss function itself, however, has received comparatively little attention, even though it plays a central role in shaping training dynamics. We show that enforcing an additional Bellman consistency objective on top of the semi-gradient loss stabilizes training at high replay ratios. Our method demonstrates competitive performance with concurrent methods on the hardest DeepMind Control Suite tasks like humanoid and dog, without relying on network resets or additional normalization techniques.

1 INTRODUCTION

In real-world settings such as robotics, collecting new interaction data can be costly and time-consuming (Kober et al., 2013), making it important to extract as much learning as possible from each transition. Off-policy reinforcement learning (RL) provides a natural way to do this: experience can be stored in a replay buffer and reused for multiple gradient updates. Increasing this replay ratio has led to substantial improvements in sample efficiency, but standard algorithms eventually become unstable as the number of updates per sample grows (Nikishin et al., 2022; D'Oro et al., 2023; Hussing et al., 2024). Recent work has addressed this problem from several directions: scaling network capacity (Nauman et al., 2024b; Lee et al., 2025a;b), introducing normalization and regularization schemes (Bhatt et al., 2024; Palenicek et al., 2025; Hussing et al., 2024; Palenicek et al., 2026), using critic ensembles or dropout (Chen et al., 2021; Hiraoka et al., 2022), and periodically resetting network parameters (Nikishin et al., 2022; D'Oro et al., 2023). These approaches have been effective, but they primarily change the function approximator or the optimization around it. Recent methods have also changed the form of the critic loss, for example by replacing squared error with distributional objectives, but the bootstrapped Bellman target is still generally treated as fixed during differentiation (Lee et al., 2025b; Palenicek et al., 2026). As deep RL moves toward higher-compute regimes, a basic question arises: *should we also revisit the critic update itself?*

Classical residual algorithms provide a natural starting point. Baird (1995) showed that standard temporal-difference methods based on semi-gradient updates can become unstable with function approximation, and instead proposed differentiating through both sides of the Bellman error to minimize the mean squared Bellman residual. Zhang et al. (2019) later revisited this idea in deep RL and showed that residual algorithms can be practical in near-deterministic continuous-control settings. We revisit this idea for modern high-replay-ratio actor-critic learning. Starting from Soft Actor-Critic (SAC) (Haarnoja et al., 2018), we augment the standard critic objective with a Bellman residual minimization (BRM) objective, in which the online critic is used for the bootstrap target and gradients flow through both sides of the Bellman error.

A remaining question is how strongly to weight the residual objective. Baird (1995) already observed that the trade-off between semi-gradient and residual-gradient updates need not remain fixed throughout training, and proposed adapting the mixing coefficient. We follow the same principle, but use a different adaptation rule suited to high-replay-ratio training. Rather than fixing the residual weight throughout training, we adapt it based on the critic's value estimates, allowing the residual objective to strengthen when the critic begins to show signs of overestimation and relax otherwise. This keeps the update close to standard SAC when critic values remain below the reference, while

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.

strengthening the residual objective when they grow excessively. We call the resulting method *BRM-SAC*.

Poor scaling at high replay ratios has been linked to overfitting and loss of plasticity, with validation TD error and value-function divergence emerging as useful indicators of training failure (Nikishin et al., 2022; D'Oro et al., 2023; Li et al., 2023; Hussing et al., 2024). We find that, during instability, validation TD error, critic value estimates, and critic gradient norms rise and fall together, suggesting that they reflect the same underlying failure. This motivates using the critic's Q-values, which are already available during training, as a simple signal for adapting the residual weight. BRM in turn suppresses the large increases in value estimates and critic gradients seen with standard SAC at high replay ratios. At a replay ratio of 32, BRM-SAC remains stable and performs competitively with recent sample-efficient methods on challenging humanoid and dog tasks from the DeepMind Control Suite, without network resets or additional normalization techniques.

In summary, our contributions are:

1. We revisit residual Bellman gradients in the high-replay-ratio regime and introduce BRM-SAC, which augments the standard SAC critic objective with a residual Bellman objective while leaving the actor and temperature updates unchanged.
2. We introduce an adaptive mixing coefficient that increases the influence of the residual objective when the critic exhibits excessive value growth and otherwise keeps training close to standard SAC.
3. We study the training dynamics of SAC at high replay ratios and show that Q-values and critic gradient norms closely track increases in held-out TD error. BRM suppresses these instability signals and enables stable training at a replay ratio of 32, with competitive performance on challenging DMC tasks without network resets or additional normalization.

2 PRELIMINARIES

Off-policy actor-critic and SAC. We focus on online off-policy actor-critic methods. Off-policy learning allows transitions in the replay buffer to be reused for many gradient updates, while in the online setting the resulting policy determines which data are collected next. This interaction between learning and data collection is important at high replay ratios (D'Oro et al., 2023). Actor-critic methods can be viewed as approximate policy iteration, alternating between policy evaluation and policy improvement (Haarnoja et al., 2018). Since repeated updates at high replay ratios are applied primarily to the critic, in this work we are essentially focusing on improving the policy-evaluation step.

We consider the standard continuous-control reinforcement learning setting, modeled as an MDP $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$ with continuous state space $\mathcal{S}$, continuous action space $\mathcal{A}$, transition dynamics $p(s' \mid s, a)$, reward function $r(s, a)$, and discount factor $\gamma \in (0, 1)$. A stochastic policy $\pi_\psi(a \mid s)$ selects actions, and the value of taking action a in state s and then following π_ψ is the Q-function, which satisfies the Bellman equation

$$Q^{\pi_\psi}(s, a) \;=\; \mathbb{E}\left[r(s, a) + \gamma\, \mathbb{E}_{a' \sim \pi_\psi(\cdot|s')}\left[Q^{\pi_\psi}(s', a')\right]\right]. \tag{1}$$

Soft Actor-Critic (SAC) (Haarnoja et al., 2018) approximates Q^{π_ψ} with two critics $Q_{\theta_1}, Q_{\theta_2}$ (parameters $\boldsymbol{\theta} = \{\theta_1, \theta_2\}$), trained off-policy on transitions (s, a, r, s') from a replay buffer $\mathcal{D}$, and takes the minimum of the two to reduce overestimation. SAC further augments Equation 1 with an entropy term, so that the policy is optimized to maximize both reward and action entropy, weighted by a temperature α. To stabilize training against the bootstrapped target, both critics maintain target networks $Q_{\bar{\theta}_1}, Q_{\bar{\theta}_2}$, Polyak-averaged copies of $\boldsymbol{\theta}$ updated after each gradient step as $\bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}$. Given a fresh next action $a' \sim \pi_\psi(\cdot \mid s')$, the critic loss regresses each $Q_{\theta_i}(s, a)$ towards

$$y(r, s') \;=\; r + \gamma\left(\min_{i=1,2} Q_{\bar{\theta}_i}(s', a') - \alpha \log \pi_\psi(a' \mid s')\right), \tag{2}$$

using a target network in place of $Q_{\boldsymbol{\theta}}$ itself. The number of such critic updates performed per environment step is the *replay ratio*.

Semi-gradient and residual updates. Because $y(r, s')$ in Equation 2 is computed with the target parameters $\bar{\boldsymbol{\theta}}$, it is treated as a constant when differentiating the critic loss, giving

$$\nabla_{\theta_i} \left(Q_{\theta_i}(s,a) - y(r,s')\right)^2 \;\approx\; -2\big(y(r,s') - Q_{\theta_i}(s,a)\big)\, \nabla_{\theta_i} Q_{\theta_i}(s,a), \tag{3}$$

a *semi-gradient* update: it corrects the prediction at (s, a) but ignores how changing θ_i would also change $y(r, s')$ through $Q_{\theta_i}(s', a')$, since that dependency has been severed by using the target network. Baird (1995) showed that such updates are not the true gradient of any fixed objective, which is part of why they can diverge under off-policy training and nonlinear function approximation (Sutton & Barto, 2018). If $y(r, s')$ instead used the online parameters θ_i directly, differentiating through both terms would give

$$\nabla_{\theta_i} \left(Q_{\theta_i}(s,a) - y(r,s')\right)^2 \;=\; -2\big(y(r,s') - Q_{\theta_i}(s,a)\big)\, \Big(\gamma\, \nabla_{\theta_i} Q_{\theta_i}(s',a') - \nabla_{\theta_i} Q_{\theta_i}(s,a)\Big), \tag{4}$$

which Baird (1995) calls a *residual gradient* update: it performs true gradient descent on the mean squared Bellman error, rather than the mean squared *projected* Bellman error that semi-gradient methods effectively minimize, and carries convergence guarantees that semi-gradient updates lack. Residual gradients see little practical use, largely because Equation 4 is a product of two terms that both depend on the stochastic transition to s'; estimating it from a single sampled s' introduces a bias, since in general $\mathbb{E}[XY] \neq \mathbb{E}[X]\mathbb{E}[Y]$ for correlated random variables X and Y. An unbiased estimate instead requires two *independent* samples of s' from the same (s, a), a requirement known as the double-sampling problem, which is difficult to satisfy from a single environment rollout in general stochastic environments. This obstacle disappears in near-deterministic environments (Zhang et al., 2019): if s' is a deterministic function of (s, a), as is approximately the case in the MuJoCo-based DeepMind Control Suite tasks we consider in this work, a single sampled transition already gives an unbiased target, and the double-sampling problem does not arise.

3 Related Work

3.1 Sample Efficiency and Replay-Ratio Scaling

Sample efficiency in model-free RL has been improved through several complementary strategies. BRO scales regularized critics (Nauman et al., 2024b), SimBa and SimBaV2 introduce architectures designed for increasing network capacity (Lee et al., 2025a;b), and Ma et al. (2025) use network sparsity to improve scaling. Parameter resets address primacy bias (Nikishin et al., 2022) and enable training at substantially higher replay ratios (D'Oro et al., 2023), while FlashSAC combines larger models and greater data throughput with explicit control of weight, feature, and gradient norms (Kim et al., 2026). Other work focuses more directly on stable data reuse: REDQ uses critic ensembles and randomized target minimization (Chen et al., 2021), while DroQ replaces the large ensemble with dropout and normalization (Hiraoka et al., 2022). High replay ratios have also been linked to overfitting (Li et al., 2023); Nauman et al. (2024a) study overestimation, overfitting, and loss of plasticity as distinct sources of instability, while Sokar et al. (2023) identify dormant neurons as a source of reduced network expressivity and propose recycling them during training. CrossQ uses batch normalization to achieve strong sample efficiency at a low replay ratio (Bhatt et al., 2024), CrossQ+WN adds weight normalization to scale to higher replay ratios (Palenicek et al., 2025), and XQC combines batch normalization, weight normalization, and a distributional critic loss to improve critic optimization (Palenicek et al., 2026). BRM-SAC takes a complementary approach by modifying the critic objective itself. It does not prescribe a particular architecture, normalization scheme, or reset mechanism, and can therefore in principle be combined with these approaches. Our results suggest that residual gradients themselves are an underused lever for improving sample efficiency.

3.2 Residual and Gradient-Based TD Methods

Residual-gradient methods were introduced by Baird (1995), who proposed directly minimizing the mean squared Bellman residual and a broader family of residual algorithms that interpolate between residual-gradient and semi-gradient updates. In stochastic environments, estimating the true residual gradient from sampled transitions faces the double-sampling problem. Zhang et al. (2019) brought residual algorithms to deep RL, focusing on near-deterministic continuous-control environments

and introducing a bidirectional target network for a residual version of DDPG. A related line of work instead uses gradient-TD methods, which optimize projected Bellman-error objectives through auxiliary correction estimates (Sutton et al., 2009; Ghiassian et al., 2020; Patterson et al., 2022; Elelimy et al., 2025). Our setting permits a simpler approach: because the DMC tasks considered here have near-deterministic transitions, we directly differentiate the Bellman residual using the online critic on both sides, without a bidirectional target network or auxiliary correction terms, and study whether this update improves sample efficiency in high-replay-ratio SAC.

4 BRM-SAC

Our contribution is confined entirely to the critic objective; the actor and temperature updates are unchanged from standard SAC. Section 4.1 introduces our residual Bellman loss and how it is combined with the standard critic loss, and Section 4.2 describes how the coefficient controlling this combination is tuned adaptively during training.

4.1 Augmenting the Critic Objective with a Bellman Residual Loss

Given a transition (s, a, r, s') sampled from $\mathcal{D}$ and a fresh next action $a' \sim \pi_\psi(\cdot \mid s')$, the standard SAC critic loss is

$$\mathcal{L}_{\text{back}}(\boldsymbol{\theta}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\sum_{i=1,2}\left(Q_{\theta_i}(s,a) - y(r,s')\right)^2\right], \tag{5}$$

$$y(r,s') = r + \gamma\left(\min_{i=1,2} Q_{\bar{\theta}_i}(s',a') - \alpha \log \pi_\psi(a' \mid s')\right). \tag{6}$$

Because $y(r, s')$ depends only on the target parameters $\bar{\boldsymbol{\theta}}$, gradients flow only through $Q_{\theta_i}(s, a)$ on the left-hand side, this is a semi-gradient update. Semi-gradient updates ignore how a change in $\boldsymbol{\theta}$ would itself change the bootstrapped target, a known source of instability when function approximation is combined with bootstrapping and off-policy learning (Sutton & Barto, 2018).

A residual Bellman loss. We introduce a second loss term that evaluates the same Bellman backup, but replaces the target network with the online network when computing the bootstrapped value at the next state. This target is then regressed against the online prediction at (s, a), so that both sides of the resulting loss depend on the same set of online parameters. Concretely, we draw a second, independently-sampled next action $\hat{a}' \sim \pi_\psi(\cdot \mid s')$, and define a residual target for each critic $i \in \{1, 2\}$,

$$\hat{y}_i(r,s';\boldsymbol{\theta}) \;=\; r + \gamma\left(Q_{\theta_i}(s',\hat{a}') - \alpha \log \pi_\psi(\hat{a}' \mid s')\right), \tag{7}$$

which mirrors $y(r, s')$ in Equation 2 but replaces the target network $Q_{\bar{\theta}_i}$ with the online network Q_{θ_i}. The residual loss is then

$$\mathcal{L}_{\text{res}}(\boldsymbol{\theta}) \;=\; \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\sum_{i=1,2}\left(\hat{y}_i(r,s';\boldsymbol{\theta}) - Q_{\theta_i}(s,a)\right)^2\right]. \tag{8}$$

Unlike $\mathcal{L}_{\text{back}}$, both sides of Equation 8 depend on the online parameters $\boldsymbol{\theta}$, so gradients flow through *both* the prediction $Q_{\theta_i}(s, a)$ and the bootstrapped target $\hat{y}_i$. This is a true residual-gradient update in the sense of Baird (1995), rather than a semi-gradient one, and it is this property that directly constrains how far the online critic is allowed to violate its own Bellman equation, rather than allowing errors to compound one-sidedly through the target network.

Combined critic objective. The two losses are combined with a scalar mixing coefficient $\phi \geq 0$, tuned adaptively as described in Section 4.2:

$$\mathcal{L}_{\text{critic}}(\boldsymbol{\theta}) \;=\; \tfrac{1}{2}\,\mathcal{L}_{\text{back}}(\boldsymbol{\theta}) + \phi \cdot \tfrac{1}{2}\,\mathcal{L}_{\text{res}}(\boldsymbol{\theta}). \tag{9}$$

When $\phi = 0$, Equation 9 reduces exactly to the standard SAC critic loss; the auxiliary term only pulls weight once the training dynamics call for it.

4.2 Adaptively Tuning the Mixing Coefficient ϕ

Fixing ϕ to a constant requires committing, in advance, to how strongly the residual loss should regularize training, independent of whether an agent is currently at risk of diverging. Instead, we treat ϕ as a dual variable that grows only when the critic shows early signs of overestimation, and shrinks back towards zero once training is well-behaved.

An upper bound on the optimal value. Under the mild assumption that per-step rewards are bounded, $r_t \in [0, r_{\max}]$, the optimal value function admits a simple upper bound,

$$Q^*(s,a) \;\le\; \frac{r_{\max}}{1-\gamma} \;\triangleq\; Q_{\text{ceil}} \tag{10}$$

Q_{ceil} depends only on the (known) reward scale and discount factor, so it can be computed once, before training begins, and used as a reference point: if the critic's own value estimates exceed Q_{ceil}, this is a direct, easily-measured symptom of overestimation.

Dual ascent on ϕ. At each update, we compute the critics' mean value on the current minibatch $\mathcal{B}$,

$$\bar{Q}_{\mathcal{B}} \;=\; \frac{1}{2}\,\mathbb{E}_{(s,a)\sim\mathcal{B}}\left[Q_{\theta_1}(s,a) + Q_{\theta_2}(s,a)\right], \tag{11}$$

and update ϕ by gradient ascent on

$$\mathcal{L}_\phi(\phi) \;=\; -\,\phi\left(\bar{Q}_{\mathcal{B}} - Q_{\text{ceil}}\right), \tag{12}$$

where $\bar{Q}_{\mathcal{B}}$ is treated as constant with respect to ϕ. A single Adam step on Equation 12, followed by projection onto the non-negative orthant, gives the update

$$\phi \;\leftarrow\; \max\left(0,\; \phi + \eta_\phi\left(\bar{Q}_{\mathcal{B}} - Q_{\text{ceil}}\right)\right), \tag{13}$$

where η_ϕ is the (Adam-derived) step size. Intuitively, whenever the critics' average value exceeds the ceiling Q_{ceil}, ϕ increases, placing more weight on the residual loss $\mathcal{L}_{\text{res}}$ and pulling training back towards stability; whenever values sit safely below the ceiling, ϕ relaxes back towards zero, and Equation 9 reduces towards the standard SAC objective. This mirrors the automatic temperature-tuning mechanism already used for α in SAC (Haarnoja et al., 2018), but here the dual variable is driven by a divergence signal rather than an entropy constraint. Because ϕ is updated using only $\bar{Q}_{\mathcal{B}}$ and Q_{ceil}, this adaptation adds negligible computational overhead, requiring no extra forward or backward passes through the critic networks beyond those already needed for Equation 9.

Connections to other adaptive dual variables. The update in Equation 13 is an instance of a broader pattern in deep RL: introducing a constraint on some quantity of interest and enforcing it by dual gradient ascent on a scalar multiplier, rather than fixing that multiplier by hand. The clearest precedent is SAC's own entropy temperature α, which is tuned by the same mechanism to enforce a target entropy constraint (Haarnoja et al., 2018). The same idea has been used to constrain other quantities of interest: Kumar et al. (2020) tune the conservatism coefficient of CQL by dual gradient ascent on a target gap between in-distribution and out-of-distribution Q-values, rather than fixing it as a constant, and report that this variant is noticeably more robust across tasks than a fixed coefficient.

It is worth noting that updates of this form, including ours, are a form of *integral* control: ϕ accumulates (integrates) the constraint violation $\bar{Q}_{\mathcal{B}} - Q_{\text{ceil}}$ over the course of training, with no direct dependence on how quickly that violation is changing. Stooke et al. (2020) show that this integral-only structure is precisely what causes the oscillations and overshoot commonly observed in Lagrangian-style adaptive coefficients, and propose augmenting the update with proportional and derivative terms to damp these dynamics. We did not find it necessary to introduce such terms for ϕ, but this connection suggests a natural direction for making the adaptation more responsive if oscillatory behavior is observed in other settings.

5 BRM Stabilizes High Replay Ratio Training

We evaluate whether adding the residual Bellman objective allows SAC to train reliably at high replay ratios, and whether the resulting increase in data reuse translates into improved sample effi-

ciency. We focus on the more challenging `humanoid` and `dog` domains from the DeepMind Control Suite, and compare BRM-SAC against recent methods designed for sample-efficient continuous-control learning.

5.1 Main Experiments

Experimental setup. Following D'Oro et al. (2023), we use a 500k-environment-step budget to evaluate sample efficiency. We evaluate on 19 state-based continuous-control tasks from the DeepMind Control Suite (DMC) (Tassa et al., 2018). In the main text, we focus on the challenging `humanoid-stand`, `humanoid-walk`, `humanoid-run`, `dog-stand`, and `dog-walk` tasks; results on all DMC15 environments are provided in Appendix A. We report the training episodic returns against the number of environment steps. Our results are averaged over 5 random seeds, with shaded regions denoting 95% confidence intervals. We focus on online off-policy actor-critic methods, where replay buffers allow multiple critic updates per environment interaction.

Baselines. We compare against several strong model-free and model-based baselines. SAC (Haarnoja et al., 2018) is the standard off-policy maximum-entropy actor-critic algorithm on which our method is built. SR-SAC (D'Oro et al., 2023) uses periodic network resets to enable SAC to train at higher replay ratios. TD-MPC2 (Hansen et al., 2024) is a model-based method that learns a latent world model and performs local trajectory optimization in the learned latent space. MRQ (Fujimoto et al., 2025) is a model-free actor-critic method that uses model-based auxiliary objectives, including reward and dynamics prediction, to learn its representations. Finally, we include BRO (Nauman et al., 2024b), which scales SAC using a substantially larger critic together with distributional Q-learning, optimistic exploration, and periodic resets. BRO is not a directly matched comparison to our setting, since increasing network capacity is central to its design, whereas BRM modifies only the critic objective and is orthogonal to network scaling. Nevertheless, we include it as a strong reference point on these tasks. For the same reason, we do not include SimBa and its variants (Lee et al., 2025a;b), whose primary contribution is an architecture designed specifically to enable parameter scaling.

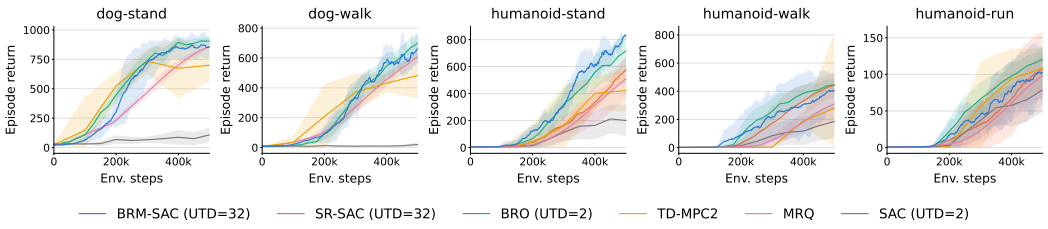


Figure 1: Results on the `dog` and `humanoid` environments

Implementation details. We use two configurations of the same implementation. The default configuration is used for all tasks except the `dog` domain and uses a two-hidden-layer critic MLP with width 256. The `dog` observations are substantially higher-dimensional, so for these tasks we use a 512-wide critic, delayed policy updates, and AdamW weight decay. These are standard implementation choices used in recent strong continuous-control methods (Bhatt et al., 2024; Palenicek et al., 2025; Lee et al., 2025a), rather than components specific to BRM. All remaining hyperparameters, including the complete settings for both configurations, are reported in Appendix B. Unless stated otherwise, BRM-SAC is trained at a replay ratio of 32. All experiments were run on NVIDIA V100 GPUs; a 500k-step run at replay ratio 32 takes approximately nine hours on a single GPU.

Results. Figure 1 shows the main comparison on the five `humanoid` and `dog` tasks considered above. BRM-SAC trains reliably at a replay ratio of 32 and remains competitive with the strongest baselines across these tasks without network resets or additional normalization. The same dog-specific configuration does not yield similarly strong performance on `dog-trot` and `dog-run`; we report these results together with the remaining DMC15 tasks in Appendix A. Because the dog setup includes a few additional changes besides BRM, we ablate them in Section 5.3. We find that the gains depend critically on BRM and cannot be explained by these changes alone.

5.2 Symptoms of High Replay Ratio Training

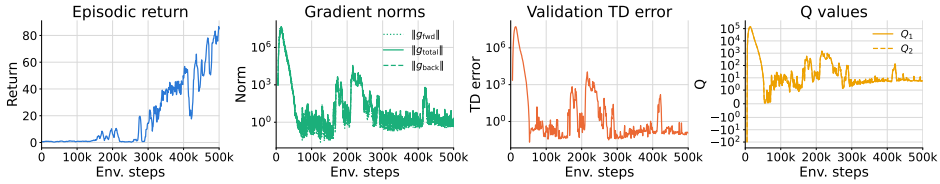


Figure 2: A training run on `humanoid-run` at a replay ratio of 64. Critic gradient norm, validation TD error, and Q-values rise and decay together during two instability episodes.

Training at high replay ratios is known to degrade performance, but there is no consensus on a cheap and reliable signal a practitioner can monitor during training to identify when a run is at risk. Li et al. (2023) identify TD error on a held-out validation set as the diagnostic most consistently correlated with this failure. Computing it requires holding out transitions and periodically evaluating the critic on them, which most training runs do not do by default. We ask whether signals already computed during ordinary training track validation TD error closely enough to replace it.

Gradient norm and Q-values track validation TD error. Across environments and random seeds, we consistently observe that the critic gradient norm and Q-values track changes in validation TD error. Figure 2 shows a representative run, where all three quantities rise and decay together during two instability episodes. Unlike the held-out TD error, gradient norm and Q-values are already available during ordinary training without an additional evaluation pass. This makes them inexpensive online proxies.

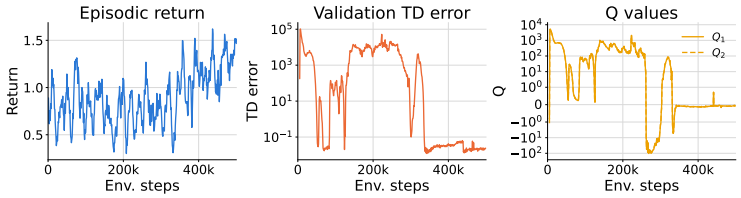


Figure 3: `humanoid-run` at replay ratio 64. Validation TD error remains low despite no improvement in episodic return.

Validation TD error is not infallible. Figure 3 shows a run where validation TD error falls to and stays near its lowest observed value while episodic return stays flat. A low validation TD error here does not mean the agent is learning: the critic's Q-values settle to a near-constant value close to zero rather than to a meaningful estimate of return. Validation TD error is informative when it is high, not when it is low. Li et al. (2023) nevertheless find validation TD error to be among the diagnostics most consistently associated with high-replay-ratio failure, and since gradient norm and Q-values track it during genuine instability, controlling Q-value growth directly through the residual loss in Section 4.1 is a reasonable substitute for controlling validation TD error itself, which cannot be optimized against without held-out data collected for that purpose alone.

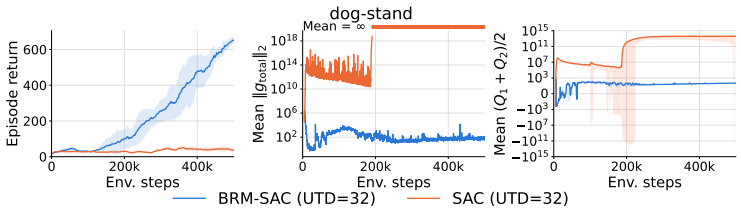


Figure 4: Comparison on `dog-stand` at replay ratio 32. BRM-SAC prevents the growth in critic gradients and Q-values observed with standard SAC.

BRM suppresses these instability signals. Across environments and random seeds, we find that BRM suppresses the large increases in gradient norm and Q-values observed with standard SAC at high replay ratios. Figure 4 shows a representative comparison on `dog-stand` at a replay ratio of 32. With standard SAC, both quantities grow by several orders of magnitude as training fails, while with BRM they remain stable and episodic return continues to improve. This shows that BRM suppresses the same instability signals that track increases in validation TD error under high replay ratios.

5.3 Ablations

We next isolate the contribution of the main components of BRM-SAC. We first remove the residual objective entirely, reducing the method to standard SAC under the same training setup, and then study the role of the adaptive coefficient ϕ.

5.3.1 The residual objective

We first ask whether the improvement comes from the residual objective itself. We set $\phi = 0$, which removes the BRM term and recovers standard SAC, while keeping all other settings unchanged. Figure 5 shows a large drop in performance on both `dog-stand` and `dog-walk` when the residual objective is removed. BRM-SAC learns faster and reaches substantially higher returns, showing that the residual objective is a critical component of the improvement at high replay ratios.

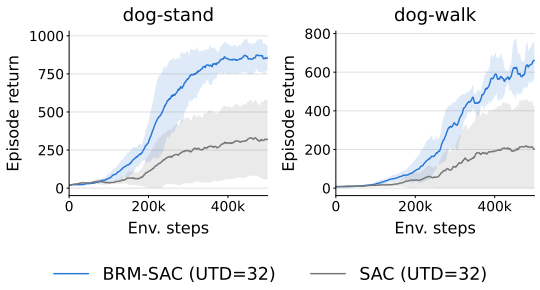


Figure 5: Ablating the residual objective on `dog-stand` and `dog-walk`. Removing BRM recovers standard SAC and substantially reduces performance at replay ratio 32.

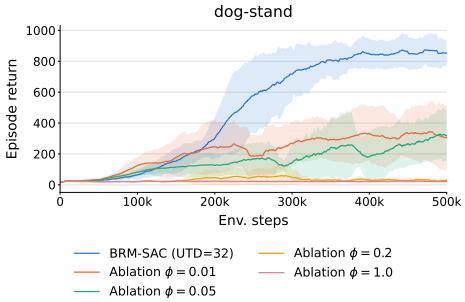


Figure 6: Adaptive versus fixed residual weights on `dog-stand`. No fixed value of ϕ matches adaptive BRM-SAC at replay ratio 32.

5.3.2 Adaptive tuning of the coefficient

We next ask whether ϕ needs to be adapted during training, or whether a fixed residual weight is sufficient. We compare BRM-SAC against fixed $\phi \in \{0.01, 0.05, 0.2, 1.0\}$, keeping all other settings unchanged. Figure 6 shows that none of the fixed values matches the performance of adaptive BRM-SAC. Small coefficients improve learning but remain substantially below the adaptive method, while larger coefficients perform poorly. This shows that the appropriate balance between the semi-gradient and residual objectives changes during training, and motivates adapting ϕ rather than treating it as a fixed hyperparameter.

6 Conclusion

We showed that revisiting the critic update itself can substantially improve high-replay-ratio training. BRM-SAC augments the standard SAC critic loss with a residual Bellman objective and adapts its weight using the critic's value scale. At high replay ratios, BRM suppresses the large growth in Q-values and critic gradients that accompanies failure in standard SAC, while achieving strong sample efficiency on challenging DMC tasks. These results suggest that residual gradients, despite receiving relatively little attention in modern deep RL, remain a useful and complementary tool

for stabilizing aggressive data reuse. More broadly, our findings motivate looking beyond architecture and normalization choices and reconsidering the learning objective itself as RL moves toward higher-compute regimes.

AI USE STATEMENT

A large language model was used solely to edit for clarity, grammar, and overall readability. We reviewed the output as necessary and take full responsibility for the content of this publication.

REPRODUCIBILITY STATEMENT

Section 4 fully specifies the BRM-SAC objective and adaptive update, while the implementation details in Section 5 and the complete hyperparameters in Appendix B provide all settings used in our experiments. Together, these details are sufficient to reproduce our results on a single NVIDIA V100 GPU.

REFERENCES

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine learning proceedings 1995*, pp. 30–37. Elsevier, 1995.

Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *International conference on learning representations*, volume 2024, pp. 55293–55311, 2024.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model. *International Conference on Learning Representations (ICLR)*, 2021.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. *International Conference on Learning Representations (ICLR)*, 2023.

Esraa Elelimy, Brett Daley, Andrew Patterson, Marlos C. Machado, Adam White, and Martha White. Deep reinforcement learning with gradient eligibility traces. *Reinforcement Learning Journal*, 6: 2530–2550, 2025.

Scott Fujimoto, Pierluca D'Oro, Amy Zhang, Yuandong Tian, and Michael Rabbat. Towards general-purpose model-free reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 83509–83552, 2025.

Sina Ghiassian, Andrew Patterson, Shivam Garg, Dhawal Gupta, Adam White, and Martha White. Gradient temporal-difference learning with regularized corrections. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3524–3534. PMLR, 2020.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *International Conference on Learning Representations*, 2022.

Marcel Hussing, Claas Voelcker, Igor Gilitschenski, Amir-massoud Farahmand, and Eric Eaton. Dissecting deep RL with high update ratios: Combatting value divergence. *Reinforcement Learning Journal (RLC)*, 2024.

Donghu Kim, Youngdo Lee, Minho Park, Kinam Kim, I Nahendra, Takuma Seno, Sehee Min, Daniel Palenicek, Florian Vogt, Danica Kragic, et al. Flashsac: Fast and stable off-policy reinforcement learning for high-dimensional robot control. *arXiv preprint arXiv:2604.04539*, 2026.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 46816–46848, 2025a.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. *arXiv preprint arXiv:2502.15280*, 2025b.

Qiyang Li, Aviral Kumar, Ilya Kostrikov, and Sergey Levine. Efficient deep reinforcement learning requires regulating overfitting. *arXiv preprint arXiv:2304.10466*, 2023.

Guozheng Ma, Lu Li, Zilin Wang, Li Shen, Pierre-Luc Bacon, and Dacheng Tao. Network sparsity unlocks the scaling potential of deep reinforcement learning. *arXiv preprint arXiv:2506.17204*, 2025.

Michal Nauman, Michał Bortkiewicz, Piotr Miłoś, Tomasz Trzcinski, Mateusz Ostaszewski, and Marek Cygan. Overestimation, overfitting, and plasticity in actor-critic: the bitter lesson of reinforcement learning. *International Conference on Machine Learning (ICML)*, 2024a.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. *Advances in neural information processing systems*, 37:113038–113071, 2024b.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. *International Conference on Machine Learning (ICML)*, 2022.

Daniel Palenicek, Florian Vogt, Joe Watson, and Jan Peters. Scaling off-policy reinforcement learning with batch and weight normalization. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2071.

Daniel Palenicek, Florian Vogt, Joe Watson, Ingmar Posner, and Jan Peters. XQC: Well-conditioned optimization accelerates deep reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2026.

Andrew Patterson, Adam White, and Martha White. A generalized projected bellman error for off-policy value estimation in reinforcement learning. *Journal of Machine Learning Research*, 23 (145):1–61, 2022.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In *International Conference on Machine Learning*, pp. 32145–32168. PMLR, 2023.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International conference on machine learning*, pp. 9133–9143. PMLR, 2020.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, second edition, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

Richard S. Sutton, Hamid R. Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 993–1000, 2009.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018. URL https://arxiv.org/abs/1801.00690.

Shangtong Zhang, Wendelin Boehmer, and Shimon Whiteson. Deep residual reinforcement learning. *arXiv preprint arXiv:1905.01072*, 2019.

A DMC CURVES

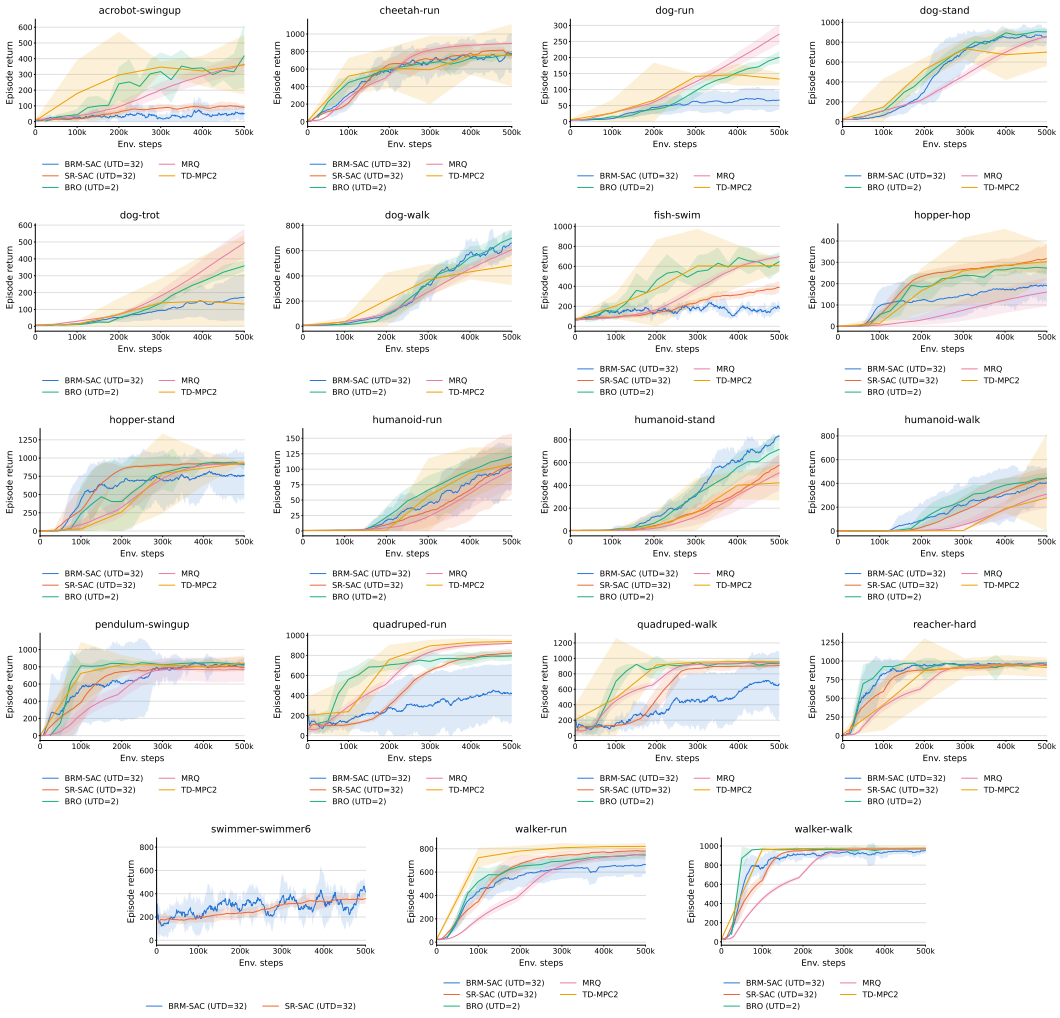


Figure 7: Results across DeepMind Control Suite environments.

B HYPERPARAMETERS

Table 1: Hyperparameters used for BRM-SAC. Values not listed are set to the JAX/Optax defaults for their respective optimizer.

Hyperparameter	**Value**
Training	
Environment steps	500,000
Replay buffer size	1,000,000
Batch size	256
Learning starts (random exploration steps)	5,000
Action repeat	1
Discount factor γ	0.99
Target network smoothing coefficient τ	0.005
Actor-Critic (SAC)	
Actor learning rate	3×10^{-4}
Critic learning rate	3×10^{-4}
Entropy temperature (α) learning rate	1×10^{-4}
Initial temperature α_0	1.0
Target entropy	$-\lvert\mathcal{A}\rvert/2$
Policy update frequency	every critic update
Temperature update frequency	every critic update
Optimizer (AdamW)	
β_1	0.9
β_2	0.999
ϵ	1.5×10^{-4}
Actor weight decay	0.0
Critic weight decay	0.0
Temperature weight decay	0.0
Network Architecture	
Actor hidden dimension	128
Actor hidden layers	2
Critic hidden dimension	256
Critic hidden layers	2
Activation	ReLU
Residual Objective (ϕ)	
Replay ratio (critic updates per env. step)	32
Initial mixing coefficient ϕ_0	0.0
Mixing coefficient learning rate η_ϕ	3×10^{-4}
Mixing coefficient optimizer	Adam ($\beta_1, \beta_2, \epsilon$ shared with critic)

Table 2: Hyperparameters used for BRM-SAC for the dog environment. Values not listed are set to the JAX/Optax defaults for their respective optimizer.

Hyperparameter	**Value**
Training	
Environment steps	500,000
Replay buffer size	1,000,000
Batch size	256
Learning starts (random exploration steps)	5,000
Action repeat	1
Discount factor γ	0.99
Target network smoothing coefficient τ	0.005
Actor-Critic (SAC)	
Actor learning rate	3×10^{-4}
Critic learning rate	3×10^{-4}
Entropy temperature (α) learning rate	1×10^{-4}
Initial temperature α_0	1.0
Target entropy	$-\lvert\mathcal{A}\rvert/2$
Policy update frequency	every 3 env steps
Temperature update frequency	every 1 critic update
Optimizer (AdamW)	
β_1	0.9
β_2	0.999
ϵ	1.5×10^{-4}
Actor weight decay	0.0
Critic weight decay*	0.01
Temperature weight decay	0.0
Network Architecture	
Actor hidden dimension	128
Actor hidden layers	2
Critic hidden dimension	512
Critic hidden layers	2
Activation	ReLU
Residual Objective (ϕ)	
Replay ratio (critic updates per env. step)	32
Initial mixing coefficient ϕ_0	0.0
Mixing coefficient learning rate η_ϕ	3×10^{-4}
Mixing coefficient optimizer	Adam

*Weight decay is applied only to the second hidden layer of the critic; all other critic parameters use plain Adam (no decay).