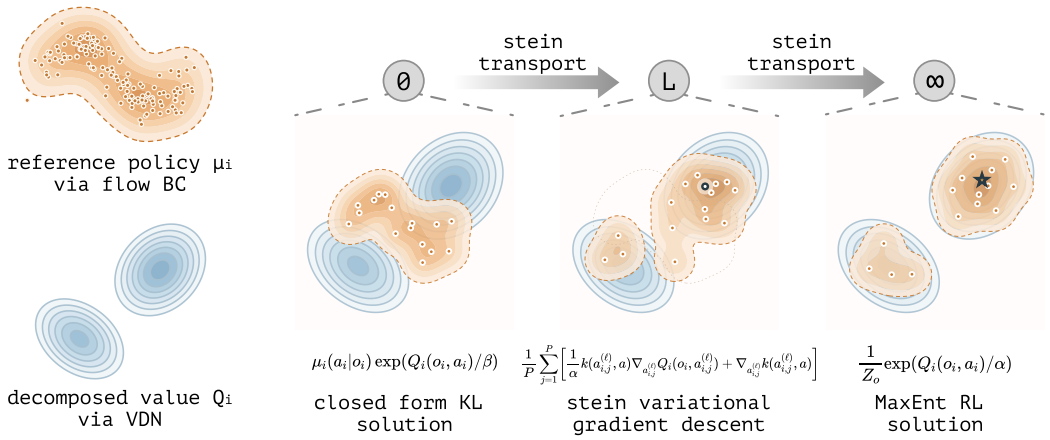
Test-time Multi-agent Coordination by Decomposed Value Gradient Flow

Dongsu Lee **Haoran Xu** **Amy Zhang**
The University of Texas at Austin
dongsu.lee@utexas.edu
https://github.com/DongsuLeeTech/scout.git

Abstract

Offline multi-agent reinforcement learning (MARL) faces a persistent trade-off. Expressive generative policies can represent multi-modal coordination in the data, but cannot distinguish high-value regions, while value-optimized policies exploit the learned Q-function but collapse the multi-modal into a single dominant mode. A single agent's mode collapse can break joint coordination, and simultaneous drift across agents can push the joint policy into unseen regions of the action space. We propose **scalable coordination via optimal unified transport (SCOUT)**, the first offline MARL framework to combine a generative foundation model with a learned value function through test-time action refinement. SCOUT trains two decoupled components: a flow-matching behavioral prior and a decomposed value function. At test-time, it transports behavioral samples toward high-value regions via Stein variational gradient descent. The number of transport steps controls adaptive test-time scaling, replacing a fixed regularization coefficient. Under the individual-global-max (IGM) principle, we prove a single-term KL bound on the joint soft-value gap that vanishes as transport converges, with an irreducible additive residual proportional to the IGM violation. Empirically, SCOUT achieves the best average performance across discrete and continuous offline MARL benchmarks and yields performance improvements in all offline-to-online configurations.



(a) Setup (b) SCOUT: test-time transport via decomposed value gradient

Figure 1: **Overview of the proposed solution.** (*a*) The prerequisite for SCOUT is to train the reference policy via flow matching and decomposed value via a value decomposition network (VDN). (*b*) SCOUT transports the behavior distribution at test time via Stein variational gradient descent.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Offline MARL is motivated by the need to learn coordinated multi-agent behaviors in safety-critical or cost-constrained domains where active data collection is infeasible or expensive, *e.g.*, autonomous driving [1], logistics [2], and robotics [3]. In this setting, the learning objective is twofold. Agents must learn to coordinate their behavior from heterogeneous and suboptimal demonstrations, while avoiding the distributional shift that arises when the learned policy visits state-action regions unsupported by the data. Early offline MARL methods [4–6] rely on simple policy classes (*i.e.*, Gaussian), which struggle to represent multi-modal and coordinated joint action distributions. When offline data contains a mixture of diverse coordination strategies, a unimodal policy often collapses onto a single mode and produces out-of-distribution joint actions that no agent in the data ever executed together.

To address this expressiveness gap, recent works adopt powerful generative models, *including* diffusion models [7, 8] and flow matching models [9, 10], as policy representations in offline MARL. Although these models improve the representational power of policy, their iterative sampling process incurs a high inference cost. MAC-Flow [10] and OM^{2}P [9] mitigate this via one-step distillation and mean-flow generation, respectively. However, improved expressiveness and faster inference do not resolve a more fundamental limitation. Since the outcome policy is optimized against a learned Q-function, it inherently tends to collapse diverse coordination modes into a single or blurred one.

Methods that parameterize a policy and optimize it to maximize a learned Q-function must balance value improvement against staying close to the behavioral distribution. A fixed regularization coefficient controls this trade-off. If it is too large, the policy stays too close to suboptimal data; if it is too small, the policy exploits errors in the learned Q-function and produces OOD actions. Although this tension is well-known in single-agent offline RL [11, 12], the multi-agent setting introduces the following compounding difficulty. Training-time value maximization deforms each agent's policy away from the behavioral prior, and the regularization coefficient bounds how far each marginal may drift, but cannot bound the compound effect on the joint. Even when every marginal remains within its individual support, the joint policy can place mass on action combinations that no trajectory in the data ever visited (*i.e.*, regions where the learned Q-function is least reliable). In fact, the data may support multiple distinct joint modes of comparably high estimated value, but committing each marginal to a single greedy mode at training time discards the others before the agent has any way to tell them apart. Once new interaction data arrives, the policy can no longer use it to discriminate which mode is truly higher-value, because the alternatives have already been suppressed.

A natural alternative is to decouple two objectives, motivated by test-time training [13–15]. Instead of learning a policy that directly maximizes value, we keep the behavioral policy as a foundational prior and defer value-based refinement to test time. At deployment, we sample action candidates from this prior and then move them toward high-value regions using value gradients. The extent of this adjustment acts as adaptive control, replacing the need for a fixed regularization coefficient.

In this work, we ask a question not yet studied by prior works:

How do we avoid explicit policy parameterization for value maximization, and instead recover coordination at test time by directly transporting behavioral samples toward high-value regions?

We answer this question with the proposed solution, **SCOUT (scalable coordination via optimal unified transport)**, an offline MARL framework that combines flow-matching behavioral priors with decomposed value gradient for test-time action refinement. Our key contributions are:

- We propose a policy-free value maximization for offline MARL that separates behavioral modeling from value improvement, replacing training-time regularization tuning with a test-time refinement budget.
- We derive decomposed value gradient transport under the IGM principle, enabling fully decentralized test-time refinement that recovers inter-agent coordination with a bounded approximation gap.
- Empirically, test-time refinement strengthens coordination without retraining, achieving the best average performance in continuous and discrete domains. Notably, unlike existing flow-based MARL, our method yields improvements in all offline-to-online RL experiments.

2 Related Works

Offline MARL. Several works have explored the data-driven sequential decision-making problem, *including* offline RL and MARL, which aims to extract a return-maximizing policy within the support of an offline dataset. Specifically, offline MARL requires learning coordinated policies for multiple agents. This is not trivial because it requires modeling dependencies among agents in a combinatorially large joint action space. The simplest baselines are multi-agent extensions of single-agent offline RL methods, either by introducing conservatism in the value function [11, 12] or the actor [16], or by performing Bellman updates without querying out-of-distribution actions [17]. To further promote coordination, more advanced works are often combined with additional multi-agent design choices, such as value decomposition [5, 18], policy optimization with a centralized critic [4, 6], or a centralized world model [19]. However, these methods often struggle to capture the multi-modal distributions present in multi-agent offline datasets because of their limited policy expressiveness. To address this issue, recent work has adopted more expressive generative policies, including an autoregressive model [20], diffusion models [7], and flow matching [10]. However, these methods couple value learning with the generative process itself: they either anchor the policy via BC-like losses [21, 22, 10, 23], which produces overly conservative behavior, or backpropagate value gradients through iterative sampling [24, 25], which is unstable. In contrast, the proposed solution bypasses explicit policy parameterization altogether. Instead, Stein variational gradient [26, 27, 14] fields refine flow-generated action candidates toward high-value regions at test time. This enables adaptive test-time scaling while retaining generative expressiveness.

Test-time action refinement. Our main idea in this work is based on test-time refinement (*i.e.*, $a \leftarrow a + \nabla_a Q(s, a)$) [13], which refines actions from a base generative policy at inference time rather than during training. This paradigm has been explored along several directions in single-agent RL. Rejection sampling methods generate multiple action candidates and select the highest-valued one [28, 29]. Gradient-based approaches directly optimize sampled actions via Q-function gradients [25, 14, 30]. Residual policies instead train a separate network to modify a base policy in noise or action space [31, 32]. Complementary to these, optimal transport (OT) offers a principled framework for moving probability mass toward high-value regions [33, 34]. More recently, particle-based methods adopt an OT viewpoint to transport samples toward value-preferred regions [35]. However, these approaches remain limited to single-agent settings. The proposed solution is motivated by this direction but specifically designed to solve the multi-agent coordination problem. This avoids explicit policy parameterization and naturally handles the combinatorial joint action space.

3 Preliminaries

Problem setting. We consider cooperative MARL problems modeled as a decentralized partially observable Markov decision process (Dec-POMDP) [36], defined by the tuple $\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, \{\mathcal{O}_i\}, \{\mathcal{A}_i\}, \mathcal{T}, \{\Omega_i\}, \{r_i\}, \gamma \rangle$, where $\mathcal{I} = \{1, 2, \ldots, I\}$ is a finite set of agents, $\mathcal{S}$ denotes the state space, $\mathcal{O}_i$ and $\mathcal{A}_i$ are the observation and action spaces of agent i, $\mathcal{T}(s' \mid s, \mathbf{a}) : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_I \to \Delta(\mathcal{S})$ is state transition function over state s and joint action $\mathbf{a} = [a_1, \ldots, a_I]$, $\Omega_i(o_i \mid s) : \mathcal{S} \to \Delta(\mathcal{O}_i)$ is the observation function, $r_i(s, \mathbf{a}) : \mathcal{S} \times \prod_i \mathcal{A}_i \to \mathbb{R}$ is an individual reward, and $\gamma \in [0, 1)$ is the discount factor. In the offline setting, agents learn from a fixed dataset $\mathcal{D} = \{\tau^{(n)}\}_{n=1}^{N}$, where each trajectory $\tau^{(n)}$ is collected by unknown behavioral policies. The behavioral policy can be empirically defined as $\mu(\mathbf{a} \mid \mathbf{o}) \approx p_{\mathcal{D}}(\mathbf{a} \mid \mathbf{o})$ and its per-agent marginals as $\mu_i(a_i \mid o_i)$. We consider the centralized training and decentralized execution (CTDE) [4, 6, 37, 10].

Offline RL methods typically aim for Q-value maximization because optimizing cumulative return at a given MDP is infeasible without additional interaction [38, 12]. To make this tractable in MARL, we decompose the global action-value function $Q_{\text{tot}}(\mathbf{o}, \mathbf{a})$ into per-agent utilities $\{Q_i(o_i, a_i)\}_{i=1}^{I}$ under the individual-global-max (IGM) principle [39, 40] as follows:

$$\arg\max_{\mathbf{a}} Q_{\text{tot}}(\mathbf{o}, \mathbf{a}) = \big(\arg\max_{a_1} Q_1(o_1, a_1), \ldots, \arg\max_{a_I} Q_I(o_I, a_I)\big). \tag{1}$$

We adopt the additive decomposition throughout. Under this, each agent independently maximizes its own value, and the resulting joint action is optimal, *i.e.*, $\max_{\Pi} \mathbb{E}_{\mathbf{o}\sim\mathcal{D}}[Q_{\text{tot}}(\mathbf{o}, \Pi(\mathbf{o}))] = \sum_{i=1}^{I} \max_{\pi_i} \mathbb{E}_{o_i\sim\mathcal{D}}[Q_i(o_i, \pi_i(o_i))]$. However, direct maximization of a learned Q-function leads to

OOD errors. Therefore, in practice, the per-agent policy extraction problem can be defined as follows:

$$\pi_i^* = \arg\max_{\pi_i} \mathbb{E}_{o_i\sim\mathcal{D}, a_i\sim\pi_i(\cdot|o_i)}\big[Q_i(o_i,a_i)\big], \quad \text{s.t.} \quad \mathbb{E}_{o_i\sim\mathcal{D}}\Big[D\big(\pi_i(\cdot|o_i), \mu_i(\cdot|o_i)\big)\Big] \le \epsilon, \tag{2}$$

where D is a divergence measure, *e.g.*, KL-divergence or ℓ_2-norm, and ϵ is a threshold.

Policy extraction. The behavior-regularized objective function in (2) can be optimized by two families of solutions in general [41], differentiated by what they focus mode-seeking and mode-covering.

For mode-seeking, a representative example is a parameterized gradient [16, 42], which extracts a Gaussian policy by converting the constraint into a penalty, following `DDPG+BC` objective [5, 41]:

$$\max_{\pi_i} \mathbb{E}_{o_i\sim\mathcal{D}}\Big[\mathbb{E}_{a_i\sim\pi_i}\big[Q_i(o_i,a_i)\big] - \beta D(\pi_i(\cdot|o_i), \mu_i(\cdot|o_i))\Big]. \tag{3}$$

An alternative for a mode-seeking family is to use distillation to compress the multi-step generative model into a one-step policy [43, 22, 10]. However, these methods output Gaussian policies, which may be insufficient because they collapse into a single mode and lack expressivity. For expressive generative policies such as diffusion [44] or flow models [45], the parameterized gradient requires backpropagation through multi-step iterations, which is expensive and numerically fragile [21].

A second family samples actions from μ and reweights them by values. A closed-form solution is $\pi^*(a|s) \propto \mu(a|s)\exp(Q(s,a)/\beta)$, approximated by `weighted BC` [46, 6, 47]:

$$\max_{\pi} \mathbb{E}_{o_i\sim\mathcal{D}, a_i\sim\mu_i}\big[\exp(Q_i(o_i,a_i)/\beta)\cdot\log\pi_i(a_i \mid o_i)\big].$$

While this has been viewed as overly conservative [48], it carries a distinct advantage for multi-agent coordination [41]. A key implication is that preserving the coordinated strategies in the dataset is often more important than aggressive mode-seeking, as independent mode selection can break inter-agent compatibility. As a theoretically near-optimal solution for this problem, best-of-N draws N candidate actions from μ and returns the highest-valued one [49–51, 14] as follows:

$$a_i^* = \arg\max_{a_i^k\sim\mu_i(\cdot|o_i), k\in[N]} Q_i(o_i, a_i^k). \tag{4}$$

This requires no policy training in terms of value maximization beyond behavioral policy modeling.

Flow-matching and behavioral policy modeling. To model behavioral policy in a multi-agent system, generative modeling offers a flexible framework that can capture multi-modal distributions [7, 8, 10, 9, 52]. Specifically, flow-matching can relax the tradeoff between coordination performance and computational cost in MARL. Flow-matching learns a time-dependent velocity field $v_\phi(t,x) : [0,1]\times\mathbb{R}^d \to \mathbb{R}^d$ that transports noise $p_0 = \mathcal{N}(0, I_d)$ to the data distribution p_1 [45]. Given the conditional linear path $x_t = (1-t)x_0 + tx_1$, the training objective is as follows:

$$\mathcal{L}_{\text{FlowMatching}}(\phi) = \mathbb{E}_{x_0\sim p_0, x_1\sim p_1, t\sim\text{Unif}([0,1])}\big[\|v_\phi(t,x_t) - (x_1 - x_0)\|_2^2\big].$$

At inference, a sample is produced by integrating the ODE from initial noise [53–55].

Under multi-agent learning, different structural assumptions on the velocity field yield a hierarchy of policy representations [9, 10]. A joint flow policy $\mu_\phi(\mathbf{o}, \mathbf{z})$ conditions the velocity field on the full joint observation $\mathbf{o}$ and joint noise $\mathbf{z} \sim \prod_i \mathcal{N}(0, I_{d_i})$, capturing inter-agent dependencies, as follows:

$$\mathcal{L}_{\text{JointFlow}}(\phi) = \mathbb{E}_{\mathbf{z}\sim p_0,\ (\mathbf{o},\mathbf{a})\sim\mathcal{D},\ t\sim\text{Unif}([0,1])}\big[\|v_\phi(t,\mathbf{o},x_t) - (\mathbf{a}-\mathbf{z})\|_2^2\big]. \tag{5}$$

While expressive, the joint policy violates CTDE at execution time. Instead, a factorized model trains individual policy $\hat{p}(\mathbf{a} \mid \mathbf{o}) = \prod_i \hat{p}_i(a_i \mid o_i)$ [8]. This is fully decentralized but discards coordination patterns. A projected model [10, 52] bridges the two by learning a joint flow and then distilling it into the factorized one, $\hat{p}_{\text{proj}} = \arg\min_{\hat{p}\in\mathcal{P}} \mathcal{L}(\hat{p}\|p)$, retaining as much coordination as the factorized structure permits. In practice, this can be written as follows:

$$\mathcal{L}_{\text{ProjectedFlow}}(\mathbf{w}) = \mathbb{E}_{\mathbf{o}\sim\mathcal{D},\ \mathbf{z}\sim\mathbf{p}_0}\left[\sum_{i=1}^{I} ||\mu_{w_i}(o_i, z_i) - [\mu_\phi(\mathbf{o},\mathbf{z})]_i||_2^2\right]. \tag{6}$$

4 Scalable Coordination via Optimal Unified Transport (SCOUT)

As motivated in Section 1, our goal is to develop an offline MARL algorithm that can extract Q-maximizing multi-agent policies without explicit policy parameterization for value maximization. Following prior works [14, 56, 57], we consider the MaxEnt surrogate of (2), $\mathbb{E}_{a\sim\pi}[Q(s,a)] + \alpha\mathcal{H}(\pi(\cdot|s))$, whose optimal policy takes the Boltzmann form $\pi^*(a|s) \propto \exp(Q(s,a)/\alpha)$.

Rather than fitting an explicit actor to approximate π^*, we construct a transport map that moves samples from μ toward π^* via variational gradient descent, using $\nabla_a Q$ as the driving force. Specifically, given a set of action candidates sampled from a flow-matching behavioral policy, we iteratively refine them at test time via the simplest possible update as $a \leftarrow a + \eta\nabla_a Q(o,a)$. The resulting refined actions constitute the implicit policy output, with the number of gradient steps L serving as a tunable knob that controls coordination strength at deployment without retraining.

Why do we need an implicit policy? This direction offers several appealing properties. First, because the initial actions are drawn from the behavioral policy, the implicit policy refines the behavioral prior rather than replacing it. Then, the transport budget provides an adaptive test-time scaling. When $L = 0$, it recovers the best sampling from the behavioral policy, while increasing L progressively pushes actions toward Q-maximizing regions. Furthermore, the gradient-based transport can discover actions beyond the reference distribution when the Q-function generalizes well.

4.1 Why is this hard in multi-agent settings?

The above idea and problem hold straightforwardly in the single-agent setting [14, 48], where the transport operates on a single action space $\mathcal{A}$ guided by a single Q-function. However, extending this to multi-agent settings is non-trivial. Under the CTDE paradigm, each agent i can only access its local observation o_i at test time. This means the action refinement must run independently for each agent, using only its local Q-function $Q_i(o_i, a_i)$ and its own action particles. The refined joint action $(a_1^*, \ldots, a_I^*)$ must be globally coherent. Namely, each agent's refined action should be coordinated with those of other agents. This tension between decentralized execution and coordinated behavior has been a long-standing challenge in MARL [39, 40].

The most common solution is the IGM principle [39] in (1), which ensures independent per-agent maximization to align with joint maximization. However, several practical questions remain. First, IGM guarantees alignment at the greedy argmax, but the gradient-based transport involves iterative updates that modify the particle distribution across multiple steps. It is unclear whether the per-agent transport dynamics remain aligned with the joint optimum throughout this process, especially when the critic must evaluate actions that are progressively shifted away from the behavioral support during training. Second, the individual behavioral prior might discard the inter-agent correlations present in the offline data. Even if the transport dynamics are correctly factorized, the initial particles may lack the coordination patterns necessary for effective joint behavior.

4.2 Proposed solution

In this subsection, we now present the full SCOUT framework for multi-agent test-time coordination.

Flow-matching behavioral prior. We learn a per-agent velocity field via flow matching [45].

$$\mathcal{L}_{\text{Flow-BC}}(w_i) = \mathbb{E}_{x_i^0\sim p_0,\ (o_i,a_i)\sim\mathcal{D},\ t\sim\text{Unif}([0,1])}\left[\|v_{w_i}(t,o_i,x_i^t) - (a_i - x_i^0)\|_2^2\right]. \tag{7}$$

This is a generative model of the behavioral policy $\mu_i(\cdot|o_i)$ that serves as the source distribution for the transport. At inference time, action samples are produced by Euler integration over K steps.

Decomposed value function. The value function has to satisfy that it must decompose under IGM so that per-agent gradient transport is globally consistent. We achieve this by training $\{Q_{\theta_i}\}_{i=1}^I$ whose TD targets are computed through an additive decomposition as $Q_{\text{tot}}(\mathbf{o},\mathbf{a}) = \sum_{i=1}^I Q_{\theta_i}(o_i,a_i)$. This is the simplest decomposition satisfying IGM. The critic is trained via TD learning with targets $y = r + \gamma Q_{\text{tot}}(\mathbf{o}',\mathbf{a}')$, where the next-state actions a_i' are obtained by policy. This global credit signal is what enables per-agent gradient transport to produce coordinated behavior at test time.

Action refinement via decomposed gradient transport. Given the pre-trained behavioral prior and the decomposed value function, we now introduce the test-time refinement as the core of SCOUT. Each

agent i begins by drawing P action particles from its behavioral flow policy, $a_{i,j}^{(0)} = \mu_{w_i}(o_i, z_{i,j})$ with $z_{i,j} \sim \mathcal{N}(0, I)$ for $j = \{1, \ldots, P\}$. These particles are then iteratively refined for L steps, motivated by Stein variational gradient descent (SVGD) as transport in Stein geometry [26, 27] as follows:

$$\hat{\phi}_i^*(a) = \frac{1}{P}\sum_{j=1}^{P}\left[\frac{1}{\alpha}k(a_{i,j}^{(\ell)}, a)\nabla_{a_{i,j}^{(\ell)}}Q_i(o_i, a_{i,j}^{(\ell)}) + \nabla_{a_{i,j}^{(\ell)}}k(a_{i,j}^{(\ell)}, a)\right] \quad \text{s.t. } a_{i,j}^{(\ell+1)} = a_{i,j}^{(\ell)} + \epsilon\hat{\phi}_i^{\text{clip}}(a_{i,j}^{(\ell)}), \tag{8}$$

where $k(x, y) = \exp(-\gamma\|x - y\|^2)$ is an RBF kernel with adaptive median bandwidth. The first term in (8) is a kernel-smoothed value gradient that drives particles toward high-Q_i regions; the second term is a repulsive force that prevents particle collapse and preserves multi-modality.

For discrete action spaces, the refinement operates in the logit space. The flow policy produces logit vectors $\ell_{i,j} \in \mathbb{R}^{|\mathcal{A}_i|}$, and the value gradient is computed as $\nabla_\ell \sum_a \text{softmax}(\ell)_a \cdot Q_i(o_i, a)$ via automatic differentiation through the softmax-weighted Q-value.

4.3 Theoretical analysis

We now provide theoretical guarantees for the SCOUT framework.

Soft value gap under approximate IGM. The learned decomposition is at best an approximation of the true joint value, so we adopt the following residual assumption.

Assumption 4.1. *The global Q-function decomposes as $Q_{\text{tot}}(\mathbf{o}, \mathbf{a}) = \sum_{i=1}^{I} Q_i(o_i, a_i) + \Delta(\mathbf{o}, \mathbf{a})$, where the residual interaction term satisfies $\|\Delta\|_\infty \le \delta$.*

When $\delta = 0$, exact additive IGM holds and decentralized transport recovers the global Boltzmann optimum. The following result quantifies the soft-value gap when $\delta > 0$.

Theorem 4.2 (Soft value gap under approximate IGM)**.** *Let $\tilde{\pi}^L = \prod_i \tilde{\pi}_i^L$ be the joint policy after L steps of independent per-agent transport, each targeting $\pi_i^* \propto \exp(Q_i/\alpha)$. Let $\pi_{\text{joint}}^* \propto \exp(Q_{\text{tot}}/\alpha)$ be the true joint Boltzmann target and $\mathcal{J}_{\text{soft}}(\pi) := \mathbb{E}_\pi[Q_{\text{tot}}] + \alpha\mathcal{H}(\pi)$. Then*

$$0 \le \mathcal{J}_{\text{soft}}(\pi_{\text{joint}}^*) - \mathcal{J}_{\text{soft}}(\tilde{\pi}^L) \le \alpha \underbrace{\text{KL}\left(\tilde{\pi}^L \middle\| \prod_i \pi_i^*\right)}_{\text{transport error}} + \underbrace{4\delta}_{\text{IGM violation}}. \tag{9}$$

The transport error is a single KL term that vanishes as $L \to \infty$ under idealized population SVGD with exact Q-functions and suitable step sizes. The SVGD performs gradient descent on $\text{KL}(\cdot\|\prod_i \pi_i^*)$ in the kernel-induced Stein geometry [26, 27]. The 4δ term decomposes as 2δ from the per-agent transport target π_i^* and 2δ from the gap between the factorized and joint Boltzmann optima; both arise from the IGM decomposition itself and apply equally to all decomposed-Q methods, including MAC-Flow's distillation [10]. What distinguishes SCOUT is that the transport error can be driven down by increasing L at test time without retraining, whereas distillation-based methods fix their approximation quality at training time.

Decomposed transport approximates joint best-of-P. Even though each agent's gradient update is entirely local, we show that the combination of decomposed transport and per-agent selection tracks the joint best-of-P outcome up to the irreducible IGM gap.

Proposition 4.3 (Decomposed transport under IGM)**.** *Consider P joint candidates with independently initialized particles $(a_{1,j}^{(0)}, \ldots, a_{I,j}^{(0)}) \sim \prod_i \mu_i$. After L steps of per-agent gradient transport: (i) when $\Delta = 0$, per-agent selection $a_i^* = \arg\max_j Q_i(o_i, a_{i,j}^{(L)})$ yields $\text{Cov}(a_i^*, a_{i'}^*) = 0$ and is jointly optimal; (ii) when $\|\Delta\|_\infty \le \delta$, per-agent selection approximates joint best-of-P within the irreducible IGM gap as $Q_{\text{tot}}(a_1^*, \ldots, a_I^*) \ge \max_{j\in[P]} Q_{\text{tot}}\left(a_{1,j}^{(L)}, \ldots, a_{I,j}^{(L)}\right) - 2\delta$.*

Herein, (*i*) follows since each agent's particles and selection depend only on its own Q_i and initial noise. Then, (*ii*) follows from $|Q_{\text{tot}} - \sum_i Q_i| \le \delta$ uniformly; full proofs are in Appendix A.

Implicit coordination via TD targets. Proposition 4.3 shows that decomposed transport is safe, *i.e.*, it does not lose more than the irreducible 2δ relative to a hypothetical centralized refiner. The complementary question is whether the per-agent gradient $\nabla_{a_i} Q_i$ itself encodes inter-agent dependencies. Recall the TD target $y = r + \gamma \sum_j Q_j(o'_j, a'_j)$. At the TD fixed point, Q_i inherits

Algorithm 1 SCOUT's training & evaluation

Initialize v_w, $\{Q_{\theta_i}\}$, $\{\bar{Q}_{\bar{\theta}_i}\}$
while not converged **do**
  Sample $(o_i, a_i, r, o_i')_{i=1}^{I} \sim \mathcal{D}$
  # INDIVIDUAL BEHAVIORAL FOUNDATION
  $x_i^0 \sim \mathcal{N}(0, I)$, $t \sim \mathrm{Unif}([0,1])$
  $x_i^t \leftarrow (1-t)\, x_i^0 + t\, a_i$
  $\mathcal{L}_{\mathrm{BC}} \leftarrow \|v_w(t, o_i, x_i^t) - (a_i - x_i^0)\|^2$
  # DECOMPOSED VALUE LEARNING
  $\{a_i'\} \leftarrow$ REFINEMENT$(o_i', v_w, \bar{Q}, L_{\mathrm{train}})$ ▷ Alg. 3
  $y \leftarrow r + \gamma \sum_i (\bar{Q}_{\bar{\theta}_i}(o_i', a_i'))$
  $\mathcal{L}_{\mathrm{critic}} \leftarrow \frac{1}{2} |\sum_i Q_{\theta_i}(o_i, a_i) - y|^2$
# EVALUATION
while episode not done **do**
  for agent $i = 1, \ldots, I$ **in parallel do**
    $\{a_{i,j}^{(0)}\}_{j=1}^{P} \leftarrow$ ODE(o_i, v_w, P) ▷ Alg. 2
    $\{a_{i,j}^{(L)}\} \leftarrow$ REFINEMENT$(o_i, Q_{\theta_i}, \{a_{i,j}^{(0)}\}, L_{\mathrm{test}})$ ▷ Alg. 3
    $a_i^* \leftarrow \arg\max_j Q_{\theta_i}(o_i, a_{i,j}^{(L)})$
  Execute $(a_1^*, \ldots, a_I^*)$, observe (o_i', r, d)

Algorithm 2 ODE solver

Require: observation o_i, velocity field v_w, particles P
function ODE(o_i, v_w, P)
  $a \leftarrow z \sim \mathcal{N}(0, I)$ ▷ P particles
  for $k = 0, \ldots, M-1$ **do**
    $a \leftarrow a + \frac{1}{M}\, v_w\big(\frac{k}{M},\, o_i,\, a\big)$
  return a ▷ behavioral samples

Algorithm 3 SCOUT's refinement

Require: observation o_i, critic Q_i, particles $\{a_{i,j}\}$, refinement steps L, refinement step size ϵ
function REFINEMENT$(o_i, Q_i, \{a_{i,j}\}, L)$
  $a_{i,j}^{(0)} \leftarrow a_{i,j}$
  for $\ell = 0, \ldots, L-1$ **do**
    $\hat{\phi}_i^* \leftarrow \frac{1}{P} \sum_j \big[\frac{1}{\alpha}\, k(a_j, \cdot)\, \nabla_{a_j} Q_i + \nabla_{a_j} k(a_j, \cdot)\big]$
    $a_{i,j} \leftarrow a_{i,j} + \epsilon\, \mathrm{clip}(\hat{\phi}_i,\, c)$
  return $\{a_{i,j}\}$ ▷ refined particles

sensitivity to other agents' actions through the joint distribution of a_j' used to form the target. Concretely, fitted Q_i can be interpreted as approximating $\nabla_{a_i} Q_i(o_i, a_i) \approx \mathbb{E}_{a_{-i} \sim \rho}[\nabla_{a_i} Q_{\mathrm{tot}}(\mathbf{o}, \mathbf{a})]$ along the training distribution ρ. Under additive decomposition, $\partial Q_{\mathrm{tot}} / \partial Q_i = 1$, so $\nabla_{a_i} Q_i$ preserves the sign of the joint gradient $\nabla_{a_i} Q_{\mathrm{tot}}$, and per-agent transport along $\nabla_{a_i} Q_i$ is consistent with joint Q-improvement. Coordination is thus inherited from training, not synthesized at test time, and decomposed transport amplifies it through repeated kernel-smoothed updates over the particle set.

4.4 Algorithm summary

We summarize the SCOUT framework in Pseudocode 1. Training alternates between two decoupled objectives: (*i*) the flow-matching loss $\mathcal{L}_{\mathrm{BC}}$ fits the individual behavioral prior with ODE in Pseudocode 2, and the TD loss $\mathcal{L}_{\mathrm{critic}}$ learns decomposed Q functions whose targets are computed through the additive decomposition $Q_{\mathrm{tot}} = \sum_i Q_i$ using actions refined by L_{train} transport steps in Pseudocode 3.

At test time, each agent independently generates P behavioral candidates via ODE integration, refines them for L_{test} steps of decomposed value gradient flow, and selects the highest-Q_i particle. The transport budget $(L_{\mathrm{test}}, \epsilon)$ need not match training. In particular, $L_{\mathrm{test}} = 0$ recovers best-of-P sampling, $L_{\mathrm{test}} = L_{\mathrm{train}}$ matches training conditions, and $L_{\mathrm{test}} > L_{\mathrm{train}}$ enables additional test-time scaling when the Q-function generalizes beyond the data support.

4.5 Didactic example: Why do we need test-time value guidance, not training time?

To illustrate the trade-off between training-time and test-time value incorporation, we design a two-agent coordination game (Figure 2 (*a*)). Each agent selects a scalar action $a_i \in [-2, 2]$, and the joint reward has two anti-correlated Gaussian modes: an optimal mode at $(-1, 1)$ with reward 4 and a sub-optimal mode at $(1, -1)$ with reward 2. The offline dataset consists of 400 transitions, with only 10% near each peak. Both methods share the same VDN critic $Q_{\mathrm{tot}} = Q_1(a_1) + Q_2(a_2)$.

The key difference is how each method incorporates the Q-function. MAC-Flow jointly optimizes distillation and Q-maximization, forcing the projected one-step policy to balance behavioral cloning and value improvement during training. In this toy example, this training-time compromise anchors each agent's output to the broad data distribution, while the decomposed Q-gradient alone cannot represent the anti-correlated structure, causing the policy to collapse near the origin (Figure 2 (*c*)). In contrast, SCOUT keeps the behavioral flow unchanged and applies value guidance only at test time. This preserves the multi-modal prior, and five refinement steps are sufficient to move candidates

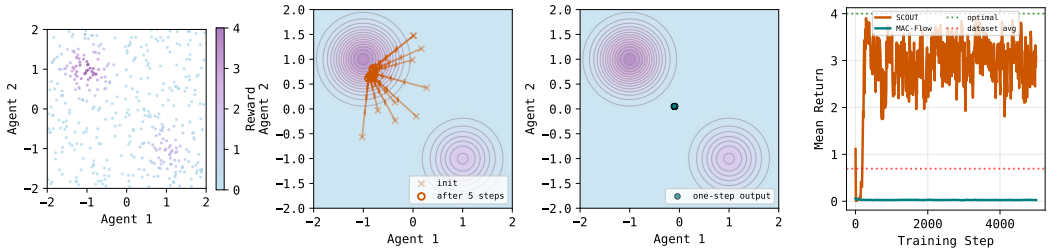


Figure 2: **Toy example.** (*a*) Offline dataset. (*b*) SCOUT's action refinement ($L_{\text{test}} = 5$). (*c*) MAC-Flow's one-step distilled policy. (*d*) Learning curve of two algorithms. We plot 10 evaluation examples.

Table 1: A performance comparison for SMACv1 is provided, covering 4 tasks, and 12 datasets. Results are averaged over 8 seeds, with two standard deviations reported after the ± sign. The superscript † denotes reported scores from [58].

Scenarios		Dataset	Gaussian policies			Diffusion policies		Flow policies		w/o policy
			BC†	MABCQ†	MACQL†	MADiff-D	DoF	Flow BC	MAC-Flow	SCOUT
SMACv1	3m	Good	16.0 ± 1.0	16.3 ± 1.5	19.6 ± 0.3	19.3 ± 0.5	19.4 ± 0.1	20.0 ± 0.0	19.8 ± 1.0	20.0 ± 0.0
		Medium	8.2 ± 0.8	18.3 ± 1.2	18.9 ± 1.2	16.4 ± 2.6	18.2 ± 0.2	14.7 ± 1.5	16.8 ± 1.3	17.8 ± 2.2
		Poor	4.4 ± 0.1	12.4 ± 2.3	5.8 ± 0.4	9.3 ± 2.1	8.9 ± 0.5	4.5 ± 0.1	6.8 ± 1.1	17.0 ± 3.6
	8m	Good	16.7 ± 0.4	12.7 ± 6.3	11.3 ± 6.1	18.9 ± 1.1	19.1 ± 0.1	19.5 ± 0.2	19.4 ± 1.3	20.0 ± 0.0
		Medium	10.7 ± 0.5	16.0 ± 1.4	16.8 ± 3.1	16.8 ± 1.6	17.5 ± 1.2	18.2 ± 0.8	19.1 ± 1.0	19.6 ± 0.4
		Poor	5.3 ± 0.1	5.8 ± 1.4	4.6 ± 2.4	5.8 ± 0.9	9.2 ± 0.3	4.9 ± 0.1	9.1 ± 1.2	8.9 ± 1.8
	2s3z	Good	18.2 ± 0.4	16.6 ± 1.2	19.0 ± 0.8	18.9 ± 1.2	18.2 ± 0.3	19.5 ± 0.1	19.5 ± 1.0	19.9 ± 0.1
		Medium	12.3 ± 0.7	13.6 ± 1.5	14.3 ± 2.0	17.5 ± 0.9	17.5 ± 1.1	15.1 ± 2.0	16.0 ± 1.2	17.5 ± 1.9
		Poor	6.7 ± 0.3	11.5 ± 1.0	10.1 ± 0.7	9.8 ± 1.3	9.8 ± 1.6	6.9 ± 0.8	8.4 ± 1.1	10.1 ± 1.1
	5m_vs_6m	Good	16.6 ± 0.6	8.3 ± 0.9	13.8 ± 3.9	16.5 ± 2.8	17.7 ± 1.3	14.7 ± 2.1	17.3 ± 1.0	16.8 ± 2.0
		Medium	12.4 ± 0.9	12.1 ± 1.3	16.9 ± 1.2	15.2 ± 2.6	16.2 ± 0.9	12.8 ± 0.8	16.6 ± 1.3	15.1 ± 1.7
		Poor	7.5 ± 0.2	11.0 ± 0.9	10.4 ± 1.0	8.9 ± 1.3	10.8 ± 0.3	7.7 ± 0.8	10.3 ± 1.2	10.5 ± 1.1
Average rewards			11.2	12.9	13.5	14.4	15.2	13.2	14.9	16.1

toward the optimal mode (Figure 2 (*b*)). In this setting, the selected per-agent actions align with the same anti-correlated mode because the additive utilities are trained from coordinated demonstrations.

5 Experiments

The subsection presents the experimental setup and answers the following research questions.

RQ1. How good is SCOUT over continuous and discrete control domains of MARL testbeds?

RQ2. How does SCOUT improve with additional interaction data?

RQ3. How sensitive is SCOUT to the refinement step parameter L_{train} and L_{test}?

5.1 Experimental setup

We evaluate SCOUT on two cooperative MARL testbeds with discrete and continuous action spaces, spanning 7 scenarios and 24 offline datasets. We first introduce the selected testbeds and baselines.

SMACv1. Agents control a team of units in StarCraft II micromanagement tasks against a built-in AI opponent. We test on four maps, *including* 3m, 8m, 2s3z, and 5m_vs_6m. Each map is paired with three offline dataset quality levels (Good, Medium, Poor) from the off-the-grid benchmark [58], resulting in 12 configurations. The quality levels differ in the proportion of near-optimal trajectories, with Poor containing mostly suboptimal behavior.

MA-MuJoCo. In this testbed, a single robot is factorized into multiple agents, with each agent responsible for a specific subset of the robot's joints. This setup requires agents to learn coordinated behaviors to accomplish shared tasks. We source the corresponding offline datasets from [6].

Baseline. We organize baselines into three families by their policy type. For the Gaussian policy, BC, MATD3BC [16] , MABCQ [11], MACQL [12], and ICQ [4] extend single-agent methods to the multi-agent setting. For the diffusion policy, MADiff [7] models the joint policy as a diffusion process over the full

Table 2: Performance evaluation for MA-MuJoCo. We present a performance comparison across 3 tasks and 12 datasets. The superscript † denotes reported scores from [10].

	Scenarios	Dataset	Offline MARL				Generative modeling		
			MATD3BC†	MACQL†	ICQ†	OMIGA†	MADiff-D	MAC-Flow	SCOUT
MA-MuJoCo	Ant	E	93.8 ± 1.4	95.8± 1.0	93.9 ± 0.9	94.2 ± 0.1	94.5 ± 0.8	94.5 ± 0.9	95.8 ± 1.0
		M	42.4 ± 1.0	11.2 ± 1.1	41.8 ± 1.2	42.1 ± 0.4	43.0± 0.9	43.4 ± 0.7	43.6 ± 0.7
		M-R	8.0 ± 4.2	-36.8 ± 8.7	9.8 ± 4.3	17.1 ± 7.2	31.6± 29.1	43.7 ± 1.4	42.5 ± 4.6
		M-E	59.2 ± 6.1	73.5 ± 0.9	55.5 ± 6.8	66.0 ± 8.9	67.7 ± 12.7	93.3 ± 0.6	94.7 ± 0.3
	Hopper	E	87.8 ± 0.1	88.9 ± 13.6	18.5 ± 19.9	21.3 ± 17.3	74.6 ± 14.3	95.2 ± 0.2	90.2 ± 0.4
		M	21.7 ± 3.8	22.5 ± 4.9	11.7 ± 0.3	30.3 ± 13.2	37.0 ± 10.9	25.8 ± 6.1	67.6 ± 8.8
		M-R	5.4 ± 1.0	-1.1 ± 0.4	3.4 ± 2.5	18.8 ± 12.0	23.4 ± 13.9	29.6 ± 10.9	53.0 ± 9.2
		M-E	76.6 ± 11.6	72.7 ± 3.0	7.7 ± 9.1	17.3 ± 14.5	74.0 ± 17.5	73.0 ± 11.7	75.5 ± 10.7
	HalfCheetah	E	111.3 ± 4.1	116.1 ± 2.4	76.6 ± 11.2	87.5 ± 13.5	120.5± 5.2	113.3 ± 6.6	111.6 ± 7.9
		M	68.3 ± 1.7	82.3 ± 7.5	66.6 ± 2.3	93.0 ± 5.8	69.1 ± 8.9	100.5 ± 9.0	88.9 ± 10.3
		M-R	90.7 ± 2.9	90.1 ± 7.2	51.5 ± 14.9	65.5 ± 2.0	73.3 ± 7.1	78.8 ± 10.6	95.1 ± 6.6
		M-E	90.4 ± 9.3	120.0 ± 4.4	74.2 ± 10.2	76.9 ± 12.6	113.6 ± 20.4	125.4 ± 2.0	126.9 ± 1.3
Average normalized score			62.97	61.27	42.60	52.50	68.52	76.37	82.12

Dataset abbreviations denote the data quality: `E (Expert)`, `M (Medium)`, `M-R (Medium-Replay)`, and `M-E (Medium-Expert)`.

action space, but we follow a distributed version (`MADiff-D`) that masks other agents' information. `DoF` [8] factorizes the diffusion into per-agent components to satisfy decentralized execution. For the flow policy, `Flow BC` trains a per-agent flow-matching behavioral prior, and `MAC-Flow` [10] learns a joint flow and distills it into one-step projected policies with Q-maximization.

5.2 Experimental results

RQ1 (Offline RL). Tables 1 and 2 report offline RL performance on SMACv1 and MA-MuJoCo. `SCOUT` achieves the highest average score on both benchmarks, outperforming all baselines across policy classes. The gain is consistent, *i.e.*, `SCOUT` matches or exceeds the best baseline on the majority of individual configurations. Notably, the advantage over `MAC-Flow`, which shares the same flow-matching backbone, isolates the benefit of test-time value transport over training-time distillation.

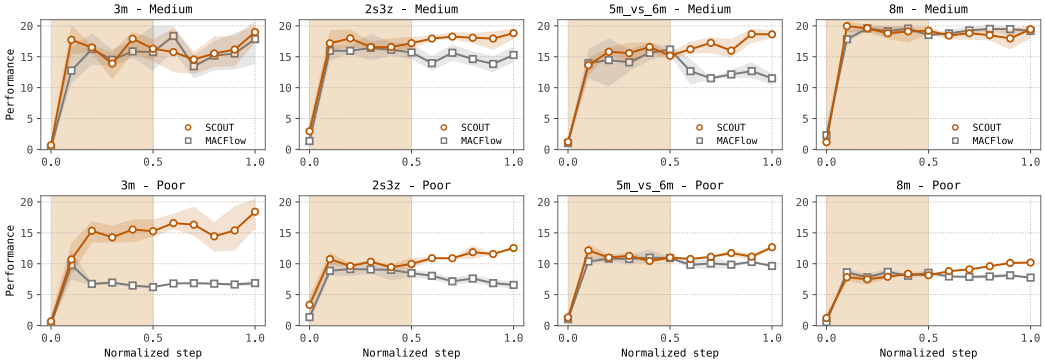


Figure 3: **Offline-to-online RL (8 seeds).** Fine-tuning starts at normalized gradient steps 0.5.

RQ2 (Offline-to-online RL). Figure 3 evaluates how well each method transitions from offline pre-training to online fine-tuning. `SCOUT` outperforms `MAC-Flow` across all four scenarios. After the online transition, `SCOUT` continues to improve steadily. Because the behavioral prior is never deformed by a training-time value loss, it provides a stable foundation that online fine-tuning can build upon rather than correct. The advantage is most pronounced on Poor-quality datasets. Here, `MAC-Flow`'s offline policy is anchored to low-quality demonstrations by its distillation objective, and online fine-tuning struggles to overcome this initialization. `SCOUT`, by contrast, treats the offline prior as a sampling distribution rather than a fixed policy, so the online phase simply adjusts the transport budget to exploit newly available data without inheriting the limitations of the offline dataset.

RQ3 (Hyperparameter sensitivity). Figure 4 varies the refinement steps for training and test across four SMACv1-Medium scenarios. First, applying even a small number of refinement steps at test time improves over the no-refinement baseline ($L_{\text{test}} = 0$), confirming that decomposed value gradient transport adds meaningful coordination beyond best-of-P selection alone. Second, the optimal L_{test} varies across scenarios. This is expected because each scenario and dataset quality presents a different Q-function landscape; the number of steps needed to reach high-value regions depends on how far the behavioral prior is from those regions and how smooth the Q-surface is. In practice, L_{test} can be

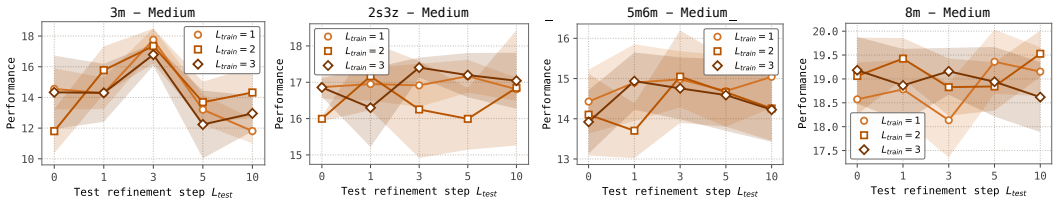


Figure 4: **Hyperparameter sweep.** The plots show how different L affect offline RL performance.

selected via a small validation sweep at deployment time without retraining, which is precisely the adaptivity that training-time methods cannot offer.

6 Conclusion

In this paper, we propose SCOUT, the first test-time policy refinement framework for offline MARL that decouples behavioral modeling from value maximization. While prior generative policies often compromise expressiveness by integrating value guidance during training, SCOUT preserves the multi-modal behavioral prior and defers value-driven refinement strictly to test-time inference via decomposed Stein variational gradient transport. We provide a single-term KL bound on the joint soft-value gap under approximate IGM and constructively explain how additive TD targets transfer inter-agent dependencies into per-agent gradients.

Limitation. While decoupling the behavioral prior from value maximization is effective across a range of tasks, the refinement hyperparameters exhibit some task- and dataset-dependent sensitivity. Therefore, a principled adaptive stopping criterion remains an open direction for future work.

Acknowledgements

This work was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. NSF 2340651, NSF 2402650, and ARO W911NF-24-1-0193. This research used the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing computational resources.

References

[1] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

[2] Dai Tamagawa, Eiichi Taniguchi, and Tadashi Yamada. Evaluating city logistics measures using a multi-agent model. *Procedia-Social and Behavioral Sciences*, 2(3):6002–6012, 2010.

[3] Jun Ota. Multi-agent robot systems as distributed autonomous systems. *Advanced engineering informatics*, 20(1):59–70, 2006.

[4] Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. In *NeurIPS*, 2021.

[5] Ling Pan, Longbo Huang, Tengyu Ma, and Huazhe Xu. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. In *ICML*, 2022.

[6] Xiangsen Wang, Haoran Xu, Yinan Zheng, and Xianyuan Zhan. Offline multi-agent reinforcement learning with implicit global-to-local value regularization. In *NeurIPS*, 2023.

[7] Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. MADiff: Offline multi-agent learning with diffusion models. In *NeurIPS*, 2024.

[8] Chao Li, Ziwei Deng, Chenxing Lin, Wenqi Chen, Yongquan Fu, Weiquan Liu, Chenglu Wen, Cheng Wang, and Siqi Shen. DoF: A diffusion factorization framework for offline multi-agent reinforcement learning. In *ICLR*, 2025.

[9] Zhuoran Li, Xun Wang, Hai Zhong, and Longbo Huang. OM2P: Offline multi-agent mean-flow policy. In *AAMAS*, 2026.

[10] Dongsu Lee, Daehee Lee, and Amy Zhang. Multi-agent coordination via flow matching. In *ICLR*, 2026.

[11] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *ICML*, 2019.

[12] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *NeurIPS*, 2020.

[13] Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline RL? In *NeurIPS*, 2024.

[14] Haoran Xu, Kaiwen Hu, Somayeh Sojoudi, and Amy Zhang. Reinforcement learning via value gradient flow. In *ICLR*, 2026.

[15] Daehee Lee, Dongsu Lee, TaeYoon Kwack, Wonje Choi, and Honguk Woo. Policy compatible skill incremental learning via lazy learning interface. In *NeurIPS*, 2025.

[16] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *NeurIPS*, 2021.

[17] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *ICLR*, 2022.

[18] Zongkai Liu, Qian Lin, Chao Yu, Xiawei Wu, Yile Liang, Donghui Li, and Xuetao Ding. Offline multi-agent reinforcement learning via in-sample sequential policy optimization. In *AAAI*, 2025.

[19] Paul Barde, Jakob Foerster, Derek Nowrouzezahrai, and Amy Zhang. A model-based solution to the offline multi-agent reinforcement learning coordination problem. In *AAMAS*, 2024.

[20] Linghui Meng, Muning Wen, Chenyang Le, Xiyun Li, Dengpeng Xing, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, Yaodong Yang, et al. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research*, 20(2):233–248, 2023.

[21] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *ICLR*, 2023.

[22] Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *ICML*, 2025.

[23] Sungyoung Lee, Dohyeong Kim, Eshan Balachandar, Zelal Su Mustafaoglu, and Keshav Pingali. Towards efficient and expressive offline RL via flow-anchored noise-conditioned Q-learning. In *ICML*, 2026.

[24] Allen Z Ren, Justin Lidard, Lars L Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *ICLR*, 2025.

[25] Max Sobol Mark, Tian Gao, Georgia Gabriela Sampaio, Mohan Kumar Srirama, Archit Sharma, Chelsea Finn, and Aviral Kumar. Policy agnostic RL: Offline RL and online RL fine-tuning of any class and backbone. *arXiv preprint arXiv:2412.06685*, 2024.

[26] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *NeurIPS*, 2016.

[27] Qiang Liu. Stein variational gradient descent as gradient flow. In *NeurIPS*, 2017.

[28] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. IDQL: Implicit Q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[29] Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *NeurIPS*, 2025.

[30] Qiyang Li and Sergey Levine. Q-learning with adjoint matching. In *ICLR*, 2026.

[31] Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. In *ICLR*, 2021.

[32] Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *CoRL*, 2025.

[33] Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *AAAI*, 2018.

[34] Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal Wasserstein imitation learning. In *ICLR*, 2021.

[35] Arip Asadulaev, Rostislav Korst, Aleksandr Korotin, Vage Egiazarian, Andrey Filchenkov, and Evgeny Burnaev. Rethinking optimal transport in offline reinforcement learning. In *NeurIPS*, 2024.

[36] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research*, 27(4):819–840, 2002.

[37] Yueheng Li, Guangming Xie, and Zongqing Lu. Revisiting cooperative off-policy multi-agent reinforcement learning. In *ICML*, 2025.

[38] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[39] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *ICML*, 2019.

[40] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

[41] Dongsu Lee, Daehee Lee, and Amy Zhang. A recipe for stable offline multi-agent reinforcement learning. *arXiv preprint arXiv:2603.08399*, 2026.

[42] Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. In *NeurIPS*, 2023.

[43] Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *ICLR*, 2024.

[44] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[45] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

[46] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[47] Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline RL with no OOD actions: In-sample learning via implicit value regularization. In *ICLR*, 2023.

[48] Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. The invisible leash: Why RLVR may or may not escape its origin. *arXiv preprint arXiv:2507.14843*, 2025.

[49] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. EMaQ: Expected-max Q-learning operator for simple yet effective offline and online RL. In *ICML*, 2021.

[50] Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. Theoretical guarantees on the best-of-n alignment policy. In *ICML*, 2025.

[51] Joy Qiping Yang, Salman Salamatian, Ziteng Sun, Ananda Theertha Suresh, and Ahmad Beirami. Asymptotics of language model alignment. In *ISIT*, 2024.

[52] David Huk, Oliver Hamelijnck, Dimitris Demiris, and Theodoros Damoulas. MAGIC: Multi-agent generative intention coordination. In *ICLR Workshop*, 2026.

[53] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[54] Jonas Wildberger, Maximilian Dax, Simon Buchholz, Stephen Green, Jakob H Macke, and Bernhard Schölkopf. Flow matching for scalable simulation-based inference. In *NeurIPS*, 2023.

[55] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

[56] Benjamin Eysenbach and Sergey Levine. Maximum entropy RL (provably) solves some robust RL problems. In *ICLR*, 2022.

[57] Michael Bloem and Nicholas Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. In *CDC*, 2014.

[58] Claude Formanek, Asad Jeewa, Jonathan Shock, and Arnu Pretorius. Off-the-grid MARL: Datasets and baselines for offline multi-agent reinforcement learning. In *AAMAS*, 2023.

Appendix

Contents

A Proofs **15**

A.1 Proof of Theorem 4.2 15

A.2 Proof of Proposition 4.3 15

B Experimental Details **16**

B.1 Computational resources 16

B.2 Implementation details 16

B.3 Hyperparameters 16

B.4 Baselines 16

C Additional Results **17**

A Proofs

In this section, we provide complete proofs for the theoretical results in Section 4.3. Throughout, we use the following standing notation. Let $\mathcal{O} = \prod_{i=1}^{I} \mathcal{O}_i$ and $\mathcal{A} = \prod_{i=1}^{I} \mathcal{A}_i$ denote the joint observation and action spaces. Let $\mu_i(\cdot \mid o_i)$ denote the per-agent behavioral prior, and let $\pi_i^*(a_i \mid o_i) \propto \exp(Q_i(o_i, a_i)/\alpha)$ denote the per-agent factorized Boltzmann target. The joint Boltzmann target is defined as follows:

$$\pi^*_{\text{joint}}(a \mid o) \propto \exp\left(\frac{Q_{\text{tot}}(o, a)}{\alpha}\right).$$

We denote by $\tilde{\pi}^L = \prod_{i=1}^{I} \tilde{\pi}_i^L$ the joint policy obtained after L steps of independent per-agent transport in (8).

A.1 Proof of Theorem 4.2

Proof. Let $\pi^*_{\text{fact}} := \prod_i \pi_i^*$ and $\mathcal{J}_{\text{add}}(\pi) := \mathbb{E}_\pi[\sum_i Q_i] + \alpha \mathcal{H}(\pi)$. The product Boltzmann distribution π^*_{fact} is the unique maximizer of $\mathcal{J}_{\text{add}}$ on the joint simplex (it is the Gibbs distribution for the additive utility), and the standard variational identity yields the gap-as-KL relation:

$$\mathcal{J}_{\text{add}}(\pi^*_{\text{fact}}) - \mathcal{J}_{\text{add}}(\pi) \;=\; \alpha\, \mathrm{KL}\big(\pi \,\|\, \pi^*_{\text{fact}}\big) \quad \text{for any joint } \pi. \tag{10}$$

Under Assumption 4.1, $|\mathcal{J}_{\text{soft}}(\pi) - \mathcal{J}_{\text{add}}(\pi)| = |\mathbb{E}_\pi[\Delta]| \le \delta$ for any π.

Combining (10) with the δ-shift between $\mathcal{J}_{\text{soft}}$ and $\mathcal{J}_{\text{add}}$:

$$\begin{aligned}\mathcal{J}_{\text{soft}}(\pi^*_{\text{fact}}) - \mathcal{J}_{\text{soft}}(\tilde{\pi}^L) &\le \big[\mathcal{J}_{\text{add}}(\pi^*_{\text{fact}}) + \delta\big] - \big[\mathcal{J}_{\text{add}}(\tilde{\pi}^L) - \delta\big] \\ &= \alpha\, \mathrm{KL}\big(\tilde{\pi}^L \,\|\, \pi^*_{\text{fact}}\big) + 2\delta. \end{aligned} \tag{11}$$

Since π^*_{joint} maximizes $\mathcal{J}_{\text{soft}}$, $\mathcal{J}_{\text{soft}}(\pi^*_{\text{joint}}) - \mathcal{J}_{\text{soft}}(\pi^*_{\text{fact}}) \ge 0$. For the upper bound, since π^*_{fact} maximizes $\mathcal{J}_{\text{add}}$, $\mathcal{J}_{\text{add}}(\pi^*_{\text{fact}}) \ge \mathcal{J}_{\text{add}}(\pi^*_{\text{joint}})$, so

$$\mathcal{J}_{\text{soft}}(\pi^*_{\text{joint}}) - \mathcal{J}_{\text{soft}}(\pi^*_{\text{fact}}) \le \big[\mathcal{J}_{\text{add}}(\pi^*_{\text{joint}}) + \delta\big] - \big[\mathcal{J}_{\text{add}}(\pi^*_{\text{fact}}) - \delta\big] \le 2\delta. \tag{12}$$

Adding (11) and (12):

$$\mathcal{J}_{\text{soft}}(\pi^*_{\text{joint}}) - \mathcal{J}_{\text{soft}}(\tilde{\pi}^L) \;\le\; \alpha\, \mathrm{KL}\big(\tilde{\pi}^L \,\|\, \pi^*_{\text{fact}}\big) + 4\delta.$$

The lower bound 0 follows directly from the $\mathcal{J}_{\text{soft}}$-optimality of π^*_{joint}. □

A.2 Proof of Proposition 4.3

Proof. Mathematical derivations are divided into two parts of Proposition 4.3. For ***(i)***, the per-agent transport for agent i as follows:

$$a_{i,j}^{(\ell+1)} = a_{i,j}^{(\ell)} + \epsilon\, \mathrm{clip}_c\big(\hat{\phi}_i^*\big(a_{i,j}^{(\ell)}; \{a_{i,k}^{(\ell)}\}_{k=1}^{P}, Q_i(o_i, \cdot)\big)\big),$$

depends only on agent-i particles and Q_i. Initialization is independent across agents and no cross-agent information enters any update, so independence is preserved:

$$\big(a_{i,j}^{(L)}\big)_{j=1}^{P} \perp\!\!\!\perp \big(a_{i',j}^{(L)}\big)_{j=1}^{P} \quad \text{for } i \ne i'. \tag{13}$$

Per-agent selection $j_i^* = \arg\max_j Q_i(o_i, a_{i,j}^{(L)})$ depends only on agent-i particles, so $a_i^* = a_{i,j_i^*}^{(L)}$ are functions of independent sets, giving $\mathrm{Cov}(a_i^*, a_{i'}^*) = 0$. When $\Delta = 0$, $Q_{\text{tot}} = \sum_i Q_i$, and per-agent argmax is jointly optimal by the IGM identity (1).

Next, for ***(ii)***, let $j^* \in [P]$ be any joint candidate index. By per-agent optimality, $\sum_i Q_i(o_i, a_{i,j_i^*}^{(L)}) \ge \sum_i Q_i(o_i, a_{i,j^*}^{(L)})$. Using $Q_{\text{tot}} = \sum_i Q_i + \Delta$ and $\|\Delta\|_\infty \le \delta$:

$$\begin{aligned} Q_{\text{tot}}\big(a_{1,j_1^*}^{(L)}, \ldots, a_{I,j_I^*}^{(L)}\big) &= \textstyle\sum_i Q_i\big(a_{i,j_i^*}^{(L)}\big) + \Delta\big(a_{1,j_1^*}^{(L)}, \ldots, a_{I,j_I^*}^{(L)}\big) \\ &\ge \textstyle\sum_i Q_i\big(a_{i,j^*}^{(L)}\big) - \delta \\ &\ge Q_{\text{tot}}\big(a_{1,j^*}^{(L)}, \ldots, a_{I,j^*}^{(L)}\big) - 2\delta. \end{aligned} \tag{14}$$

Maximizing over $j^* \in [P]$ on the right-hand side yields the claim. □

B Experimental Details

B.1 Computational resources

All experiments were conducted on a single server equipped with an `Intel Xeon E5-2650 v3` CPU and an `NVIDIA A40` GPU. The software stack consists of `CUDA 12.2, cuDNN 8.9.7, Python 3.9, and JAX 0.4.30.`

B.2 Implementation details

Flow BC prior. We instantiate the foundation as a conditional flow-matching model that learns the behavior distribution of the offline dataset. Given an observation and a Gaussian noise sample, the actor parameterizes a velocity field that transports x_0 to a data action x_1 along the linear interpolation, trained by minimizing the flow-matching loss. At inference time, candidate actions are produced by a fixed-step Euler integration of the learned ODE, so the prior remains an explicit, multimodal generative model of the dataset rather than a unimodal collapse. Because the resulting samples are biased toward the modeled behavioral support, they encourage the agent to remain near in-distribution behaviors, while still preserving the diverse coordination patterns present in the offline corpus.

Sinusoidal time embedding. The flow conditioning variable $t \in [0, 1]$ carries information about how close the current sample x_t is to data; feeding it as a raw scalar concatenated to the network input yields poor expressivity, since a single-channel signal interacts weakly with high-dimensional weights and forces the actor to learn very different behaviors at neighboring t values from a one-dimensional cue. We therefore lift t into a Fourier feature space $\varphi(t) = \left[\sin(2^k \pi t), \cos(2^k \pi t)\right]_{k=0}^{K-1} \in \mathbb{R}^{2K}$ with $K = 8$, echoing the positional encodings widely used in diffusion and neural-field models. The exponentially spaced frequencies provide a multi-resolution decomposition of the time axis: low-frequency components encode the coarse phase of the integration trajectory while high-frequency components let the velocity field model sharp transitions near $t = 0$ and $t = 1$.

Online fine-tuning. After offline gradient steps, we roll out the policy and store the resulting trajectories in an online replay buffer. After an additional warm-up step with no online updates, each gradient step samples a $50 : 50$ mixture of offline data and newly collected online data.

B.3 Hyperparameters

We select L_{train} and L_{test} from a small grid using a held-out validation split of the offline dataset; no environment interaction is used for selection. Baselines use their officially reported hyperparameters with comparable tuning budgets.

Table 3: Hyperparameter for RL optimization

Hyperparameter	Default
`Flow prior network`	$(256, 256, 256, 256)$
`Value prior network`	$(256, 256, 256, 256)$
`Layer normalization`	`value`
`Learning rate`	0.0003
`Q value Aggregation`	`mean`
`Flow steps`	10
t `embedding frequency`	8
τ	0.005
γ	0.99

Table 4: Hyperparameter for `SCOUT`

Hyperparameter	Default
ϵ `refinement step size`	0.02
P `# of particles`	10
L_{train}	0
L_{test}	0
`Particle selection (train)`	`mean`
`Particle selection (test)`	`max`
α `temperature`	1.0

B.4 Baselines

`MADiff-D` [7]. We utilize the official open-source implementation, specifically adopting the decentralized variant, `MADiff-D`, to align with our Dec-POMDP formulation. The model trains a conditional diffusion policy on centralized joint demonstrations but executes decentrally; each agent samples actions via reverse diffusion based solely on local observations while predicting teammates' trajectories. We retain the original architecture and hyperparameters, applying a Gaussian DDPM scheduler with 200 denoising steps across all environments.

Table 5: Hyperparameter selection of SCOUT for offline RL configuration

Configuration	$L_{\texttt{train}}$	$L_{\texttt{test}}$	Configuration	$L_{\texttt{train}}$	$L_{\texttt{test}}$
3m - Good	1	1	2s3z - Good	1	0
3m - Medium	1	3	2s3z - Medium	2	1
3m - Poor	2	3	2s3z - Poor	3	10
8m - Good	3	0	5m_vs_6m - Good	3	3
8m - Medium	2	10	5m_vs_6m - Medium	2	3
8m - Poor	3	10	5m_vs_6m - Poor	1	3
ant - Expert	2	0	ant - Medium-Expert	0	0
ant - Medium	2	0	ant - Medium-Replay	0	0
hopper - Expert	2	10	hopper - Medium-Expert	2	3
hopper - Medium	2	3	hopper - Medium-Replay	3	0
halfcheetah - Expert	2	3	halfcheetah - Medium-Expert	3	0
halfcheetah - Medium	0	1	halfcheetah - Medium-Replay	3	1

DoF [8]. We use the official implementation of the DoF-Trajectory agent equipped with the W-concat factorization. All training and deployment configurations strictly follow the original paper. For action sampling, we similarly employ a Gaussian DDPM scheduler with 200 denoising steps for all environments.

MAC-Flow [10]. We utilize the official open-source implementation. MAC-Flow models multi-modal data distributions via flow matching, utilizes a learned Q-function for guidance, and distills the guided flow into one-step decentralized policies. We maintain the original architecture and hyperparameters for all experiments.

C Additional Results

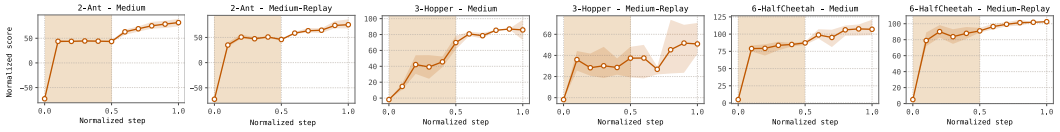


Figure 5: **Offline-to-online RL in MA-MuJoCo (8 seeds.)**

Offline-to-online RL. Figure 5 extends the offline-to-online evaluation to the continuous control domain. Across all six configurations, SCOUT demonstrates consistent improvement after the online transition at normalized step 0.5. These empirical results suggest that the undeformed flow prior allows the transport budget to flexibly exploit newly collected interaction data. Together with the SMACv1 results in Figure 3, SCOUT achieves positive online improvement in all offline-to-online RL configurations across both benchmarks, confirming that the test-time transport paradigm generalizes across discrete and continuous action space types.

Hyperparameter sweep. Figure 6 reports the sensitivity of SCOUT to the refinement steps L_{train} and L_{test} across all four dataset qualities in MA-MuJoCo. In the Expert and Medium-Expert datasets, performance is largely insensitive to the choice of L_{test}, which is expected because the behavioral prior already closely matches high-value regions, leaving little room for gradient transport to improve upon. In contrast, the Medium and Medium-Replay datasets exhibit a clearer benefit from moderate refinement steps, where increasing L_{test} from 0 to 3 yields substantial gains. However, excessive refinement $L_{\text{test}} = 10$ can degrade performance in some cases, suggesting that the critic's accuracy degrades when actions are transported too far from the behavioral support.

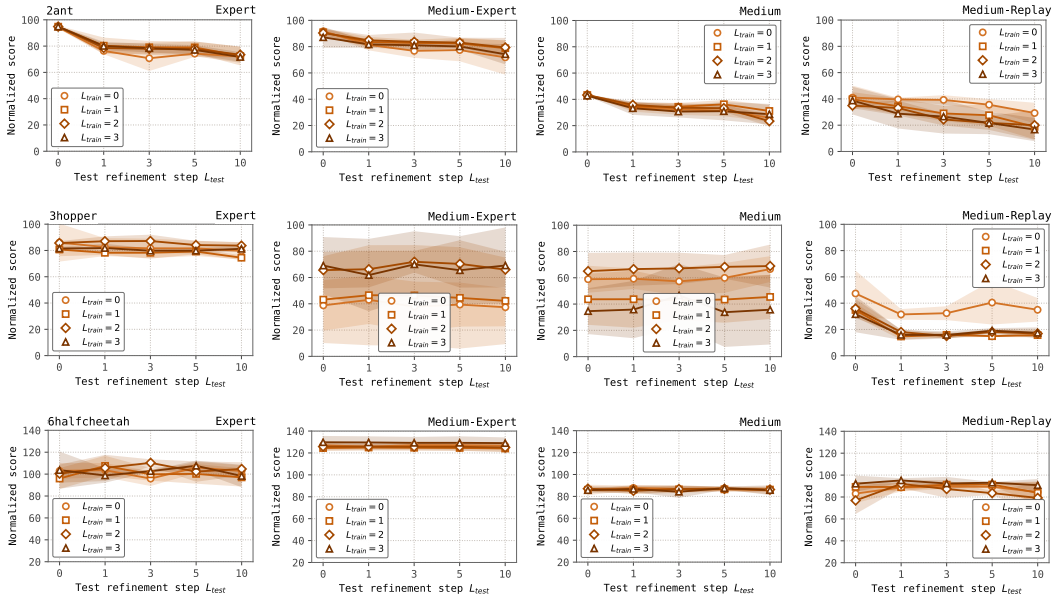


Figure 6: **Hyperparameter sweep on MA-MuJoCo.**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state three contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated limitation discussion in the Conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper states Assumption 4.1 explicitly and provides three theoretical results (Theorem 4.2, Proposition 4.3) with complete proofs in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides detailed algorithm pseudocode (Algorithm 1), complete hyperparameter tables, implementation details, and baseline descriptions in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides an anonymous code repository link in the Abstract. The offline datasets used from the off-the-grid dataset [58] are publicly available benchmarks.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

Justification: Section 5 describes the experimental setup, *including* testbeds, datasets, and baselines. Appendix B provides comprehensive implementation details, default and per-configuration hyperparameters, and baseline configurations.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All results in Tables 1 and 2 report mean $\pm$ two standard deviations over 8 random seeds.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the type of GPU/CPU hardware used for the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The work conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This work is foundational research on offline MARL. It does not have a direct path to negative societal applications.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not pose risks for misuse comparable to pretrained language models or image generators.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the original papers for all benchmarks and datasets used.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper releases code via an anonymous repository linked in the abstract.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.