RAM-A: BOUNDED ADVANTAGE REGRESSION FOR FLOW POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-matching policies provide an expressive class of action distributions for robotic control, but reinforcement learning with such policies remains challenging. Existing approaches such as Flow Policy Optimization (FPO) construct policy updates through exponentiated differences of flow-matching losses and rely on proximal clipping to stabilize optimization. We introduce RAM-A, an advantage-based extension of Reinforce Adjoint Matching (RAM) that casts reinforcement learning as regression toward advantage-shifted flow-velocity targets. We show that RAM-A and FPO share the same first-order policy-improvement direction at the behavior policy, but differ under repeated optimization of a rollout batch, where unbounded advantage regression diverges for negative advantages. RAM-A instead bounds the norm of the regression target, which makes each update self-limiting and robust to corrupted advantage estimates. We evaluate RAM-A across dense-reward continuous control and sparse-reward robotic manipulation, together with targeted experiments on multimodality and multitask retention. While matching existing methods in task performance, RAM-A retains both modes of a multimodal pretrained policy and better preserves skills not targeted by the fine-tuning reward. Pretrained generative policies can thus be improved one skill at a time without erasing their diversity or capabilities.

1 INTRODUCTION

Diffusion and flow-matching models are expressive policy representations for continuous control, capable of modeling complex, multimodal action distributions through iterative generative processes (Chi et al., 2023; Lipman et al., 2023). These models are typically trained through supervised denoising or velocity regression rather than direct optimization of action probabilities. In robotics, this has enabled policies trained from demonstrations to generate coordinated, high-dimensional actions and action sequences (Physical Intelligence et al., 2025; Kim et al., 2024; Brohan et al., 2023). Yet, as the success of reward-based post-training in large language models has demonstrated, supervised learning can provide a powerful starting point without fully optimizing task performance (Ouyang et al., 2022). For control policies, reinforcement learning (RL) offers a natural next step by allowing them to improve from the outcomes of their own interactions with the environment. This is useful both for fine-tuning pretrained policies beyond the behaviors present in demonstrations and for learning generative policies directly from reward feedback.

Applying on-policy reinforcement learning to flow policies, however, exposes a mismatch between two training paradigms. Policy-gradient methods rely on action log-likelihoods and their ratios (Schulman et al., 2017). Flow policies are instead trained by regressing a velocity field and define their action distribution only implicitly through its integration. Existing methods bridge this gap in one of two ways. They either make the sampling process stochastic so that transition likelihoods become tractable (Ren et al., 2025; Zhang et al., 2025; Liu et al., 2025), or they replace the likelihood ratio by a surrogate built from flow-matching losses (McAllister et al., 2025; Yi et al., 2026). Both routes yield policy-gradient-style objectives that differ from the regression objective used in pretraining. A third route keeps the regression objective and moves the reward into its target (Fan et al., 2025; Xue et al., 2026; Bergmeister et al., 2026). These methods have so far been used to optimize rewards on complete generated outputs, such as images, rather than for sequential decision-making.

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.

Translating regression-based policy improvement to on-policy actor–critic learning raises two distinct optimization challenges. First, on-policy methods reuse each rollout batch for several optimization epochs. The batch is collected by the behavior policy, that is, the policy at the start of the iteration, and its advantage estimates are valid for that policy. As the actor deviates from the behavior policy over the course of these epochs, the estimates no longer reflect the consequences of the actor's own actions. Regression-based objectives do not inherently control this movement. Second, incorporating signed, critic-estimated advantages introduces additional difficulties. Negative advantages induce repulsive updates, while large or erroneous estimates can produce excessive displacements of the regression target. These issues are particularly relevant in sparse-reward robotic control, where advantage estimates are obtained from finite rollouts and a learned critic.

To address these challenges, we build on Reinforce Adjoint Matching (RAM), which expresses reward-based generative-model fine-tuning as regression toward reward-dependent velocity targets (Bergmeister et al., 2026). We adapt this formulation to sequential on-policy actor–critic learning using advantages estimated through generalized advantage estimation and investigate its behavior under repeated actor optimization. We establish its relationship to regularized advantage-weighted flow matching and FPO, and analyze the effect of signed advantages on the resulting regression updates. Our analysis reveals that sufficiently negative advantages can render the effective sample-level regression objective unbounded below, while large advantage estimates can induce excessive target displacements even when a finite optimum exists. Motivated by these observations, we introduce RAM-A, which incorporates norm-bounded regression targets to limit the influence of individual advantage estimates. To additionally control accumulated policy movement during repeated actor optimization, we introduce a separate proximal update mechanism relative to the behavior policy. Together, these mechanisms distinguish three aspects of policy optimization: anchoring to a reference policy, bounding individual reward-dependent regression corrections, and controlling movement relative to the policy that generated the current rollout batch. We evaluate RAM-A in dense-reward continuous control and sparse-reward robotic manipulation, investigating optimization stability, robustness to advantage outliers, and the effects of reference-policy anchoring. Our main contributions are:

- **Sequential advantage regression and optimization analysis.** We adapt RAM to on-policy actor–critic learning with temporally estimated advantages, establish its relationship to existing flow policy optimization objectives, and characterize the optimization behavior of signed-advantage regression.
- **Bounded and proximal policy optimization.** We introduce norm-bounded regression targets and combine them with separate behavior-policy proximity control to regulate individual advantage-dependent corrections and accumulated policy movement during repeated actor updates.
- **Empirical evaluation.** We show that RAM-A matches existing flow-policy reinforcement-learning methods in task performance while preserving the solution modes of a multimodal pre-trained policy and the skills of a multitask policy that the fine-tuning reward does not exercise.

2 Related Work

Diffusion and flow matching. Diffusion and flow-matching models have emerged as powerful generative modeling frameworks for representing complex, high-dimensional distributions (Ho et al., 2020; Lipman et al., 2023), with applications spanning image generation (Rombach et al., 2022), sequence modeling (Chen et al., 2024), and control (Chi et al., 2023). While diffusion models learn to reverse a stochastic noising process, flow matching learns a time-dependent velocity field $v_\theta(x_\tau, \tau)$ that transports samples from a simple source distribution to a target data distribution over flow time $\tau \in [0, 1]$ (Lipman et al., 2023). For a source sample $x_0 \sim \mathcal{N}(0, I)$ and a data sample x_1, the linear conditional path $x_\tau = \tau x_1 + (1 - \tau)x_0$ induces the conditional target velocity $u = x_1 - x_0$, and the velocity field is trained through the supervised regression objective $\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{x_0, x_1, \tau}\|v_\theta(x_\tau, \tau) - u\|_2^2$ with $\tau \sim \mathcal{U}[0, 1]$. After training, samples are generated by integrating the learned velocity field from $\tau = 0$ to $\tau = 1$, starting from source samples.

Reinforcement learning for diffusion and flow models. Applying conventional policy-gradient methods to diffusion and flow policies is challenging because their objectives are typically formulated in terms of action log-likelihoods or likelihood ratios (Schulman et al., 2017). While these

quantities are readily available for simple Gaussian policies, evaluating the marginal likelihood of an action generated through an iterative denoising or flow process can be computationally expensive (Song et al., 2021; Ren et al., 2025). Existing methods differ in how they obtain a policy-improvement signal without this likelihood.

Transition likelihoods. One family of approaches introduces tractable stochastic transitions within the generative sampling process. Diffusion Policy Policy Optimization (DPPO) treats the denoising process as an inner Markov decision process and applies PPO using the Gaussian densities of individual denoising transitions (Ren et al., 2025). ReinFlow enables policy-gradient fine-tuning of flow-matching policies by injecting learnable stochasticity into their sampling trajectories, enabling tractable transition-likelihood evaluation (Zhang et al., 2025). Similarly, Flow-GRPO converts the deterministic flow ODE into an SDE to enable online policy-gradient optimization of flow-matching models (Liu et al., 2025). These approaches avoid evaluating the marginal endpoint likelihood, but their policy-gradient objectives depend on the stochastic transition model used during optimization.

Likelihood-ratio surrogates. A second family constructs policy-improvement objectives from the supervised matching losses used during generative pretraining. Flow Policy Optimization (FPO) constructs an exponential surrogate ratio from differences in conditional flow-matching losses and incorporates it into a PPO-style clipped objective (McAllister et al., 2025). FPO++ extends this approach with per-sample ratios and asymmetric policy optimization to regulate policy updates (Yi et al., 2026).

Reward-dependent regression. A third family incorporates the reward directly into the matching objective, so that the model is trained with the squared-error loss of pretraining and its likelihood is never evaluated. ORW-CFM-W2 weights the flow-matching loss by a nonnegative function of the reward and regularizes the velocity field toward a reference model (Fan et al., 2025). AWM derives an advantage-weighted regression gradient from a GRPO-style objective (Xue et al., 2026); its signed weights allow both attractive and repulsive updates, and it considers reference-model regularization but no separate trust region toward the behavior policy for repeated optimization on a fixed batch. RAM, on which we build, considers the KL-regularized fine-tuning of a reference distribution p_{ref} with velocity field v_{ref} under a scalar reward $R(x)$ (Bergmeister et al., 2026),

$$p^* = \arg\max_p \left\{ \mathbb{E}_{x\sim p}[R(x)] - \beta D_{\mathrm{KL}}\left(p \| p_{\mathrm{ref}}\right) \right\}, \qquad p^*(x) \propto p_{\mathrm{ref}}(x) \exp\left(R(x)/\beta\right), \tag{1}$$

and derives from its stochastic-optimal-control formulation a regression objective for the fine-tuned velocity field instead of fitting p^* through its likelihood. Stated in the convention above (RAM places the data at flow time zero), the regression target is

$$\hat{v}_{\mathrm{RAM}} = \mathrm{sg}\Big[v_{\mathrm{ref}}(x_\tau, \tau) + \tfrac{1}{\beta} R(x_1) \left(u - v_\theta(x_\tau, \tau)\right) \Big], \tag{2}$$

where the stop-gradient $\mathrm{sg}[\cdot]$ evaluates the target with the current parameters and treats it as a constant during differentiation. The model minimizes $\|v_\theta(x_\tau, \tau) - \hat{v}_{\mathrm{RAM}}\|_2^2$ over samples x_1 drawn from the current model, with x_0 and τ sampled as in $\mathcal{L}_{\mathrm{FM}}$. The target shifts the reference velocity along the flow-matching residual $u - v_\theta$, toward the conditional velocity of high-reward samples and away from that of low-reward ones, without reward gradients or likelihood evaluation. Two properties of this construction matter below: each target is used for a single gradient step, since RAM draws fresh samples at every optimizer step, and the reward enters the target linearly.

The first two families target sequential decision-making, whereas AWM, ORW-CFM-W2, and RAM optimize rewards of complete generated outputs. We adapt RAM to on-policy actor–critic learning with signed, critic-estimated advantages, characterize the resulting optimization behavior, and bound the displacement of each target. A separate proximal mechanism, following the adaptive penalty of PPO (Schulman et al., 2017) and Lagrangian constraint enforcement (Achiam et al., 2017), controls movement relative to the behavior policy and so separates reference anchoring from within-iteration proximity.

3 METHOD

RAM-A performs policy improvement for flow policies by velocity-field regression. Each actor update regresses the velocity field onto an advantage-dependent target, so that the policy is opti-

mized with the same squared-error loss as during pretraining and its likelihood is never evaluated. The target specifies the direction of the update relative to a reference policy, and a bound on its displacement limits the influence of any single advantage estimate. A separate proximity penalty limits movement away from the behavior policy while a rollout batch is reused. We refer to the unbounded objective as advantage regression (AR), to its bounded form as bounded AR, and to the complete procedure as RAM-A.

3.1 Problem Setup

We formulate the control problem as a discounted Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma)$ with state space $\mathcal{S}$, continuous action space $\mathcal{A} \subseteq \mathbb{R}^d$, transition kernel $P(s' \mid s, a)$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, initial state distribution ρ_0, and discount factor $\gamma \in [0, 1)$. Our goal is to find a policy $\pi_\theta(a \mid s)$ that maximizes the expected discounted return $J(\pi_\theta) = \mathbb{E}_{\xi\sim\pi_\theta}\big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\big]$, where $\xi = (s_0, a_0, s_1, a_1, \ldots)$ denotes a trajectory induced by the policy and the environment. We write $V^\pi(s)$, $Q^\pi(s, a)$, and $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$ for the discounted value, action-value, and advantage functions of π, and $d^\pi(s)$ for its normalized discounted state-visitation distribution.

We parameterize π_θ as a state-conditioned flow-matching model (Lipman et al., 2023), whose velocity field $v_\theta(x_\tau, \tau, s)$ implicitly defines the action distribution. Actions are sampled by integrating this field from Gaussian noise $x_0 = \epsilon \sim \mathcal{N}(0, I)$ to $a = x_1$. Here, $\tau \in [0, 1]$ denotes flow time, distinct from the environment timestep t. Throughout, we use the linear conditional path $x_\tau = \tau a + (1-\tau)\epsilon$ with conditional target velocity $u = a - \epsilon$, and behavior cloning from demonstrations minimizes the state-conditioned form of $\mathcal{L}_{\mathrm{FM}}$. We train π_θ with on-policy actor–critic learning. At iteration k, the behavior policy $\pi_k := \pi_{\theta_k}$ collects a batch of transitions, and a learned critic $V_\phi \approx V^{\pi_k}$ provides advantage estimates $\hat{A}_i$ for them through generalized advantage estimation (GAE) (Schulman et al., 2016). This yields the rollout batch $\mathcal{B}_k = \{(s_i, a_i, \hat{A}_i)\}_{i=1}^N$ on which the actor is updated. Since environment interaction is expensive, the actor is optimized for several epochs on $\mathcal{B}_k$ before the next batch is collected.

3.2 Advantage Regression for Flow Policies

KL-regularized policy improvement. Our goal is to increase the probability of actions with higher advantage while keeping the updated policy close to a reference policy π_{ref}. Following Peng et al. (2020), we express this trade-off through the KL-regularized policy-improvement objective

$$\max_q \ \mathbb{E}_{s\sim d^{\pi_k}}\Big[\mathbb{E}_{a\sim q(\cdot|s)}[A^{\pi_k}(s, a)] - \eta\, D_{\mathrm{KL}}(q(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s))\Big], \tag{3}$$

where q is a candidate policy and $\eta > 0$ controls the strength of the penalty. Optimizing over unrestricted action distributions yields

$$q_{k+1}(a \mid s) = \frac{1}{Z_k(s)}\, \pi_{\mathrm{ref}}(a \mid s)\, \exp\big(A^{\pi_k}(s, a)/\eta\big), \tag{4}$$

where $Z_k(s)$ normalizes the distribution. The standard choice $\pi_{\mathrm{ref}} = \pi_k$ keeps the improved policy close to the policy under which the advantages were estimated; Section 3.3 also considers a slowly moving reference. For policies with tractable densities, π_θ can be fitted to equation 4 through advantage-weighted maximum likelihood (Peng et al., 2020). For flow policies, evaluating $\log \pi_\theta(a \mid s)$ requires integrating the divergence of the velocity field along the sampling trajectory, making likelihood-based updates computationally expensive.

Regression target. The advantage-weighted policy in equation 4 has the same form as RAM's reward-tilted distribution in equation 1, with the advantage under the behavior policy replacing the reward and the reference policy replacing the reference model. This correspondence motivates adapting RAM's regression target (equation 2) to advantage-based policy improvement, which avoids likelihood evaluation altogether. We define $A(s, a) := A^{\pi_k}(s, a)/\eta$ and $\tilde{A}_i := \hat{A}_i/\eta$, with $\eta > 0$. Throughout, A denotes the scaled advantage entering the regression target. Let v_{ref} denote the velocity field of the reference policy, held fixed during actor optimization; the choice $\pi_{\mathrm{ref}} = \pi_k$ corresponds to $v_{\mathrm{ref}} = v_{\theta_k}$. The advantage-regression objective is

$$\mathcal{L}_{\mathrm{AR}}(\theta) = \mathbb{E}_{s\sim d^{\pi_k},\, a\sim\pi_k(\cdot|s)}\ \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I),\, \tau\sim\mathcal{U}[0,1]}\left[\left\|v_\theta(x_\tau, \tau, s) - \hat{v}\right\|_2^2\right], \tag{5}$$

with x_τ and u defined in Section 3.1 and regression target

$$\hat{v} = \mathrm{sg}\Big[v_{\mathrm{ref}}(x_\tau, \tau, s) + A(s,a)\big(u - v_\theta(x_\tau, \tau, s)\big)\Big], \tag{6}$$

with the stop-gradient operator as in equation 2. The regression target in equation 6 admits a simple interpretation. It combines the reference velocity with an advantage-scaled correction along the flow-matching residual $u - v_\theta$. For $A(s,a) > 0$, this correction points toward the conditional velocity u. For $A(s,a) < 0$, it points away from u. When $A(s,a) = 0$, the target reduces to v_{ref}. Subtracting a state-dependent baseline from Q^{π_k} leaves the tilt equation 4 unchanged and, when the behavior field is the flow-matching-optimal field for its own samples, also the population gradient of equation 5 at θ_k (Appendix D); the critic thus serves to reduce the variance of the regression coefficient.

In practice, we estimate equation 5 from the rollout batch $\mathcal{B}_k$. The outer expectation is approximated by the transitions in the batch, using the GAE estimates $\hat{A}_i$ in place of A^{π_k}, as in PPO. The inner expectation is approximated for each transition by N_{MC} noise–time pairs $(\epsilon_{ij}, \tau_{ij})$, $j = 1, \ldots, N_{\mathrm{MC}}$, drawn from the same distributions as in equation 5. Each pair defines an interpolant $x_{ij} = \tau_{ij}\, a_i + (1 - \tau_{ij})\, \epsilon_{ij}$ and a conditional velocity $u_{ij} = a_i - \epsilon_{ij}$. The noise ϵ_{ij} is independent of the noise that generated a_i, as in RAM. The empirical objective is

$$\widehat{\mathcal{L}}_{\mathrm{AR}}(\theta) = \frac{1}{N N_{\mathrm{MC}}} \sum_{i=1}^{N} \sum_{j=1}^{N_{\mathrm{MC}}} \big\| v_\theta(x_{ij}, \tau_{ij}, s_i) - \hat{v}_{ij} \big\|_2^2, \tag{7}$$

with per-sample targets $\hat{v}_{ij} = \mathrm{sg}\left[v_{\mathrm{ref}}(x_{ij}, \tau_{ij}, s_i) + \tilde{A}_i\big(u_{ij} - v_\theta(x_{ij}, \tau_{ij}, s_i)\big)\right]$. Within iteration k, the transitions, raw advantage estimates, and noise–time pairs are sampled once. Because the target depends on the current prediction, it is re-evaluated with the current parameters at every optimizer step and treated as a constant during differentiation. Unlike RAM, which draws fresh samples from the updated model at every step, AR reuses the actions of the behavior policy throughout the iteration, as is standard in on-policy actor–critic learning; Section 3.3 analyzes the consequences of this reuse.

Relation to the policy tilt. The exponential tilt equation 4 is the distribution the update is meant to approximate. What the regression converges to can be characterized exactly for the unbounded objective with the behavior policy as reference. Take $v_{\mathrm{ref}} = v_{\theta_k}$, assume that the behavior field is the flow-matching-optimal field for its own samples, that is, $v_{\theta_k}(z) = \mathbb{E}[u \mid z]$ for $z = (x_\tau, \tau, s)$ under $a \sim \pi_k(\cdot \mid s)$ and independent noise, and optimize equation 5 over unrestricted velocity fields. If $1 + A(s,a) > 0$ the stationary field is the conditional flow-matching field of the linearly tilted policy $q_{\mathrm{lin}}(a \mid s) \propto \pi_k(a \mid s)\,[1 + A(s,a)]$ (Appendix D). Unbounded advantage regression therefore targets a linear rather than an exponential tilt of the behavior policy. The two agree to first order in the coefficient, since $\exp(A) = 1 + A + O(A^2)$, and the linear form shows directly why coefficients below -1 are problematic: the weight $1 + A$ turns negative, and q_{lin} ceases to be a distribution. This characterization describes the core update; the mechanisms introduced below modify its fixed point.

3.3 Bounded Advantage Regression

The regression target $\hat{v}$ is the reference velocity shifted by $A\,(u - v_\theta)$, an advantage-weighted step from the current prediction toward the conditional velocity u. Two things can go wrong when the actor is optimized for several epochs on the same batch. First, the target moves with the prediction. For a positive advantage, each step brings the prediction closer to u, the residual $u - v_\theta$ shrinks, and the updates settle. For a negative advantage, each step pushes the prediction away from u and the residual grows, which enlarges the next shift; if the advantage is large enough in magnitude, the steps grow with every epoch and the prediction diverges from the reference. Second, the shift is proportional to the advantage, and the residual depends on the sampled noise through $u = a - \epsilon$. A single erroneous advantage estimate therefore displaces the targets of all of its noise–time samples arbitrarily far from the reference. The first issue does not arise in RAM, where every target is used for a single gradient step; the second is inherited from RAM's linear dependence on the reward.

To make the first issue precise, consider a single sample (x_τ, τ, s, u) and write $v := v_\theta(x_\tau, \tau, s)$ for the current prediction and $r := v_{\mathrm{ref}}(x_\tau, \tau, s)$ for the reference velocity at that sample. Each gradient

step on $\|v - \hat{v}\|_2^2$ moves v part of the way toward its current target $\hat{v} = r + A\,(u - v)$. Because the target depends on v, the iteration can only come to rest where the prediction equals its own target, $v = r + A\,(u - v)$, that is, at

$$v^\star = \frac{r + A\,u}{1 + A}. \tag{8}$$

For $A > 0$, $v^\star$ is a weighted average of r and u, and every step brings v closer to it. For $-1 < A < 0$, $v^\star$ lies on the far side of r from u, and the steps still contract toward it. For $A < -1$, the steps move v away from $v^\star$ and grow with every epoch, so the iteration never settles. On rollout samples, $\tilde{A}_i < -1$ is equivalent to $\hat{A}_i < -\eta$, so the advantage scale η controls how often this case occurs. Equivalently, $v - \hat{v}(v) = (1 + A)\,v - (r + Au)$ is the gradient of

$$\Phi_A(v) = \tfrac{1}{2}\|v - r\|_2^2 + \tfrac{A}{2}\|v - u\|_2^2, \tag{9}$$

so repeated optimization on a fixed batch performs gradient descent on Φ_A, which has a minimizer for $A > -1$ and is unbounded below for $A < -1$ (Appendix E). The first term anchors the prediction to the reference; the second is the flow-matching loss of the sample weighted by its advantage, which on its own gives advantage-weighted matching (Xue et al., 2026).

Bounded target. Both failure modes arise from the displacement $A\,(u - v)$ of the target from the reference. One alternative is to compute the target once from the behavior policy v_{θ_k} and keep it fixed during the iteration. This avoids the divergence, but the fixed displacement $A\,(u - v_{\theta_k})$ still grows linearly in A and overshoots u for $A > 1$. Recomputing the target from the current prediction at every step, as in equation 6, instead converges to a point between r and u for every $A > 0$, so we keep this target and limit its displacement instead. Clipping the scalar advantage would bound the coefficient but not the residual, which varies across noise–time samples. We therefore bound the displacement itself, constraining the target to a ball around the reference velocity:

$$\hat{v} = \mathrm{sg}\Big[r + \mathrm{clip}_b\big(A\,(u - v)\big)\Big], \qquad \mathrm{clip}_b(z) := z \min\left(1, \frac{b}{\|z\|_2}\right), \tag{10}$$

with $\mathrm{clip}_b(0) := 0$ and radius $b = c\,\|r\|_2$, where $c > 0$ sets the maximum displacement relative to the magnitude of the reference velocity; a small absolute floor on b can be added for $r \approx 0$. The bound preserves the direction of the correction and leaves the target unchanged whenever $\|A(u - v)\|_2 \le b$. The objective remains the squared regression loss of equation 5, with equation 10 in place of equation 6; we denote it by $\mathcal{L}_{\mathrm{bAR}}$. Figure 6 in Appendix J illustrates the construction at a single sample.

Since $\|\,\mathrm{clip}_b(\cdot)\|_2 \le b$, the target always lies in the ball $\mathcal{B}_b(r) = \{v : \|v - r\|_2 \le b\}$, and each gradient step pulls the prediction toward this ball for every $A \in \mathbb{R}$ (Appendix I). In the sample-level model, the bound thus removes the runaway dynamics for every advantage value. It also limits the influence of any single advantage estimate, since the displacement of each target is at most b whatever the magnitude of the estimate. The bounded objective corresponds to an anchored Huber-type loss (Appendix F). Because shared parameters couple updates across samples, these statements describe each prediction rather than the network as a whole. We test these predictions empirically: Figure 4 shows that bounding the target suppresses the growth in policy drift under repeated optimization of a fixed rollout batch, while Appendix B provides additional fixed-batch analyses and shows that the bounded update is substantially less sensitive to corrupted advantage estimates.

Reference policy. The construction in Section 3.2 with $\pi_{\mathrm{ref}} = \pi_k$ uses $v_{\mathrm{ref}} = v_{\theta_k}$, so that the regression targets are centered at the behavior policy. More generally, as in KL-regularized fine-tuning (Ouyang et al., 2022; Bergmeister et al., 2026), we let the reference be a separate, slowly evolving policy $v_{\mathrm{ref}} = v_{\theta_{\mathrm{ref}}}$. Each prediction is then an advantage-dependent interpolation from the reference and, with the bound, is confined to a ball around it (Propositions 3(i) and 7), so that the reference determines which behavior the update retains. We initialize θ_{ref} from the initial actor and update it by Polyak averaging (Polyak & Juditsky, 1992), $\theta_{\mathrm{ref}}^{k+1} = (1 - \rho)\,\theta_{\mathrm{ref}}^{k} + \rho\,\theta_{k+1}$ with $\rho \in [0, 1]$. For $\rho = 1$, the reference coincides with the behavior policy at the start of every iteration, recovering the update motivated by the proximal problem. For $\rho = 0$, the initial policy is a persistent anchor, and intermediate values trade off retention of the initial behavior against adaptation, which Section 4.2 evaluates.

3.4 BEHAVIOR-POLICY PROXIMITY

The surrogate in equation 3 uses the state distribution d^{π_k} of the behavior policy, so its relation to the return depends on controlling how far the updated policy moves from π_k. This control is complementary to the bound, which acts on individual targets, and to the reference, which differs from π_k for $\rho < 1$. We therefore penalize the squared difference between the current and the behavior velocity field, integrated along the stored sampling path of each action; for a Lipschitz velocity field, this path divergence bounds the displacement of the action generated from the same noise. The penalty enforces a batch-average budget δ through a Lagrange multiplier updated by dual ascent. In addition, samples of transitions that exceed the budget while still moving in their advantage-favored direction are regressed onto the reference instead of the advantage-shifted target. Appendix G gives the definitions, the displacement bound, and the gating rule, and Algorithm 1 in Appendix C summarizes the complete procedure.

Relation to ratio-based updates. FPO (McAllister et al., 2025) maximizes a clipped version of $A\,\hat{\varrho}_\theta$ with the surrogate ratio $\hat{\varrho}_\theta = \exp(\ell_{\theta_k} - \ell_\theta)$, built from the same conditional flow-matching losses $\ell_\theta = \|v_\theta(x_\tau, \tau, s) - u\|_2^2$. At the behavior policy, its gradient coincides with that of unbounded AR, so a single step cannot distinguish the two (Appendix H). Under repeated optimization they differ. In the sample-level model, the ratio objective drives the prediction to u for every $A > 0$, regardless of the magnitude of the advantage, and has no minimizer for any $A < 0$; clipping halts the resulting drift once the ratio leaves the band but does not reverse it. The bound removes the corresponding drift for advantage regression.

4 EXPERIMENTS

Our experiments first establish that advantage regression is a competitive policy-improvement mechanism on standard continuous-control and robotic fine-tuning benchmarks (Section 4.1). The remaining experiments concern what fine-tuning leaves intact: whether a multitask policy fine-tuned on a single task retains the skills the reward does not exercise (Section 4.2), and whether a multimodal pretrained policy keeps its solution modes (Section 4.3). We compare against one representative of each family in Section 2, ReinFlow (Zhang et al., 2025) for transition-likelihood methods and FPO++ (Yi et al., 2026) for likelihood-ratio surrogates, with all methods sharing the same policy architectures, initializations, critics, and environment interaction budgets. Implementation details and hyperparameters are given in Appendix A.

4.1 CONTINUOUS-CONTROL AND ROBOTIC FINE-TUNING BENCHMARKS

We first evaluate whether the regression-based update of RAM-A provides an effective general-purpose policy-improvement mechanism. We consider two settings that differ in initialization and reward: fine-tuning a behavior-cloned policy under a sparse task reward on ManiSkill manipulation tasks, and training a policy from a random initialization under dense rewards on the DeepMind Control Suite.

Sparse-reward robotic fine-tuning. We consider PickCube, StackCube, and PegInsertionSide from ManiSkill. Each method is initialized from the same behavior-cloned flow policy and subsequently fine-tuned using a sparse success reward. Figure 1 (top) reports the success rate as a function of environment interactions, averaged over 5 seeds, for RAM-A, FPO++, ReinFlow. All methods substantially improve over their pretrained initialization. ReinFlow is generally the most sample efficient, particularly on PickCube and StackCube, while RAM-A and FPO++ exhibit similar learning behavior. On PegInsertionSide, the differences between the methods are smaller over the evaluated interaction budget.

Dense-reward continuous control. We additionally evaluate RAM-A, FPO++, and ReinFlow on five tasks from the DeepMind Control Suite. Figure 1 (bottom) reports average reward as a function of environment interactions. RAM-A is particularly effective in the low-interaction regime, reaching strong performance with comparatively fewer environment samples across several tasks. As the interaction budget increases, the performance gap narrows on some environments, suggesting that its advantage is primarily in early policy improvement rather than uniformly higher asymptotic

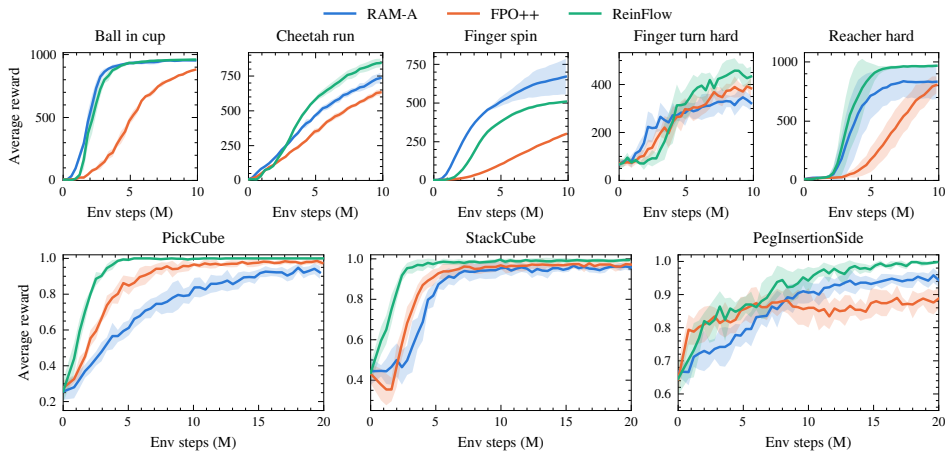


Figure 1: **Policy optimization on sparse-reward robotic fine-tuning and dense-reward continuous control. Top:** Learning from scratch on dense-reward DeepMind Control Suite tasks. **Bottom:** Fine-tuning behavior-cloned flow policies under sparse task rewards on ManiSkill manipulation tasks. Curves show the mean over 5 seeds; bands show ± 1 std.

return. Overall, these experiments demonstrate that advantage regression can serve as an effective policy-improvement mechanism across both pretrained robotic policies and standard continuous-control domains.

4.2 Preserving Multitask Skills During Single-Task Fine-Tuning

We next evaluate whether reinforcement fine-tuning preserves behavior which is not part of the current optimization objective. We first train a single flow policy jointly on PegInsertion, StackCube, and PickCube using behavior-cloning. Starting from this we then fine-tune sequentially on individual tasks while evaluating performance on all three throughout. Figure 2 shows how performance evolves across the fine-tuning sequence. The first column corresponds to the shared behavior-cloned initialization, while each subsequent column reports performance after fine-tuning on the indicated task. All methods improve performance on the task currently receiving reward feedback, but differ in how much previously learned behavior they preserve. FPO++ and ReinFlow show substantial degradation on tasks that are not part of the current fine-tuning objective, whereas RAM-A retains more of the pretrained skills while still adapting to the target task.

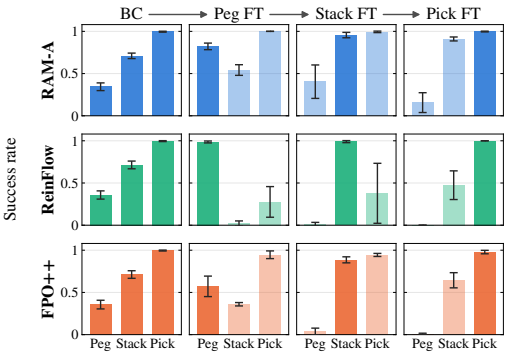


Figure 2: **Retention of pretrained skills during sequential single-task fine-tuning.** A shared behavior-cloned policy is sequentially fine-tuned on individual tasks and evaluated on all three after each stage. Dark bars indicate the currently optimized task; lighter bars show retention on the remaining tasks. Error bars and markers denote variation across random seeds.

4.3 Preserving Multimodal Behavior During Reinforcement Fine-Tuning

A central motivation for generative policies is their ability to represent multiple valid solutions. We therefore investigate whether this multimodality is preserved during reinforcement fine-tuning using a symmetric two-goal PickCube task, where moving the cube to either a left or right target yields identical reward. The behavior-cloned initialization represents both modes, and we fine-tune the same policy using RAM-A, FPO++, and ReinFlow. Figure 3 shows that all methods improve success from 81% to approximately 100%, but differ substantially in mode coverage. RAM-A remains

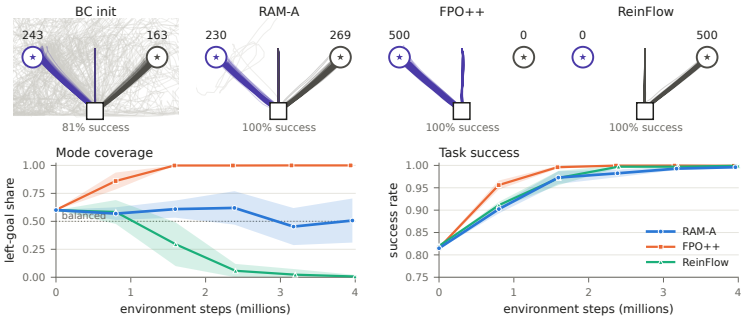


Figure 3: **Multimodality under reinforcement fine-tuning.** All methods achieve near-perfect success on symmetric two-goal PickCube, but only RAM-A preserves both pretrained modes; FPO++ and ReinFlow collapse to a single mode.

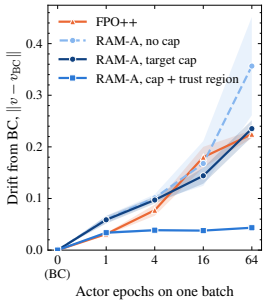


Figure 4: **Policy drift under batch reuse.** Mean over 5 rollout seeds, band = 95% CI.

approximately balanced between both targets throughout fine-tuning, whereas FPO++ progressively concentrates on the left target and ReinFlow collapses to the right. Since both modes receive identical reward, this symmetry breaking is induced by the optimization dynamics rather than the task objective. The experiment therefore shows that similar task success can mask substantial differences in how well fine-tuning preserves the behavioral diversity learned during pretraining.

Together with the multitask experiment, this evaluates two complementary forms of preservation. The two-goal task tests whether multiple solutions *within a task* are retained, while the multitask experiment tests whether skills *across tasks* survive sequential specialization. Both probe whether reinforcement fine-tuning can improve the target behavior without unnecessarily overwriting structure learned during pretraining. Figure 4 shows the mechanism behind this preservation on a single frozen rollout batch: the drift of the actor from the behavior-cloned policy grows with every epoch for unbounded advantage regression and FPO++, while the bounded target removes the accelerating growth of the unbounded variant and the proximity penalty holds the drift near its value after the first epoch for the remaining 63. Repeated optimization of a batch thus moves the policy by a controlled amount rather than by an amount that grows with the number of epochs, which is what leaves untouched skills and modes intact.

5 Conclusion, Limitations and Future Work

We presented RAM-A, which performs on-policy reinforcement learning for flow policies by regressing the velocity field toward advantage-shifted targets, so that the policy is optimized with the same squared-error loss as during pretraining and its likelihood is never evaluated. This update shares its first-order direction with FPO at the behavior policy, but under repeated optimization of a rollout batch both unbounded advantage regression and the ratio objective drift without bound for negative advantages. Bounding the norm of the regression target removes this drift and limits the influence of any single advantage estimate. On robotic manipulation tasks, RAM-A is competitive with existing flow-policy reinforcement-learning methods in task performance and differs from them in what it preserves: when a multitask policy is fine-tuned on a single task, it retains the skills the reward does not exercise, and when a pretrained policy has several equally rewarded solution modes, it keeps them, where FPO++ and ReinFlow lose both. Both effects follow from the same construction, a bounded target centered at a reference that moves slowly away from the pretrained policy, so that pretrained generative policies can be improved one skill at a time without erasing their diversity or capabilities.

This retention comes at a price in adaptation speed: the bound and the proximity budget limit how far the policy moves per iteration, and on some tasks ReinFlow reaches high success with fewer environment interactions. All experiments are conducted in simulation; deploying RAM-A on real robots, where large policy changes are costly and demonstrated skills should not be lost, and extending it toward safe real-world fine-tuning of large vision-language-action models are the natural next steps.

AI USE STATEMENT

In this work, we used generative AI tools for different subtasks. For polishing writing by letting AI tools do a pass over grammar and suggest possible stylistic improvements. For retrieval and discovery to find related works which solved similar tasks in different domains as works in molecules generation which were doing flow matching over rotations. For research ideation and execution, specifically for writing code of visualizers and efficiently test novel ideas. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work by reviewing the final code used for this work, and all the suggested literature. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. Pmlr, 2017.

Andreas Bergmeister, Stefanie Jegelka, Nikolas Nüsken, Carles Domingo-Enrich, and Jakiw Pidstrigach. Reinforce adjoint matching: Scaling rl post-training of diffusion and flow-matching models. *arXiv preprint arXiv:2605.10759*, 2026.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2aee1c4159e48407d68fe16ae8e6e49e-Abstract-Conference.html.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu (eds.), *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://doi.org/10.15607/RSS.2023.XIX.026.

Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=2IoFFexvuw.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online RL. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/3a10c46572628d58cb44fb705f25cbbf-Abstract-Conference.html.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. *CoRR*, abs/2507.21053, 2025. doi: 10.48550/ARXIV.2507.21053. URL https://doi.org/10.48550/arXiv.2507.21053.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. In *International Conference on Learning Representations*, 2020.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992. doi: 10.1137/0330046. URL https://doi.org/10.1137/0330046.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 10674–10685. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01042. URL https://doi.org/10.1109/CVPR52688.2022.01042.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria,*

May 3-7, 2021. OpenReview.net, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI. *arXiv preprint arXiv:2410.00425*, 2025. URL https://arxiv.org/abs/2410.00425.

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. *arXiv preprint arXiv:1801.00690*, 2018. URL https://arxiv.org/abs/1801.00690.

Shuchen Xue, Chongjian Ge, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning rl with pretraining in diffusion models. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2509.25050.

Brent Yi, Hongsuk Choi, Himanshu Gaurav Singh, Xiaoyu Huang, Takara E. Truong, Carmelo Sferrazza, Yi Ma, Rocky Duan, Pieter Abbeel, Guanya Shi, C. Karen Liu, and Angjoo Kanazawa. Flow policy gradients for robot control. *CoRR*, abs/2602.02481, 2026. doi: 10.48550/ARXIV.2602.02481. URL https://doi.org/10.48550/arXiv.2602.02481.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/98d6f928497d9eaad9f320fb2db3040d-Abstract-Conference.html.

A Experimental Details

The experiments address three properties of flow-policy fine-tuning: task performance, retention of pretrained skills across tasks, and preservation of multiple solutions within a task. Appendix A.1 gives the benchmark protocols and hyperparameters; Appendices A.2 and A.3 describe the two preservation experiments.

A.1 Continuous-Control and Robotic Fine-Tuning Benchmarks

The benchmarks in Section 4.1 evaluate policy improvement in two settings: fine-tuning pretrained manipulation policies and learning continuous control from scratch. Within each task, RAM-A, FPO++ (Yi et al., 2026), and ReinFlow (Zhang et al., 2025) share the policy and critic architectures, observation inputs, action chunking, and rollout budget.

A.1.1 Tasks and Initialization

Sparse-reward manipulation. We fine-tune state-based policies on PickCube, StackCube, and PegInsertionSide in ManiSkill3 (Tao et al., 2025), using sparse success rewards and end-effector delta-pose control. Policies observe task-relevant poses and relative positions; critics additionally receive robot joint states. Observations and actions retain the demonstration-derived normalization. Policies predict and execute four actions before replanning ($T_p = T_a = 4$). Table 1 summarizes the task settings.

Table 1: ManiSkill setup. Rollout sizes count action chunks; episode limits count environment steps.

Setting	PickCube	StackCube	PegInsertionSide
Successful demonstrations	25	100	500
Parallel training environments	700	400	400
Transitions per iteration	21,000	20,000	20,000
Training / evaluation episode limit	120 / 200	200 / 200	200 / 300
Training iterations	500	500	500

Behavior-cloning initialization. All methods start from the same task-specific flow policy, trained on successful ManiSkill motion-planning demonstrations by conditional flow matching. Pretraining uses AdamW with batch size 256, weight decay 10^{-6}, and dropout 0.1. The learning rate warms up over 100 dataset passes to 10^{-4}, then decays toward 10^{-5} with a cosine schedule over 1000 passes. Fine-tuning loads the policy weights and initializes the critic afresh; dropout is disabled during RL.

Learning continuous control from scratch. We train from scratch on Ball-in-Cup Catch, Cheetah Run, Finger Spin, Finger Turn Hard, and Reacher Hard from the DeepMind Control Suite (Tassa et al., 2018), using MJX implementations and state observations. Policies predict one action ($T_p = T_a = 1$), with action repeat two. Each run uses 256 parallel environments, 61,440 transitions per iteration, and 200 iterations (12.288 million policy decisions).

Architectures. The velocity field is a time-conditioned residual MLP with width 128, four blocks, and time-embedding dimension 32 on ManiSkill; the corresponding DMC values are 24, two, and eight. All methods use a separate ReLU value MLP with three width-256 hidden layers.

A.1.2 Sampling and Evaluation

All benchmark policies use ten Euler steps. Training starts from unit Gaussian noise; ReinFlow additionally learns stochastic transitions. On ManiSkill, all methods evaluate with the same ODE sampler initialized at zero. On DMC, evaluation retains Gaussian initialization but removes transition noise; ReinFlow also clips intermediate actions to $[-1, 1]$.

Evaluation uses 100 episodes on ManiSkill and 64 on DMC. We measure success at the end of each episode on ManiSkill and mean undiscounted episode return on DMC.

A.1.3 Optimization and Hyperparameters

Tables 2 and 3 list the main hyperparameters. Learning rates are constant during RL, advantages are standardized, and value estimates bootstrap at time-limit truncations.

Table 2: RL optimization settings. R/F denotes RAM-A and FPO++; RF denotes ReinFlow. Epochs and data repetitions refer to each collected batch.

Setting	ManiSkill	DMC
Actor / critic learning rate	10^{-5} / 10^{-4}	10^{-4} / 10^{-4}
Discount γ / GAE λ	0.99 / 0.95	0.99 / 0.95
Minibatches per pass	8	8
Actor / critic epochs (R/F)	10 / 10	8 / 8
Update epochs / data repetitions (RF)	10 / 1	5 / 10

RAM-A refreshes its reference after each iteration ($\rho = 1$), reuses flow samples across epochs, and recomputes the regression targets at each update. Its displacement bound is $b = 2\|v_{\text{ref}}\|_2$ on ManiSkill and $b = \max\{2\|v_{\text{ref}}\|_2, 8\}$ on DMC, where the floor permits learning from a zero-initialized velocity field. FPO++ uses per-sample ratios, with asymmetric policy optimization disabled on ManiSkill and enabled on DMC. ReinFlow uses learned transition noise and approximate-KL early stopping; its DMC data repetitions increase optimization effort without adding environment interactions.

Table 3: Method-specific hyperparameters for the benchmark experiments.

Setting	ManiSkill	DMC
Flow-matching samples N_{MC} (R/F)	8	8
RAM-A bound multiplier c / minimum radius	2 / 0	2 / 8
RAM-A reference rate ρ	1	1
RAM-A path budget δ	0.005	0.005
RAM-A dual initial value / maximum	10 / 100	1 / 10
RAM-A dual step size κ	30	30
FPO++ ratio clip	0.02	0.05
FPO++ asymmetric policy optimization	disabled	enabled
FPO++ residual loss	Huber ($h = 0.5$)	squared
ReinFlow ratio clip	0.0018	0.00453312
ReinFlow approximate-KL threshold	0.01	1
ReinFlow entropy / value-loss coefficient	0.01 / 2	0.03 / 0.5
ReinFlow transition std. bounds (initial)	$[0.08, 0.14]$	$[0.10, 0.24]$
ReinFlow noise schedule	learned	learned with decay

A.2 Retention of Pretrained Skills

Section 4.2 measures whether adapting a shared policy to one task degrades its other pretrained skills. All methods use the same multitask initialization, task order, and training budget.

Multitask behavior cloning. We train one flow policy jointly on PickCube, StackCube, and PegInsertionSide, using 500 successful motion-planning demonstrations per task. Demonstrations are converted into overlapping four-action windows and pooled for conditional flow-matching training. Minibatches are sampled from this combined dataset, so tasks contribute in proportion to their numbers of training windows. All three tasks use a shared observation layout for the robot, manipulated object, and target, together with a task identifier. A common observation and action normalization is computed from the combined demonstrations and retained throughout fine-tuning. The shared representation lets one policy perform all three tasks.

Sequential fine-tuning. Starting from the shared BC checkpoint, each method follows a single sequence:

$$\text{PegInsertionSide} \longrightarrow \text{StackCube} \longrightarrow \text{PickCube}.$$

Each stage uses a fixed budget corresponding to 10 million environment steps. Only the current task supplies training trajectories and sparse success rewards; demonstrations and trajectories from the other tasks are not replayed. At the end of a stage, its final policy initializes the next stage, so changes accumulate across the sequence. Only policy weights are transferred between stages. The critic, optimizer states, and rollout buffers are reinitialized for the new task; ReinFlow also initializes a fresh exploration-noise network. The observation representation, normalization, and action interface remain fixed across all three stages.

Retention evaluation. We evaluate the shared policy on all three tasks before fine-tuning, throughout each stage, and at every stage boundary, using 100 episodes per task and the same deterministic, zero-initialized ODE sampler. Evaluation episode limits are 300 steps for PegInsertionSide and StackCube and 200 for PickCube. We report each task's success rate separately: improvement on the active task measures adaptation, while performance on the other two tasks measures retention. Figure 2 compares the BC initialization with each stage endpoint, distinguishing the active task from the two tasks receiving no training data.

A.3 Preservation of Multimodal Behavior

Section 4.3 tests whether a policy retains multiple successful behaviors within a task. A symmetric two-goal variant of ManiSkill PickCube provides two equally rewarded solutions, allowing mode coverage to be measured separately from task success.

Task and observations. The cube starts at a fixed pose in front of a Panda robot. Two goal sites are placed symmetrically to its left and right, each 15 cm from the cube's centerline and 15 cm above the table. Reaching either goal yields the same sparse success reward. Success requires the cube to lie within 2.5 cm of a goal and the robot to be stationary. The policy observes robot joint states, the end-effector and cube poses, their relative displacement, and both goal positions. No input specifies which goal to choose. The cube and goals remain fixed, while training randomizes the robot's initial joint configuration.

Paired demonstrations and pretraining. The BC dataset contains 250 pairs of successful motion-planning demonstrations, for 500 trajectories in total. Each pair starts from the same robot and object configuration and contains one trajectory to each goal. A pair is retained only when both trajectories succeed, giving equal numbers of left- and right-goal demonstrations without correlating the desired side with the initial state. A single flow policy is trained by conditional flow matching on overlapping four-action windows, using normalization computed from these demonstrations.

Reinforcement fine-tuning. All methods load the same pretrained policy and normalization and train on the symmetric task with varied robot starting configurations. The policy predicts and executes four-action chunks. Training uses only the task reward, with no additional mode-balancing reward or demonstration-replay objective. This isolates changes in mode coverage induced by reward-based adaptation.

Evaluation and mode coverage. We evaluate a fixed-start condition to test whether the policy produces multiple solutions from the same state, using 500 episodes with a 200-step limit. All methods use the same Euler ODE sampler with Gaussian initialization; ReinFlow's learned transition noise is disabled for evaluation. Sampling remains stochastic to expose both modes even when the initial environment state is identical.

Completed episodes are classified as left-goal success, right-goal success, or failure using the terminal state.

B Finite-Update Experiments

B.1 Robustness to Advantage Corruption

We test whether bounding the regression target reduces the sensitivity of the actor-gradient direction to extreme advantage estimates. For PickCube, StackCube, and PegInsertionSide, we collect one rollout batch from the pretrained policy and compute standardized advantages using a critic fitted to that batch. All methods start from the same pretrained actor weights and use the same transitions, advantages, and flow-matching samples. We compare FPO++ with bounded and unbounded advantage regression; both regression variants use the same FPO-style ratio gate, so their difference isolates the target bound. This diagnostic does not include the path-divergence penalty.

For a uniformly selected fraction $p \in \{0.1\%, 0.5\%, 1\%, 5\%\}$ of transitions, we replace the standardized advantage by $\tilde{A}'_i = -20|\tilde{A}_i|$, creating large negative outliers. The remaining advantages are unchanged, and we do not renormalize after corruption. We compute the full-batch actor gradients g and g' before any optimizer step and measure their cosine similarity, $\langle g, g' \rangle / (\|g\|_2 \|g'\|_2)$. Each corrupted gradient is compared with that method's own clean gradient. The selected transitions are shared across methods; results use one rollout batch and one corruption draw per condition (seed zero).

Figure 5 shows that the target bound better preserves the clean gradient direction on all three tasks. At 5% corruption, bounded regression retains cosine similarities of 0.74, 0.84, and 0.64 on PickCube, StackCube, and PegInsertionSide, respectively, compared with -0.11, 0.65, and 0.13 without the bound, and 0.06, 0.67, and 0.35 for FPO++. This supports the bounded-influence prediction for sparse advantage outliers; it measures local gradient sensitivity rather than downstream policy performance.

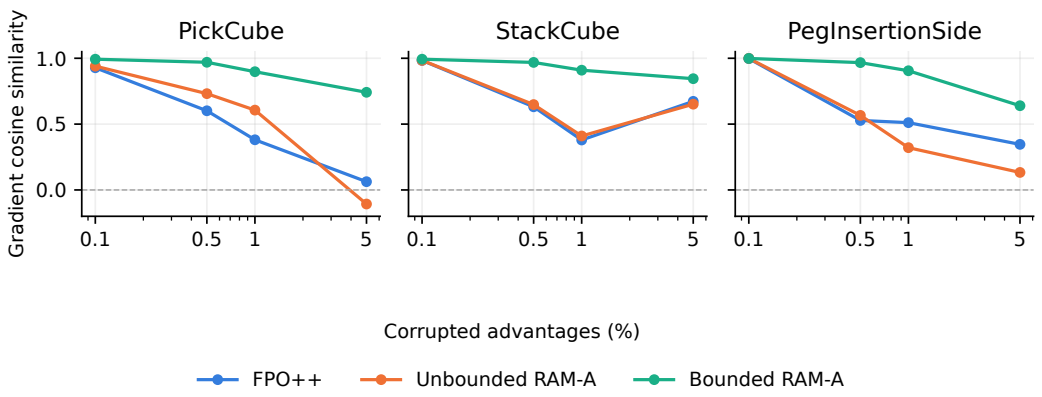


Figure 5: Actor-gradient direction under corrupted advantages. Selected standardized advantages are replaced by $-20|\tilde{A}_i|$. The target bound reduces the rotation away from each method's clean gradient at the pretrained weights. Curves show a single fixed-batch diagnostic per task.

C Algorithm

Algorithm 1 summarizes the complete on-policy actor–critic procedure of Section 3. Two quantities determine where the actor may move during an iteration: the reference v_{ref}, which centers the regression targets, and the behavior policy π_k, under which the batch and its advantages were collected. The algorithm treats the two separately: the reference enters through the targets and is updated by Polyak averaging, and the behavior policy enters through the path-divergence constraint and the stored flow-matching losses used for the direction test.

D Advantage Regression as Conditional Reinforce Adjoint Matching

RAM (Bergmeister et al., 2026) is derived for an unconditional generative model with reference distribution p_{ref} and scalar reward $R(x)$, whereas advantage regression applies it to a state-conditioned policy with the advantage $A^{\pi_k}(s,a)$ in place of the reward. This appendix records the two steps that make the transfer valid: the per-state structure of the policy-improvement problem, and the invariance of the target distribution and of the first-order update to a state-dependent baseline. Throughout, we use the notation of Section 3.1, with the scaled advantage $A(s,a) = A^{\pi_k}(s,a)/\eta$.

Per-state structure. The objective equation 3 is an expectation over $s \sim d^{\pi_k}$ of the per-state problems

$$\max_{q(\cdot|s)} \mathbb{E}_{a\sim q(\cdot|s)}[A^{\pi_k}(s,a)] - \eta\, D_{\mathrm{KL}}(q(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s))\,, \tag{11}$$

which are coupled only through the shared parameterization of q. For a fixed state s, equation 11 is RAM's problem equation 1 with $p_{\mathrm{ref}} = \pi_{\mathrm{ref}}(\cdot \mid s)$, $R = A^{\pi_k}(s,\cdot)$, and $\beta = \eta$, and its solution is the per-state tilt equation 4. Since the state enters the velocity field only as a conditioning input, RAM's regression target for this problem is equation 6 evaluated at the conditioning state, and the population objective equation 5 is the d^{π_k}-weighted sum of the per-state RAM objectives. What the regression converges to is characterized next. Shared parameters do not attain every per-state minimizer simultaneously in general, but this is the same approximation that any amortized conditional model makes.

Population fixed point of unbounded advantage regression. Let $v_{\mathrm{ref}} = v_{\theta_k}$ and write $z = (x_\tau, \tau, s)$ for the regression input, with $(s,a) \sim d^{\pi_k} \times \pi_k$, $\epsilon \sim \mathcal{N}(0, I)$, $\tau \sim \mathcal{U}[0,1]$, $x_\tau = \tau a + (1-\tau)\epsilon$, and $u = a - \epsilon$. Assume that the behavior field is the flow-matching-optimal field for its own samples,

$$v_{\theta_k}(z) = \mathbb{E}[u \mid z]\,, \tag{12}$$

ALGORITHM 1: RAM-A: Bounded Advantage Regression

Input: initial actor θ_0, critic ϕ, advantage scale η, bound c, path budget δ, dual step size κ, reference rate ρ, samples per transition N_{MC}, step size α

$\theta \leftarrow \theta_0$; $\theta_{\mathrm{ref}} \leftarrow \theta_0$; $\lambda \leftarrow 0$

for *iteration* $k = 0, 1, 2, \ldots$ **do**

  /* Rollout with the behavior policy; the batch is frozen for the iteration */

  $\theta_k \leftarrow \theta$

  Collect $\mathcal{B}_k = \{(s_i, a_i)\}_{i=1}^{N}$ with π_k; store each sampling path $x_{i,\tau}^k$ and its velocities $v_{\theta_k}(x_{i,\tau}^k, \tau, s_i)$

  Estimate advantages $\hat{A}_i$ with GAE and V_ϕ; $\tilde{A}_i \leftarrow \hat{A}_i / \eta$

  for *each transition* i *and* $j = 1, \ldots, N_{\mathrm{MC}}$ **do**

    $\epsilon_{ij} \sim \mathcal{N}(0, I)$, $\tau_{ij} \sim \mathcal{U}[0, 1]$; $x_{ij} \leftarrow \tau_{ij} a_i + (1 - \tau_{ij})\, \epsilon_{ij}$; $u_{ij} \leftarrow a_i - \epsilon_{ij}$

    $\ell_{ij}^k \leftarrow \| v_{\theta_k}(x_{ij}, \tau_{ij}, s_i) - u_{ij} \|_2^2$ // FM loss under π_k

  /* Actor update: several epochs on the frozen batch */

  for *each epoch and each minibatch* $\mathcal{B} \subseteq \mathcal{B}_k$ **do**

    for *each transition* $i \in \mathcal{B}$ **do**

      $D_i \leftarrow \int_0^1 \| v_\theta(x_{i,\tau}^k, \tau, s_i) - v_{\theta_k}(x_{i,\tau}^k, \tau, s_i) \|_2^2 \, \mathrm{d}\tau$ // displacement from π_k, Eq. equation 23

      for $j = 1, \ldots, N_{\mathrm{MC}}$ **do**

        $v_{ij} \leftarrow v_\theta(x_{ij}, \tau_{ij}, s_i)$; $r_{ij} \leftarrow v_{\theta_{\mathrm{ref}}}(x_{ij}, \tau_{ij}, s_i)$ // current / reference velocity

        $\hat{v}_{ij} \leftarrow \mathrm{sg}\big[r_{ij} + \mathrm{clip}_{c \| r_{ij} \|_2}\big(\tilde{A}_i \, (u_{ij} - v_{ij}) \big) \big]$ // bounded target, Eq. equation 10

        $f_{ij} \leftarrow \big[\tilde{A}_i \, (\ell_{ij}^k - \| v_{ij} - u_{ij} \|_2^2) > 0 \big]$ // advantage-favored direction?

        if $D_i > \delta$ ***and*** f_{ij} **then** $\mathcal{L}_{ij} \leftarrow \| v_{ij} - \mathrm{sg}[r_{ij}] \|_2^2$ // over budget: regress onto reference

        else $\mathcal{L}_{ij} \leftarrow \| v_{ij} - \hat{v}_{ij} \|_2^2$ // bounded advantage regression

    $\bar{D} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} D_i$

    $\theta \leftarrow \theta - \alpha \nabla_\theta \Big[\frac{1}{|\mathcal{B}| \, N_{\mathrm{MC}}} \sum_{i,j} \mathcal{L}_{ij} + \lambda \, (\bar{D} - \delta) \Big]$ // regression loss + Lagrangian penalty

    $\lambda \leftarrow \mathrm{clip}_{[0, \lambda_{\max}]} \big(\lambda + \kappa \, (\bar{D} - \delta) \big)$ // dual ascent ($\bar{D}$ after the step)

  /* Critic and reference */

  Update V_ϕ on $\mathcal{B}_k$

  $\theta_{\mathrm{ref}} \leftarrow (1 - \rho)\, \theta_{\mathrm{ref}} + \rho \, \theta$ // Polyak update, Sec. 3.3

which holds at a flow-matching optimum, for instance after behavior cloning, and is the condition RAM's derivation relies on in its fresh-sample regime.

Proposition 1 (Linear tilt)**.** *Under equation 12, suppose* $1 + A(s, a) > 0$ *almost surely,* $\mathbb{E}[A(s, a) \mid s] = 0$*, and* $\mathbb{E}\big[(1 + A) \, \|u\|_2\big] < \infty$*. Then a velocity field* v *is stationary for equation 5, optimized over all measurable fields with the target held fixed, if and only if*

$$v(z) = \frac{\mathbb{E}\big[(1 + A)\, u \mid z\big]}{\mathbb{E}\big[1 + A \mid z\big]} \qquad \textit{almost everywhere}, \tag{13}$$

which is the conditional flow-matching field of the linearly tilted policy $q_{\mathrm{lin}}(a \mid s) = \pi_k(a \mid s)\, [1 + A(s, a)]$ *under the same state distribution, noise, and path.*

Proof. With the target held fixed, the stationarity condition of equation 5 at v is, for almost every z,

$$\begin{aligned} & \mathbb{E}\big[v(z) - v_{\theta_k}(z) - A\,(u - v(z)) \mid z \big] = 0 \\ \iff \quad & v(z)\, \mathbb{E}[1 + A \mid z] = v_{\theta_k}(z) + \mathbb{E}[Au \mid z] = \mathbb{E}[(1 + A)\, u \mid z], \end{aligned} \tag{14}$$

where the last equality uses equation 12. Since $\mathbb{E}[1 + A \mid z] > 0$, this gives equation 13. The right-hand side is the conditional expectation of u given z under the joint distribution of (s, a, ϵ, τ) reweighted by the nonnegative density $1 + A(s, a)$. Because $\mathbb{E}[A \mid s] = 0$, the reweighting has conditional mean one given s, so it leaves the state distribution unchanged and tilts only the conditional action distribution, from $\pi_k(a \mid s)$ to $q_{\mathrm{lin}}(a \mid s)$. The conditional flow-matching field of a data

distribution is exactly the conditional expectation of u given z under that distribution with the same noise and path, which completes the proof. □

The zero-mean condition holds for the true advantage, since $\mathbb{E}_{a\sim\pi_k}[A^{\pi_k}(s,a)] = 0$; with a learned critic it holds only approximately, and the residual mean $\mathbb{E}[A \mid s]$ additionally reweights the state distribution by $1 + \mathbb{E}[A \mid s]$. Proposition 1 shows that unbounded advantage regression targets the linear tilt $\pi_k\,(1+A)$ rather than the exponential tilt equation 4; the two agree to first order in A, and the positivity requirement $1 + A > 0$ is the population counterpart of the sample-level threshold in Proposition 3.

Baseline invariance of the target distribution. Advantage regression uses the advantage $A^{\pi_k} = Q^{\pi_k} - V^{\pi_k}$ rather than the action value or the return as the reward of the per-state problem. The following observation shows that this choice does not change the improved policy.

Proposition 2 (Baseline invariance)**.** *Let $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $b : \mathcal{S} \to \mathbb{R}$. For every state s,*

$$\frac{\pi_{\mathrm{ref}}(a \mid s)\,\exp\big((f(s,a) - b(s))/\eta\big)}{\int \pi_{\mathrm{ref}}(a' \mid s)\,\exp\big((f(s,a') - b(s))/\eta\big)\,\mathrm{d}a'} = \frac{\pi_{\mathrm{ref}}(a \mid s)\,\exp\big(f(s,a)/\eta\big)}{\int \pi_{\mathrm{ref}}(a' \mid s)\,\exp\big(f(s,a')/\eta\big)\,\mathrm{d}a'}. \tag{15}$$

Proof. The factor $\exp(-b(s)/\eta)$ does not depend on a and cancels between numerator and denominator. □

Applied with $f = Q^{\pi_k}$ and $b = V^{\pi_k}$, Proposition 2 shows that the tilt equation 4 is the same whether it is written with the advantage or with the action value. The same holds for any other state-dependent baseline, including a learned critic V_ϕ that differs from V^{π_k}: the baseline changes the coefficient of the regression target but not the distribution it is meant to approximate.

Baseline invariance of the first-order update. The advantage enters RAM's target linearly rather than through the tilt, so Proposition 2 does not by itself imply that the regression update is baseline invariant. Replacing A by $A - b(s)/\eta$ in equation 6 changes the target by $-\frac{b(s)}{\eta}\,(u - v_\theta)$, and by Proposition 5 the per-sample gradient at the behavior policy changes by

$$\frac{2\,b(s)}{\eta}\,\big(\nabla_\theta v_{\theta_k}\big)^\top\big(u - v_{\theta_k}(x_\tau, \tau, s)\big). \tag{16}$$

Under the flow-matching optimality condition equation 12, the conditional expectation of $u - v_{\theta_k}$ given (x_τ, τ, s) vanishes, so equation 16 has zero expectation and the population gradient of equation 5 at θ_k is invariant to the baseline. The baseline does change the per-sample gradient, and choosing $b = V^{\pi_k}$ removes the state-dependent component of the coefficient, reducing its variance in the usual way. Importantly, equation 12 is an assumption on the behavior velocity field: it holds when v_{θ_k} is the flow-matching-optimal field for its induced action distribution, but is not guaranteed after arbitrary reinforcement-learning updates. We therefore use this condition only for the population-level characterization of the update at θ_k. The finite-update analysis in Section 3.3, as well as the local equivalence to FPO in Proposition 5, does not rely on equation 12.

Summary. Advantage regression is RAM applied per state to the KL-regularized improvement problem. Its population fixed point is the flow-matching field of the linear tilt $\pi_k\,(1 + A)$, which agrees with the exponential tilt equation 4 to first order in the advantage, and no likelihood or reward gradient is required. The critic contributes a state-dependent baseline that leaves the target distribution and the first-order update unchanged and only reduces the variance of the regression coefficient.

E Sample-Level Stability of Advantage Regression

Let $g_A(v) := v - \hat{v}(v) = (1 + A)\,v - (r + Au) = \nabla_v \Phi_A(v)$ denote half of the gradient of the unbounded loss with the target held fixed, with Φ_A from equation 9.

Proposition 3 (Sample-level stability of unbounded advantage regression)**.** *For the scaled advantage A, under the gradient flow $\dot{v} = -g_A(v)$: (i) if $A > -1$, v converges exponentially to $v^\star = r + \frac{A}{1+A}(u - r)$; (ii) if $A < -1$, $v^\star$ is a repelling fixed point and $\|v - r\|_2 \to \infty$ for every $v(0) \neq v^\star$; (iii) if $A = -1$ and $u \neq r$, v drifts linearly with constant velocity $r - u$; if $A = -1$ and $u = r$, every v is stationary. In case (ii), and in case (iii) with $u \neq r$, Φ_A is unbounded below.*

Proof of Proposition 3. The gradient flow $\dot{v} = -g_A(v)$ with g_A from equation 9 is the linear system

$$\dot{v} = -(1 + A)\, v + (r + Au). \tag{17}$$

For $A \neq -1$, its unique fixed point is

$$v^\star = \frac{r + Au}{1 + A} = r + \frac{A}{1 + A}\,(u - r), \tag{18}$$

and the deviation $w := v - v^\star$ satisfies $\dot{w} = -(1 + A)\, w$, so that $w(t) = e^{-(1+A)t}\, w(0)$.

Case (i), $A > -1$. The deviation decays as $\|w(t)\|_2 = e^{-(1+A)t}\|w(0)\|_2$, so v converges exponentially to $v^\star$ at rate $1 + A$. The potential Φ_A is strictly convex, since its Hessian is $(1 + A)I$, and $v^\star$ is its unique minimizer.

Case (ii), $A < -1$. Now $\|w(t)\|_2 = e^{|1+A|\,t}\|w(0)\|_2$ grows without bound for every $w(0) \neq 0$, so $v^\star$ is a repelling fixed point. By the triangle inequality, $\|v(t) - r\|_2 \geq \|w(t)\|_2 - \|v^\star - r\|_2 \to \infty$. The Hessian of Φ_A is $(1 + A)I$ with $1 + A < 0$, so Φ_A is strictly concave and unbounded below.

Case (iii), $A = -1$. Equation equation 17 reduces to $\dot{v} = r - u$, so $v(t) = v(0) + t\,(r - u)$ drifts linearly with constant velocity $r - u$ whenever $u \neq r$. The potential reduces to $\Phi_{-1}(v) = -\langle r - u, v\rangle$, which is linear and unbounded below for $r \neq u$. □

Fixed-point displacement. For $A > -1$, the displacement of the fixed point from the reference is

$$\|v^\star - r\|_2 = \frac{|A|}{1 + A}\,\|u - r\|_2, \tag{19}$$

which is bounded by $\|u - r\|_2$ for $A \geq 0$, since then $A/(1+A) \in [0, 1)$, and diverges as $A \to -1^+$.

Discrete updates. With step size $h > 0$, the discrete update $v_{n+1} = v_n - h\, g_A(v_n)$ gives $w_{n+1} = \big(1 - h(1 + A)\big)\, w_n$ for the deviation $w_n = v_n - v^\star$. It therefore converges to $v^\star$ if and only if $|1 - h(1 + A)| < 1$, that is, $0 < h(1 + A) < 2$, and it diverges for every $h > 0$ when $A < -1$. The gradient-flow statements of 3 thus carry over to gradient descent for sufficiently small steps, while large positive advantages can destabilize the discrete update even though the flow converges. In a shared network, the per-sample curvature $1 + A$ is replaced by a data-weighted curvature in parameter space, so these thresholds describe isolated samples rather than the full actor.

F The Bounded Objective as an Anchored Huber-Type Loss

We show that the sample-level gradient flow of the bounded objective is the gradient flow of a potential that combines a quadratic anchor at the reference with a Huber-type term in the residual. Let $\tilde{g}_A(v) = v - r - \mathrm{clip}_b\big(A\,(u - v)\big)$ denote half of the gradient of the bounded loss with the target held fixed, as in Proposition 7, and write $w := u - v$ for the residual.

Unbounded case. The potential of the unbounded objective can be rewritten as

$$\Phi_A(v) = \tfrac{1+A}{2}\|v\|_2^2 - \langle r + Au, v\rangle = \tfrac{1}{2}\|v - r\|_2^2 + \tfrac{A}{2}\|u - v\|_2^2 + \mathrm{const}, \tag{20}$$

which is a quadratic anchor at r plus an advantage-signed quadratic term in the residual. For $A < 0$, the second term is concave and dominates the first once $|A| > 1$, which is the divergence of Proposition 3.

Bounded case. Let H_δ denote the Huber function,

$$H_\delta(x) = \begin{cases} \tfrac{1}{2}x^2, & |x| \leq \delta, \\ \delta\,\big(|x| - \tfrac{\delta}{2}\big), & |x| > \delta, \end{cases} \tag{21}$$

which is continuously differentiable with $H'_\delta(x) = x$ for $|x| \leq \delta$ and $H'_\delta(x) = \delta\ \mathrm{sign}(x)$ otherwise. Define, for $A \neq 0$,

$$\Psi_A(v) := \tfrac{1}{2}\|v - r\|_2^2 + A\, H_{b/|A|}\big(\|u - v\|_2\big). \tag{22}$$

We claim that $\nabla_v \Psi_A = \tilde{g}_A$. Inside the region $\|Aw\|_2 \leq b$, that is, $\|w\|_2 \leq b/|A|$, the clip is inactive, $\tilde{g}_A(v) = v - r - Aw$, and $\nabla_v \Psi_A(v) = (v - r) + A\,(v - u) = v - r - Aw$, since there

$A\,H_{b/|A|}(\|w\|_2) = \frac{A}{2}\|w\|_2^2$. Outside this region, $\mathrm{clip}_b(Aw) = b\,\mathrm{sign}(A)\,w/\|w\|_2$, so $\tilde{g}_A(v) = v - r + b\,\mathrm{sign}(A)\,(v-u)/\|v-u\|_2$, while $\nabla_v\big[A\,H_{b/|A|}(\|v-u\|_2)\big] = A \cdot \frac{b}{|A|} \cdot \frac{v-u}{\|v-u\|_2} = b\,\mathrm{sign}(A)\,\frac{v-u}{\|v-u\|_2}$. The two expressions agree on both regions, and Ψ_A is continuously differentiable across the boundary because H_δ is.

Hence the sample-level gradient flow of the bounded objective is the gradient flow of Ψ_A: a quadratic anchor at the reference velocity, plus an attraction toward ($A > 0$) or repulsion from ($A < 0$) the conditional velocity that is quadratic within radius $b/|A|$ of u and grows only linearly beyond it. Because the linear term cannot dominate the quadratic anchor, Ψ_A is bounded below and coercive for every $A \in \mathbb{R}$, which is the potential-level counterpart of the forward invariance in Proposition 7. For $A > 0$, the bound replaces the quadratic attraction of equation 20 by a Huber attraction, so that samples far from the current prediction exert a force of magnitude at most b; for $A < 0$, it replaces an unbounded quadratic repulsion by a bounded one. For $A = 0$, both potentials reduce to the anchor $\frac{1}{2}\|v-r\|_2^2$.

G Behavior-Policy Proximity

This section details the proximity mechanism summarized in Section 3.4.

The surrogate in equation 3 uses the state distribution d^{π_k} of the behavior policy in place of that of the updated policy, so its relationship to the return depends on controlling policy movement even with exact advantages (Schulman et al., 2016), and repeated optimization on $\mathcal{B}_k$ additionally reuses actions and estimates collected under π_k. Neither the bound nor the reference controls the displacement from π_k. The bound constrains individual targets rather than accumulated movement, and for $\rho < 1$ the reference differs from the behavior policy, so that the ball is centered at $v_{\mathrm{ref}} \neq v_{\theta_k}$. We therefore measure the displacement from π_k along the stored sampling path $x^k_{i,\tau}$ that generated each action,

$$D_i(\theta) = \int_0^1 \big\| v_\theta(x^k_{i,\tau}, \tau, s_i) - v_{\theta_k}(x^k_{i,\tau}, \tau, s_i) \big\|_2^2 \,\mathrm{d}\tau, \tag{23}$$

which bounds the displacement of the action generated from the same noise, $\|a_i^\theta - a_i\|_2 \le e^L\sqrt{D_i}$, if the current velocity field is uniformly L-Lipschitz in its spatial argument (Proposition 4 below). It is a path-based proximity measure rather than a divergence between action distributions. We enforce the batch-average budget $\bar{D} := \frac{1}{N}\sum_i D_i \le \delta$ with a Lagrangian penalty $\lambda\,(\bar{D} - \delta)$, whose multiplier λ is updated by projected dual ascent; finite minibatch updates with a capped multiplier need not satisfy the budget exactly.

Since the penalty acts on the batch average, individual transitions may exceed the budget. For a transition with $D_i > \delta$, we test whether its samples have moved in the direction favored by their advantage, using the change of the conditional flow-matching loss relative to the behavior policy:

$$m_{ij}(\theta) = \mathbb{I}\Big[D_i(\theta) > \delta \;\wedge\; \tilde{A}_i\big(\ell^k_{ij} - \ell^\theta_{ij}\big) > 0\Big], \qquad \ell^\theta_{ij} = \|v_\theta(x_{ij}, \tau_{ij}, s_i) - u_{ij}\|_2^2, \tag{24}$$

where ℓ^k_{ij} is the loss at the behavior policy. Samples with $m_{ij} = 1$ are regressed onto v_{ref} instead of $\hat{v}_{ij}$, which halts further advantage-driven movement, while samples that have moved back toward the behavior policy keep their targets; the mask is recomputed at every step and treated as fixed during differentiation. The sign test is equivalent to $\tilde{A}_i \log \hat{\varrho}_{ij} > 0$ for the FPO surrogate ratio $\hat{\varrho}_{ij} = \exp(\ell^k_{ij} - \ell^\theta_{ij})$, but unlike in FPO the ratio enters only through its sign: it determines neither the update direction nor the size of the trust region, and the objective remains a squared regression loss throughout. The gate removes the advantage correction for the selected samples and does not by itself guarantee a decrease in D_i, in particular when $v_{\mathrm{ref}} \neq v_{\theta_k}$; the path penalty supplies the proximity signal.

G.1 Path Divergence and Action Displacement

We show that the path divergence equation 23 bounds the displacement of the action generated from the same noise. Fix a transition i and drop the index. Let x_τ denote the stored sampling path of the behavior policy, that is, the solution of $\dot{x}_\tau = v_{\theta_k}(x_\tau, \tau, s)$ from $x_0 = \epsilon$, with $a = x_1$, and let y_τ denote the path of the current policy from the same noise, $\dot{y}_\tau = v_\theta(y_\tau, \tau, s)$ with $y_0 = \epsilon$ and $a^\theta = y_1$. Assume that $v_\theta(\cdot, \tau, s)$ is L-Lipschitz in its first argument, uniformly in τ and s.

Proposition 4 (Action displacement)**.** *Under the above assumptions,*

$$\|a^\theta - a\|_2 \le e^L \sqrt{D(\theta)}, \qquad D(\theta) = \int_0^1 \big\|v_\theta(x_\tau,\tau,s) - v_{\theta_k}(x_\tau,\tau,s)\big\|_2^2 \,\mathrm{d}\tau. \tag{25}$$

Proof. Let $e_\tau := y_\tau - x_\tau$, so that $e_0 = 0$ and $e_1 = a^\theta - a$. Then

$$\dot e_\tau = v_\theta(y_\tau,\tau,s) - v_{\theta_k}(x_\tau,\tau,s) = \big[v_\theta(y_\tau,\tau,s) - v_\theta(x_\tau,\tau,s)\big] + \big[v_\theta(x_\tau,\tau,s) - v_{\theta_k}(x_\tau,\tau,s)\big]. \tag{26}$$

The first bracket has norm at most $L\,\|e_\tau\|_2$ by the Lipschitz assumption; denote the norm of the second by δ_τ. Hence $\frac{\mathrm{d}}{\mathrm{d}\tau}\|e_\tau\|_2 \le L\,\|e_\tau\|_2 + \delta_\tau$, and Grönwall's inequality with $e_0 = 0$ gives

$$\|e_1\|_2 \le \int_0^1 e^{L(1-\tau)}\,\delta_\tau \,\mathrm{d}\tau \le e^L \int_0^1 \delta_\tau \,\mathrm{d}\tau \le e^L \Big(\int_0^1 \delta_\tau^2 \,\mathrm{d}\tau\Big)^{1/2} = e^L\sqrt{D(\theta)}, \tag{27}$$

where the last inequality is Cauchy–Schwarz on $[0,1]$. □

The bound is evaluated along the stored path of the behavior policy only, so it requires no integration of the current velocity field. In practice, the sampler takes K Euler steps; the discrete analogue of the argument, with the discrete Grönwall inequality and the quadrature of D over the stored steps, yields the same bound with e^L replaced by $(1 + L/K)^K \le e^L$. The converse does not hold: a small action displacement does not imply a small path divergence, since velocity changes that cancel along the path leave the endpoint unchanged. The constraint on D is therefore the more conservative of the two.

H Relation to Ratio-Based Policy Optimization

The natural point of comparison for advantage regression is the ratio-based objective of FPO, which is driven by the same advantages and the same flow-matching losses. We compare the two under the sample-level model of Section 3.3. FPO (McAllister et al., 2025) constructs the surrogate ratio $\hat\varrho_\theta = \exp\big(\ell_{\theta_k} - \ell_\theta\big)$ from the conditional flow-matching loss $\ell_\theta = \|v_\theta(x_\tau,\tau,s) - u\|_2^2$ and maximizes a PPO-style clipped version of $A\,\hat\varrho_\theta$. At the behavior policy, this objective and advantage regression move the policy in the same direction.

Proposition 5 (Local equivalence to FPO)**.** *For $v_{\mathrm{ref}} = v_{\theta_k}$ and the same scaled advantage A in both objectives, the per-sample gradients satisfy*

$$\nabla_\theta \big\|v_\theta - \hat v\big\|_2^2\Big|_{\theta=\theta_k} = -\nabla_\theta\big(A\,\hat\varrho_\theta\big)\Big|_{\theta=\theta_k} = -2A\,\big(\nabla_\theta v_{\theta_k}\big)^\top (u - v_{\theta_k}). \tag{28}$$

Proof of Proposition 5. We consider a single sample (x_τ,τ,s,u) with scaled advantage A, write $v_\theta := v_\theta(x_\tau,\tau,s)$, and denote by $\nabla_\theta v_\theta$ the Jacobian of the velocity prediction with respect to the parameters.

AR gradient. Since the target $\hat v$ is detached, the chain rule gives

$$\nabla_\theta \big\|v_\theta - \hat v\big\|_2^2 = 2\,\big(\nabla_\theta v_\theta\big)^\top \big(v_\theta - \hat v\big). \tag{29}$$

At $\theta = \theta_k$, we have $v_\theta = v_{\theta_k} = v_{\mathrm{ref}}$, so that equation 6 yields $\hat v = v_{\theta_k} + A\,(u - v_{\theta_k})$ and hence $v_{\theta_k} - \hat v = -A\,(u - v_{\theta_k})$. Therefore,

$$\nabla_\theta \big\|v_\theta - \hat v\big\|_2^2\Big|_{\theta=\theta_k} = -2A\,\big(\nabla_\theta v_{\theta_k}\big)^\top (u - v_{\theta_k}). \tag{30}$$

FPO gradient. Differentiating the surrogate ratio,

$$\nabla_\theta\big(A\,\hat\varrho_\theta\big) = A\,\hat\varrho_\theta\,\nabla_\theta\big(\ell_{\theta_k} - \ell_\theta\big) = -A\,\hat\varrho_\theta\,\nabla_\theta\ell_\theta, \tag{31}$$

since ℓ_{θ_k} does not depend on θ. The flow-matching loss has gradient $\nabla_\theta\ell_\theta = 2\,(\nabla_\theta v_\theta)^\top (v_\theta - u)$. At $\theta = \theta_k$, the ratio equals $\hat\varrho_{\theta_k} = \exp(0) = 1$, and thus

$$\nabla_\theta\big(A\,\hat\varrho_\theta\big)\Big|_{\theta=\theta_k} = -2A\,\big(\nabla_\theta v_{\theta_k}\big)^\top (v_{\theta_k} - u) = 2A\,\big(\nabla_\theta v_{\theta_k}\big)^\top (u - v_{\theta_k}). \tag{32}$$

Comparing both expressions establishes the claim. Since AR minimizes its loss while FPO maximizes its surrogate, the resulting descent and ascent directions coincide. At $\theta = \theta_k$, the ratio lies within any PPO clipping range, so the clipped FPO objective has the same gradient. □

Proposition 5 shows that the two objectives cannot be distinguished by a single gradient step; the difference lies in what repeated optimization does afterwards. For advantage regression, Proposition 3 gives the answer: the prediction converges to an interpolation between the reference and the conditional velocity for $A > -1$ and diverges only below that threshold. The ratio objective behaves differently, but not more favorably.

Proposition 6 (Sample-level dynamics of the ratio objective)**.** *Let $\hat{\varrho}(v) = \exp(\ell_k - \|v-u\|_2^2)$ with ℓ_k fixed, and consider gradient ascent on the unclipped FPO objective, $\dot{v} = \nabla_v\big(A\,\hat{\varrho}(v)\big) = -2A\,\hat{\varrho}(v)\,(v-u)$. (i) If $A > 0$, $\|v-u\|_2$ decreases monotonically and $v \to u$. (ii) If $A < 0$, $\|v-u\|_2$ increases monotonically and without bound, $\hat{\varrho}(v) \to 0$, and the objective approaches its supremum 0 without attaining it. Under PPO clipping with band $[1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}]$ and $A < 0$, the clipped objective is constant on the set $\{v : \hat{\varrho}(v) < 1-\epsilon_{\text{clip}}\}$, so that its gradient vanishes there and no restoring force acts on v.*

Proof of Proposition 6. With $d := \|v-u\|_2$, the dynamics give $\frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{2}d^2 = -2A\,\hat{\varrho}(v)\,d^2$, whose sign is that of $-A$; this yields (i) and the monotonicity in (ii). For $A < 0$, $\dot{d} = 2|A|e^{\ell_k - d^2}d > 0$, and the time to reach any $D > d(0)$ is $\int_{d(0)}^{D} e^{d^2-\ell_k}/(2|A|d)\,\mathrm{d}d < \infty$, so $d \to \infty$. The remaining claims follow from $\hat{\varrho} = e^{\ell_k - d^2}$ and from the form of the clipped objective, $\min\{A\hat{\varrho}, A\,\mathrm{clip}(\hat{\varrho}, 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\} = A(1-\epsilon_{\text{clip}})$ for $A < 0$ and $\hat{\varrho} < 1-\epsilon_{\text{clip}}$. □

The two objectives thus differ in both regimes. For positive advantages, advantage regression stops at an interpolation with coefficient $A/(1+A) < 1$, whereas the ratio objective drives the prediction all the way to u, regardless of the magnitude of the advantage. For negative advantages, advantage regression diverges only for $A < -1$, whereas the ratio objective has no minimizer for any $A < 0$: the prediction drifts away from u at a rate that decays with the distance, and clipping stops the drift once the ratio leaves the band but does not reverse it, so that updates driven by other samples through shared parameters can carry it further. In both objectives, the source of the drift is a negative-advantage sample whose effective objective has no minimizer; the bound of Section 3.3 removes it for advantage regression, while clipping leaves it in place for the ratio objective.

I Bounded Dynamics

We use the sample-level model of Section 3.3: a single sample (x_τ, τ, s, u) with prediction v treated as a free variable and reference r held fixed. The bounded target equation 10 yields, with the target held fixed during differentiation, half of the gradient of the bounded loss

$$\tilde{g}_A(v) = v - r - \mathrm{clip}_b\big(A\,(u-v)\big). \tag{33}$$

Proposition 7 (Bounded dynamics)**.** *Consider the gradient flow $\dot{v} = -\tilde{g}_A(v)$. For every $A \in \mathbb{R}$ and every v with $\|v-r\|_2 > b$,*

$$\frac{\mathrm{d}}{\mathrm{d}t}\|v-r\|_2 \le -\big(\|v-r\|_2 - b\big). \tag{34}$$

Hence the ball $\mathcal{B}_b(r) = \{v : \|v-r\|_2 \le b\}$ is forward invariant, and $\|v(t)-r\|_2 - b \le \big(\|v(0)-r\|_2 - b\big)\,e^{-t}$ for every initialization outside the ball. The same holds for gradient descent $v^+ = v - h\,\tilde{g}_A(v)$ with step size $0 < h \le 1$, with contraction factor $1-h$ per step.

Proof. Let $w = v-r$ and $z = \mathrm{clip}_b(A(u-v))$, so that $\tilde{g}_A(v) = w - z$ and $\|z\|_2 \le b$ by construction. For $w \neq 0$,

$$\frac{\mathrm{d}}{\mathrm{d}t}\|w\|_2 = \frac{\langle w, \dot{w}\rangle}{\|w\|_2} = -\frac{\langle w, w-z\rangle}{\|w\|_2} = -\|w\|_2 + \frac{\langle w, z\rangle}{\|w\|_2} \le -\|w\|_2 + b, \tag{35}$$

where the inequality is Cauchy–Schwarz. This is the stated bound. On the boundary $\|w\|_2 = b$ the derivative is nonpositive, so trajectories cannot leave $\mathcal{B}_b(r)$. Outside the ball, $d(t) := \|w(t)\|_2 - b > 0$ satisfies $\dot{d} \le -d$, and Grönwall's inequality gives $d(t) \le d(0)\,e^{-t}$. For gradient descent,

$w^+ = (1-h)\,w + h\,z$, so for $0 < h \le 1$ the triangle inequality gives $\|w^+\|_2 \le (1-h)\|w\|_2 + hb$, that is, $\|w^+\|_2 - b \le (1-h)\big(\|w\|_2 - b\big)$. □

The proposition holds for every value of A, including $A \le -1$, where the unbounded potential Φ_A of Appendix E has no minimizer. It does not assert convergence to a single point inside the ball: for $\|A(u-v)\|_2 \le b$ the clipping is inactive and the dynamics reduce to those of Proposition 3, so that for $A > -1$ the prediction converges to $v^\star$ whenever $v^\star \in \mathcal{B}_b(r)$, and for $A \le -1$ it settles on the boundary of the ball, where the outward pull of the advantage term and the inward pull of the anchor balance.

J Geometry of the Bounded Target

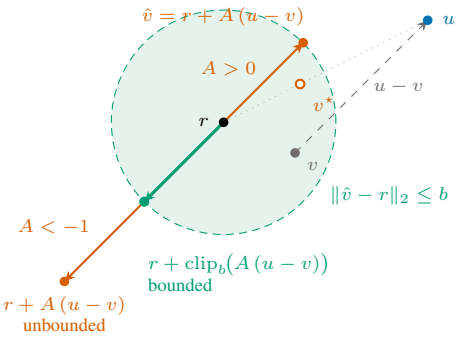


Figure 6: **Bounded advantage regression at one sample.** The target is the reference velocity r shifted along the residual $u - v$ by the scaled advantage A. For $A > 0$, the shift points toward the conditional velocity u, and repeated optimization converges to $v^\star$ between r and u. For $A < -1$, it points away from u and grows with $|A|$ and with the residual, so that repeated optimization has no fixed point; the bound clips the shift to the ball of radius b around r (Proposition 7).