CODA: Coordination via On-Policy Diffusion for Multi-Agent Offline Reinforcement Learning

Marcel Hedman [1 2] **Kale-ab Abebe Tessera** [3] **Juan Claude Formanek** [4] **Anya Sims** [2] **Riccardo Zamboni** [5]
Trevor McInroe [3] **John Torr** [1] **Elliot Fosong** [1]

Abstract

Offline multi-agent reinforcement learning (MARL) enables policy learning from fixed datasets, but is prone to **coordination failure**: agents trained on static, off-policy data converge to suboptimal joint behaviours because they cannot co-adapt as their policies change. We introduce **CODA** (***C**oordination via **O**n-Policy **D**iffusion for Multi-**A**gent Reinforcement Learning*), a diffusion-based multi-agent trajectory generator for data augmentation that samples conditioned on the current joint policy, producing synthetic experience which better reflects the evolving behaviours of the agents, thereby providing a mechanism for co-adaptation. We find that previous diffusion-based augmentation approaches are insufficient for fostering multi-agent coordination because they produce static augmented datasets that do not evolve as the current joint policy changes during training; CODA tackles this by more closely simulating on-policy learning and is a meaningful step toward coordinated behaviours in the offline setting. CODA is algorithm-agnostic and can be layered onto both model-free and model-based offline reinforcement learning pipelines as an augmentation module. Empirically, CODA resolves canonical coordination pathologies in continuous polynomial games.

1. Introduction

Multi-agent reinforcement learning (MARL) has emerged as a key framework for training intelligent agents that must operate in shared environments (Albrecht et al., 2024; Lowe et al., 2017; Peng et al., 2021). From coordinated robot teams (Matignon et al., 2012) to large-scale network control and economic resource allocation (Mi et al., 2024), MARL has been used in applications in which learning optimal *joint* behaviour is critical.

While online MARL remains the dominant research paradigm, it requires interactive exploration which may be unsafe, costly, or infeasible in real-world systems. Offline reinforcement learning (RL), instead learns from fixed datasets, promising improved safety and sample efficiency (Levine et al., 2020). However, shifting from single-agent to cooperative multi-agent settings introduces additional complexities that compound the standard offline issues of distributional shift and extrapolation error (Fu et al., 2020). Specifically, it requires resolving per-agent credit assignment (Albrecht et al., 2024) and coordination demands absent in single-agent environments (Tilbury et al., 2024).

In this work, we focus on tackling **offline coordination failure** in cooperative tasks: where distinct agents converge to incompatible behaviours despite individually taking optimal actions under the pre-collected dataset (Barde et al., 2024; Tilbury et al., 2024). During online training, an agent's local optimisation not only shapes its own behaviour, but also alters the experiences of its teammates, who can in turn adapt. By contrast, in *offline* MARL, the dataset is static, so cross-agent co-adaptation cannot occur even with sufficient marginal coverage of the joint state-action space in the dataset.

A standard approach to mitigating coordination challenges online is Centralised Training with Decentralised Execution (CTDE), which uses global information during training while keeping decentralised policies at test time. CTDE underpins successful online MARL algorithms, including MAPPO (Yu et al., 2022), MADDPG (Lowe et al., 2017) and MASAC (Haarnoja et al., 2018). However, recent work (Barde et al., 2024; Tilbury et al., 2024) shows that naively porting these model-free CTDE methods to offline MARL does not prevent coordination failures. In particular, so-called Best Response under Data (BRUD) algorithms (Tilbury et al., 2024), which include MADDPG and

[1]Recurvia Ltd [2]Department of Statistics, University of Oxford [3]University of Edinburgh [4]University of Cape Town, South Africa [5]Politecnico di Milano, Milan, Italy. Correspondence to: Marcel Hedman <marcel.hedman@jesus.ox.ac.uk>.

Accepted at ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning. Copyright 2026 by the author(s).

MASAC, independently optimise each agent against the fixed dataset, and can actively induce miscoordination even with centralised critics. The issue is not missing centralised information but missing *co-adaptation*. Restoring this adaptation signal, without additional environment interaction, is the challenge we address.

We propose **CODA (Coordination via On-Policy Diffusion for Multi-Agent Offline Reinforcement Learning)**, a trajectory-level augmentation method for offline MARL. We argue that offline coordination failure arises fundamentally from the absence of *endogenous joint policy evolution* with respect to the dataset. In online MARL, each agent's update alters its teammates' effective learning environment, producing a feedback-driven drift in the joint trajectory distribution.

CODA approximates this missing co-adaptation in the offline setting by learning a diffusion model over joint trajectories and using *on-policy guidance at sampling time* to condition synthetic rollouts on the current joint policy. Unlike approaches that rely on Q-values or behavioural statistics (Tilbury et al., 2024; Oh et al., 2024), CODA produces policy-dependent joint trajectories that track the behaviours agents are currently learning, while remaining strictly offline.

As a data augmentation method, CODA is agnostic to the MARL algorithm used, so it can be layered onto existing model-free or model-based offline MARL approaches. By uniting trajectory-level diffusion with on-policy conditioning, CODA provides a step toward resolving coordination failures in offline MARL. Empirically, we demonstrate that CODA resolves explicit coordination pathologies in continuous polynomial games and we also test performance on complex continuous control benchmarks.

2. Background

2.1. Multi-Agent Reinforcement Learning

We consider a fully cooperative multi-agent reinforcement learning (MARL) setting modelled as a Decentralised Partially Observable Markov Decision Process (Dec-POMDP, Oliehoek & Amato, 2016). A Dec-POMDP is defined by the tuple, $G = \langle I, S, \{A^i\}, T, R, \{O^i\}, \Omega, \gamma \rangle$, with agents $I = \{1, \ldots, N\}$. At time t, agent i observes $o_t^i \in O^i$ and takes actions $a_t^i \in A^i$. Let $\mathbf{o}_t = (o_t^i)_{i \in I} \in \mathbf{O} := \prod_i O^i$ and $\mathbf{a}_t = (a_t^i)_{i \in I} \in \mathbf{A} := \prod_i A^i$ denote the joint quantities. Dynamics follow $T(s_{t+1} \mid s_t, \mathbf{a}_t)$, observations $\Omega(\mathbf{o}_t \mid s_t)$, and shared team reward $r_t = R(s_t, \mathbf{a}_t)$. Finally, decentralised policies factorise as $\boldsymbol{\pi}(\mathbf{a}_t \mid \mathbf{o}_t) = \prod_i \pi^i(a_t^i \mid o_t^i)$. The induced distribution over finite-horizon trajectories $\tau = (s_0, \mathbf{o}_0, \mathbf{a}_0, r_0, \ldots, s_H)$ is

$$p_{\boldsymbol{\pi}}(\tau) = p_0(s_0) \prod_{t=0}^{H-1} \Omega(\mathbf{o}_t \mid s_t) \boldsymbol{\pi}(\mathbf{a}_t \mid \mathbf{o}_t) T(s_{t+1} \mid s_t, \mathbf{a}_t), \quad (1)$$

where $\boldsymbol{\pi}$ is the trainable joint policy (Levine, 2018; Jackson et al., 2024). We optimise $J(\boldsymbol{\pi}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\pi}}(\tau)}[\sum_{t=0}^{H-1} \gamma^t r_t]$ where γ is the discount factor.

Offline MARL. In the offline setting, learning is restricted to a fixed dataset $\mathcal{D}_{\text{off}} \sim p_{\boldsymbol{\pi}_{\text{off}}}(\tau)$, collected under an unknown behaviour policy $\boldsymbol{\pi}_{\text{off}}$. No additional environment interaction is permitted during policy learning. The objective is to learn a policy $\boldsymbol{\pi}^*$ that maximises the expected discounted return $J(\boldsymbol{\pi}^*)$. Throughout, we adopt the centralised training with decentralised execution (CTDE) learning paradigm, wherein centralised critics may condition on global state and joint actions during training, but at execution time each agent selects actions using only its local observation o^i.

Beyond standard offline single-agent RL challenges (e.g. partial observability or distributional shift between $\boldsymbol{\pi}_{\text{off}}$ and $\boldsymbol{\pi}_{\text{curr}}$), offline MARL introduces a structural difficulty: agent coordination must be inferred from static off-policy data, without interactive co-adaptation among agents.

2.2. Diffusion Models

In this work we seek to generate synthetic RL trajectories. We do so using diffusion models (DM) — a class of generative model noted for their ability to sample from complex multimodal distributions. DMs generate samples by reversing a gradual noising process that corrupts data through iterative Gaussian convolutions. Here, we leverage their ability to conduct trajectory-level modelling (Lu et al., 2023). Starting from clean data (trajectories) $\tau_0 \sim p(\tau)$, a forward process produces progressively noisier variables τ_x according to a predefined noise schedule, such that at sufficiently large noise levels $\sigma(x)$, the corrupted distribution $p(\tau; \sigma(x))$ approaches an isotropic Gaussian.

We adopt the probability flow ODE formulation of Karras et al. (2022), a deterministic special case of the general SDE-based framework. Given a continuous noise schedule $\sigma(x)$, the data evolves as:

$$d\tau = -\dot{\sigma}(x)\sigma(x)\nabla_\tau \log p(\tau; \sigma(x))\, \mathrm{d}x, \quad (2)$$

where $\nabla_\tau \log p(\tau; \sigma(x))$ is the noise-conditioned score function and $\dot{\sigma}(x) := \frac{d\sigma(x)}{dx}$ denotes the derivative of the noise schedule with respect to x. Learning reduces to estimating the score function with a neural network trained on trajectories from $\mathcal{D}_{\text{off}}$. Sampling is then performed by numerically integrating the ODE from Gaussian noise back to the data space.

2.3. Conditional Generation

As we will see in Section 4, ultimately our goal is to conduct conditional generation such that we are able to generate trajectories that are policy-conditioned. Two standard guidance mechanisms to conduct such conditioning are Classifier-Free Guidance and Classifier Guidance.

Classifier-Free Guidance (CFG). CFG trains a single model to predict both conditional and unconditional scores by randomly dropping the conditioning variable y during training (Ho & Salimans, 2022). At sampling time, conditional and unconditional scores are linearly combined:

$$\begin{aligned}\hat{\nabla}_\tau \log p(\tau;\sigma \mid y) \approx (1+w)\nabla_\tau \log p(\tau;\sigma \mid y) \\ - w\nabla_\tau \log p(\tau;\sigma),\end{aligned} \tag{3}$$

where $w \geq 0$ is a guidance scale controlling the strength of conditioning. Larger w increases adherence to the condition at the cost of reduced diversity.

Classifier Guidance. Alternatively, classifier guidance augments the unconditional score using gradients from a separately trained discriminative model $p(y \mid \tau;\sigma)$ (Dhariwal & Nichol, 2021).

$$\begin{aligned}\hat{\nabla}_\tau \log p(\tau;\sigma \mid y) \approx \nabla_\tau \log p(\tau;\sigma) \\ + \lambda\nabla_\tau \log p(y \mid \tau;\sigma),\end{aligned} \tag{4}$$

where λ is a guidance scale, $\nabla_\tau \log p(\tau;\sigma)$ is the standard unconditional diffusion score and $\nabla_\tau \log p(y \mid \tau;\sigma)$ is provided by a classifier (or more generally, any differentiable scoring model). In reinforcement learning contexts, such a classifier may correspond to value functions (Oh et al., 2024; Janner et al., 2022), policy likelihoods (Jackson et al., 2024), or other objectives defined over trajectories (He et al., 2023).

3. Offline Coordination Failure

Recent work has studied the challenge of coordination in cooperative offline MARL. Notably, Barde et al. (2024) decompose offline coordination into two components: *strategy agreement*, which entails selecting among multiple compatible optima in the joint policy space, and *strategy fine-tuning*, which refers to calibrating behaviours to realise the chosen strategy. Both components rely on agents adapting to one another during learning. This in turn requires that individual policy updates are conditioned on the *current joint policy*.

BRUD updates in offline MARL. Tilbury et al. (2024) formalised the prevailing approach to offline MARL as the Best Response Under Data (BRUD) paradigm. Multi-agent actor-critic methods have an objective of the form:

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\mathbf{a}\sim\boldsymbol{\pi}}\big[Q(\mathbf{o},\mathbf{a}) + \alpha\mathcal{R}(\boldsymbol{\pi})\big], \tag{5}$$

where Q is a centralised critic over joint observations and actions, and $\mathcal{R}$ is a policy regularisation term with strength parameter $\alpha \in \mathbb{R}$. This formulation directly captures off-policy CTDE actor-critic methods such as MASAC (Haarnoja et al., 2018) and MADDPG (Lowe et al., 2017).

Applying this objective in the offline setting requires constructing a joint action using both the current policy and the dataset. Following Tilbury et al. (2024), the updated agent samples its action from the current policy, while teammate actions are drawn directly from the dataset:

$$a^i \sim \pi^i_{\text{curr}}(\cdot \mid o^i), \qquad \mathbf{a}^{-i} \sim \mathcal{D}_{\text{off}}. \tag{6}$$

Ignoring the regularisation term, the resulting offline policy gradient for agent i, with parameters θ^i, under a deterministic actor $\boldsymbol{\mu}_\theta$ becomes:

$$\nabla_{\theta^i} J(\boldsymbol{\mu}_\theta) = \mathop{\mathbb{E}}_{(\mathbf{o},\mathbf{a})\sim\mathcal{D}_{\text{off}}}\big[\nabla_{\theta^i}\mu_{\theta^i}(o^i)\nabla_{\tilde{a}^i}Q\big(\mathbf{o},(\mu_{\theta^i}(o^i),\mathbf{a}^{-i})\big)\big]. \tag{7}$$

Thus, the update for agent i is a best response to dataset actions rather than to teammates' current behaviours.

Coordination challenges with BRUD updates. In online MARL, data is continually collected under the current joint policy $\boldsymbol{\pi}_{\text{curr}}$. Policy updates therefore change the distribution of future experience, enabling teammates to respond to one another's behavioural changes. This feedback-driven shift in the joint trajectory distribution across optimisation iterations constitutes *endogenous joint policy evolution*. Under idealised conditions (sufficient exploration, centralisation, realisable function approximation, and stable optimisation), this interaction loop allows agents to repeatedly adapt to one another, enabling both strategy agreement and fine-tuning (Barde et al., 2024).

By contrast, offline MARL breaks this interaction loop due to learning from a fixed dataset

$$\mathcal{D}_{\text{off}} \sim p_{\boldsymbol{\pi}_{\text{off}}}(\tau),$$

collected under a typically unknown behaviour policy $\boldsymbol{\pi}_{\text{off}}$. The sampled experience is therefore drawn from the fixed distribution $p_{\boldsymbol{\pi}_{\text{off}}}(\tau)$ even as the current policy $\boldsymbol{\pi}_{\text{curr}}$ changes during training.

Consequently, each agent updates using trajectories that reflect teammates' randomly sampled past behaviour rather than their current policies. Agents may therefore observe teammate actions that are inconsistent with the joint distribution induced by the evolving current policy, leading to incorrect adaptation and miscoordination.

This induces a structural distribution mismatch. Training implicitly optimises a surrogate objective under $p_{\boldsymbol{\pi}_{\text{off}}}(\tau)$,

whereas evaluation is under $p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$:

$$J(\boldsymbol{\pi}_{\text{curr}}) = \mathbb{E}_{\tau \sim p_{\boldsymbol{\pi}_{\text{curr}}}} \left[\sum_{t=0}^{H-1} \gamma^t r_t \right]. \tag{8}$$

Crucially, because $p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$ does not influence the sampled data distribution during training, the endogenous joint policy evolution present in online learning is absent in the offline setting.

Implications for coordination. Under BRUD updates, each agent effectively learns a best response to teammate actions recorded in the dataset. Since those actions are sampled from a potentially very different policy than the current policy, the agents' policies do not necessarily co-evolve during training. As a result, the learning dynamics required for reliable strategy agreement and fine-tuning may not emerge.

Why dataset coverage is not enough. Increasing dataset coverage can mitigate value-function extrapolation error, but it does not alone resolve this coordination issue. Even if $\text{supp}(p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)) \subseteq \text{supp}(p_{\boldsymbol{\pi}_{\text{off}}}(\tau))$, gradients computed under the offline distribution may still differ from $\nabla J(\boldsymbol{\pi}_{\text{curr}})$ because teammate actions in the dataset are not drawn from their current policies.

Each agent therefore optimises against effectively non-adapting teammates. Marginal distribution coverage alone cannot ensure consistent joint strategies, and Section C shows that in some settings offline learning can even collapse to a constant non-adaptive gradient signal.

Taken together, the offline training process fails to reflect how the current joint policy reshapes the trajectory distribution. In this work, we address this limitation through *data augmentation*, generating trajectories more consistent with the evolving current policy while remaining strictly offline.

4. Towards Joint Policy Evolution via Generative Modelling

Offline MARL in the BRUD paradigm lacks the feedback loop whereby as the joint policy $\boldsymbol{\pi}_{\text{curr}}$ changes, the distribution of trajectories sampled changes accordingly. CODA approximately restores this coupling, *without further environment interaction*, by defining a policy-dependent target trajectory distribution and using conditional generation to approximately sample from it.

4.1. The Desired Target Distribution

Ideally, training would use trajectories from the current joint policy trajectory distribution, namely

$$p_{\boldsymbol{\pi}_{\text{curr}}}(\tau) \;=\; p_0(s_0) \prod_{t=0}^{H-1} \Omega(\mathbf{o}_t \mid s_t)\, \boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t)\, T(s_{t+1} \mid s_t, \mathbf{a}_t). \tag{9}$$

If we could sample $\tau \sim p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$ (or equivalently evaluate expectations under it), then training would reflect how the joint policy $\boldsymbol{\pi}_{curr}$ reshapes the joint trajectory distribution, thereby restoring endogenous joint policy evolution and enabling co-adaptation.

4.2. Sampling From the Target Distribution

CODA uses conditional diffusion to generate trajectories approximately from $p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$. Operationally, diffusion sampling requires a noise-conditioned score $\nabla_{\hat{\tau}} \log p_{\boldsymbol{\pi}_{\text{curr}}}(\hat{\tau}; \sigma)$. We denote by $\hat{\tau}$ a noised trajectory at noise level σ.

Two complementary routes are used to sample from this distribution: (i) **policy-guided sampling** via classifier guidance when the policy is not easily encoded as an explicit conditioning variable; and (ii) **direct conditioning** via classifier-free guidance (CFG) when a compact conditioning representation of the policy is available.

(A) Policy-guided score-based sampling (implicit conditioning). We begin by leveraging the score-based derivation of Policy-Guided Diffusion (PGD, Jackson et al., 2024), and adapt it to the multi-agent setting. The ideal objective in Equation (8) requires expectations under $p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$. Given we only have access to offline data samples, trajectory-level importance sampling can be invoked. For any test function $f(\tau)$,

$$\begin{aligned} \mathbb{E}_{\tau \sim p_{\boldsymbol{\pi}_{\text{curr}}}}[f(\tau)] \;&=\; \mathbb{E}_{\tau \sim p_{\boldsymbol{\pi}_{\text{off}}}}[\mathbf{w}(\tau)\, f(\tau)]\,, \\ \mathbf{w}(\tau) \;&\triangleq\; \prod_{t=0}^{H-1} \frac{\boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t)}{\boldsymbol{\pi}_{\text{off}}(\mathbf{a}_t \mid \mathbf{o}_t)}. \end{aligned} \tag{10}$$

Equivalently,

$$p_{\boldsymbol{\pi}_{\text{curr}}}(\tau) = p_{\boldsymbol{\pi}_{\text{off}}}(\tau) \prod_{t=0}^{H-1} \frac{\boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t)}{\boldsymbol{\pi}_{\text{off}}(\mathbf{a}_t \mid \mathbf{o}_t)}. \tag{11}$$

This change-of-measure identity holds provided the standard importance sampling support condition is satisfied, namely that $\text{supp}(p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)) \subseteq \text{supp}(p_{\boldsymbol{\pi}_{\text{off}}}(\tau))$, so that the ratio is well-defined. In offline MARL this assumption is rarely guaranteed in practice, as the offline dataset may not provide full coverage of the possible trajectories from the on-policy

distribution. Consequently, directly implementing the exact importance ratio can be unstable or undefined when the current policy places mass outside the support of the data. We address this below.

Taking logs and differentiating Equation (11) yields

$$\nabla_\tau \log p_{\boldsymbol{\pi}_{\text{curr}}}(\tau) = \nabla_\tau \log p_{\boldsymbol{\pi}_{\text{off}}}(\tau) + \sum_{t=0}^{H-1} \Big(\nabla_\tau \log \boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \,|\, \mathbf{o}_t) - \nabla_\tau \log \boldsymbol{\pi}_{\text{off}}(\mathbf{a}_t \,|\, \mathbf{o}_t) \Big). \tag{12}$$

As in prior work (Jackson et al., 2024), in the limit $\sigma \to 0$ the noise-conditioned score approaches the score of the clean distribution. Approximating the noise-conditioned current score therefore gives

$$\nabla_{\hat{\tau}} \log p_{\boldsymbol{\pi}_{\text{curr}}}(\hat{\tau}; \sigma) \approx \nabla_{\hat{\tau}} \log p_{\boldsymbol{\pi}_{\text{off}}}(\hat{\tau}; \sigma) + \sum_{t=0}^{H-1} \Big(\nabla_{\hat{\tau}} \log \boldsymbol{\pi}_{\text{curr}}(\hat{\mathbf{a}}_t \,|\, \hat{\mathbf{o}}_t) - \nabla_{\hat{\tau}} \log \boldsymbol{\pi}_{\text{off}}(\hat{\mathbf{a}}_t \,|\, \hat{\mathbf{o}}_t) \Big). \tag{13}$$

The diffusion model trained on $\mathcal{D}_{\text{off}}$ provides $\nabla_{\hat{\tau}} \log p_{\boldsymbol{\pi}_{\text{off}}}(\hat{\tau}; \sigma)$. The current-policy term is also computable since $\boldsymbol{\pi}_{\text{curr}}$ is known and differentiable in most RL settings. However, $\boldsymbol{\pi}_{\text{off}}$ is typically unknown, so as in prior work, we drop the behaviour-policy term.

Beyond $\boldsymbol{\pi}_{\text{off}}$ being unknown, dropping this term is appropriate offline: The importance ratio implicitly assumes adequate support overlap between the current and behaviour distributions. Subtracting $\nabla \log \boldsymbol{\pi}_{\text{off}}$ can be unstable where the behaviour policy has negligible density. Keeping only the current-policy term lets the diffusion prior $p_{\boldsymbol{\pi}_{\text{off}}}$ serve as a support constraint, biasing samples toward regions likely under $\boldsymbol{\pi}_{\text{curr}}$ while staying on the empirical data manifold. As noted in Jackson et al. (2024), it also limits model error or out of sample challenges.

The practical guided score becomes:

$$\nabla_{\hat{\tau}} \log p(\hat{\tau}; \sigma) + \lambda \sum_{t=0}^{H-1} \nabla_{\hat{\mathbf{a}}_t} \log \boldsymbol{\pi}_{\text{curr}}(\hat{\mathbf{a}}_t \mid \hat{\mathbf{o}}_t), \tag{14}$$

where $\lambda \geq 0$ is the guidance scale and gradients are taken only with respect to action components for stability (Jackson et al., 2024). This score corresponds to sampling from the following *policy-tilted* distribution:

$$\tilde{p}_{\boldsymbol{\pi}_{\text{curr}}}(\tau) \;\propto\; p_{\boldsymbol{\pi}_{\text{off}}}(\tau) \exp\Big(\lambda \sum_{t=0}^{H-1} \log \boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t) \Big). \tag{15}$$

Thus, rather than recovering $p_{\boldsymbol{\pi}_{\text{curr}}}$ exactly, we sample from a *regularised surrogate* that reweights the offline trajectory distribution toward trajectories likely under the current joint policy. The diffusion prior constrains sampling to the empirical data support, while the policy term biases mass toward $\boldsymbol{\pi}_{\text{curr}}$-consistent behaviour. As $\boldsymbol{\pi}_{\text{curr}}$ evolves, this tilt updates immediately, inducing policy-dependent drift in the generated joint trajectory distribution.

(B) Direct policy-conditioning via CFG (explicit conditioning). When the policy can be represented as a compact conditioning variable (e.g., task embedding, parameter vector, or probing representation (Chandak et al., 2019)), we train a conditional trajectory diffusion model $p_\theta(\tau; \sigma \mid y)$ with condition dropout and apply classifier-free guidance (CFG) at sampling time. This directly produces trajectories whose distribution depends on the current joint policy descriptor y.

This route is preferred when a stable, low-dimensional conditioning interface exists.

Under standard conditional diffusion training (denoising score matching with condition dropout), the learned model approximates the data conditional

$$p_\theta(\tau \mid y) \;\propto\; p_{\boldsymbol{\pi}_{\text{off}}}(\tau) p(y \mid \tau).$$

If y encodes the current joint policy, then $p(y \mid \tau)$ acts as a compatibility term measuring how likely the trajectory is under that policy. In particular, when $p(y \mid \tau) \propto \exp\big(\sum_t \log \boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t)\big)$, the induced conditional coincides with the same policy-tilted distribution in Equation (15). Thus, CFG likewise samples from a distribution that re-weights $p_{\boldsymbol{\pi}_{\text{off}}}(\tau)$ toward trajectories that are probable under the current joint policy.

In this sense, both score-based policy guidance and explicit conditioning via CFG target the same pseudo-distribution Equation (15), which serves as a tractable, support-constrained approximation to the ideal on-policy target $p_{\boldsymbol{\pi}_{\text{curr}}}(\tau)$.

Why this is inherently multi-agent Equation (14) is joint in two critical ways: 1) The diffusion prior is trained on *joint trajectories*, capturing cross-agent correlations; 2) the guidance term is the gradient of the *joint* policy log-likelihood, $\log \boldsymbol{\pi}_{\text{curr}}(\mathbf{a}_t \mid \mathbf{o}_t) = \sum_{i=1}^{N} \log \pi^i_{\text{curr}}(a^i_t \mid o^i_t)$, so all agents' actions are nudged simultaneously toward regions jointly consistent with their *current* decentralised policies.

5. Implementing CODA

We now instantiate this idea concretely. **CODA** moves towards restoring endogenous joint policy evolution by (i) learning a diffusion prior over offline joint trajectories, and (ii) sampling synthetic trajectories from a policy-tilted distribution that depends on the current joint policy. These trajectories can then be used to augment the offline dataset used by any offline MARL algorithm. In Appendix A, we provide algorithmic descriptions of the CODA data gener-

ation process (Algorithm 1) and use of CODA in MARL training (Algorithm 2).

5.1. Full-Horizon Joint Trajectory Diffusion

Object being diffused. CODA generates finite-horizon joint trajectories $\tau = \left(s_0, \mathbf{o}_0, \mathbf{a}_0, r_0, \ \ldots, \ s_H\right)$, of length H, with $\tau \in \mathbb{R}^{d_\tau}$ after flattening and CDF normalisation. Thus, diffusion is performed over the *entire horizon at once*, rather than auto-regressively. This allows the model to capture: (i) **temporal correlations** across time indices $t = 0, \ldots, H$, and (ii) **cross-agent correlations** through the joint quantities $\mathbf{o}_t$ and $\mathbf{a}_t$ at each time-step.

Centralised diffusion backbone. We use a single centralised diffusion network that takes the noised trajectory $\hat{\tau}$ and noise level σ and predicts the joint denoising direction. States, joint observations, joint actions, and rewards are concatenated into a unified representation, with convolutions taken over the temporal dimension. Unlike MADiff (Zhu et al., 2024), we use no explicit cross-agent attention; coordination is captured by modelling the *joint* trajectory distribution over $(s_t, \mathbf{o}_t, \mathbf{a}_t, r_t)$ across the horizon. Both temporal dynamics and inter-agent dependencies are encoded directly in the learned score. We train this diffusion prior on $\mathcal{D}_{\text{off}}$ to estimate $\nabla_{\hat{\tau}} \log p_{\boldsymbol{\pi}_{\text{off}}}(\hat{\tau}; \sigma)$, using the EDM framework (Karras et al., 2022). At sampling time, we apply the objectives from Section 4 to sample approximately from the target distribution Equation (9); the two variants differ only in how conditioning is implemented.

6. Experiments

We evaluate CODA across two continuous control environments: Polynomial Games (Tilbury et al., 2024) and MA-MuJoCo (Peng et al., 2021; Formanek et al., 2023). CODA is complementary to any offline MARL algorithm; a flexibility inherited from the algorithmic structure of methods such as (Jackson et al., 2024) and Lu et al. (2023). We use MADDPG (Lowe et al., 2017) as the underlying model-free MARL algorithm used in each baseline, in order to ensure fair comparison.

6.1. Polynomial Games

We first consider two-player polynomial games in an offline setting, following the setup of Tilbury et al. (2024). These games provide a differentiable, continuous analogue of classical discrete matrix games (Barde et al., 2024). Two agents choose continuous actions $\mathbf{a} = (a^x, a^y) \in [-1, 1]^2$ and receive a shared reward $R(a^x, a^y)$ given by a polynomial. The environment is single-step and stateless, so an *episode* consists of a single joint action, $\mathbf{a}$, and $Q(\mathbf{a}, \mathbf{o}) = R(\mathbf{a})$. The objective is to learn policies that select actions $\mathbf{a}$, maximising the joint return $R(\mathbf{a})$.

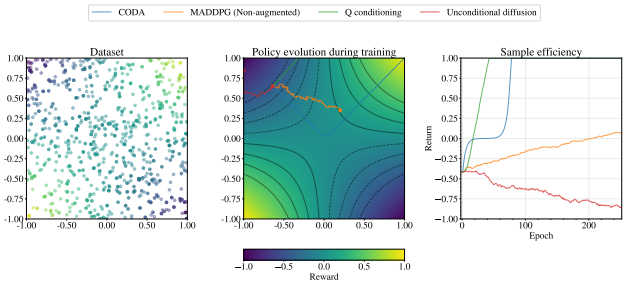

Figure 1. **Multiplication game ($R = a^x a^y$).** Reward takes values in $[-1, 1]$. Left: offline dataset used to train diffusion models and the baseline MADDPG policy. Middle: policy evolution during training. Right: returns during training (the final epoch equals the final test return for deterministic policies). Q-conditioned augmentation appears optimal only due to action-boundary effects ($\mathbf{a} \in [-1, 1]^2$): the y-agent quickly saturates at the action bounds, so the deterministic gradient update acts solely through the x-agent, inflating returns (see Figure 2 where exploiting boundary-effects cannot reach an optimum). Sample efficiency plot contains no error bars as in this environment deterministic policies create deterministic rewards at test-time.

Multiplication game. Tilbury et al. (2024) showed that BRUD algorithms such as MADDPG fail to coordinate in offline polynomial games, converging deterministically to suboptimal behaviours determined by the overall statistics in the fixed offline dataset (see Section C for discussion). Using MADDPG as the base offline learning algorithm, we compare CODA against three baselines: (i) using a non-augmented offline dataset; (ii) augmentation using an unconditional diffusion generator inspired by (Lu et al., 2023); and (iii) augmentation using a Q-conditioned diffusion generator inspired by (Oh et al., 2024). For this setting, we implement conditioning in CODA via CFG.

Figure 1 shows that baseline MADDPG with a non-augmented dataset converges suboptimally despite full coverage of the action space in the offline dataset, showing that coordination is not purely a coverage issue. Unconditional diffusion approximately reproduces the empirical data distribution $p_{\boldsymbol{\pi}_{\text{off}}}$ and therefore inherits the same deterministic training pathology as under the static dataset. Conditioning on high Q generates a high-return biased dataset but the data distribution remains static, leaving the coordination problem unresolved. The resulting training exhibits deterministic behaviour and the policy quickly collides with the action boundary during training. The convergence toward the optimum is therefore driven by the action space constraints rather than correct gradient following: The policy is dragged along the boundary. In contrast, CODA's on-policy conditioning alters the effective training distribution, allowing agents to follow the correct joint gradient and converge to the optimum.

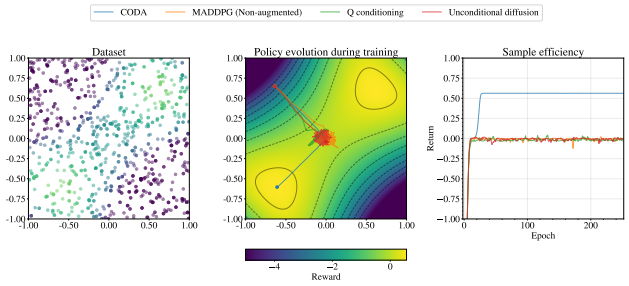

Figure 2. **Twin Peaks game.** Left: offline dataset used to train diffusion models and the baseline MADDPG policy. Middle: policy evolution during training. Right: returns during training (the final epoch equals the final test return for deterministic policies). CODA's on-policy conditioning again mitigates offline miscoordination. Sample efficiency plot contains no error bars as in this environment deterministic policies create deterministic rewards at test-time.

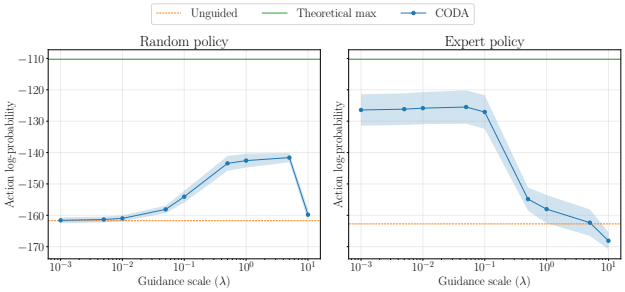

Figure 3. **On-policy steering in MaMuJoCo (2HalfCheetah).** Mean action log-likelihood of CODA-generated trajectories under the current policy $\boldsymbol{\pi}_{\text{current}}$ (Replay dataset). "Random" and "Expert" denote policies at initialisation and after 300k training steps on the Replay dataset. Standard error across 8 seeds; horizon $H = 20$; batch size 2048 trajectories. Unguided diffusion corresponds to $\lambda = 0$.

Robustness to increased agent interaction. To test robustness under stronger agent coupling, we consider the Twin Peaks game with reward $R = -A((a^x)^2 + (a^y)^2) - B(a^x a^y)^2 + Ca^x a^y, \quad A > 0, \ B > 0, \ C > 2A$. Figure 2 shows that CODA again recovers coordinated learning (A=1, B=4, C=5) while the baselines converge to $(0, 0)$. See Section C.3 for a theoretical treatment explaining the convergence to $(0, 0)$.

6.2. MaMuJoCo

Next, we evaluate CODA in MaMuJoCo, a higher-dimensional continuous-control benchmark in which success can depend on multiple interacting factors rather than coordination alone. We test two questions: (i) can CODA steer generated trajectories toward the target on-policy distribution; and (ii) how does this steering translate into varying degrees of downstream performance when training with a standard MARL algorithm (MADDPG) on synthetic data?

We use the BRUD-style MADDPG+BC algorithm (Lowe et al., 2017) as the base learner and follow the MaMuJoCo setup of Formanek et al. (2023). We sweep to select the behaviour cloning coefficient. To reduce wall-clock cost, we generate synthetic trajectories periodically rather than continuously, with $N_{\textbf{epochs}} = 4$ epochs between generation rounds, following the update frequency used in Jackson et al. (2024). Because MaMuJoCo policies are represented by high-dimensional neural networks, we use the classifier-guidance variant of CODA to target Equation (15).

6.2.1. Steering trajectories on-policy

Before reporting downstream performance, we verify that the guidance scale λ has the intended effect. For synthetic trajectories τ generated by CODA, we compute the mean action log-likelihood under the current policy $\boldsymbol{\pi}_{\text{curr}}$. For deterministic policies, we evaluate this under a Gaussian surrogate centered at $\boldsymbol{\mu}_{\text{current}}(\mathbf{o}_t)$ (summed over the horizon and agents), with fixed standard deviation 1. See Section B for the theoretical analysis of how λ affects this metric.

Results. Figure 3 shows that unconditional diffusion ($\lambda = 0$) produces trajectories with low likelihood under $\boldsymbol{\pi}_{\text{curr}}$, while conditioning increases on-policy likelihood and exhibits the expected peak-and-collapse shape as λ grows (See Section B for a theoretical explanation). Notably, for the expert target, very small λ yields a sharp discontinuous jump in likelihood. We hypothesise this is because the expert's action distribution is concentrated; any guidance, regardless of how small, is sufficient to move samples into a high-density region of $\boldsymbol{\pi}_{\text{curr}}$.

6.3. Investigating downstream performance

Having verified that guidance can increase likelihood under $\boldsymbol{\pi}_{\text{curr}}$, we evaluate whether this improves performance when learning with MADDPG+BC using synthetic trajectories. We report test-time episodic return (mean over 10 episodes per seed) and aggregate over 16 seeds.

Results. Figure 4 and Table 1 summarise final performance on 2HalfCheetah datasets and on 4Ant-Replay for (i) MADDPG+BC, (ii) MADDPG+BC with unconditional diffusion augmentation, and (iii) CODA. CODA yields the strongest gains on the **Replay** and **Good** datasets. This matches CODA's objective: sampling from the policy-tilted distribution in Equation (15) reweights offline data toward trajectories that remain under the support of the empirical dataset while being high-likelihood under the current policy. These datasets have examples of expert trajectories. On **Medium** and especially **Poor**, CODA does not outperform unconditional diffusion. A plausible explanation is limited dataset support: as the learned policy improves, trajectories

Table 1. **MaMuJoCo performance (mean ± s.e.).** Evaluation across 16 seeds; each seed averaged over 10 episodes.

Task	Dataset	MADDPG + BC	MADDPG + BC + Diffusion	CODA
2HalfCheetah	Replay	**6854.97** ± 79.74	6438.85 ± 135.27	**6925.02** ± 72.94
	Good	**7187.77** ± 22.94	7185.59 ± 20.96	**7208.19** ± 25.35
	Medium	4515.81 ± 8.91	**4554.11** ± 10.28	4530.19 ± 9.41
	Poor	-64.79 ± 106.92	**2713.69** ± 92.70	715.90 ± 196.56
4Ant	Replay	**1858.39** ± 60.19	1742.14 ± 89.28	**1875.51** ± 80.90

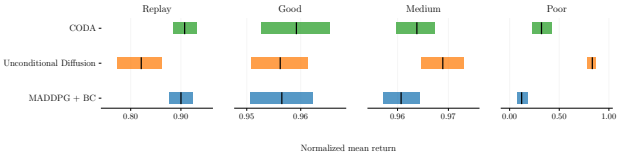

Figure 4. **2HalfCheetah mean normalised performance across datasets.** Normalised within each dataset. Standard error across 16 seeds; each seed averaged over 10 episodes.

consistent with π_{curr} may increasingly fall outside the behavioural support represented in these datasets, constraining the extent to which policy-tilting can help while remaining faithful to offline support. In contrast, unconditional diffusion can still help by providing broader transition diversity without explicitly concentrating probability mass near π_{curr}.

7. Related Work

Offline MARL and offline RL baselines. Offline MARL learns coordinated policies from fixed datasets, often by extending single-agent offline RL methods such as BCQ (Fujimoto et al., 2019), CQL (Kumar et al., 2020), and TD3+BC (Fujimoto & Gu, 2021) to the decentralised MARL setting (Jiang & Lu, 2021). To limit out-of-distribution generalisation error, ICQ (Yang et al., 2021) imposes an implicit constraint that keeps learning near the dataset support, mitigating extrapolation error that can amplify with more agents. Complementarily, OMAR (Pan et al., 2021) addresses optimisation issues from conservative value estimates by combining first-order policy gradients with a zeroth-order actor update to escape poor local optima.

Coordination and offline coordination failure. Cooperative MARL introduces per-agent credit assignment and coordination challenges absent in single-agent RL (Albrecht et al., 2024; Tilbury et al., 2024), exacerbated by partial observability and limited information sharing (Zhang & Lesser, 2013). Recent work shows that offline cooperative MARL can fail even under CTDE because offline training is off-policy (Tilbury et al., 2024; Barde et al., 2024). Related perspectives on coordination and generalisation include plan- and learning-based coordination (Boutilier, 1996), zero-shot coordination (Hu et al., 2020), and augmentation/self-play mechanisms (Lerer & Peysakhovich, 2019).

Model-based offline MARL. A complementary direction to ours approximates online interaction through learned dynamics and imagined rollouts. MOMA-PPO (Barde et al., 2024) uses a world model to produce policy-dependent experience and reduce offline coordination failures. Beyond tackling coordination, model-based MARL learns dynamics for rollout-based updates and planning (Willemsen et al., 2021; Zhang et al., 2022; 2021). These approaches simulate environment transitions forward, whereas we generate joint trajectories directly from the offline data distribution with conditioning on current policies.

Diffusion models for RL and MARL augmentation. Diffusion models have been used in RL for trajectory planning (Janner et al., 2022; Liang et al., 2023), as expressive policy parameterisations in offline RL (Ajay et al., 2022; Wang et al., 2022; Pearce et al., 2023), and as world models for imagination and long-horizon prediction (Alonso et al., 2025; Ding et al., 2024). They have also been explored for single-task (Lu et al., 2023) & multi-task trajectory generation (He et al., 2023) and trust-region style control of policy updates in offline RL (Chen et al., 2025). In offline MARL, diffusion-based generation has been used for episode augmentation and structured trajectory modelling: EAQ (Oh et al., 2024) generates synthetic episodes via reward-conditioned diffusion to bias augmentation toward high-return trajectories, while MADiff (Zhu et al., 2024) and related methods model multi-agent trajectories with diffusion backbones (Yuan et al., 2025; Li et al., 2025; 2023). Our work builds most directly on policy-guided diffusion for single-agent RL (Jackson et al., 2024; Rigter et al., 2024), and extends the idea of conditioning generation on the *current* policy to the multi-agent regime to tackle coordination.

8. Conclusion

This paper argues that a core driver of coordination failure in cooperative offline MARL is the loss of the feedback loop that allows teammates to adapt to one another during learning. Focusing on BRUD-style algorithms (e.g. MAD-

DPG and MASAC), we show that existing diffusion-based augmentation approaches still yield effectively static training distributions: agents update against stale teammate behaviours and can converge to jointly suboptimal solutions, even when the offline data provides broad marginal coverage.

CODA takes a step towards addressing this by making augmentation policy-dependent: the generated joint trajectories are reweighted toward samples the current joint policy would most likely produce, while remaining anchored to the empirical dataset support. In controlled polynomial games, this is sufficient to recover the desired joint optimisation dynamics and avoid the deterministic miscoordination observed under BRUD-style learning and existing diffusion augmentation approaches. In MaMuJoCo, increasing guidance reliably steers generated trajectories toward the current policy. Downstream performance varies depending on dataset quality.

There are a number of future work directions including adaptive guidance selection, and exploring hybrid approaches that combine policy-conditioned generation with model-based rollouts to expand effective coverage beyond the offline data while mitigating compounding model error from autoregressive world modelling.

Acknowledgements

MH and this research was supported by Recurvia Ltd alongside Novo Nordisk and the EPSRC Centre for Doctoral Training in Modern Statistics and Statistical Machine Learning (EP/S023151/1). AS is supported by the EPSRC Centre for Doctoral Training in Modern Statistics and Statistical Machine Learning (EP/S023151/1).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Ajay, A., Du, Y., Gupta, A., Tenenbaum, J. B., Jaakkola, T., and Agrawal, P. Is conditional generative modeling all you need for decision-making? *ArXiv*, abs/2211.15657, 2022. URL https://api.semanticscholar.org/CorpusID:254044710.

Albrecht, S. V., Christianos, F., and Schäfer, L. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. URL https://www.marl-book.com.

Alonso, E., Jelley, A., Micheli, V., Kanervisto, A., Storkey, A., Pearce, T., and Fleuret, F. Diffusion for world modeling: visual details matter in atari. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.

Bansal, A., Chu, H., Schwarzschild, A., Sengupta, S., Goldblum, M., Geiping, J., and Goldstein, T. Universal guidance for diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=pzpWBbnwiJ.

Barde, P., Foerster, J., Nowrouzezahrai, D., and Zhang, A. A model-based solution to the offline multi-agent reinforcement learning coordination problem. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '24, pp. 141–150, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400704864.

Boutilier, C. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '96, pp. 195–210, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc. ISBN 1558604179.

Chandak, Y., Theocharous, G., Kostas, J. E., Jordan, E., and Thomas, P. S. Learning action representations for reinforcement learning. In *International Conference on Machine Learning*, 2019. URL https://api.semanticscholar.org/CorpusID:59553460.

Chen, T., Wang, Z., and Zhou, M. Diffusion policies creating a trust region for offline reinforcement learning. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Ding, Z., Zhang, A., Tian, Y., and Zheng, Q. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning, 2024. URL https://arxiv.org/abs/2402.03570.

Formanek, C., Jeewa, A., Shock, J., and Pretorius, A. Off-the-grid marl: Datasets and baselines for offline multi-agent reinforcement learning. In *Proceedings of the 2023*

International Conference on Autonomous Agents and Multiagent Systems, AAMAS '23, pp. 2442–2444, Richland, SC, 2023. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450394321.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *ArXiv*, abs/2004.07219, 2020. URL https://api.semanticscholar.org/CorpusID:215827910.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/fujimoto19a.html.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ArXiv*, abs/1801.01290, 2018. URL https://api.semanticscholar.org/CorpusID:28202810.

He, H., Bai, C., Xu, K., Yang, Z., Zhang, W., Wang, D., Zhao, B., and Li, X. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Ho, J. and Salimans, T. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. "other-play " for zero-shot coordination. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Jackson, M., Matthews, M., Lu, C., Ellis, B., Whiteson, S., and Foerster, J. N. Policy-guided diffusion. *ArXiv*, abs/2404.06356, 2024. URL https://api.semanticscholar.org/CorpusID:269009692.

Janner, M., Du, Y., Tenenbaum, J. B., and Levine, S. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022. URL https://api.semanticscholar.org/CorpusID:248965046.

Jiang, J. and Lu, Z. Offline decentralized multi-agent reinforcement learning. *ArXiv*, abs/2108.01832, 2021. URL https://api.semanticscholar.org/CorpusID:236912690.

Karras, T., Aittala, M., Laine, S., and Aila, T. Elucidating the design space of diffusion-based generative models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33: 1179–1191, 2020.

Lerer, A. and Peysakhovich, A. Learning existing social conventions via observationally augmented self-play. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, pp. 107–114, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450363242. doi: 10.1145/3306618.3314268. URL https://doi.org/10.1145/3306618.3314268.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *ArXiv*, abs/1805.00909, 2018. URL https://api.semanticscholar.org/CorpusID:19077536.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv*, abs/2005.01643, 2020. URL https://api.semanticscholar.org/CorpusID:218486979.

Li, C., Deng, Z., Lin, C., Chen, W., Fu, Y., Liu, W., Wen, C., Wang, C., and Shen, S. Dof: A diffusion factorization framework for offline multi-agent reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OTFKVkxSlL.

Li, Z., Pan, L., and Huang, L. Beyond conservatism: Diffusion policies in offline multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2307.01472.

Liang, Z., Mu, Y., Ding, M., Ni, F., Tomizuka, M., and Luo, P. Adaptdiffuser: diffusion models as adaptive self-evolving planners. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6382–6393, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Lu, C., Ball, P. J., Teh, Y. W., and Parker-Holder, J. Synthetic experience replay. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Matignon, L., Jeanpierre, L., and Mouaddib, A.-I. Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pp. 2017–2023. AAAI Press, 2012.

Mi, Q., Xia, S., Song, Y., Zhang, H., Zhu, S., and Wang, J. Taxai: A dynamic economic simulator and benchmark for multi-agent reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '24, pp. 1390–1399, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400704864.

Oh, J., Kim, S., Kim, G., Kim, S., and Yun, S.-Y. Diffusion-based episodes augmentation for offline multi-agent reinforcement learning, 2024. URL https://arxiv.org/abs/2408.13092.

Oliehoek, F. A. and Amato, C. *A Concise Introduction to Decentralized POMDPs*. Springer Publishing Company, Incorporated, 1st edition, 2016. ISBN 3319289276.

Pan, L., Huang, L., Ma, T., and Xu, H. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. *CoRR*, abs/2111.11188, 2021. URL https://arxiv.org/abs/2111.11188.

Pearce, T., Rashid, T., Kanervisto, A., Bignell, D., Sun, M., Georgescu, R., Macua, S. V., Tan, S. Z., Momennejad, I., Hofmann, K., and Devlin, S. Imitating human behaviour with diffusion models. *ArXiv*, abs/2301.10677, 2023. URL https://api.semanticscholar.org/CorpusID:256231177.

Peng, B., Rashid, T., de Witt, C. A. S., Kamienny, P.-A., Torr, P. H. S., Böhmer, W., and Whiteson, S. Facmac: factored multi-agent centralised policy gradients. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Rigter, M., Yamada, J., and Posner, I. World models via policy-guided trajectory diffusion. *Transactions on Machine Learning Research*, 2024(2), 2024.

Tilbury, C. R., Formanek, C., Beyers, L., Shock, J. P., and Pretorius, A. Coordination failure in cooperative offline marl, 2024. URL https://arxiv.org/abs/2407.01343.

Wang, Z., Hunt, J. J., and Zhou, M. Diffusion policies as an expressive policy class for offline reinforcement learning. *ArXiv*, abs/2208.06193, 2022. URL https://api.semanticscholar.org/CorpusID:251554821.

Willemsen, D., Coppola, M., and de Croon, G. C. Mambpo: Sample-efficient multi-robot reinforcement learning using learned world models. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5635–5640. IEEE Press, 2021. doi: 10.1109/IROS51168.2021.9635836. URL https://doi.org/10.1109/IROS51168.2021.9635836.

Yang, Y., Ma, X., Li, C., Zheng, Z., Zhang, Q., Huang, G., Yang, J., and Zhao, Q. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:10299–10312, 2021.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of PPO in cooperative multi-agent games. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=YVXaxB6L2Pl.

Yuan, L., Bian, Y., Li, L., Zhang, Z., Guan, C., and Yu, Y. Efficient multi-agent offline coordination via diffusion-based trajectory stitching. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EpnZEzYDUT.

Zhang, C. and Lesser, V. Coordinating multi-agent reinforcement learning with limited communication. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS '13, pp. 1101–1108, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450319935.

Zhang, Q., Lu, C., Garg, A., and Foerster, J. Centralized model and exploration policy for multi-agent rl. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '22, pp. 1500–1508, Richland, SC, 2022. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

Zhang, W., Wang, X., Shen, J., and Zhou, M. Model-based multi-agent policy optimization with adaptive opponent-wise rollouts. *ArXiv*, abs/2105.03363, 2021. URL https://api.semanticscholar.org/CorpusID:234093673.

Zhu, Z., Liu, M., Mao, L., Kang, B., Xu, M., Yu, Y., Ermon, S., and Zhang, W. Madiff: Offline multi-agent learning with diffusion models. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 4177–4206. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/07e278a120830b10aae20cc600a8c07b-Paper-Conference.pdf.

A. Algorithms

Algorithm 1 CODA: Trajectory Sampling via Joint Policy–Guided Diffusion (Multi-Agent)

Parameters: noise schedule $\{\sigma_n\}_{n=0}^{N_{\text{diff}}}$, guidance schedule $\{\lambda_n\}_{n=0}^{N_{\text{diff}}}$, temporary noise factor γ_n, noise level S_{noise}, diffusion steps N_{diff}
Require: trajectory denoiser D_θ trained on offline data; joint target policy $\boldsymbol{\pi}_\phi = \{\pi_\phi^i\}_{i=1}^N$
Notation: a trajectory $\tau = (s_0, \mathbf{o}_0, \mathbf{a}_0, r_0, \ldots, s_H)$; bar $\bar{\cdot}$ denotes denoised estimates; hat $\hat{\cdot}$ noised variables
Init: sample $\tau_0 \sim \mathcal{N}(0, \sigma_0^2 I)$ ▷ EDM/score-based ODE init
for $n = 0$ **to** $N_{\text{diff}} - 1$ **do**
  sample $\epsilon_n \sim \mathcal{N}(0, S_{\text{noise}}^2 I)$
  $\hat{\sigma}_n \leftarrow \sigma_n + \gamma_n \sigma_n; \quad \hat{\tau}_n \leftarrow \tau_n + \sqrt{\hat{\sigma}_n^2 - \sigma_n^2}\, \epsilon_n$
  $\bar{\tau}_n \leftarrow D_\theta(\hat{\tau}_n; \hat{\sigma}_n)$ ▷ Denoised trajectory, converted to constrained space
  $d_n \leftarrow (\hat{\tau}_n - \bar{\tau}_n)/\hat{\sigma}_n$ ▷ EDM preconditioning: $\partial\tau/\partial\sigma$ Score approx
  // Joint policy guidance on actions only (multi-agent)
  Extract $(\bar{\mathbf{a}}_t, \bar{\mathbf{o}}_t)$ for all t from $\bar{\tau}_n$
  for each time $t = 0, \ldots, H-1$ **do**
    $g_t \leftarrow \sum_{i=1}^N \nabla_{\bar{a}_t^{(i)}} \log \pi_\phi^{(i)}\left(\bar{a}_t^{(i)} \mid \bar{o}^{(i)}\right)$ ▷ Per-agent scores
  end for
  $g_n \leftarrow \text{stack}(g_t)_{t=0}^{H-1}; \quad g_n \leftarrow g_n / \|g_n\|_2$ ▷ Normalize for stability
  $\hat{\mathbf{a}}_n \leftarrow \text{actions}(\hat{\tau}_n); \quad \hat{\mathbf{a}}_n \leftarrow \hat{\mathbf{a}}_n + \lambda_n\, g_n$ ▷ Apply guidance to *noised* actions
  $\hat{\tau}_n \leftarrow \text{replace_actions}(\hat{\tau}_n, \hat{\mathbf{a}}_n)$
  $\tau_{n+1} \leftarrow \hat{\tau}_n + (\sigma_{n+1} - \hat{\sigma}_n)\, d_n$ ▷ Euler step
  if $\sigma_{n+1} \neq 0$ **then** ▷ 2nd-order correction (EDM)
    $d'_n \leftarrow \left(\tau_{n+1} - D_\theta(\tau_{n+1}; \sigma_{n+1})\right)/\sigma_{n+1}$ ▷ Score approx
    $\hat{\tau}_{n+1} \leftarrow \hat{\tau}_n + (\sigma_{n+1} - \hat{\sigma}_n)\left(\frac{1}{2} d_n + \frac{1}{2} d'_n\right)$
    $\tau_{n+1} \leftarrow \hat{\tau}_{n+1}$
  end if
end for
return $\tau_{N_{\text{diff}}}$ ▷ Map back to original domains if using logit/sigmoid transforms

Algorithm 2 Training an Agent with CODA

Parameters: epochs N_{epochs}, policy update steps per epoch N_{policy}, synthetic set size B (trajectories), guidance schedule $\{\lambda_n\}_{n=0}^{N_{\text{diff}}}$, diffusion steps N_{diff}, mix ratio $\alpha \in [0, 1]$
Require: offline dataset $\mathcal{D}_{\text{off}}$; diffusion sampler $\mathcal{F} \equiv$ Algorithm 1; base offline RL algorithm (e.g., MADDPG-BC)
Train trajectory diffusion D_θ on $\mathcal{D}_{\text{off}}$ ▷ Behavior prior over full trajectories
Initialize joint policy $\boldsymbol{\pi}_\phi$ (and critics if applicable)
for $e = 1$ **to** N_{epochs} **do**
  // Generate on-policy synthetic experience under current joint policy
  $\mathcal{D}_{\text{syn}} \leftarrow \{\tau^{(j)} \sim \mathcal{F}(\,\cdot\,|\boldsymbol{\pi}_\phi, D_\theta; \{\lambda_n\}_{n=0}^{N_{\text{diff}}})\}_{j=1}^B$
  // Replay mix with real data
  $\mathcal{B} \leftarrow \text{concat}\big(\text{sample}(\mathcal{D}_{\text{syn}}, \alpha),\ \text{sample}(\mathcal{D}_{\text{off}}, 1-\alpha)\big)$
  for $k = 1$ **to** N_{policy} **do**
    Sample a mini-batch of sub-trajectories/transitions from $\mathcal{B}$ (or $\mathcal{D}_{\text{syn}}$ if α=1)
    Update critics and each agent policy $\pi_\phi^{(i)}$ via chosen offline RL algo ;
  end for
end for
return $\boldsymbol{\pi}_\phi$

B. Geometric Intuition for Guidance Coefficient

Convex–contractiveness of policy conditioning. The below investigates policy conditioning when implementing CODA using the score based classifier-based conditioning. For a deterministic policy $\mu(\mathbf{o})$ with a surrogate quadratic log-likelihood model

$$\log p(\mathbf{a} \mid \mathbf{o}) \;=\; -\tfrac{1}{2}\|\mathbf{a} - \mu(\mathbf{o})\|^2,$$

the gradient of the log-likelihood is

$$\nabla_{\mathbf{a}} \log p(\mathbf{a} \mid \mathbf{o}) \;=\; -(\mathbf{a} - \mu(\mathbf{o})).$$

Which is a vector pointing towards $\mu(\mathbf{o})$. A single policy-conditioned update of magnitude λ therefore takes the form

$$\mathbf{a}' \;=\; \mathbf{a} + \lambda \nabla_{\mathbf{a}} \log p(\mathbf{a} \mid \mathbf{o}) \;=\; (1-\lambda)\mathbf{a} + \lambda\mu(\mathbf{o}).$$

For $\lambda \in (0, 2)$ this update is a *convex combination* of $\mathbf{a}$ and $\mu(\mathbf{o})$, hence strictly contractive:

$$\|\mathbf{a}' - \mu(\mathbf{o})\| \;=\; |1-\lambda|\|\mathbf{a} - \mu(\mathbf{o})\|.$$

Where the above is the distance from μ following the update step.

- If $\lambda \in (0, 1)$, then $1 - \lambda \in (0, 1)$, so $\mathbf{a}'$ is a convex combination of $\mathbf{a}$ and $\mu(\mathbf{o})$ and the distance shrinks strictly:

$$\|\mathbf{a}' - \mu(\mathbf{o})\| \;=\; (1-\lambda)\,\|\mathbf{a} - \mu(\mathbf{o})\| \;<\; \|\mathbf{a} - \mu(\mathbf{o})\| \qquad (\mathbf{a} \neq \mu(\mathbf{o})).$$

- If $\lambda \in (1, 2)$, then $1 - \lambda < 0$, so the update overshoots to the other side of $\mu(\mathbf{o})$ (the direction flips), but it is still contractive because $|1 - \lambda| = \lambda - 1 \in (0, 1)$:

$$\|\mathbf{a}' - \mu(\mathbf{o})\| \;=\; (\lambda-1)\,\|\mathbf{a} - \mu(\mathbf{o})\| \;<\; \|\mathbf{a} - \mu(\mathbf{o})\|.$$

- If $\lambda = 1$, then $\mathbf{a}' = \mu(\mathbf{o})$ in one step.

- If $\lambda > 2$ (or $\lambda < 0$), then $|1 - \lambda| > 1$ and the update becomes expansive:

$$\|\mathbf{a}' - \mu(\mathbf{o})\| \;>\; \|\mathbf{a} - \mu(\mathbf{o})\|.$$

Thus, for $\lambda \in (0, 2)$, the update always increases the Gaussian log-likelihood.

In diffusion-based planners such as EDM, this idealised threshold is substantially altered by three factors. (i) Gradients are normalised in our implementation, making the effective overshoot condition depend on the current distance $\|\mathbf{a} - \mu(\mathbf{o})\|$ rather than $\lambda = 1$. (ii) Guidance is applied repeatedly across K denoising steps, and later steps may partially correct earlier overshoots. (iii) Trajectory likelihood aggregates over time and agents, so local overshoot effects are smoothed. Consequently, we often observe that policy likelihood continues to improve even for $\lambda > 1$, followed by a sharp collapse once the cumulative expansive effect dominates the corrective capacity of subsequent denoising steps.

C. BRUD Updates can Induce Constant Gradients

We focus this theoretical analysis within the polynomial game environment. In order to test coordination as defined by Barde et al. (2024), we must test *strategy agreement*. Consequently, it is critical that the games selected admit *multiple optima*: agreement is only meaningful when there are several distinct optimal joint strategies that agents must select consistently. In contrast, in a single-optimum setting, many methods would resolve the task trivially, since no coordination (in the form of selecting among competing optimal conventions) is required.

Tilbury et al. (2024) show that in offline MARL, BRUD-style policy updates can induce *non-adaptive* (constant) gradient fields: the direction of improvement is determined by dataset statistics rather than by the local geometry at the current joint policy. The below treatment is largely a restatement of their arguments.

C.1. BRUD Updates in Stateless Polynomial Two-Player Games

Consider the setup from Section 6.1 with continuous actions $a = (a^x, a^y) \in [-1, 1]^2$ and shared reward $R(a_x, a_y)$. Since the environment is single-step, stateless and the reward is assumed known, the centralised critic reduces to the reward: $Q(\mathbf{o}, \mathbf{a}) = R(a^x, a^y)$. Note a learned critic can also be implemented if desired.

We use state-independent deterministic policies $\mu_{\theta^x}, \mu_{\theta^y}$ defined by

$$\mu_{\theta^x} \equiv \theta^x, \qquad \mu_{\theta^y} \equiv \theta^y,$$

so the chosen actions are $a^x = \mu_{\theta^x}$ and $a^y = \mu_{\theta^y}$. Then $\nabla_{\theta^x} \mu_{\theta^x} = 1$, and following the action sampling approach of Equation (6), we find that Equation (7) becomes the following when updating agent x:

$$\begin{aligned} \nabla_{\theta^x} J &= \mathbb{E}_{a^y \sim \mathcal{D}_{\text{off}}} \left[\nabla_{\theta^x} \mu_{\theta^x} \; \nabla_{\tilde{a}^x} R(\tilde{a}^x, a^y) \big|_{\tilde{a}^x = \mu_{\theta^x}} \right] \\ &= \mathbb{E}_{a^y \sim \mathcal{D}_{\text{off}}} \left[\partial_{\tilde{a}^x} R(\tilde{a}^x, a^y) \big|_{\tilde{a}^x = \theta^x} \right]. \end{aligned} \tag{16}$$

Similarly,

$$\nabla_{\theta^y} J = \mathbb{E}_{a^x \sim \mathcal{D}_{\text{off}}} \left[\partial_{a^y} R(a^x, a^y) \big|_{a^y = \theta^y} \right]. \tag{17}$$

Equations (16)–(17) illustrate a key mismatch in offline BRUD updates. In coordinated settings, we would like agent x's update to respond to agent y's *current behaviour*, so that learning adapts as y's policy changes. Instead, BRUD takes an expectation over the offline dataset for the other agent's action, which marginalises out y's current policy entirely. As a result, agent x does not adapt to μ_{θ^y}; any influence of agent y enters only through fixed statistics of the dataset.

In the special case where the relevant term is linear in a^y, this dependence reduces simply to the dataset mean action of agent y. More generally, for higher-order interactions it reduces to higher-order dataset moments (e.g. mean, variance, etc.). In all cases, BRUD couples agents through static dataset statistics rather than through the teammate's evolving behaviour.

C.2. Multiplication Game: A Constant Vector Field

Here, we specifically consider the multiplication polynomial game $R(a^x, a^y) = a^x a^y$.

Online objective (on-policy joint distribution). The online objective is

$$J_{\text{on}}(\mu_{\theta^x}, \mu_{\theta^y}) \;\triangleq\; \mathbb{E}_{a^x \sim \mu_{\theta^x},\, a^y \sim \mu_{\theta^y}} [R(a^x, a^y)] = \mathbb{E}_{a^x \sim \mu_{\theta^x}} [a^x] \; \mathbb{E}_{a^y \sim \mu_{\theta^y}} [a^y], \tag{18}$$

where the factorisation follows because $R(a^x, a^y) = a^x a^y$ and the sampling is independent.

Equation (18) therefore reduces to

$$J_{\text{on}}(\mu_{\theta^x}, \mu_{\theta^y}) = \theta^x \theta^y, \qquad \Rightarrow \qquad \nabla_\theta J_{\text{on}}(\mu_{\theta^x}, \mu_{\theta^y}) = (\theta^y, \theta^x). \tag{19}$$

Thus the online gradient field is *policy-adaptive*: the update direction varies with the current joint policy (θ^x, θ^y) and matches the true surface gradient $\nabla R(a^x, a^y) = (a^y, a^x)$ evaluated at $(a^x, a^y) = (\theta^x, \theta^y)$.

Constant vector field under BRUD. Under offline BRUD-style updates, using Equation (16)–Equation (17), we see the below in the multiplication game setting

$$\nabla_{\theta^x} J = \mathbb{E}_{a^y \sim \mathcal{D}_{\text{off}}} [a^y] \equiv \bar{a}^y, \qquad \nabla_{\theta^y} J = \mathbb{E}_{a^x \sim \mathcal{D}_{\text{off}}} [a^x] \equiv \bar{a}^x.$$

Hence the induced BRUD "gradient field" in policy-parameter space is

$$\nabla_\theta J(\theta^x, \theta^y) = (\bar{a}^y, \bar{a}^x), \tag{20}$$

a *constant* (unidirectional) vector determined entirely by dataset means. In practice we sample mini-batches, so we follow an empirical estimate $(\hat{\mathbb{E}}[a^y], \hat{\mathbb{E}}[a^x])$, which introduces small stochastic deviations around the straight-line trajectory implied by Equation (20) (consistent with the slight wobble observed in Figure 1 by non-CODA approaches).

Action bounds and boundary effects. When actions are constrained (e.g. $\mathbf{a} \in [-1, 1]^2$), a constant drift direction can push policies quickly to the boundary; subsequent updates may then "slide" along the constraint surface. This can create the appearance of improvement even when the underlying update direction is actually constant because the policy slides along the boundary.

C.3. Twin Peaks: Dependence on Dataset Mean and Variance

Again following the treatment within Tilbury et al. (2024), now consider the Twin Peaks reward:

$$R(a^x, a^y) = -A((a^x)^2 + (a^y)^2) \; - \; B(a^x a^y)^2 \; + \; C a^x a^y, \qquad A > 0, \; B > 0, \; C > 2A. \tag{21}$$

The partial derivative for agent x is

$$\partial_{a^x} R(a^x, a^y) = -2Aa^x \; - \; 2B\, a^x\, (a^y)^2 \; + \; Ca^y.$$

Plugging into Equation (16) and writing dataset moments $\bar{a}^y \equiv \mathbb{E}_{\mathcal{D}_{\text{off}}}[a^y]$ and $\mathbb{E}_{\mathcal{D}_{\text{off}}}[(a^y)^2] = (\bar{a}^y)^2 + (\sigma^y)^2$ (where σ^y is the sample standard deviation of the a^y in $\mathcal{D}_{\text{off}}$), we obtain the BRUD update for x:

$$\begin{aligned} \nabla_{\theta^x} J(\theta^x) &= \mathbb{E}_{a^y \sim \mathcal{D}_{\text{off}}} \left[-2A\theta^x - 2B\, \theta^x\, (a^y)^2 + Ca^y \right] \\ &= -2A\theta^x \; - \; 2B\, \theta^x ((\bar{a}^y)^2 + (\sigma^y)^2) \; + \; C\bar{a}^y. \end{aligned} \tag{22}$$

By symmetry,

$$\nabla_{\theta^y} J(\theta^y) = -2A\theta^y \; - \; 2B\, \theta^y ((\bar{a}^x)^2 + (\sigma^x)^2) \; + \; C\bar{a}^x. \tag{23}$$

Fixed point (converged policy) under offline BRUD. Setting Equation (22) to zero yields the stationary point for agent x:

$$\nabla_{\theta^x} J = 0 \quad \Longleftrightarrow \quad \theta^{x*} = \frac{C\,(\bar{a}^y)}{2A + 2B((\bar{a}^y)^2 + (\sigma^y)^2)}. \tag{24}$$

This highlights that the offline BRUD solution once again does not depend on the current policy of the other agent. It depends on the dataset mean $\bar{a}^y$ alongside the dataset *spread* $(\sigma^y)^2$ through the interaction term $(a^x a^y)^2$.

Origin-centred datasets cannot recover the true optimum. If the dataset is centred so that $\bar{a}^y = 0$, then Equation (24) gives $\theta^{x*} = 0$ regardless of $(\sigma^y)^2$. Thus, increasing action diversity (variance) cannot recover the true optimum under BRUD when the dataset is centred at the origin; learning collapses to $(0, 0)$ (Figure 2).

Increasing diversity can be harmful even when centred near the optimum. Even if the dataset mean is centred at an optimal action (or near it), Equation (24) shows that as $(\sigma^y)^2$ grows, the denominator increases and $\theta^{x*} \to 0$. Hence, greater diversity in teammate actions within the dataset can drive the BRUD fixed point *away* from the true optimum and back toward the origin, a counter-intuitive effect induced by higher-order interaction terms. Increased action space coverage becomes detrimental.

How this relates to "constant" gradients. Unlike the multiplication game, Twin Peaks does not produce a strictly constant vector field. However, it is non-adaptive in relation to the current policy of teammates. Equation (22)–Equation (23) show the same structural issue: the *cross-agent* signal entering each agent's update is mediated only through low-order dataset moments (mean and variance), rather than through the teammate's *current* action. As a result, the induced learning dynamics becomes largely dataset-determined and only weakly responsive to the evolving joint policy, which is the core non-adaptivity phenomenon highlighted by Tilbury et al. (2024).

D. MaMuJoCo Additional Results

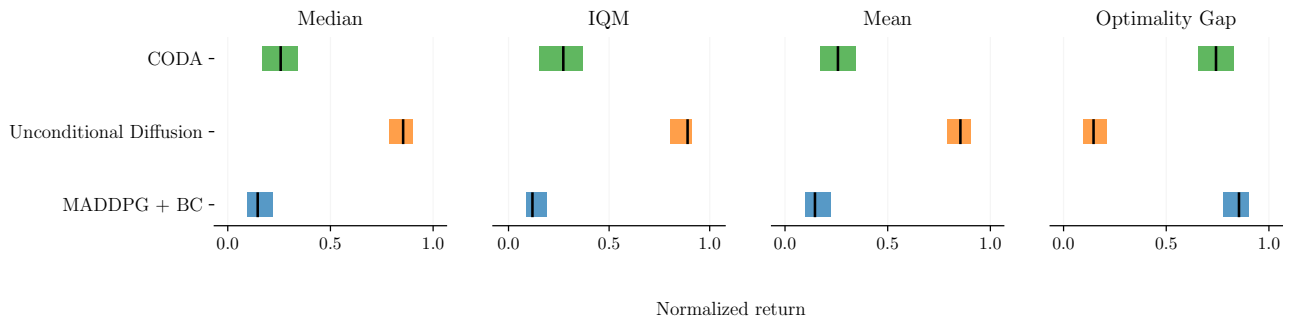

Figure 5. 2HalfCheetah, Poor Dataset, BC 2.5, CODA guidance scale 0.6

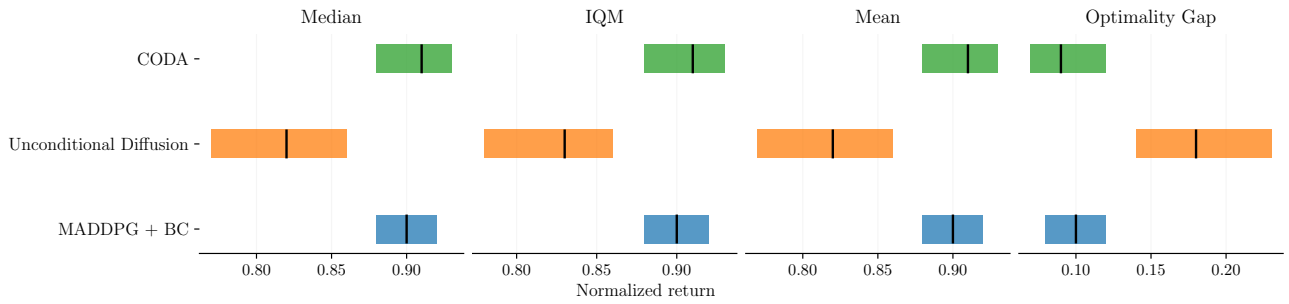

Figure 6. 2HalfCheetah, Replay Dataset, BC 2.5, CODA guidance scale 0.6

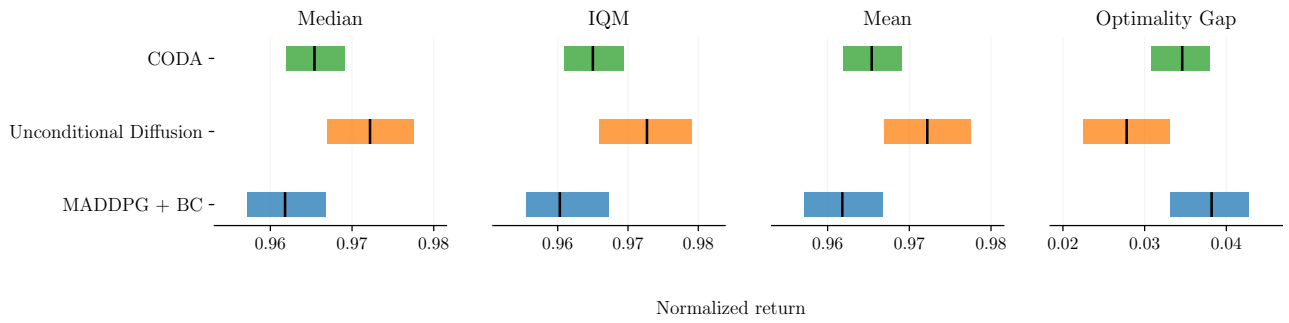

Figure 7. 2HalfCheetah, Medium Dataset, BC 2.5, CODA guidance scale 0.6

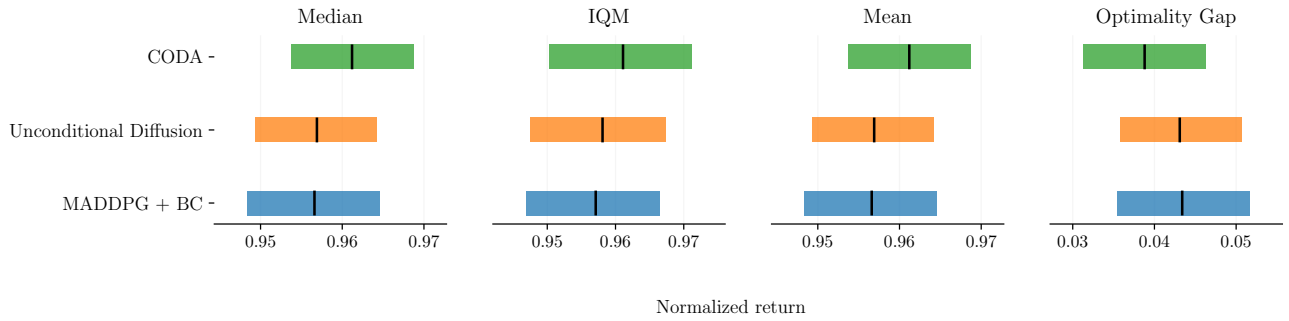

Figure 8. 2HalfCheetah, Good Dataset, BC 2.5, CODA guidance scale 0.6

E. CODA Implementation Considerations

E.1. Policy Guided Diffusion Derivations

Stabilisation Techniques. CODA inherits several stabilisation techniques from prior work:

- Cosine guidance schedules to modulate policy influence over the denoising trajectory (Jackson et al., 2024).
- Score normalisation to mitigate high-variance gradients.
- Computing the guidance gradient from the denoised samples, rather than directly on noisy samples, following best practices in classifier-guided diffusion (Bansal et al., 2024).
- Computing guidance gradients only with respect to the action components of the trajectory. Jackson et al. (2024) observe that excluding state guidance improves stability. Intuitively, adapting actions while conditioning on plausible states is natural, whereas directly shifting states can move samples away from the data manifold of realistically encountered environment states. This can produce out-of-distribution states and lead to unstable guidance.

E.2. Ensuring Diffusion Occurs in Unconstrained Space

To enable stable and well-defined diffusion dynamics, we first map all constrained variables (e.g., bounded actions, observations, rewards, and terminal indicators) into an unconstrained real space before applying the noise perturbation and denoising processes. This transformation allows the diffusion model to operate over $\mathbb{R}^n$, where Gaussian noise assumptions hold naturally. If this step is omitted, samples tend to cluster or "stick" at the domain boundaries (e.g., actions saturating at their minimum or maximum values), since Gaussian noise can easily push constrained variables outside their valid range, leading to biased gradients and degenerate dynamics near the edges. By transforming into an unconstrained space, we ensure that diffusion proceeds in a smooth, consistent domain where additive Gaussian perturbations remain valid across all sample magnitudes.

Constrained to unconstrained transformation. For any variable x bounded to $[\ell, h]$, we normalise it into the unit interval $[0, 1]$ and then apply the *logit* transform:

$$u = \mathrm{clip}\left(\frac{x-\ell}{h-\ell}, \varepsilon, 1-\varepsilon\right), \qquad z = \mathrm{logit}(u) = \log\frac{u}{1-u}, \tag{25}$$

where $\varepsilon > 0$ is a small constant (e.g., 10^{-6}) included to prevent numerical overflow at the boundaries $u \in \{0, 1\}$, where the logit function diverges. This ensures finite gradients during optimisation and avoids instability when values approach the domain limits. Note, this affine normalisation maps each variable into the unit interval $[0, 1]$ while preserving its empirical distribution shape (i.e., skewness and modality). It does not render the data uniform; it simply rescales the support to a standardised bounded domain. In practice, we actually apply a uniformisation by applying an empirical CDF transform. This removes data skew and makes diffusion more efficient in general due to isotropic diffusion dynamics. For unbounded quantities, we apply the identity mapping albeit still applying uniformisation typically.

Diffusion in $\mathbb{R}^n$. After transformation, diffusion proceeds in the unconstrained space:

$$z = z_0 + \sigma\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{26}$$

where z_0 is the clean sample, σ is the noise scale, and ϵ is Gaussian noise. The denoiser predicts $\hat{z}_0$ using the EDM preconditioning scheme, and the training loss is computed as a simple mean-squared error:

$$\mathcal{L} = \|\hat{z}_0 - z_0\|_2^2. \tag{27}$$

No Jacobian correction is required since the diffusion process and loss are defined entirely in the transformed (unconstrained) space, not in the original variable domain. We do not compute any densities which would require a Jacobian adjustment.

Inverse transformation. After sampling, the generated unconstrained outputs are mapped back to the original bounded domain using the sigmoid inverse:

$$u = \sigma(z) = \frac{1}{1 + e^{-z}}, \qquad x = \ell + (h - \ell)\, u. \tag{28}$$

For terminal indicators ("dones"), u can be interpreted as a probability in $[0, 1]$ and optionally thresholded for binary execution (Lu et al., 2023).

Summary. This two-way transformation (logit–sigmoid pair) ensures that the diffusion model operates in a mathematically unconstrained space, while respecting the physical or environmental bounds of the underlying variables. The inclusion of ε terms guarantees numerical stability near the domain boundaries without altering the effective data distribution.

E.3. Multi-agent gradients

The implication of Equation (14) is that CODA derives guidance from a joint trajectory objective while applying gradients independently to each agent's action component. This preserves decentralised policy updates while still encouraging globally coordinated samples due to the coordination signal coming from the joint signal.

F. Hyperparameters

F.1. Diffusion Models

For the base diffusion model, we follow the EDM design of Karras et al. (2022) and use a 1D temporal U-Net denoiser over full trajectory horizon. The denoiser operates on inputs of shape $[B, C, T]$, where C is the joint trajectory feature dimension (concatenating the feature dimensions of $\tau = \left(s_0, \mathbf{o}_0, \mathbf{a}_0, r_0, \ \ldots, \ s_H\right)$, and H is the horizon. This concatenation enables multi-agent information sharing across the joint actions and joint observations. Noise-level embeddings, learned positional embeddings, and conditioning embeddings are fused in a shared latent space and injected into each encoder and decoder block through learned linear projections.

Table 2. **Polynomial games: diffusion model and CFG configuration.** Diffusion architecture, conditioning setup, training noise distribution, and sampling schedule. The denoiser is a 1D temporal U-Net trained with classifier-free conditioning dropout and sampled with CFG guidance scale 1.0. Noise schedule and sigma parameterisation are based on Karras et al. (2022). Learning rate was ablated over the range 1×10^{-1} to 1×10^{-3}.

Diffusion architecture	**Value**
Denoiser backbone	1D temporal U-Net
U-Net embedding	32
U-Net blocks	2
Kernel size	3
Trajectory horizon H	1
Positional embeddings	Learned
Condition embedding dimension	32
Condition dropout	0.1
CFG guidance scale	1.0
Noise schedule (sampling)	
Schedule type	Karras (EDM)
Minimum sampling noise $\sigma_{\min}$	0.002
Maximum sampling noise $\sigma_{\max}$	80.0
Schedule curvature ρ	7.0
Diffusion steps	40
Noise distribution (training)	
Distribution	Log-normal over σ
Log-mean of training noise $\mu_{\log \sigma}$	-1.2
Log-std of training noise $\sigma_{\log \sigma}$	1.2
Training noise range	$[\sigma_{\min}, \sigma_{\max}]$
Training	
Batch size	1000
Learning rate	1×10^{-2}
Training epochs	10000
Normaliser	CDF Normaliser
Sampling	
Generated samples	1000

Table 3. **MaMuJoCo: Diffusion model and Classifier-based configuration.** The denoiser is a 1D temporal U-Net. Noise schedule and sigma parameterisation are based on Karras et al. (2022). Learning rate was ablated over the range 1×10^{-1} to 1×10^{-3}. Guidance scale is ablated over range 1×10^{-3} to 1×10^{1} to find optimal performance in each environment.

Diffusion architecture	**Value**
Denoiser backbone	1D temporal U-Net
U-Net embedding	128
U-Net blocks	4
Kernel size	3
Trajectory horizon H	20
Positional embeddings	Learned
Classifier guidance scale	[0.6, 1.0]
Noise schedule (sampling)	
Schedule type	Karras (EDM)
Minimum sampling noise $\sigma_{\min}$	0.002
Maximum sampling noise $\sigma_{\max}$	80.0
Schedule curvature ρ	7.0
Diffusion steps	128
Noise distribution (training)	
Distribution	Log-normal over σ
Log-mean of training noise $\mu_{\log \sigma}$	-1.2
Log-std of training noise $\sigma_{\log \sigma}$	1.2
Training noise range	$[\sigma_{\min}, \sigma_{\max}]$
Training	
Batch size	64
Learning rate	1×10^{-3}
Training epochs	100000
Normaliser	CDF Normaliser
Sampling	
Generated trajectories	20000

F.2. MADDPG

Table 4. **MADDPG configuration for polynomial games.** In the stateless environment setting, each agent's policy is parameterised as a single trainable action bias independent of observations. Learning rate ablated over 1×10^{-3} to 1×10^{-1}

MADDPG Agent	**Value**
Agent type	Stateless deterministic policy
Number of agents	2
Observation dependence	None (stateless environment)
Action dimension	1 per agent
Actor parameterisation	Trainable bias per agent
Policy initialisation	$[-0.64,\ 0.65]$
Action bounds	$[-1, 1]$
Critic (centralised)	
Critic usage	Not used for policy learning in stateless setting
Training	
Optimiser	SGD
Learning rate	1×10^{-1}
Gradient clipping	1.0
Training batch size	64
Burn-in iterations	2
Number of transitions sampled	4000

For the MuJoCo experiments, we use MADDPG augmented with a behaviour cloning loss (Formanek et al., 2023). Each agent is parameterised by a recurrent policy network operating on local observations, while training uses centralised state–joint-action critics and soft target network updates.

Table 5. **MaMuJoCo experiments: MADDPG + BC configuration.** Hyperparameters and network architecture used for the MuJoCo setting. Policies are parameterised by recurrent networks operating on local observations, while critics are centralised state–joint-action value networks. Behaviour cloning is incorporated with coefficient $\alpha_{\mathrm{BC}} = 2.5$. This coefficient was ablated over the range 0 to 2.5.

MuJoCo MADDPG + BC	**Value**
Discount factor γ	0.99
Target update rate τ	0.005
Critic learning rate	3×10^{-4}
Policy learning rate	3×10^{-4}
BC coefficient α_{BC}	2.5
Add agent ID to observation	True
Policy network	
Policy type	Recurrent policy network
Input	Local observations
Linear layer width	64
Recurrent hidden dimension	64
Output	Agent action
Critic network	
Critic type	Centralised state–joint-action critic
Number of critics	2
Critic inputs	State and joint action