MNEMO: POLICY LEARNING ACCELERATED BY EXPERIENCE

Xingrui Gu*
Electrical Engineering and Computer Sciences
University of California, Berkeley
xingrui_gu@berkeley.edu

Chuyi Jiang *
Department of Electrical Engineering
Columbia University
cj2792@columbia.edu

ABSTRACT

This paper introduces Mnemo, an experience-augmented operator framework for deep reinforcement learning that treats the history of state-dependent policy updates as an evlolution trace object in policy space. Instead of viewing experience only as replayed transition tuples, Mnemo logs action-level policy update signals at visited states and compresses them into a compact, state-indexed trace that biases future updates. For deterministic policies, this trace is instantiated as a similarity-gated gradient field: historical action-gradients are retrieved and smoothed across neighbouring states to produce a state-conditioned update operator. For stochastic policies, past action distributions are aggregated via a Wasserstein barycenter with a KL-type proximal anchor, yielding a data-driven region like operator on policies. We analyse the resulting operators theoretically: in the deterministic case, we show that memory fusion can reduce gradient variance under controlled bias, and in the stochastic case we prove that the Wasserstein–KL alignment operator is a pseudo-contraction that preserves the optimal Bellman fixed point. Empirically, integrating Mnemo into DDPG, TD3, PPO and mappo on MuJoCo, Box2D, Atari and SC2 benchmarks improves sample efficiency or final return on several tasks and matches the baselines elsewhere, while adding no trainable parameters and only a bounded memory overhead comparable to standard replay buffers. These results support a new learning design paradigm in which experience is operationalised as a state-indexed operator on policy updates, an action-centric trace that persistently biases future optimisation, rather than being used only as replayed data for one-step gradient estimation.

1 INTRODUCTION

Reinforcement learning (RL) provides a general framework for training agents to optimise behavioural strategies through interaction with an environment, typically formulated as a Markov Decision Process (MDP) Sutton et al. (1998); Watkins & Dayan (1992); Watkins et al. (1989); Puterman (2014); Bertsekas & Tsitsiklis (1995). Over the past decade, RL has evolved from tabular Q-learning (Watkins & Dayan, 1992) to deep RL (DRL) (Mnih et al., 2015), extending its reach to complex, high-dimensional control problems via deep neural networks (Lillicrap et al., 2015; Schulman et al., 2017). DRL agents have achieved human-level or superhuman performance in Atari and board games such as Go (Mnih et al., 2015; Silver et al., 2016), enabled dexterous robotic manipulation and locomotion (Levine et al., 2016; Gu et al., 2017; Haarnoja et al., 2018), advanced autonomous driving and traffic control (Sallab et al., 2017), and powered large-scale recommendation and dialogue systems (Chen et al., 2019; Li et al., 2016).

Despite this progress, most policy optimisation frameworks remain *myopic*: updates rely almost exclusively on immediate rewards and gradients from the current iteration (Sutton et al., 1998; Schulman et al., 2017). Such one-step lookahead overlooks how policies evolve over time, contributing to sample inefficiency, unstable convergence, and catastrophic forgetting. As argued by Silver et al. (2021), genuine reinforcement intelligence arises not only from instantaneous error correction, but

*These authors contributed equally to this work.

from learning over extended interaction, where accumulated experience persistently shapes current behaviour.

In this work we adopt a similar stance at the algorithmic level. Because what ultimately matters in RL is which action the policy takes in each state, we treat the state-dependent adjustments of this choice as the primitive units of experience, rather than raw transition tuples alone. Concretely, we maintain a compact, state-indexed trace $\mathcal{H}_t$ of policy-update signals in action space, summarising how the policy has previously been adjusted around different states. An experience operator then consults $\mathcal{H}_t$ to bias future updates, so that past interaction leaves a persistent, geometry-aware imprint on the policy gradient dynamics instead of influencing learning only through the sampling distribution of a replay buffer.

Conceptually, Mnemo differs from both replay-based heuristics and optimizer-side momentum: it does not merely change which data are sampled, nor does it only modify step sizes in parameter space. Instead, it injects *structured, state-indexed experience* into the update dynamics in action/policy space, turning past interaction into a geometry-aware prior over *how* the policy should move. This viewpoint unifies deterministic and stochastic policies under a single principle, *history-consistent yet locally anchored updates* and makes the resulting module orthogonal to the choice of optimizer and baseline algorithm. Concretely, Mnemo instantiates this principle with a bounded evolution trace $\mathcal{H}_t$ via two plug-in operators: *Mnemo-G* (similarity-gated smoothing of historical action-gradients) and *Mnemo-π* (Wasserstein barycentric alignment with a KL anchor) that stabilise actor–critic updates with no trainable parameters and preserve the optimal fixed point. We summarise our main contributions as follows:

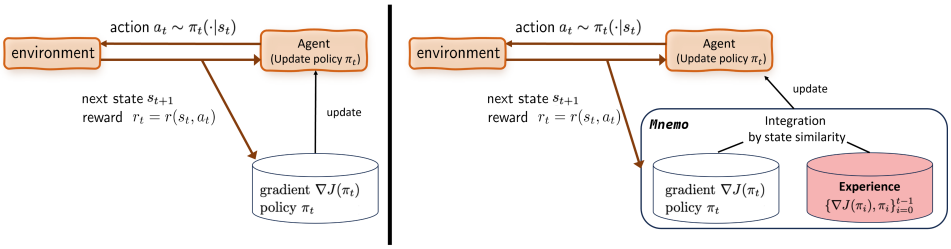

Figure 1: Standard policy-based methods (left) vs. our MnemoCore-enhanced architecture (right).

- **Experience-as-Operator.** We propose Mnemo, which converts interaction history into a *state-indexed operator* that directly reshapes policy updates in action/policy space, rather than reweighting replay data or relying on parameter, space momentum, making it plug-and-play with standard actor–critic methods and requiring no additional trainable parameters.
- **Unified geometric update rule with bounded memory.** Mnemo instantiates a single principle, *history-consistent yet locally anchored updates*, via *Mnemo-G* (similarity-gated fusion of historical action-gradients for deterministic policies) and *Mnemo-π* (Wasserstein barycentric alignment with a KL anchor for stochastic policies), using only a fixed-capacity trace $\mathcal{H}_t$ with replay-comparable overhead.
- **Theory + evidence with fixed-point preservation.** We analyze these operators jointly with Bellman/policy-gradient updates, establishing (pseudo-)contraction and non-asymptotic bounds that quantify stability–adaptation trade-offs while preserving the optimal fixed point, and empirically validate consistent gains in sample efficiency and/or final returns across DDPG/TD3/PPO/MAPPO on MuJoCo, Box2D, Atari and SC2 with ablations supporting the predicted mechanisms.

2 Related Work

2.1 Experience in deep reinforcement learning

Experience is classically incorporated into deep RL through replay buffers and short-horizon regularisation. Off-policy methods store past transitions (s, a, r, s') in a buffer and reuse them to estimate

one-step value and policy gradients (Sutton et al., 1998). Trust-region algorithms such as TRPO and PPO stabilise these gradients via local KL or clipping constraints (Schulman et al., 2015; 2017), but the underlying update operator at iteration t still depends only on the current gradient estimate computed from a minibatch: the sequence of past policy updates does not enter the update except through its effect on the current parameters.

A number of works aim to make richer use of past interaction. Methods for policy alignment and RLHF incorporate human or preference feedback (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022), but experience is still encoded primarily as additional loss terms on the current policy rather than as a temporally structured object. Memory-based and trajectory-resampling approaches such as HI-RL, SRPO and episodic control-style methods (Gu et al., 2024; Zhang et al., 2025) revisit past trajectories or policies to shape the objective, often with task-specific heuristics. Offline and batch RL methods (Wu et al., 2019; Fujimoto & Gu, 2021) rely entirely on static datasets, introducing conservative constraints to guard against distributional shift.

Across these lines of work, experience is typically treated as *data*: past interaction modifies the sampling distribution or the loss function used to compute a one-step update. What is generally absent is a view of *policy evolution itself*, for example, state-indexed records of how action choices have been adjusted over time, as an explicit object that constrains the form of subsequent update operators. This gap motivates approaches that go beyond replaying transitions or reweighting losses to consider longer-horizon structure in the sequence of policy updates.

2.2 State similarity and nonparametric structure

State similarity has been widely studied as a tool for abstraction and sample-efficient learning. Bisimulation metrics and Kantorovich distances (Ferns et al., 2004; 2012) provide strong guarantees by grouping states with equivalent value and transition structure, but computing these metrics can be prohibitive in large or continuous domains and sensitive to model misspecification. Scalable relaxations and learned metrics (Castro, 2020; Pavse & Hanna, 2023) improve tractability and adaptivity by parameterising similarity, yet typically require additional optimisation objectives and regularisation to remain stable. Trajectory-chain and pseudometric approaches (Girgin et al., 2007; Dadashi et al., 2021) capture finer behavioural notions of similarity, but often at the cost of increased storage or auxiliary computations.

Most of these methods define similarity in terms of values, dynamics, or trajectory statistics, and deploy it for state aggregation, exploration, or offline evaluation. They rarely act directly on the *policy update* mechanism itself, for instance, by using similarity to decide which past action-level update signals should influence a new update at a given state. Consequently, the potential of state similarity as a way to organise and reuse policy-update experience, rather than only state occupancy or returns, remains relatively unexplored.

3 Preliminaries

A reinforcement learning (RL) problem is typically formalised as a Markov Decision Process (MDP) (Sutton et al., 1998; Puterman, 2014), denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$. Here, $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $P(s' \mid s, a)$ represents the transition probability of moving to state $s' \in \mathcal{S}$ after taking action $a \in \mathcal{A}$ based on current state $s \in \mathcal{S}$, $r(s, a)$ is the immediate reward for (s, a), and $\gamma \in (0, 1)$ is a discount factor controlling the weight of future rewards. We denote π_θ as a policy (action selection rule) parameterized by θ, namely $\pi_\theta(a \mid s)$ denote the chosen action for a deterministic policy (the probability distribution of choosing action a) conditioned on the state s, for any $(s, a) \in \mathcal{S} \times \mathcal{A}$.

The objective is to maximize the expected discounted return $J(\theta)$ as below:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta, P}\left[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t)\right] \tag{1}$$

where $\tau = (s_0, a_0, s_1, a_1, \cdots)$ denotes a trajectory starting from some initial state distribution, generated by following the policy π_θ with transitions governed by $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ at each time step t. In addition, $Q_\theta(s, a) := \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t) \mid s_0 = s, a_0 = a]$ represents the *state-*

action value function starting from state s and taking action a under policy π_θ, similarly, $V_\theta(s) := \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t) \mid s_0 = s]$ denotes the *state value function* starting from state $s \in \mathcal{S}$.

Policy gradient and actor-critic. Throughout the paper, we denote $\nabla_x f(\cdot)$ as the gradient of the function f with respect to the variable x. For both stochastic and deterministic policy, there are standard policy gradient and actor-critic algorithms for them. Here we introduce selective basics of them which are useful throughout the paper.

A deterministic policy π_θ means for any state $s \in \mathcal{S}$, there is only one selected action $a = \pi_\theta(s) \in \mathcal{A}$. To learn the optimal deterministic policy, the *deterministic actor-critic algorithm* has the following update rule for the policy (actor) (Silver et al., 2014) at any iteration t:

$$\nabla_\theta J(\theta_t) \approx \mathbb{E}_{s\sim\mathcal{B}_t}\Big[\nabla_\theta \pi_{\theta_t}(s)\, \nabla_a Q(s,a)\big|_{a=\pi_{\theta_t}(s)}\Big] \tag{2}$$

where $\mathcal{B}_t = \{s_i, a_i, r_i, s_{i+1}\}_{i=0}^{t-1}$ denotes the replay buffer until time step t.

While to learn a stochastic policy $\pi_\theta \in \Delta(\mathcal{A})$ (a probability distribution over action space $\mathcal{A}$), the *stochastic policy gradient algorithm* (Sutton et al., 1999; Schulman et al., 2015) executes the following update at time step t:

$$\nabla_\theta J(\theta_t) \approx \mathbb{E}_{s,a\sim\pi_{\theta_t}}\Big[\nabla_\theta \log \pi_{\theta_t}(a \mid s)\, \hat{A}_{\theta_t}(s,a)\Big] \tag{3}$$

where $\hat{A}_{\theta_t}(s,a) = Q_{\theta_t}(s,a) - V_{\theta_t}(s)$ is an estimate of the advantage function.

4 Experience-Augmented Policy Optimization

Standard policy optimisation can be written as

$$\pi_{t+1} = \mathcal{U}_t(\pi_t; \mathcal{T}), \tag{4}$$

where π_t is the current policy, $\mathcal{T}$ is an RL operator (e.g. Bellman or policy gradient), and $\mathcal{U}_t$ a one-step update that depends only on π_t and fresh data. This *myopic* rule ignores how the policy has evolved over time.

Mnemo instead introduces an evolution-informed policy operator that reshapes this update using a compact trace of past evolution:

$$\pi_{t+1} = \mathcal{M}_t\big(\mathcal{H}_t, \mathcal{U}_t(\pi_t; \mathcal{T})\big), \tag{5}$$

where $\mathcal{H}_t$ is a bounded capacity trace of policy updates up to iteration t and $\mathcal{M}_t$ acts directly in policy space.

Definition 4.1 (Policy evolution trace)**.** At optimisation iteration t, we log an evolution sample

$$\mathcal{E}_t = (s_t, \xi_t), \tag{6}$$

where s_t is a visited state and ξ_t encodes policy-update information (e.g. an action-gradient $\nabla_a Q_\phi(s_t, a_t)$ or a policy distribution $\pi_{\theta_t}(\cdot \mid s_t)$). The evolution trace is $\mathcal{H}_t = \{\mathcal{E}_i\}_{i\le t}$, stored with a fixed capacity in practice.

Formally, $\mathcal{H}_t$ is an $\mathcal{F}_t$-measurable functional of the interaction history $\mathbf{e}_{0:t}$ as shown in Appendix D, where each element $\mathcal{E}_i = (s_i, \xi_i)$ encodes a visited state and an action-related policy-update signal computed from past data. Rather than an auxiliary optimiser state or an arbitrary memory buffer, $\mathcal{H}_t$ constitutes an action-centric experience trace over policy evolution. At update time, Mnemo queries this trace with the current state and retrieves relevant past signals via a state similarity kernel, using the retrieved summary as a prior to bias the next policy update.

Mnemo instantiates this principle by constructing operators from $\mathcal{H}_t$ that integrate seamlessly into standard RL frameworks; Algorithm 1 provides a general example of how such operators are applied. We specify two instantiations: a gradient-field operator for deterministic policies and a Wasserstein–KL alignment operator for stochastic policies.

Algorithm 1 General Markov Decision Process (MDP) with Mnemo

1: Initialize policy π_{θ_0}
2: **for** Steps $t = 1$ to T **do**
3: Sample action $a_t \sim \pi_{\theta_t}(\cdot \mid s_t)$
4: Execute a_t, observe reward r_t and next state s_{t+1}
5: Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$ and evolution sample $\mathcal{E}_t = (s_t, \xi_t)$ in $\mathcal{H}$ (Eq. 6)
6: Update policy $\pi_{\theta_{t+1}} = \mathcal{M}_t\big(\mathcal{H}_t, \mathcal{U}_t(\pi_{\theta_t}; \mathcal{T})\big)$(Eq. 5)
7: **end for**

4.1 MNEMO-G: GRADIENT-FIELD OPERATOR FOR DETERMINISTIC POLICIES

Recall Definition 4.1, where each memory item is $\mathcal{E}_t = (s_t, \xi_t)$. For MNEMO-G, we instantiate the policy-update signal as the deterministic action-gradient,

$$\xi_t \;:=\; \nabla_t \;:=\; \nabla_a Q_\phi(s_t, a)\big|_{a=\pi_{\theta_t}(s_t)} \in \mathbb{R}^{\dim(\mathcal{A})}. \tag{7}$$

At a high level, MNEMO-G augments deterministic actor-critic methods (e.g., DDPG, TD3) by retrieving state-relevant historical action-gradients from a gradient memory, fusing them with the instantaneous gradient to obtain a smoothed update direction, and using this fused direction to construct a perturbed action for the actor update. Concretely, at optimisation iteration t we estimate the deterministic policy-gradient signal:

$$\nabla_t := \nabla_a Q_\phi(s_t, a)\big|a = \pi_{\theta_t}(s_t) \in \mathbb{R}^{\dim(\mathcal{A})}, \tag{8}$$

and treat the collection of state-conditioned gradients as a nonparametric *gradient field*: each pair (s_t, ∇_t) is stored in a gradient memory $\mathcal{H}$ and later queried at similar states to bias subsequent updates.

Given the gradient memory $\mathcal{H}_t = \{(s_i, \nabla_i)\}_{i=1}^t$, MNEMO-G queries $\mathcal{H}_t$ with the current state s_t and retrieves a state-conditioned historical direction ∇_t^{hist} by aggregating past action-gradients with relevance weights computed by the similarity mechanism described in Section F. This retrieved direction is then fused with the instantaneous action-gradient ∇_t to form a smoothed update signal:

$$\nabla_t^{\text{all}} = \beta_t \, \nabla_{t-1}^{\text{all}} + (1-\beta_t) \, \nabla_t, \qquad \nabla_0^{\text{all}} := 0, \tag{9}$$

where the decay weight β_t is adapted based on the agreement between the retrieved and instantaneous directions at s_t:

$$\beta_t = \text{clip}\left(\sigma\Big(\alpha \frac{\langle \nabla_t^{\text{hist}}, \nabla_t \rangle}{\|\nabla_t^{\text{hist}}\| \cdot \|\nabla_t\| + \epsilon}\Big), \epsilon, 1-\epsilon\right). \tag{10}$$

Intuitively, β_t increases when memory and instant feedback agree, promoting stability via stronger reuse, and decreases when they disagree, allowing rapid adaptation to new local signals.

With the integrated policy-update ∇_t^{all} in hand, we can know that. Towards this, the updated chosen action is constructed as

$$a_{\text{pert}} = \pi_{\theta_t}(s_t) + \frac{\lambda_t \nabla_t^{\text{all}}}{\|\nabla_t^{\text{all}}\| + \epsilon}, \tag{11}$$

with the update magnitude λ_t increasing as $\lambda_t = \lambda_{\min} + \frac{t}{T_{\max}} (\lambda_{\max} - \lambda_{\min})$. Here, $\lambda_{\max}$ and $\lambda_{\min}$ are some positive threshold, and $T_{\max}$ denote the maximum iteration numbers. Finally, we can plug the experience-augmented into the original policy update rule of deterministic actor critic:

$$\theta_{t+1} = \theta_t + \eta_t \Big[\nabla_\theta \pi_{\theta_t}(s) \nabla_a Q(s, a_{\text{pert}})\Big]. \tag{12}$$

With capacity M, MNEMO-G stores tuples (s_i, ∇_i) with $\nabla_i \in \mathbb{R}^{d_a}$, incurring $O(M(d_s + d_a))$ memory, where d_s and d_a is the dimension of state and action space, where d_s and d_a are the state and action dimensions, and does not store full network parameters or parameter-gradients of size $O(\dim(\theta))$.

4.2 POLICY MEMORY FOR STOCHASTIC POLICIES

Recall Definition 4.1, where each memory item is $\mathcal{E}_t = (s_t, \xi_t)$. For stochastic policies, we instantiate

$$\xi_t \;:=\; \pi_{\theta_t}(\cdot \mid s_t) \in \mathcal{P}(\mathcal{A}), \tag{13}$$

i.e., the local action distribution induced by the current policy at the visited state. The resulting trace is $\mathcal{H}_t = \{(s_i, \pi_{\theta_i}(\cdot \mid s_i))\}_{i \le t}$ (stored with fixed capacity in practice).

Given the query state s_t, we retrieve a subset of relevant items from $_t$ and form a weighted aggregate of their action distributions,

$$\pi_t^{\text{hist}}(\cdot \mid s_t) \;:=\; \sum_{i \in \mathcal{I}_t(s_t)} w_i(s_t)\, \pi_{\theta_i}(\cdot \mid s_i), \qquad \sum_{i \in \mathcal{I}_t(s_t)} w_i(s_t) = 1, \tag{14}$$

where the relevance weights $w_i(s_t)$ are computed by the state-similarity retrieval mechanism defined in Section F (or Eq. equation 58). Equation equation 14 defines a state-conditioned *distributional memory* in policy space.

To integrate the distributional memory with the current policy at s_t, Mnemo computes a conservative alignment that stays close to the current policy while moving toward the historical aggregate in Wasserstein geometry:

$$\pi_t^{\text{align}}(\cdot \mid s_t) \in \arg\min_{\pi \in \mathcal{P}(\mathcal{A})} \Big\{ W_2^2\big(\pi, \pi_t^{\text{hist}}(\cdot \mid s_t)\big) + \rho\, \mathrm{KL}\big(\pi \,\|\, \pi_{\theta_t}(\cdot \mid s_t)\big) \Big\}. \tag{15}$$

In practice, the Wasserstein term is approximated by entropic optimal transport computed on samples. For a finite action space, given a cost matrix $C \in \mathbb{R}_+^{|\mathcal{A}| \times |\mathcal{A}|}$, we compute the entropy-regularised transport plan

$$P^\star = \arg\min_{P \in \mathcal{U}(\pi_{\theta_t}, \pi_t^{\text{hist}})} \langle C, P \rangle + \varepsilon H(P), \tag{16}$$

where $\langle C, P \rangle := \sum_{i,j} C_{ij} P_{ij}$ denotes the Frobenius inner product, $H(P)$ is the entropy of P, $\varepsilon > 0$ is the entropic regularisation strength, and $\mathcal{U}(\mu, \nu) = \{P \ge 0 : P\mathbf{1} = \mu,\ P^\top \mathbf{1} = \nu\}$ is the transport polytope, with $\mathbf{1}$ denoting a vector of ones of appropriate dimension. The same computation extends to continuous actions by using samples from $\pi_{\theta_t}(\cdot \mid s_t)$ and $\pi_t^{\text{hist}}(\cdot \mid s_t)$ and applying Sinkhorn in the sampled space.

We then use $\pi_t^{\text{align}}(\cdot \mid s_t)$ as a memory-informed reference in standard stochastic policy optimisation. For PPO, this yields the clipped surrogate objective

$$\max_\theta L(\theta) = \mathbb{E}_{\mathcal{B}_t}\left[\min\left(\frac{\pi_t^{\text{hist}}(a_t \mid s_t)}{\pi_{\theta_t}(a_t \mid s_t)} \hat{A}(s_t, a_t),\ \text{clip}\left(\frac{\pi_t^{\text{hist}}(a_t \mid s_t)}{\pi_{\theta_t}(a_t \mid s_t)},\ 1-\epsilon,\ 1+\epsilon \right) \hat{A}(s_t, a_t) \right) \right]. \tag{17}$$

where π_t^{align} plays the role of a conservative, experience-shaped anchor distribution.

For MNEMO-π, the stored signal is a local distribution representation (e.g. Gaussian mean/variance in $\mathbb{R}^{d_a}$), where d_s and d_a are the state and action dimensions, yielding the same $O(M(d_s + d_a))$ scaling.

5 EXPERIMENTS AND EVALUATION

In this section, we evaluate Mnemo by integrating it into representative reinforcement learning (RL) and multi-agent reinforcement learning (MARL) algorithms, including deterministic-policy methods DDPG and TD3, as well as stochastic-policy methods PPO and MAPPO. All algorithms are implemented in the high-performance reinforcement learning framework `Xuance` (Liu et al., 2023). For a fair comparison, all methods use the same optimizer; Mnemo is added as an experience operator on top of the baseline updates. This setup allows us to assess their generality and performance gains across distinct policy optimization paradigms. Specifically, DDPG, a deterministic policy gradient method sensitive to gradient noise, examines improvements in stability; TD3, which extends DDPG with delayed updates and target smoothing, evaluates convergence smoothness in

high-precision control; PPO and MAPPO, a stochastic policy gradient method with probabilistic updates, highlights advantages in distributional smoothing and long-term dependency exploitation. For controlled comparison, we run both original and Mnemo-augmented variants under identical hyperparameters across MuJoCo, Box2D and Starcraft 2 tasks, measuring final performance, convergence speed, and variance evolution. Results are reported as Average Cumulative Rewards, averaged over three random seeds (123, 321, 666).

5.1 Experiments of Deterministic Policies

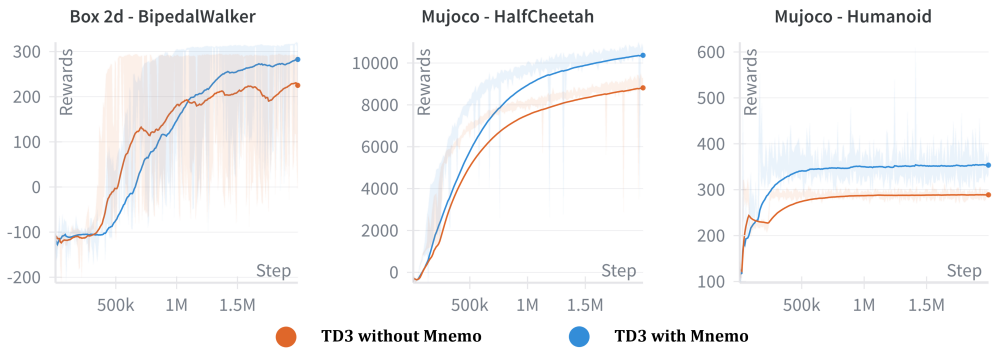

Figure 2: Learning curves of TD3 with and without the proposed Mnemo across three continuous control tasks.

Table 1: Performance comparison across continuous control environments for deterministic policy optimization (DDPG and TD3) with and without Mnemo (mean ± std over runs). Best means are in bold.

Environment	DDPG	DDPG + Mnemo
MuJoCo – Hopper	1089.81 ± 261.51	$\mathbf{1143.87 \pm 639.81}$
MuJoCo – HumanoidStandup	10128.78 ± 1508.89	$\mathbf{10308.15 \pm 1344.99}$
MuJoCo – Ant	2.06 ± 146.52	$\mathbf{197.08 \pm 277.99}$
MuJoCo – Walker2d	1331.47 ± 1460.30	$\mathbf{1654.58 \pm 1657.74}$
Box2D – BipedalWalker	-92.64 ± 12.42	$\mathbf{-37.61 \pm 275.35}$
Environment	**TD3**	**TD3 + Mnemo**
MuJoCo – Hopper	3007.61 ± 682.26	$\mathbf{3089.00 \pm 748.61}$
MuJoCo – Humanoid	289.02 ± 3.67	$\mathbf{322.34 \pm 2.48}$
MuJoCo – Ant	2595.91 ± 1159.40	$\mathbf{4194.72 \pm 1568.33}$
MuJoCo – HalfCheetah	8813.85 ± 393.89	$\mathbf{10367.48 \pm 331.46}$
Box2D – BipedalWalker	225.21 ± 37.82	$\mathbf{282.64 \pm 71.73}$

To evaluate the effectiveness of Mnemo in deterministic policy optimization, we integrate it into two representative baselines, Deep Deterministic Policy Gradient (DDPG) and Twin Delayed DDPG (TD3), and conduct experiments using the `Xuance` framework. Benchmarks cover a diverse set of MuJoCo and Box2D continuous control tasks, including high dimensional locomotion (Humanoid, Ant), high-speed cyclic dynamics (HalfCheetah, Hopper), and challenging contact-rich environments with sparse rewards (BipedalWalker). Across all tasks, Mnemo-augmented agents consistently outperform their vanilla counterparts in both mean return and training stability, as summarised in Table 1. Beyond overall gains, the improvements exhibit environment-specific patterns:in unstable high-dimensional environments, such as Humanoid, Mnemo converges more quickly to higher performance in the early stages and achieves superior asymptotic performance ; in relatively stable locomotion tasks, such as HalfCheetah, gains gradually accumulate in the mid-to-late stages of

training, resulting in final returns approximately 10% higher than the baseline ; in contact-dominant BipedalWalker, Mnemo enables faster recovery from early failures and converges to superior policies at last. These results empirically substantiate our theoretical claim: by fusing attenuated historical gradients with new updates, Mnemo mitigates variance, dampens policy oscillations, and improves global convergence in deterministic policy optimization.

5.2 Experiments of Stochastic Policies

We evaluate Proximal Policy Optimization (PPO) with and without Mnemo across MuJoCo, Box2D, and Atari benchmarks. As reported in Table 2 in Appendix and illustrated in Figure 4, incorporating Mnemo yields consistent improvements in both learning dynamics and final outcomes. In environments with sparse rewards or intricate dynamics, it accelerates the transition from initial exploration to effective policy improvement, substantially reducing instability during early training. In addition, across tasks characterized by high-dimensional state spaces or volatile gradient landscapes, Mnemo achieves higher asymptotic returns and markedly lower variance across trials. These results demonstrate that Mnemo extends policy optimization from the parameter space to the distributional measure space, providing a stabilizing mechanism that balances behavioural consistency with adaptability in diverse control and decision-making domains.

Motivated by the strong empirical performance of PPO with Mnemo, we further evaluate whether Mnemo can be effectively extended to multi-agent reinforcement learning (MARL) settings. Building upon the architectural design of PPO, we directly apply the overall Mnemo framework to MAPPO. We then assess the proposed method on StarCraft II benchmarks, with the experimental results presented in Figure 3. By evaluating the adversarial win rates among agents during training, our algorithm is able to overcome the suboptimal convergence of traditional methods and achieve superior performance. During training, we observed that incorporating the Mnemo framework increases the entropy of the learning process, further encouraging agents to explore the environment and escape suboptimal solutions.

Table 2: Performance comparison across continuous control and Atari environments for stochastic policy optimization (PPO) with and without Mnemo (mean $\pm$ std over runs). Best means are in bold. Training budgets: Mujoco/Box2D = 2M steps, Atari = 30M frames.

Environment	**PPO**	**PPO + Mnemo**
	Mujoco (2M steps)	
Mujoco - Walker2d	3209.14 ± 1374.56	$\mathbf{4586.33 \pm 2110.27}$
Mujoco - Reacher	-5.52 ± 0.58	$\mathbf{-5.26 \pm 1.53}$
Mujoco - Swimmer	92.29 ± 0.87	$\mathbf{111.10 \pm 5.88}$
Mujoco - HalfCheetah	5052.39 ± 1999.43	$\mathbf{6659.97 \pm 1316.39}$
Mujoco - HumanoidStandup	58355.59 ± 2351.22	$\mathbf{117143.71 \pm 48593.62}$
Mujoco - Humanoid	311.98 ± 28.98	$\mathbf{444.68 \pm 62.63}$
Mujoco - Ant	2250.26 ± 270.81	$\mathbf{2667.34 \pm 297.72}$
Mujoco - Hopper	1851.90 ± 1522.27	$\mathbf{2378.05 \pm 862.16}$
	Box2D (2M steps)	
Box2D - BipedalWalker	95.27 ± 78.52	$\mathbf{264.86 \pm 36.13}$
	Atari (30M steps)	
Atari - Air Raid	7147.20975 ± 2829.97	$\mathbf{7818.44865 \pm 4187.28}$
Atari - Alien	1468.51 ± 278.25	$\mathbf{2256.74 \pm 672.02}$
Atari - Amidar	1135.31636 ± 192.64	$\mathbf{1382.9635 \pm 584.51}$
Atari - Assault	8761.00 ± 2956.74	$\mathbf{9773.18 \pm 2737.39}$
Atari - Asterix	4794.55998 ± 1820.52	$\mathbf{5156.96704 \pm 2033.43}$
Atari - Breakout	291.36 ± 105.37	$\mathbf{314.32 \pm 94.85}$
Atari - Space Invader	908.11 ± 153.75	$\mathbf{1273.56 \pm 496.69}$
Atari - Star Gunner	46474.40 ± 15348.79	$\mathbf{52526.61 \pm 11053.73}$

6 CONCLUSION

In this work, we present Mnemo, which encodes the policy evolution history as a structured prior that can be integrated into standard policy optimization. In deterministic policy, it takes the form of gradient memory, using a recursive decay–injection–normalization scheme to compress and reuse historical gradients. In stochastic policy, it performs distributional alignment in Wasserstein space, combining state-conditioned retrieval with KL-regularized barycentric interpolation, yielding closed-form updates for diagonal Gaussians and Sinkhorn-based support for general cases. Theoretically, we show that these memory-augmented updates are (quasi-)contractive in W_2 space and jointly convergent with the Bellman operator, enabling the entire policy evolution to be approximated using a fixed amount of memory without compromising optimality. In empirical evaluations, Mnemo integrates seamlessly into both deterministic and stochastic policy optimization algorithms, achieving earlier stable improvement, higher final returns, and lower variance on benchmark environments including MuJoCo, Box2D, and Atari, with negligible additional computational and hyperparameter tuning overhead. Moreover, by extending PPO to MAPPO, we demonstrate that Mnemo also attains strong performance in the StarCraft II environment, further highlighting its compatibility and robust adaptability across different optimization paradigms and task domains. These results position Mnemo as a promising core primitive for experience-driven lifelong optimization.

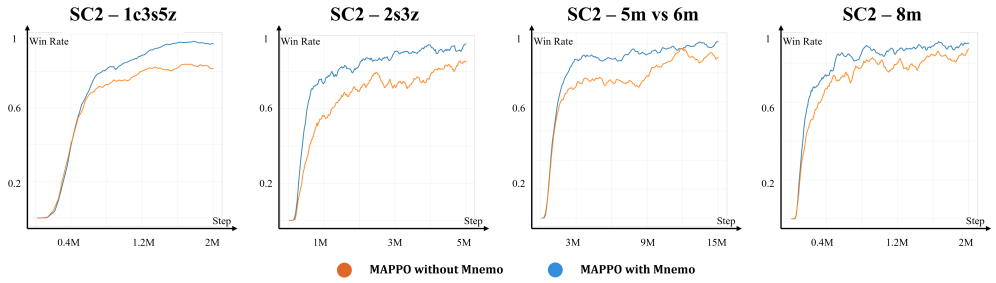

Figure 3: Learning curves of Multi-Agent Proximal Policy Optimization (MAPPO) with and without the proposed Mnemo framework on representative StarCraft II (SC2) tasks, illustrating how Mnemo enables agents to escape suboptimal solutions and achieve higher asymptotic performance.

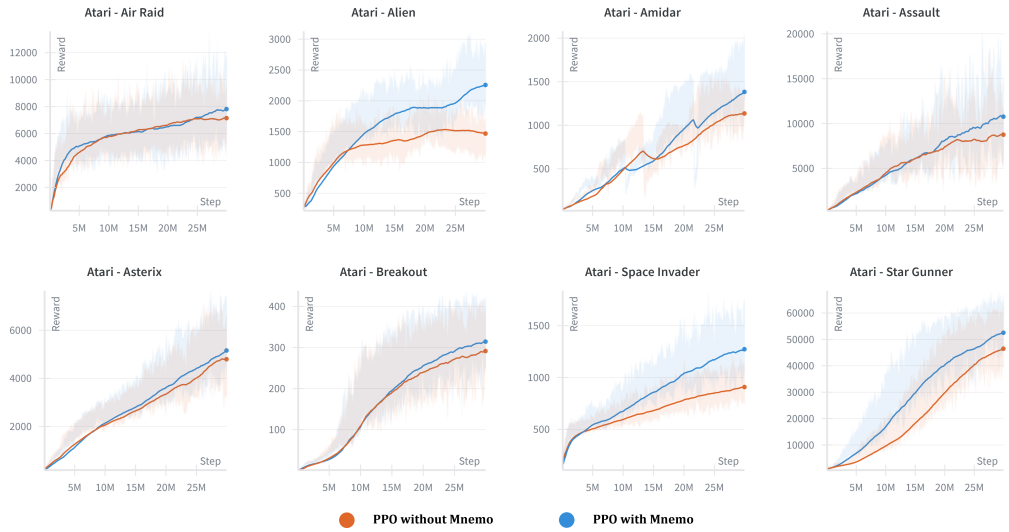

Figure 4: Learning curves of Proximal Policy Optimization with and without the proposed Mnemo across representative Atari games, illustrating how the operator accelerates early-stage learning, reduces training variance, and achieves higher asymptotic performance.

REFERENCES

Yuntao Bai, Saurav Kadavath, Amanda Askell, et al. Training a helpful and harmless assistant with rlhf. *arXiv preprint arXiv:2204.05862*, 2022.

Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *Proceedings of 1995 34th IEEE conference on decision and control*, volume 1, pp. 560–564. IEEE, 1995.

Colin Camerer and Teck Hua Ho. Experience-weighted attraction learning in normal form games. *Econometrica*, 67(4):827–874, 1999.

Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 10069–10076, 2020.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pp. 456–464, 2019.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Robert Dadashi, Shideh Rezaeifar, Nino Vieillard, Léonard Hussenot, Olivier Pietquin, and Matthieu Geist. Offline reinforcement learning with pseudometric learning. In *International Conference on Machine Learning*, pp. 2307–2318. PMLR, 2021.

Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pp. 162–169, 2004.

Norman Ferns, Prakash Panangaden, and Doina Precup. Metrics for markov decision processes with infinite state spaces. *arXiv preprint arXiv:1207.1386*, 2012.

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

Samuel J Gershman and Nathaniel D Daw. Reinforcement learning and episodic memory in humans and animals: an integrative framework. *Annual review of psychology*, 68(1):101–128, 2017.

Sertan Girgin, Faruk Polat, and Reda Alhajj. State similarity based approach for improving performance in rl. In *IJCAI*, volume 7, pp. 817–822, 2007.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396. IEEE, 2017.

Xingrui Gu, Guanren Qiao, and Chuyi Jiang. Mimicking human intuition: Cognitive belief-driven reinforcement learning. In *2nd Workshop on Models of Human Feedback for AI Alignment*, 2024.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Wenzhang Liu, Wenzhe Cai, Kun Jiang, Guangran Cheng, Yuanda Wang, Jiawei Wang, Jingyu Cao, Lele Xu, Chaoxu Mu, and Changyin Sun. Xuance: A comprehensive and unified deep reinforcement learning library. *arXiv preprint arXiv:2312.16248*, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Brahma Pavse and Josiah Hanna. State-action similarity-based representations for off-policy evaluation. *Advances in Neural Information Processing Systems*, 36:42298–42329, 2023.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *arXiv preprint arXiv:1704.02532*, 2017.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

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

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pp. 387–395. Pmlr, 2014.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artificial intelligence*, 299:103535, 2021.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

Christopher John Cornish Hellaby Watkins et al. Learning from delayed rewards. 1989.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, et al. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm. *arXiv preprint arXiv:2504.14286*, 2025.

A PSEUDOCODE

Algorithm 2 Deterministic policy optimisation with Mnemo (gradient memory)

Hyperparameters: update period d
Initialise actor π_θ, critic Q_ϕ, target networks $\theta' \leftarrow \theta$, $\phi' \leftarrow \phi$
Initialise replay buffer $\mathcal{B}$ and gradient memory $\mathcal{H}$
for environment steps $t = 1$ to T **do**
  Observe s_t, select $a_t = \pi_\theta(s_t) + \epsilon_t$, $\epsilon_t \sim \mathcal{N}(0, \sigma)$
  Execute a_t, receive (r_t, s_{t+1}), store (s_t, a_t, r_t, s_{t+1}) into $\mathcal{B}$
  Sample mini-batch $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^N \sim \mathcal{B}$
  $y_i \leftarrow r_i + \gamma Q_{\phi'}(s'_i, \pi_{\theta'}(s'_i))$
  $\phi \leftarrow \phi - \eta_\phi \nabla_\phi \frac{1}{N} \sum_i \big(Q_\phi(s_i, a_i) - y_i\big)^2$
  if $k \bmod d = 0$ **then** ▷ actor / memory update
    Let t be the current actor-update index
    Compute instantaneous action-gradient $\nabla_t \leftarrow \nabla_a Q_\phi(s_t, a)|_{a=\pi_{\theta_t}(s_t)}$ (Eq. equation 8)
    Retrieve historical gradient ∇_t^{hist} from $\mathcal{H}$
    Fuse ∇_t and ∇_t^{hist} into ∇_t^{all} (Eq. equation 9)
    Construct perturbed action a_{pert} (Eq. equation 11)
    Update actor parameters θ_t using a_{pert} (Eq. equation 12)
    $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$, $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$
    Store (s_t, ∇_t) into $\mathcal{H}$ (with fixed capacity)
  end if
end for

B CONVERGENCE ANALYSIS OF MNEMO WITH GRADIENT EXPERIENCE

This appendix consolidates the assumptions, auxiliary lemmas, and a full convergence proof in the main text. We use the short-hands $g_t := \nabla J(\theta_t)$, $\hat{g}_t := \nabla_t$ for a (possibly stochastic) gradient estimator with $\mathbb{E}[\hat{g}_t \mid \theta_t] = g_t$ and $\mathbb{E}\big[\|\hat{g}_t - g_t\|^2 \mid \theta_t\big] \le \sigma^2$, and $\bar{g}_t := \bar{\nabla}_{\text{hist},t}$ for the historical (memory) estimator. The memory-fused search direction is

$$d_t \;=\; (1-\beta)\,\hat{g}_t \;+\; \beta\,\bar{g}_t, \qquad \beta \in [0,1].$$

The iterate update is $\theta_{t+1} = \theta_t + \eta\, d_t$ with stepsize $\eta > 0$.

B.1 ASSUMPTIONS

(A1) **L-smoothness and bounded gradients:** J is L-smooth: $J(\theta') \le J(\theta) + \langle \nabla J(\theta), \theta' - \theta\rangle + \frac{L}{2}\|\theta' - \theta\|^2$. Moreover $\|g_t\| \le G$ for all t.

(A2) **Historical estimator bias/variance:** There exist sequences $B_t \ge 0$, $\kappa \in [0,1)$ and $\rho_t \in [-1,1]$ such that

$$\|\mathbb{E}[\bar{g}_t \mid \theta_t] - g_t\| \;\le\; B_t, \qquad \mathbb{E}\big[\|\bar{g}_t - \mathbb{E}[\bar{g}_t \mid \theta_t]\|^2 \big| \theta_t\big] \;\le\; \kappa\,\sigma^2,$$

and the correlation coefficient between zero-mean noises $\xi_t := \hat{g}_t - g_t$ and $\zeta_t := \bar{g}_t - \mathbb{E}[\bar{g}_t \mid \theta_t]$ satisfies $\mathrm{Corr}(\xi_t, \zeta_t) = \rho_t$.

(A3) **State-discrepancy induced bias (optional decomposition):** If $\bar{g}_t$ is computed at states s' while g_t is at s, and the empirical Lipschitz constant in state is L_{emp}, then

$$B_t \;\le\; \sqrt{\mathrm{Var}(\bar{g}_t)} + L_{\text{emp}}\delta_s \quad \text{with} \quad \delta_s := \|s' - s\|.$$

(This recovers the B you used in the main text as a convenient bound.)

B.2 AUXILIARY VARIANCE AND SECOND-MOMENT BOUNDS

Lemma B.1 (Exact variance factor under correlation)**.** *Let κ and ρ_t be as in (A2). Then, for each coordinate (and hence for the vector under isotropic scaling),*

$$\frac{\mathrm{Var}((1-\beta)\hat{g}_t + \beta\,\bar{g}_t)}{\mathrm{Var}(\hat{g}_t)} \;=\; (1-\beta)^2 \;+\; \beta^2\,\kappa \;+\; 2\beta(1-\beta)\rho_t\sqrt{\kappa} \;\in\; \big[(1-\beta-\beta\sqrt{\kappa})^2,\; 1\big].$$

Algorithm 3 Stochastic Policy Optimization with Mnemo (Distributional Memory)

Hyper: period d; neighbor count k; sim. mix $\alpha \in (0,1)$; Sinkhorn reg. ε; Gaussian samples n
Initialize stochastic policy $\pi_\theta(\cdot|s)$ with `dist_type` $\in \{$`gaussian`, `categorical`$\}$
Initialize replay buffer $\mathcal{B}$ and distribution memory $\mathcal{H}$
for $t = 1$ to T **do**
  Observe s_t, take $a_t \sim \pi_\theta(\cdot|s_t)$, receive (r_t, s_{t+1}); store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$
  Extract current params θ_t from $\pi_\theta(\cdot|s_t)$:
  if Gaussian **then**
    $\theta_t = (\mu_t, \sigma_t)$
  else
    $\theta_t = p_t$
  end if
  Push (s_t, θ_t) into $\mathcal{H}$ and update $\mathcal{L}_{\mathrm{RL}}(\theta)$ (e.g., PPO)
  if $t \bmod d = 0$ **then** ▷ distributional memory coupling
    Retrieve: compute similarity $sim(s_t, s_i)$ for $(s_i, \theta_i) \in \mathcal{H}_\pi$; pick top-$k$ neighbors $\mathcal{N}_k$
    Weight: $w_i = \dfrac{\exp(-(1 - sim(s_t, s_i))/\tau)}{\sum_{j \in \mathcal{N}_k} \exp(-(1 - sim(s_t, s_j))/\tau)}$
    OT (Sinkhorn):
    if gaussian **then**
      $\mu_{\mathrm{hist}} = \sum w_i \mu_i,\ \sigma_{\mathrm{hist}} = \sum w_i \sigma_i,\ \pi_{\mathrm{hist}} = \mathcal{N}(\mu_{\mathrm{hist}}, \mathrm{diag}(\sigma^2_{\mathrm{hist}}))$;
      sample $\{a_p\}_{p=1}^n \sim \pi_t$ and $\{b_q\}_{q=1}^n \sim \pi_{\mathrm{hist}}$, set $C_{pq} = \|a_p - b_q\|_2^2$;
    else
      $p_{\mathrm{hist}} = \sum w_i p_i$
      set $C_{ij} = \mathrm{JS}(\delta_i, \delta_j)$ ($C_{ii} = 0$, $C_{ij} = \log 2$)
    end if
    $K = \exp(-C/\varepsilon)$; iterate $u = p_t/(Kv),\ v = p_{\mathrm{hist}}/(K^\top u)$; $P^\star = \mathrm{diag}(u)\, K\, \mathrm{diag}(v)$
    Fuse policy:
    if gaussian **then**:
      transport source samples $A \leftarrow \{a_p\}$ via $P^{\star\top}$,
      set $\tilde{\mu}_t = \mathrm{mean}(P^{\star\top} A)$, $\tilde{\sigma}_t = \mathrm{std}(P^{\star\top} A)$, define $\pi_t^{\mathrm{fuse}} = \mathcal{N}(\tilde{\mu}_t, \mathrm{diag}(\tilde{\sigma}_t^2))$;
    else:
      $\tilde{p}_t = P^{\star\top} p_t$ (normalize), define π_t^{fuse}
    end if
    Policy update with fusion: use π_t^{fuse} in the policy ratio for the current update step
    Update distribution memory: store s_i with (μ_i, σ_i) or p_i into $\mathcal{H}_\pi$
  end if
end for

In particular, if $\rho_t \geq 0$ and $\kappa < 1$, memory fusion strictly reduces variance for any $\beta \in (0,1)$.

Lemma B.2 (Second moment of the fused direction)**.** *With (A1)–(A2),*

$$\mathbb{E}\big[\|d_t\|^2 \big| \theta_t\big] \;\leq\; \underbrace{\Big\|(1-\beta) g_t + \beta\big(g_t + b_t\big)\Big\|^2}_{\leq\, 2\|g_t\|^2 + 2\beta^2 B_t^2} \;+\; \underbrace{\Big((1-\beta)^2 + \beta^2\kappa + 2\beta(1-\beta)\rho_t\sqrt{\kappa}\Big)}_{=:\,\chi_t(\beta,\kappa,\rho_t)\,\leq\,1}\,\sigma^2,$$

where $b_t \coloneqq \mathbb{E}[\bar{g}_t \mid \theta_t] - g_t$ and $\|b_t\| \leq B_t$.

B.3 Main Convergence Result (Deterministic Policy, Biased Estimator)

Theorem B.3 (Convergence to a noise/bias neighborhood)**.** *Under (A1)–(A2) and update $\theta_{t+1} = \theta_t + \eta d_t$, if $\eta \leq \frac{1}{2L}$, then*

$$\min_{0 \leq t \leq T-1} \mathbb{E}\big[\|g_t\|^2\big] \;\leq\; \frac{2\big(J(\theta_0) - J^*\big)}{\eta T} \;+\; 2\beta\, G\, \bar{B}_T \;+\; L\eta\Big(\bar{\chi}_T\, \sigma^2 + 2\beta^2 \overline{B_T^2}\Big), \tag{18}$$

where $\bar{B}_T \coloneqq \frac{1}{T}\sum_{t=0}^{T-1} B_t$, $\overline{B_T^2} \coloneqq \frac{1}{T}\sum_{t=0}^{T-1} B_t^2$ and $\bar{\chi}_T \coloneqq \frac{1}{T}\sum_{t=0}^{T-1} \chi_t(\beta,\kappa,\rho_t) \leq 1$.

Proof. L-smoothness yields

$$\mathbb{E}[J(\theta_{t+1})] \le \mathbb{E}[J(\theta_t)] + \eta\,\mathbb{E}[\langle g_t, d_t\rangle] + \tfrac{L\eta^2}{2}\,\mathbb{E}\big[\|d_t\|^2\big]\,.$$

Decompose the inner product using $\mathbb{E}[\hat{g}_t \mid \theta_t] = g_t$ and $\mathbb{E}[\bar{g}_t \mid \theta_t] = g_t + b_t$:

$$\mathbb{E}[\langle g_t, d_t\rangle|\theta_t] = (1-\beta)\|g_t\|^2 + \beta\,\langle g_t, g_t + b_t\rangle \;=\; \|g_t\|^2 + \beta\,\langle g_t, b_t\rangle \;\ge\; \|g_t\|^2 - \beta\,\|g_t\|\,\|b_t\| \;\ge\; \|g_t\|^2 - \beta G B_t.$$

Taking total expectation and applying Lemma B.2,

$$\mathbb{E}[J(\theta_{t+1})] \le \mathbb{E}[J(\theta_t)] - \eta\,\mathbb{E}\big[\|g_t\|^2\big] + \eta\beta G B_t + \tfrac{L\eta^2}{2}\Big(2\,\mathbb{E}\big[\|g_t\|^2\big] + \chi_t\sigma^2 + 2\beta^2 B_t^2\Big).$$

Rearrange:

$$\Big(\eta - L\eta^2\Big)\mathbb{E}\big[\|g_t\|^2\big] \le \mathbb{E}[J(\theta_t)] - \mathbb{E}[J(\theta_{t+1})] + \eta\beta G B_t + \tfrac{L\eta^2}{2}\Big(\chi_t\sigma^2 + 2\beta^2 B_t^2\Big).$$

With $\eta \le \frac{1}{2L}$ we have $\eta - L\eta^2 \ge \frac{\eta}{2}$, so summing $t = 0$ to $T-1$ and dividing by ηT gives

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\|g_t\|^2\big] \le \frac{2\big(J(\theta_0) - J(\theta_T)\big)}{\eta T} + 2\beta\,G\,\bar{B}_T + L\eta\Big(\bar{\chi}_T\,\sigma^2 + 2\beta^2\overline{B_T^2}\Big).$$

Use $J(\theta_T) \ge J^*$ and the standard $\min \le$ average inequality to obtain equation 18. □

B.4 Policy Improvement Lower Bound (Deterministic Policy)

We quantify first-order improvement for a deterministic policy with an action correction $\delta_t(s) = \lambda_t\, v_t(s)$ where $v_t(s) = \nabla_{\mathrm{mem}}(s)/\|\nabla_{\mathrm{mem}}(s)\|$ and $\lambda_t = c\eta$.

Lemma B.4 (One-step policy improvement)**.** *Assume $Q(\cdot,\cdot)$ is L_Q-smooth in action. Then*

$$J(\pi + \delta_t) - J(\pi) \;\ge\; \lambda_t\,\mathbb{E}_{s\sim d^\pi}\Big[\big\langle \nabla_a Q(s,\pi(s)),\, v_t(s)\big\rangle\Big] - \frac{L_Q}{2}\,\lambda_t^2\,\mathbb{E}_{s\sim d^\pi}\Big[\|v_t(s)\|^2\Big].$$

If, moreover, $\big\langle \nabla_a Q(s,\pi(s)),\, v_t(s)\big\rangle \ge \beta\,\|\bar{g}_t(s)\|\,\cos\theta_t(s)$ for all s (where $\theta_t(s)$ is the angle between $\bar{g}_t(s)$ and $g_t(s)$), then

$$J(\pi + \delta_t) - J(\pi) \;\ge\; \lambda_t\,\beta\,\mathbb{E}_s\big[\|\bar{g}_t(s)\|\cos\theta_t(s)\big] \;-\; \frac{L_Q}{2}\,\lambda_t^2.$$

Hence a sufficient condition for improvement is $\lambda_t \;\le\; \dfrac{2\beta}{L_Q}\,\mathbb{E}_s\big[\|\bar{g}_t(s)\|\cos\theta_t(s)\big]$.

Proof. First-order Taylor with Lipschitz remainder in action gives $Q(s,\pi+\delta) \;\ge\; Q(s,\pi) + \langle\nabla_a Q(s,\pi),\delta\rangle - \frac{L_Q}{2}\|\delta\|^2$. Average over s and plug $\delta_t = \lambda_t v_t$; then apply the stated alignment lower bound. □

C Convergence Analysis of Mnemo with Distributional Experience

This appendix provides a complete convergence analysis for the *distributional* (stochastic-policy) memory used in Mnemo. We equip the per-state policy space with the 2-Wasserstein metric and work with the *state-averaged* distance

$$\overline{W}_2(\pi,\pi')^2 \;:=\; \mathbb{E}_{s\sim d}\big[\,W_2^2\big(\pi(\cdot|s),\,\pi'(\cdot|s)\big)\,\big],$$

where d is a fixed reference state distribution (e.g., on-policy visitation).

C.1 Preliminaries and Assumptions

(B1) **MDP regularity.** $\mathcal{S}, \mathcal{A}$ are compact metric spaces; rewards $r(s,a)$ are bounded and Lipschitz; the discount factor is $\gamma \in (0,1)$.

(B2) **Policy class.** $\Pi = \{\pi_\theta(\cdot|s) : \theta \in \Theta\}$ are *Gaussian* policies with diagonal covariance: $\pi_\theta(\cdot|s) = \mathcal{N}\big(\mu_\theta(s), \mathrm{diag}(\sigma^2_{\theta,1}(s), \ldots, \sigma^2_{\theta,d}(s))\big)$. Maps $s \mapsto (\mu_\theta(s), \sigma_\theta(s))$ are Lipschitz in both s and θ, and Θ is compact and convex.

(B3) **Memory retrieval via mixed similarity and aggregation error.** At iteration t and state s, compute a mixed similarity with each stored s_i:

$$\mathrm{sim}(s, s_i) = \alpha \cos(s, s_i) + (1-\alpha)\left(1 - \frac{\|s - s_i\|_2}{\max(\|s\|_2, \|s_i\|_2) + \varepsilon}\right), \quad \alpha \in (0,1),\ \varepsilon > 0, \tag{19}$$

define $d_i(s) = 1 - \mathrm{sim}(s, s_i)$, and select

$$k_t = \min\big(K_{\max}, \max(K_{\min}, \lfloor\sqrt{|\mathcal{H}_\pi|}\rfloor)\big). \tag{20}$$

Weights over $\mathcal{N}_{k_t}(s)$ are the median-normalized softmax

$$w_{t,i}(s) = \frac{\exp\big(-d_i(s)/\mathrm{median}(\{d_j(s)\}_{j\in\mathcal{N}_{k_t}(s)})\big)}{\sum_{j\in\mathcal{N}_{k_t}(s)} \exp\big(-d_j(s)/\mathrm{median}(\{d_\ell(s)\}_{\ell\in\mathcal{N}_{k_t}(s)})\big)}. \tag{21}$$

Let $\nu_t(\cdot|s) = \sum_{i\in\mathcal{N}_{k_t}(s)} w_{t,i}(s)\, \pi_i(\cdot|s)$ be the aggregated historical distribution, and let $\mathcal{B}_t(\cdot|s)$ denote the (ideal) Wasserstein barycenter of the retrieved set. There exists $\varepsilon_t \geq 0$ such that

$$\overline{W}_2\big(\nu_t, \mathcal{B}_t\big) \ \leq\ \varepsilon_t, \qquad \text{with either } \ \varepsilon_t \to 0 \ \text{ or } \ \sum_t \eta_t \varepsilon_t < \infty. \tag{22}$$

(B4) **Smoothness of retrieval weights.** The map $s \mapsto w_{t,i}(s)$ is locally Lipschitz a.e. with a modulus bounded uniformly in t and i; possible non-smoothness due to top-k selection occurs on measure-zero boundaries (ties), or is removed by a soft top-k relaxation. Consequently, the state-averaged fluctuation of $\nu_t(\cdot|s)$ induced by replacing π with π' is controlled linearly by $\overline{W}_2(\pi, \pi')$ up to the aggregation error equation 22.

(B5) **Transport–entropy control.** There exists a constant $c_{\mathrm{TE}} > 0$ (uniform in s) such that for diagonal Gaussians

$$W_2^2\big(\mathcal{N}(\mu, \Sigma),\, \mathcal{N}(\mu', \Sigma')\big) \ \leq\ c_{\mathrm{TE}}\, \mathrm{KL}\big(\mathcal{N}(\mu, \Sigma) \,\|\, \mathcal{N}(\mu', \Sigma')\big).$$

(This is a standard transport inequality on sub-Gaussian families and is satisfied by diagonal Gaussians with bounded moments.)

(B6) **Bellman/prox policy operator.** Let $\mathcal{T}$ denote the per-iteration *policy update* operator induced by policy improvement (e.g., trust-region or proximal step w.r.t. a value surrogate). Assume $\mathcal{T}$ is a contraction under $\overline{W}_2$ with modulus $\gamma_{\mathcal{T}} \in [0,1)$:

$$\overline{W}_2\big(\mathcal{T}\pi, \mathcal{T}\pi'\big) \ \leq\ \gamma_{\mathcal{T}}\, \overline{W}_2\big(\pi, \pi'\big).$$

Distributional Experience in Mnemo. Given current policy $\pi_t(\cdot|s)$ and retrieval $\nu_t(\cdot|s)$, define

$$M_\pi(\cdot|s) \ \in\ \arg\min_{\pi'(\cdot|s)\in\mathcal{G}(\mathcal{A})} \Big\{ W_2^2\big(\pi'(\cdot|s),\, \nu_t(\cdot|s)\big) + \lambda\, \mathrm{KL}\big(\pi_t(\cdot|s) \,\|\, \pi'(\cdot|s)\big) \Big\},$$

with $\lambda > 0$ and $\mathcal{G}(\mathcal{A})$ the family of diagonal Gaussians. We update policies via the *joint* operator

$$\Phi_{\eta_t} \ :=\ (1-\eta_t)\, \mathcal{T} \ +\ \eta_t\, M_\pi, \qquad \eta_t \in (0,1].$$

C.2 Step 1: Contraction of Experience

Lemma C.1 (Contractivity of M_π up to retrieval error)**.** *Under (B2)–(B5), for any two policies π, π',*

$$\overline{W}_2\big(M_\pi[\pi],\, M_\pi[\pi']\big) \ \leq\ \gamma_M\, \overline{W}_2(\pi, \pi') \ +\ c_1\, \varepsilon_t, \qquad \gamma_M \ =\ \frac{1}{1+\lambda} \ <\ 1,$$

for a constant c_1 that depends only on the Lipschitz moduli of the retrieval weights and on c_{TE}.

Proof. Fix s. The map

$$\Pi \ni \pi' \;\mapsto\; \underbrace{W_2^2\big(\pi'(\cdot|s),\, \nu_t(\cdot|s)\big)}_{\text{1-strongly convex in } W_2} \;+\; \lambda\, \underbrace{\mathrm{KL}\big(\pi(\cdot|s)\,\|\,\pi'(\cdot|s)\big)}_{\lambda\text{-strongly convex in KL}}$$

is strongly convex in the product geometry $W_2^2 + \lambda\,\mathrm{KL}$. Firm non-expansiveness of the corresponding prox implies that changing the *anchor* from $\pi(\cdot|s)$ to $\pi'(\cdot|s)$ contracts distances by at most $1/(1+\lambda)$ in the mixed metric. Using (B5) (transport–entropy) transfers KL control to W_2 up to the constant c_{TE}, absorbed into γ_M. Finally, replacing the ideal barycenter by the retrieved mixture incurs an additive perturbation bounded by ε_t per state; averaging over s yields the claim with some c_1 depending on the Lipschitz moduli of $w_{t,i}(\cdot)$. □

C.3 Step 2: Pseudo-Contraction of the Joint Operator

Lemma C.2 (Pseudo-contraction of Φ_{η_t})**.** *Let $\Phi_{\eta_t} = (1-\eta_t)\mathcal{T} + \eta_t M_\pi$. Under (B2)–(B6),*

$$\overline{W}_2\big(\Phi_{\eta_t}\pi,\ \Phi_{\eta_t}\pi'\big) \;\le\; \Big[(1-\eta_t)\gamma_{\mathcal{T}} + \eta_t\gamma_M\Big]\,\overline{W}_2(\pi,\pi') \;+\; c_1\,\eta_t\,\varepsilon_t.$$

Proof. By convexity of $\overline{W}_2$ and contractivity of $\mathcal{T}$ and M_π,

$$\overline{W}_2\big(\Phi_{\eta_t}\pi, \Phi_{\eta_t}\pi'\big) \le (1-\eta_t)\overline{W}_2\big(\mathcal{T}\pi, \mathcal{T}\pi'\big) + \eta_t\,\overline{W}_2\big(M_\pi[\pi], M_\pi[\pi']\big).$$

Apply $\overline{W}_2(\mathcal{T}\pi,\mathcal{T}\pi') \le \gamma_{\mathcal{T}}\overline{W}_2(\pi,\pi')$ and Lemma C.1. □

C.4 Step 3: Main Convergence Theorem

Theorem C.3 (Almost sure convergence under diminishing stepsizes)**.** *Let $\{\pi_t\}$ evolve by $\pi_{t+1} = \Phi_{\eta_t}\pi_t$ with $\eta_t \in (0,1]$. Assume (B1)–(B6), $\sum_{t=0}^{\infty}\eta_t = \infty$, $\sum_{t=0}^{\infty}\eta_t^2 < \infty$, and $\sum_{t=0}^{\infty}\eta_t\varepsilon_t < \infty$. Then there exists a unique fixed point $\pi^\star$ of $\mathcal{T}$ such that*

$$\lim_{t\to\infty}\overline{W}_2(\pi_t,\pi^\star) \;=\; 0 \qquad \textit{almost surely.}$$

Proof. Let $V_t := \overline{W}_2^2(\pi_t,\pi^\star)$. By Lemma C.2 with $\mathcal{T}\pi^\star = \pi^\star$,

$$\overline{W}_2(\pi_{t+1},\pi^\star) \;\le\; \rho_t\,\overline{W}_2(\pi_t,\pi^\star) + c_1\,\eta_t\,\varepsilon_t, \qquad \rho_t := (1-\eta_t)\gamma_{\mathcal{T}} + \eta_t\gamma_M.$$

Since $\gamma_{\mathcal{T}}, \gamma_M < 1$ and $\eta_t \in (0,1]$, there exists $c > 0$ such that $1-\rho_t \ge c\,\eta_t$. Squaring and using $(x+y)^2 \le (1+\delta)x^2 + (1+\delta^{-1})y^2$ for any $\delta > 0$, we obtain

$$\mathbb{E}[V_{t+1} \mid \mathcal{F}_t] \;\le\; (1 - 2c\eta_t + \tilde{c}\,\eta_t^2)\,V_t \;+\; C\,\eta_t^2 \;+\; \tilde{C}\,\eta_t^2\varepsilon_t^2,$$

for some constants $C, \tilde{c}, \tilde{C}$. By the Robbins–Siegmund almost supermartingale lemma, with $\sum_t \eta_t = \infty$, $\sum_t \eta_t^2 < \infty$, and $\sum_t \eta_t\varepsilon_t < \infty$ implying $\sum_t \eta_t^2\varepsilon_t^2 < \infty$, we conclude $V_t \to 0$ almost surely. □

Remarks. (i) *Constant stepsize neighborhood.* With a constant η and bounded $\varepsilon_t \le \bar{\varepsilon}$, Lemma C.2 yields a linear convergence to an $O(\eta + \bar{\varepsilon})$ neighborhood. (ii) *Role of λ.* The regularizer λ controls $\gamma_M = \frac{1}{1+\lambda}$: larger λ makes M_π more contractive but more conservative (closer to π_t), matching the stability–plasticity tradeoff. (iii) *Approximation fidelity.* Any practical alignment surrogate (finite-sample moment matching, finite-iteration Sinkhorn, etc.) is absorbed into ε_t; either $\varepsilon_t \to 0$ or $\sum_t \eta_t\varepsilon_t < \infty$ suffices. (iv) *Link to trust-region PI.* If $\mathcal{T}$ is a KL-proximal policy improvement (TRPO/PPO-style), standard KL-contractivity together with (B5) implies $\gamma_{\mathcal{T}} < 1$ in $\overline{W}_2$.

D Experience-theoretic view of $\mathcal{H}_t$.

We make the notion of *experience* explicit. Let the environment interaction at time t be

$$e_t \;:=\; (s_t, a_t, r_t, s_{t+1}) \in \mathcal{S}\times\mathcal{A}\times\mathbb{R}\times\mathcal{S}, \tag{23}$$

and define the raw experience history and its filtration by

$$\mathbf{e}_{0:t} \;:=\; (e_0,\ldots,e_t), \qquad \mathcal{F}_t \;:=\; \sigma(e_0,\ldots,e_t). \tag{24}$$

Definition D.1 (Experience trace)**.** A random object X_t is called an *experience trace* if it is $\mathcal{F}_t$-measurable, i.e. there exists a measurable mapping F_t such that

$$X_t = F_t(\mathbf{e}_{0:t}). \tag{25}$$

In actor-critic, the parameters (θ_i, ϕ_i) and the replay minibatches used to compute the policy-update signal ξ_i are themselves $\mathcal{F}_i$-measurable. Hence there exists a measurable mapping

$$\Xi_i : (\mathcal{S} \times \mathcal{A} \times \mathbb{R} \times \mathcal{S})^{i+1} \to \mathcal{X} \tag{26}$$

such that

$$\xi_i = \Xi_i(\mathbf{e}_{0:i}), \tag{27}$$

where $\mathcal{X}$ is the space of update signals (e.g. $\mathbb{R}^{\dim(\mathcal{A})}$ for action-gradients or $\Delta(\mathcal{A})$ for policy distributions).

Lemma D.2. *The policy evolution trace $\mathcal{H}_t = \{\mathcal{E}_i\}_{i \le t}$ with $\mathcal{E}_i = (s_i, \xi_i)$ is an experience trace.*

Proof. By construction,

$$\mathcal{E}_i = (s_i, \xi_i) = \big(s_i, \Xi_i(\mathbf{e}_{0:i})\big), \tag{28}$$

so $\mathcal{E}_i$ is $\mathcal{F}_i$-measurable for each $i \le t$. Therefore the collection

$$\mathcal{H}_t = \{\mathcal{E}_i\}_{i \le t} = F_t(\mathbf{e}_{0:t}) \tag{29}$$

for some measurable F_t, and H_t is $\mathcal{F}_t$-measurable by closure of σ-algebras. By Definition 4.1, $\mathcal{H}_t t$ is an experience trace. □

Consequently, Mnemo's update

$$\pi_{t+1} = \mathcal{M}_t\big(\mathcal{H}_t, \mathcal{U}_t(\pi_t; \mathcal{T})\big) \tag{30}$$

can be written equivalently as

$$\pi_{t+1} = \Phi_t^{\mathrm{Mnemo}}\big(\pi_t, \mathbf{e}_{0:t}\big), \qquad \Phi_t^{\mathrm{Mnemo}}(\pi_t, \mathbf{e}_{0:t}) := \mathcal{M}_t\big(F_t(\mathbf{e}_{0:t}), \mathcal{U}_t(\pi_t; \mathcal{T})\big), \tag{31}$$

showing that Mnemo augments the standard one-step update with an explicit experience-derived statistic H_t.

E Action-gradient vs. parameter-gradient levels and coexistence with Adam.

Consider a deterministic policy $\pi_\theta : \mathcal{S} \to \mathcal{A}$ and a critic $Q_\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Define the actor objective

$$J(\theta) \;=\; \mathbb{E}_{s \sim d^{\pi_\theta}}\big[Q_\phi(s, \pi_\theta(s))\big]. \tag{32}$$

Denote the Jacobian of the policy by

$$J_\pi(s) \;:=\; \nabla_\theta \pi_\theta(s) \;\in\; \mathbb{R}^{d_\theta \times d_a}, \tag{33}$$

where $d_\theta = \dim(\theta)$ and $d_a = \dim(\mathcal{A})$. The *action-gradient* and *parameter-gradient* are

$$h(s; \theta, \phi) \;:=\; \nabla_a Q_\phi(s, a)\big|_{a=\pi_\theta(s)} \;\in\; \mathbb{R}^{d_a}, \tag{34}$$

$$g(\theta, \phi) \;:=\; \nabla_\theta J(\theta) \;=\; \mathbb{E}_{s \sim d^{\pi_\theta}}\big[J_\pi(s)^\top h(s; \theta, \phi)\big] \;\in\; \mathbb{R}^{d_\theta}, \tag{35}$$

so that g is obtained from h via the chain rule through $J_\pi(s)$.

Adam-level update (parameter space). At optimisation step t, let

$$g_t \;:=\; g(\theta_t, \phi_t) = \mathbb{E}_{s \sim d^{\pi_{\theta_t}}}\big[J_\pi(s)^\top h(s; \theta_t, \phi_t)\big]. \tag{36}$$

An Adam-type update maintains moment estimates (m_t, v_t) in *parameter space* $\mathbb{R}^{d_\theta}$:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \tag{37}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t \odot g_t, \tag{38}$$

$$\theta_{t+1}^{\mathrm{Adam}} = \theta_t - \eta_t \, \frac{m_t}{\sqrt{v_t} + \varepsilon}, \tag{39}$$

where $\odot$ denotes element-wise product. Thus Adam defines an operator

$$\mathcal{A}_t : \mathbb{R}^{d_\theta} \times (\mathbb{R}^{d_\theta})^t \to \mathbb{R}^{d_\theta}, \qquad \theta_{t+1}^{\mathrm{Adam}} = \mathcal{A}_t\big(\theta_t; g_1, \ldots, g_t\big), \tag{40}$$

which lives entirely in the parameter space.

MNEMO-G-level update (action-gradient field). At step t, let the *logged action-gradients* be

$$h_i \;:=\; h(s_i;\theta_i,\phi_i) \in \mathbb{R}^{d_a}, \qquad i = 1,\dots,t, \tag{41}$$

and the gradient memory be

$$\mathcal{H}_t \;:=\; \{(s_i,h_i)\}_{i=1}^{m_t} \subset \mathcal{S}\times\mathbb{R}^{d_a}. \tag{42}$$

For a current state s_t, define attention weights

$$w_i(s_t) \;=\; \frac{\exp\big(\tau^{-1}\langle q(s_t),k(s_i)\rangle\big)}{\sum_{j=1}^{m_t}\exp\big(\tau^{-1}\langle q(s_t),k(s_j)\rangle\big)}, \qquad \sum_{i=1}^{m_t} w_i(s_t) = 1, \tag{43}$$

and the *retrieved historical action-gradient* at s_t:

$$h_t^{\mathrm{hist}} \;:=\; \sum_{i=1}^{m_t} w_i(s_t)\, h_i \;\in\; \mathbb{R}^{d_a}. \tag{44}$$

MNEMO-G defines a fused action-gradient

$$h_t^{\mathrm{all}} \;:=\; (1-\lambda_t)\, h(s_t;\theta_t,\phi_t) \;+\; \lambda_t\, h_t^{\mathrm{hist}} \;\in\; \mathbb{R}^{d_a}, \tag{45}$$

with $0 \le \lambda_t \le 1$ (possibly state- and time-dependent). The corresponding *Mnemo-modified parameter gradient* is

$$\tilde{g}_t \;:=\; \mathbb{E}_{s\sim d^{\pi_{\theta_t}}}\big[J_\pi(s)^\top h_t^{\mathrm{all}}(s)\big] \;\in\; \mathbb{R}^{d_\theta}. \tag{46}$$

Running Adam on top of MNEMO-G yields

$$m_t^{\mathrm{M}} = \beta_1 m_{t-1}^{\mathrm{M}} + (1-\beta_1)\tilde{g}_t, \tag{47}$$

$$v_t^{\mathrm{M}} = \beta_2 v_{t-1}^{\mathrm{M}} + (1-\beta_2)\tilde{g}_t \odot \tilde{g}_t, \tag{48}$$

$$\theta_{t+1}^{\mathrm{M+Adam}} = \theta_t - \eta_t\, \frac{m_t^{\mathrm{M}}}{\sqrt{v_t^{\mathrm{M}}+\varepsilon}}. \tag{49}$$

Thus Adam and MNEMO-G act on *different levels*:

- MNEMO-G defines an operator

$$\mathcal{M}_t^{\mathrm{G}} : (\mathcal{S}\times\mathbb{R}^{d_a})^t \to \big(\mathcal{S}\to\mathbb{R}^{d_a}\big), \qquad h_t^{\mathrm{all}}(\cdot) = \mathcal{M}_t^{\mathrm{G}}(\mathcal{H}_t)(\cdot), \tag{50}$$

  i.e. a state-dependent transformation of action-gradients.
- Adam defines $\mathcal{A}_t$ on parameter gradients $\{g_k\},\{\tilde{g}_k\} \subset \mathbb{R}^{d_\theta}$.

Non-equivalence to any Adam-style momentum. Consider a single-sample update (no expectation over s for simplicity). Define

$$g_t = J_\pi(s_t)^\top h_t, \qquad \tilde{g}_t = J_\pi(s_t)^\top\Big[(1-\lambda_t)h_t + \lambda_t \sum_{i=1}^{m_t} w_i(s_t)h_i\Big]. \tag{51}$$

Suppose there exists an Adam-style momentum scheme that reproduces $\tilde{g}_t$ using only $\{g_i\}_{i\le t}$, i.e. suppose there exist scalars $\alpha_{t,i}$ (depending on t but not on the particular $\{h_i,s_i\}$) such that for all sequences $\{(s_i,h_i)\}_{i\le t}$,

$$\tilde{g}_t = \sum_{i=1}^{t} \alpha_{t,i} g_i = \sum_{i=1}^{t} \alpha_{t,i} J_\pi(s_i)^\top h_i. \tag{52}$$

On the other hand, expanding $\tilde{g}_t$ gives

$$\tilde{g}_t = (1-\lambda_t)J_\pi(s_t)^\top h_t + \lambda_t \sum_{i=1}^{m_t} w_i(s_t) J_\pi(s_t)^\top h_i. \tag{53}$$

Comparing the coefficients of each h_i in equation 52 and equation 53, we require, for all i,

$$\alpha_{t,i} J_\pi(s_i)^\top = \begin{cases} (1-\lambda_t)J_\pi(s_t)^\top + \lambda_t w_t(s_t) J_\pi(s_t)^\top, & i = t, \\ \lambda_t w_i(s_t) J_\pi(s_t)^\top, & i \ne t. \end{cases} \tag{54}$$

Unless

$$J_\pi(s_i) = J_\pi(s_t) \quad \text{for all } i \text{ with } w_i(s_t) > 0, \tag{55}$$

and the weights $w_i(s_t)$ are independent of s_t, this system has no solution for $\{\alpha_{t,i}\}$ that is valid for all possible trajectories. For generic neural policies, $J_\pi(s)$ varies with s, and the attention weights $w_i(s_t)$ explicitly depend on the current state. Hence, in general, no Adam-style momentum that operates only on $\{g_i\}$ can reproduce the MNEMO-G update.

Therefore:

$$\text{Adam acts on } g_t \in \mathbb{R}^{d_\theta} \quad \text{(parameter space)}, \qquad \text{MNEMO-G acts on } h_t \in \mathbb{R}^{d_a} \quad \text{(action-gradient field)}, \tag{56}$$

and the two mechanisms are mathematically distinct yet composable via the chain rule:

$$\theta_{t+1}^{\mathrm{M+Adam}} = \mathcal{A}_t\big(\theta_t; \tilde{g}_1, \ldots, \tilde{g}_t\big), \qquad \tilde{g}_t = \mathbb{E}_s\big[J_\pi(s)^\top \mathcal{M}_t^{\mathrm{G}}(\mathcal{H}_t)(s)\big]. \tag{57}$$

F STATE SIMILARITY FOR EXPERIENCE RETRIEVAL

Given an evolution trace (memory) $H_t = \{(s_i, \xi_i)\}_{i=1}^{m_t}$, Mnemo retrieves a relevance-weighted summary of past update signals conditioned on the current query state s:

$$w_i(s) \;:=\; \frac{\exp\big(\tau^{-1}\kappa(s, s_i)\big)}{\sum_{j=1}^{m_t} \exp\big(\tau^{-1}\kappa(s, s_j)\big)}, \qquad \xi^{\mathrm{hist}}(s) \;:=\; \sum_{i=1}^{m_t} w_i(s)\, \xi_i, \tag{58}$$

where $\kappa : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a state similarity score and $\tau > 0$ is a temperature. Optionally, for efficiency we restrict Eq. 58 to a subset $\mathcal{I}_k(s) \subset [m_t]$ (e.g. top-k scores) and renormalise weights over $\mathcal{I}_k(s)$.

F.1 METRIC-BASED SIMILARITY

A simple instantiation uses Euclidean proximity in state space:

$$\kappa_{\mathrm{Euc}}(s, s_i) \;:=\; -\|s - s_i\|_2^2, \tag{59}$$

or equivalently inverse-distance weighting over the selected neighbours:

$$w_i(s) \;\propto\; \frac{1}{\|s - s_i\|_2 + \epsilon}, \qquad i \in \mathcal{I}_k(s), \tag{60}$$

with $\epsilon > 0$ for numerical stability.

F.2 ATTENTION-BASED SIMILARITY

Alternatively, we compute similarity in a learned representation via query–key maps $q, k : \mathcal{S} \to \mathbb{R}^d$:

$$\kappa_{\mathrm{Attn}}(s, s_i) \;:=\; \frac{\langle q(s),\, k(s_i)\rangle}{\sqrt{d}}, \tag{61}$$

leading to $w_i(s) = \mathrm{softmax}(\tau^{-1}\kappa_{\mathrm{Attn}}(s, s_i))$. In practice, q, k can be instantiated as lightweight projections on top of existing actor/critic features (or fixed projections), so retrieval depends on experience-induced representations rather than raw state coordinates.

G POLICY MEMORY: DETAILED FORMULATION

G.1 ELIGIBILITY TRACES

In cognitive neuroscience, *eligibility traces* model how the brain integrates temporally extended action–reward signals via exponential decay (Sutton et al., 1998; Gershman & Daw, 2017). The trace at time step t is:

$$e_t(s, a) = \gamma\lambda\, e_{t-1}(s, a) + \mathbb{1}\{(s_t, a_t) = (s, a)\}, \tag{62}$$

where:

- $e_t(s, a)$: credit assigned to state–action pair (s, a) at time t.
- $\gamma \in (0, 1]$: reward discount factor, controlling temporal horizon of credit.
- $\lambda \in (0, 1]$: trace decay parameter, controlling retention of past eligibility.
- $e_{t-1}(s, a)$: previous eligibility trace for (s, a).
- $\mathbb{1}\{(s_t, a_t) = (s, a)\}$: indicator function, equals 1 if the current pair matches (s, a), else 0.
- (s_t, a_t): state–action pair observed at time t.

This assigns more credit to recently visited state–action pairs, decaying exponentially with time.

G.2 Behavioural Economics Analogy (EWA)

The Experience-Weighted Attraction (EWA) model, introduced by Camerer and Ho (Camerer & Hua Ho, 1999), is a behavioural learning rule widely applied in experimental economics to capture both reinforcement and belief learning. The *Experience-Weighted Attraction* (EWA) model updates preference scores $A_t(a)$:

$$A_t(a) = \frac{\phi N_{t-1} A_{t-1}(a) + [\delta + (1-\delta)\mathbb{1}\{a_t = a\}]\pi_{t-1}(a)}{N_t}, \tag{63}$$

where:

- $A_t(a)$: attraction score for action a at time t.
- N_t: experience count (normalisation term).
- $\phi \in (0, 1]$: memory retention rate for past attractions.
- $\delta \in [0, 1]$: weight given to forgone payoffs (unplayed actions).
- $\pi_{t-1}(a)$: probability of taking action a at $t-1$.

H Running Setting

A single NVIDIA RTX 4060 (8 GB) was used for all reported runs. On **Atari** (e.g., *Breakout*, 30M frames with PPO), end-to-end training typically takes $\approx 12 \sim 18$ hours; incorporating Mnemo (stochastic) adds $\approx 5\% \sim 10\%$ wall-clock time due to distributional retrieval, OT alignment, and KL-based fusion. On **MuJoCo** (e.g., *Ant*, 2M steps with DDPG/TD3 or PPO), a full run completes in $\approx$ 30 minutes on the same GPU; adding Mnemo (deterministic) incurs $\approx 10\% \sim 20\%$ extra time from attention-based retrieval and adaptive gradient fusion. All overheads are measured with identical random seeds and dataloading settings; absolute times may vary slightly with driver and BLAS/cuDNN versions.