Prediction-Based Markov Violation Scores for Detecting Non-Markovian Observations in Reinforcement Learning

Naveen Mysore

Keywords: Markov Property, Conditional Independence Testing, Reinforcement Learning, Non-Markovian Detection.

Summary

A prediction-based Markov Violation Score (MVS) is proposed for detecting non-Markovian observation structure in reinforcement learning. The method works in two stages: a random forest strips out nonlinear Markov-compliant dynamics, then ridge regression checks whether historical observations improve prediction of what remains. MVS is evaluated across six environments (CartPole, Pendulum, Acrobot, HalfCheetah, Hopper, Walker2d), three algorithms (PPO, A2C, SAC), and AR(1) noise at six intensities (16 pairs $\times$ 6 noise levels $\times$ 10 seeds = 960 condition-seed runs). The score successfully detects violations in 7 of 16 environment–algorithm pairs, with the strongest signal in high-dimensional environments (HalfCheetah: $\rho = 0.78$). Phase 2 experiments confirm that violations degrade reward in 13 of 16 conditions. An inversion phenomenon that limits detection in low-dimensional settings is also documented, and its root cause is analyzed as a guide for future work. In a practical utility experiment, MVS is shown to provide actionable guidance for architecture selection: when partial observability renders observations non-Markov, the score identifies the problem and a history-augmented policy chosen accordingly recovers the lost performance.

Contribution(s)

1. A prediction-based Markov Violation Score (MVS) using a two-stage pipeline (random forest residualization + ridge comparison) that detects non-Markovian structure in RL trajectories without causal graph construction
   Context: Prior methods rely on causal discovery (e.g., PCMCI) or assume linearity. MVS handles nonlinear dynamics and produces a bounded scalar in $[0, 1]$.
2. Evaluation across 6 environments, 3 algorithms, 6 noise levels, and 10 seeds (960 runs), showing MVS detects violations with Spearman ρ up to 0.78 and that 13 of 16 pairs exhibit significant reward degradation under noise
   Context: Previous work was limited to few environments and a single algorithm. This is the first large-scale evaluation with significance tests and confidence intervals.
3. Analysis of an MVS inversion phenomenon where the score decreases as violations increase, characterizing when prediction-based detection fails
   Context: The inversion reveals that flexible first-stage models can absorb violation signals in low-dimensional environments, guiding future diagnostic design.
4. Utility demonstration: MVS identifies partial observability and guides architecture selection—switching to a history-augmented policy fully recovers lost performance
   Context: Shows MVS provides actionable guidance beyond detection, enabling evidence-based policy architecture decisions.

Prediction-Based Markov Violation Scores for Detecting Non-Markovian Observations in Reinforcement Learning

Naveen Mysore[1,2]
nmysore@ucsb.edu

[1]Department of Electrical and Computer Engineering, University of California, Santa Barbara, USA
[2]Dyssonance.ai

Abstract

Reinforcement learning algorithms assume that observations satisfy the Markov property, yet real-world sensors frequently violate this assumption through correlated noise, latency, or partial observability. Standard performance metrics conflate Markov breakdowns with other sources of suboptimality, leaving practitioners without tools to detect such violations. This paper introduces a prediction-based Markov Violation Score (MVS) that quantifies non-Markovian structure in observation trajectories. A random forest first removes nonlinear Markov-compliant dynamics; ridge regression then tests whether historical observations reduce prediction error on the residuals beyond what the current observation provides. The resulting score is bounded in $[0, 1]$ and requires no causal graph construction. Evaluation spans six environments (CartPole, Pendulum, Acrobot, HalfCheetah, Hopper, Walker2d), three algorithms (PPO, A2C, SAC), controlled AR(1) noise at six intensity levels, and 10 seeds per condition. In post-hoc detection, 7 of 16 environment–algorithm pairs—primarily high-dimensional locomotion tasks—show significant positive monotonicity between noise intensity and MVS (Spearman ρ up to 0.78, confirmed under repeated-measures analysis); under training-time noise, 13 of 16 pairs exhibit statistically significant reward degradation. An inversion phenomenon is documented in low-dimensional environments where the random forest absorbs the noise signal, causing MVS to decrease as true violations grow—a failure mode analyzed in detail. A practical utility experiment demonstrates that MVS correctly identifies partial observability and guides architecture selection, fully recovering performance lost to non-Markovian observations. Source code to reproduce all results is available at https://github.com/NAVEENMN/Markovianes.

1 Introduction

Reinforcement learning (RL) algorithms overwhelmingly assume that observations satisfy the Markov property: the current observation, together with the current action, is sufficient to predict the distribution over next observations and rewards (Sutton & Barto, 1998). This assumption underpins convergence guarantees, the policy gradient theorem, and the design of virtually all model-free and model-based methods. In practice, though, real-world observations are routinely corrupted by sensor noise, communication delays, and partial observability (Wisniewski et al., 2024), any of which can introduce temporal correlations that break the Markov property.

When that happens, value functions conditioned on the current state alone become systematically biased, and policy gradients may point in wrong directions. The trouble is that standard evaluation metrics—episodic return, sample efficiency, convergence rate—cannot tell a practitioner *why* an

agent is underperforming. An agent struggling because its observations are non-Markovian looks identical, by these metrics, to one struggling with reward sparsity or function approximation error. Despite the practical importance of this failure mode, the RL community lacks standard diagnostic tools for detecting Markov violations in observation trajectories.

This paper proposes a *prediction-based Markov Violation Score* (MVS) to address this gap. The core idea is straightforward: first, a random forest captures whatever structure is predictable from the current state–action pair alone; then, ridge regression checks whether lagged observations carry additional predictive signal about the residuals. If they do, history matters and the Markov property is violated. The resulting score lies in $[0, 1]$ and requires no causal graph construction.

Three main contributions are made. First, the two-stage MVS pipeline is developed and evaluated across six Gymnasium environments (CartPole, Pendulum, Acrobot, HalfCheetah, Hopper, Walker2d), three algorithms (PPO, A2C, SAC), six first-order autoregressive (AR(1)) noise intensities, and 10 seeds per condition—960 runs in total (Sections 4 and 5). Second, in the 7 environment–algorithm pairs where MVS correctly tracks violations (Spearman ρ up to 0.78), higher MVS corresponds to worse policy performance, and 13 of 16 pairs show statistically significant reward degradation under noise. However, an inversion phenomenon is also observed in low-dimensional environments where the random forest absorbs the noise signal rather than leaving it for the second stage (Section 5.3). This inversion was initially surprising, and its root cause is analyzed in detail, since understanding when the method fails is as important as demonstrating when it works. Third, MVS is shown to be actionable: in a controlled partial-observability experiment on CartPole, an MVS-guided strategy that switches to a history-augmented policy when violations are detected fully recovers the lost performance (Section 5.6).

The primary focus is *diagnosis*—giving practitioners a tool to answer "are my observations non-Markovian?"—rather than a complete remedy. That said, the utility experiment demonstrates that even a simple threshold on MVS can drive architecture decisions with substantial impact on reward.

Paper organization. Section 2 surveys related work. Section 3 introduces the Markov property and its connection to conditional independence. Section 4 defines the prediction-based MVS. Section 5 presents experiments, including the utility demonstration. Section 6 discusses limitations, and Section 7 concludes.

2 Related Works

Robust RL and observation noise. Real-world RL deployments frequently encounter noisy or corrupted observations. The robust RL literature addresses this through adversarial training (Pinto et al., 2017), model-based methods for incomplete or noisy observations (Wang et al., 2019), and distributionally robust formulations (Panaganti et al., 2022; Liu et al., 2022b). Separately, the effect of correlated action noise on exploration and performance has been studied in continuous control (Hollenstein et al., 2024; 2022), and selective noise injection has been proposed as a regularizer to improve generalization (Igl et al., 2019). These lines of work all evaluate robustness or exploration quality through downstream task performance; none of them tell a practitioner *whether* or *how severely* the Markov property has been violated.

Partial observability and POMDPs. When the full state is not directly observable, the problem becomes a POMDP (Lauri et al., 2023). Solutions range from belief-state methods to identifying tractable POMDP subclasses with provable sample efficiency (Liu et al., 2022a). Allen et al. (2024) detect and mitigate partial observability during learning via the lambda discrepancy—disagreement between TD(λ) value estimates at different λ—operating during training rather than as a post-hoc diagnostic. The present work is complementary: MVS diagnoses the violation; POMDP methods and training-time approaches address it.

Conditional independence testing and Granger causality. The statistics literature provides many tools for testing conditional independence, from kernel-based tests (Zhang et al., 2011) to

classifier-based approaches. Granger causality (Granger, 1969) asks whether past values of one series improve prediction of another—closely related to the approach taken here. Nonlinear extensions via neural networks (Tank et al., 2022) or random forests handle richer dynamics than linear methods. MVS builds on this tradition but targets the RL setting specifically, where the question reduces to whether *any* historical observation helps predict the next state beyond what the current state already provides.

Markov property testing in RL. Directly testing the Markov property in RL trajectories has received limited attention. Shi et al. (2020) proposed a Forward-Backward Learning procedure that tests the Markov assumption in sequential decision making without parametric assumptions on the joint distribution, with theoretical validity guarantees. However, that test is designed for binary hypothesis testing (Markov or not) and does not produce a graded severity score. Constraint-based causal discovery methods such as PCMCI (Runge et al., 2019) can detect multi-lag dependencies; the framework supports both linear and nonlinear conditional independence tests, though the default partial-correlation test assumes linearity. Mysore (2025) applied PCMCI with partial correlation to quantify first-order Markov violations in noisy RL, demonstrating that such diagnostics are useful but inheriting the linearity constraints of that specific test. The prediction-based approach proposed here sidesteps these issues: the random forest first stage is nonparametric, and history dependence is tested directly through prediction error comparison rather than graph construction.

Positioning. Robust RL assumes violations exist and builds defenses; causal discovery infers graphical structure. MVS sits between the two, providing a single scalar answer to a more focused question: does the current observation suffice, or does history help? For an RL practitioner, that question is often more actionable than a full causal graph, and the prediction-based formulation handles nonlinear dynamics naturally.

3 Background

3.1 Markov Property and Markov Decision Processes

A discrete-time stochastic process $\{X_t\}_{t=0}^{\infty}$ satisfies the *Markov property* if the future state X_{t+1} is conditionally independent of all prior states given the current state:

$$P\big(X_{t+1} \mid X_t, X_{t-1}, \ldots, X_0\big) \;=\; P\big(X_{t+1} \mid X_t\big).$$

In RL, this applies to the state variable S_t. When the environment is Markov,

$$P\big(S_{t+1} = s', R_{t+1} = r \;\mid\; S_t = s, A_t = a, \ldots, S_0, A_0\big) \\ = P\big(S_{t+1} = s', R_{t+1} = r \;\mid\; S_t = s, A_t = a\big),$$

so only the current state S_t and action A_t determine what happens next.

When noise or partial observability corrupts S_t, the observed signal O_t may no longer carry enough information, and higher-order dependencies can emerge in the observation stream—even though the underlying state dynamics remain Markov. Detecting such observation-level dependencies is the central goal of this work. Throughout, "Markov violation" refers to non-Markovian structure in observations, not a breakdown of the latent state dynamics.

3.2 Conditional Independence as a Test for Markov Structure

Two random variables X and Y are *conditionally independent* given Z if $P(X \mid Y, Z) = P(X \mid Z)$. The Markov property is exactly such a statement: $S_{t+1} \perp\!\!\!\perp \{S_0, \ldots, S_{t-1}\} \mid S_t$. Whenever knowledge of past states improves prediction of the next state beyond what the current state provides, this independence is violated.

This observation suggests a practical detection strategy. Rather than constructing a causal graph or computing partial correlations, one can directly test whether historical observations carry predictive information that the current observation misses. The next section formalizes this idea.

4 Prediction-Based Markov Violation Score

This section describes the prediction-based Markov Violation Score (MVS), a scalar that quantifies how much an observation trajectory departs from Markov. The method has two stages: first strip out whatever the current state can predict, then check whether past observations help explain what remains.

4.1 Stage 1: Nonlinear Markov Removal

Given a trajectory of observations $\{o_1, o_2, \ldots, o_T\}$ and actions $\{a_1, a_2, \ldots, a_T\}$, two sets of features are constructed. The *Markov features* $\mathbf{x}_t^{(M)} = [o_t, a_t]$ contain only the information that would suffice if the process were Markov. The *history features* $\mathbf{x}_t^{(H)} = [o_t, a_t, o_{t-1}, a_{t-1}, \ldots, o_{t-k+1}, a_{t-k+1}]$ additionally include $k-1$ lagged observation–action pairs. The target is always the next observation $y_t = o_{t+1}$.

In Stage 1, a random forest regressor predicts y_t from the Markov features alone:

$$\hat{y}_t^{\mathrm{RF}} = f_{\mathrm{RF}}\big(\mathbf{x}_t^{(M)}\big).$$

The forest uses 200 trees with max depth 10 and a minimum of 5 samples per leaf. To avoid information leakage, residuals are computed using out-of-bag (OOB) predictions:

$$r_t = y_t - \hat{y}_t^{\mathrm{OOB}}.$$

After this stage, the residuals $\{r_t\}$ have been stripped of everything predictable from the current observation and action. Whatever predictable structure remains must come from history—exactly the signal that indicates a Markov violation.

4.2 Stage 2: Ridge Comparison

The residuals are split into training (first 70%) and test (last 30%) sets chronologically, and two ridge regressions are fit:

Markov ridge model. g_M predicts residuals from Markov features:

$$\hat{r}_t^{(M)} = g_M\big(\mathbf{x}_t^{(M)}\big).$$

This baseline picks up any linear Markov structure that the random forest may have missed.

History ridge model. g_H predicts residuals from history features:

$$\hat{r}_t^{(H)} = g_H\big(\mathbf{x}_t^{(H)}\big).$$

If the process is truly Markov, g_H should do no better than g_M.

Both models use RidgeCV with 20 regularization values $\alpha \in \{10^{-2}, 10^{-1.6}, \ldots, 10^{5}\}$ selected via leave-one-out cross-validation (LOO). LOO on sequential data can leak temporal information through shared lagged features; blocked or rolling cross-validation would provide a stricter guard against this, at the cost of less efficient use of the training set. Test-set errors are:

$$\mathrm{MSE}_M = \frac{1}{n_{\mathrm{test}}} \sum_{t \in \mathcal{T}_{\mathrm{test}}} \big(r_t - \hat{r}_t^{(M)}\big)^2, \qquad \mathrm{MSE}_H = \frac{1}{n_{\mathrm{test}}} \sum_{t \in \mathcal{T}_{\mathrm{test}}} \big(r_t - \hat{r}_t^{(H)}\big)^2.$$

4.3 MVS Definition

The Markov Violation Score is the fractional reduction in prediction error from adding history:

$$\mathrm{MVS} = \mathrm{clip}\left(\frac{\mathrm{MSE}_M - \mathrm{MSE}_H}{\mathrm{MSE}_M},\ 0,\ 1\right). \tag{1}$$

A few properties are worth noting. The score is bounded in $[0, 1]$ by construction: negative values (where the history model performs worse, typically due to estimation noise in finite samples) are clipped to zero. When the process is Markov, the history model gains nothing and $\mathrm{MVS} \approx 0$. Intuitively, MVS measures the relative reduction in test-set prediction error from adding historical observations. The random forest first stage is important because it handles nonlinear-but-Markov dynamics (e.g., the trigonometric relationships in Pendulum) that would otherwise produce false positives. The ridge second stage, with cross-validated regularization, guards against overfitting to spurious history correlations in finite samples.

Design choices. A history depth of $k = 3$ (two additional lags beyond the current) is used, along with a 70/30 train–test split and per-dimension MVS computation averaged across observation dimensions. AR(1) noise induces dependence at all lags (decaying as α^ℓ); $k = 3$ captures the strongest portion (lags 1–2) while keeping computation manageable.

4.4 AR(1) Noise as a Controlled Markov Violation

To evaluate MVS under violations of known severity, autoregressive noise is injected into observations. At each time step, each dimension i is corrupted:

$$\tilde{o}_t^{(i)} = o_t^{(i)} + z_t^{(i)}, \qquad z_t^{(i)} = \alpha \cdot z_{t-1}^{(i)} + \epsilon_t^{(i)}, \quad \epsilon_t^{(i)} \sim \mathcal{N}(0, 1),$$

where $\alpha \in [0, 1)$ controls the autocorrelation. When $\alpha = 0$, the noise is i.i.d. and introduces no temporal correlation into the observation stream. (Strictly, even i.i.d. additive noise can make observations non-Markov in the hidden-Markov-model sense, since $\tilde{o}_t$ is a noisy function of the latent state; however, no *additional* history dependence is created by the noise process itself, and MVS is empirically near zero in this condition.) For $\alpha > 0$, the noise process z_t carries information from previous time steps into $\tilde{o}_t$, introducing temporal dependence that grows with α.

An important distinction: the underlying environment dynamics remain Markov in the augmented state (S_t, z_t). However, because the noise process z_t is hidden from the agent, the *observation stream* $\{\tilde{o}_t\}$ is non-Markov—the current corrupted observation alone does not determine the conditional distribution of the next. MVS targets exactly this observation-level non-Markovity, which is what a practitioner encounters when working with sensor data.

This setup gives two experimental phases:

1. **Phase 1 (Post-hoc detection):** Policies are trained on clean observations. AR(1) noise is injected into collected trajectories afterward, and MVS is computed. This isolates detection capability from any policy adaptation effects.
2. **Phase 2 (Training under noise):** Policies are trained from scratch with AR(1) noise present throughout, measuring how Markov violations affect learning.

5 Experiments and Results

MVS is evaluated across six RL environments, three algorithms, and six noise intensities. Phase 1 asks whether MVS can detect controlled Markov violations in post-hoc trajectories; Phase 2 measures how those same violations affect policy learning.

5.1 Experimental Setup

Environments. Table 1 summarizes the six environments, which span classic control and continuous locomotion from OpenAI Gymnasium (Towers et al., 2024).

Table 1: Environment summary. Observation dimensionality ranges from 3 (Pendulum) to 17 (HalfCheetah, Walker2d). SAC is used only for continuous-action environments.

Environment	Obs. Dim	Action Space	Algorithms	Training Steps
CartPole-v1	4	Discrete (2)	PPO, A2C	50k
Pendulum-v1	3	Continuous (1)	PPO, A2C, SAC	450k
Acrobot-v1	6	Discrete (3)	PPO, A2C	50k
HalfCheetah-v4	17	Continuous (6)	PPO, A2C, SAC	1M
Hopper-v4	11	Continuous (3)	PPO, A2C, SAC	1M
Walker2d-v4	17	Continuous (6)	PPO, A2C, SAC	1M

Algorithms. PPO (Schulman et al., 2017) and A2C (Mnih et al., 2016) are used across all six environments, with SAC (Haarnoja et al., 2018) additionally applied to the four continuous-action environments. SAC is excluded from CartPole and Acrobot because the `stable-baselines3` (Raffin et al., 2021) SAC implementation requires continuous action spaces. All agents use two-hidden-layer multi-layer perceptron (MLP) policies, giving 16 environment–algorithm pairs in total.

Noise protocol. AR(1) noise (Section 4.4) is applied to all observation dimensions at six autocorrelation levels: $\alpha \in \{0.0, 0.1, 0.3, 0.5, 0.7, 0.9\}$. The $\alpha = 0$ condition serves as the uncorrelated-noise baseline (i.i.d. additive noise with no temporal correlation).

Seeds and evaluation. Every condition is run with 10 independent seeds. Means with 95% confidence intervals are reported, and significance is assessed via Spearman rank correlation (Phase 1) and Welch's t-test (Phase 2) at the $p < 0.05$ level. Because multiple environment–algorithm pairs are tested (16 in Phase 1, 16 in Phase 2), Benjamini–Hochberg false discovery rate (FDR) correction is applied; significance counts reported throughout refer to FDR-adjusted q-values.

5.2 Phase 1: MVS Sensitivity to Noise Intensity

Policies are first trained on clean observations. After training, trajectories are collected, AR(1) noise is injected post-hoc at each α level, and MVS is computed. This design isolates MVS detection from any confounding policy adaptation.

Figure 1 plots MVS against α for all 16 pairs. Table 2 gives the Spearman correlations.

Repeated-measures validation. Because Phase 1 injects six noise levels into the same trajectory per seed, the 60 points per pair are not independent. To verify that the pooled p-values are not artifacts, Spearman ρ was computed within each seed (across the 6 α values). For all 7 positive pairs, all 10 seeds show positive ρ (sign-test $p = 0.002$). Page's L trend test—which directly tests ordered alternatives under repeated measures—confirms significance for all 7 ($p < 10^{-4}$), with within-seed median ρ from 0.54 to 0.83.

Specificity check. MVS was also computed on trajectories from random (untrained) policies under the same noise. Random-policy MVS is near zero everywhere, confirming that the score does not fire on unstructured trajectories. The nonzero scores seen in trained-policy runs reflect a genuine interaction between the AR(1) noise and the structure that the learned policy imposes on trajectories. A stronger specificity test—Markov but nonlinear dynamics with i.i.d. noise under trained policies—is left to future work.

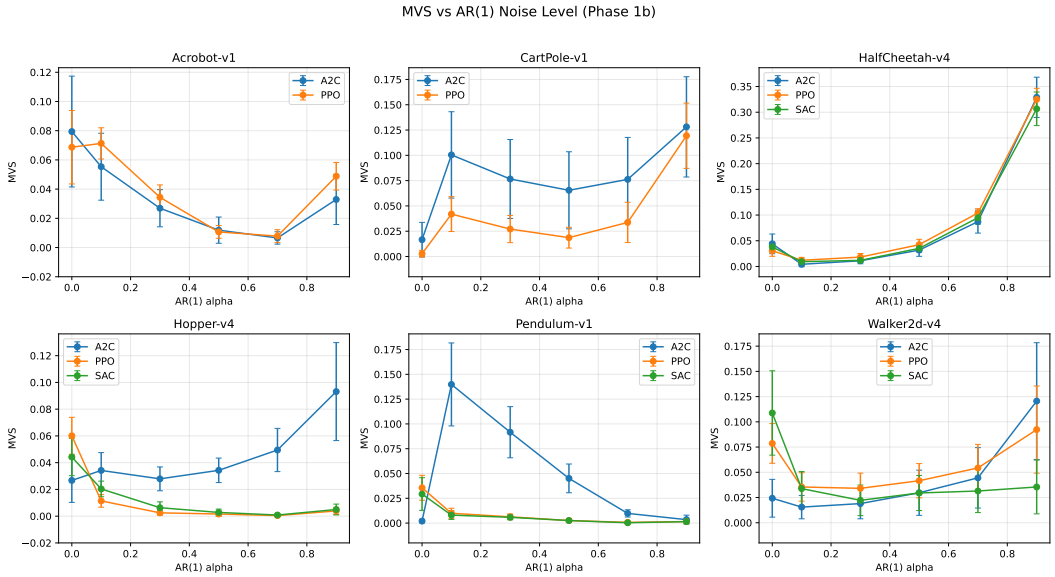

Figure 1: **Phase 1: MVS vs. noise intensity.** Each panel shows one environment; lines represent different algorithms. Error bars are 95% CIs over 10 seeds. MVS increases monotonically with α in HalfCheetah and CartPole. In Pendulum, Hopper (PPO/SAC), and Acrobot it *decreases*—the inversion phenomenon discussed in Section 5.3.

Table 2: **Phase 1 monotonicity.** Spearman ρ between noise intensity α and MVS, pooled across seeds ($n = 60$ per pair). Seven pairs show the expected positive trend (HalfCheetah strongest at $\rho = 0.78$). Eight show significant inversion. See text for repeated-measures analysis that accounts for within-seed dependence.

Environment	Algorithm	Spearman ρ	p-value	Direction
Positive monotonicity (7 pairs):				
HalfCheetah-v4	PPO	0.776	$< 10^{-12}$	✓
HalfCheetah-v4	SAC	0.700	$< 10^{-9}$	✓
HalfCheetah-v4	A2C	0.664	$< 10^{-8}$	✓
CartPole-v1	PPO	0.554	$< 10^{-5}$	✓
Hopper-v4	A2C	0.520	$< 10^{-4}$	✓
Walker2d-v4	A2C	0.507	$< 10^{-4}$	✓
CartPole-v1	A2C	0.447	$< 10^{-3}$	✓
Inverted (8 pairs):				
Pendulum-v1	PPO	-0.760	$< 10^{-12}$	×
Pendulum-v1	SAC	-0.756	$< 10^{-12}$	×
Hopper-v4	SAC	-0.724	$< 10^{-10}$	×
Hopper-v4	PPO	-0.643	$< 10^{-7}$	×
Acrobot-v1	PPO	-0.497	$< 10^{-4}$	×
Acrobot-v1	A2C	-0.466	$< 10^{-3}$	×
Walker2d-v4	SAC	-0.330	0.010	×
Pendulum-v1	A2C	-0.280	0.030	×
Not significant (1 pair):				
Walker2d-v4	PPO	0.070	0.596	—

5.3 Inverted MVS: When Detection Fails

In eight of 16 pairs, MVS moves in the *wrong* direction: it *decreases* as α increases (Table 2). This was initially surprising and prompted a detailed investigation.

Root cause. The problem lies in Stage 1. In low-dimensional environments with highly regular trained-policy trajectories, the random forest is flexible enough to fit not just the true Markov dynamics but also the AR(1) noise pattern riding on top of them. Once the RF captures that noise structure, the residuals are scrubbed of the very signal Stage 2 needs. As α grows and the noise becomes a larger fraction of the total signal, the RF fits it more aggressively—hence the inverted relationship.

When it happens. Two factors predict inversion: (1) low observation dimensionality, which gives the RF fewer features and makes noise patterns easier to memorize, and (2) highly structured clean-policy trajectories, which provide a regular backdrop against which AR(1) noise stands out. HalfCheetah, with 17 dimensions and noisier dynamics, is largely immune.

Implications. This is a fundamental limitation of any two-stage design where the first stage is flexible enough to absorb the violation signal. Potential mitigations—restricting RF capacity, using a linear first stage in low-dimensional settings, ensemble strategies—are discussed in Section 6.

5.4 Phase 2: Impact of Markov Violations on Reward

Phase 2 asks whether the violations MVS is designed to detect actually matter for learning. Agents are trained from scratch with AR(1) noise present throughout. Baseline ($\alpha = 0$) versus heavily noised ($\alpha = 0.9$) final reward is compared using Welch's t-test across 10 seeds.

Figure 2 plots reward against α; Table 3 gives the statistical comparisons.

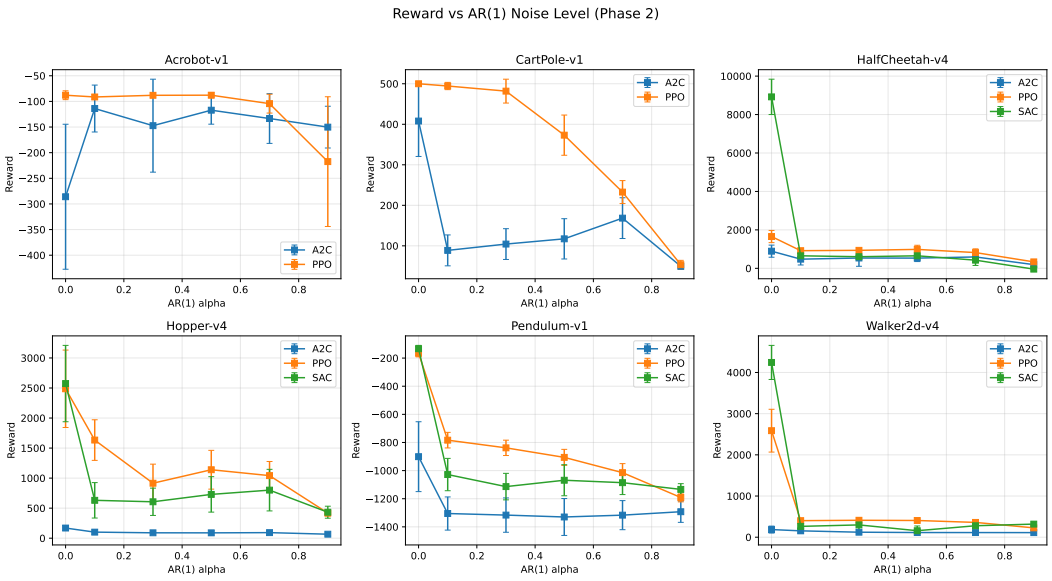

Figure 2: **Phase 2: Reward vs. noise intensity.** AR(1) noise during training degrades final performance across nearly all conditions. The worst collapses: HalfCheetah-SAC drops from 8920 to -42; Walker2d-SAC from 4244 to 318; CartPole-PPO from 500 to 53.

The damage is substantial. The worst-hit condition, HalfCheetah-SAC, collapses from 8920 to -42—essentially complete failure. Walker2d-SAC drops 93% (4244 to 318), CartPole-PPO 89% (500 to 53). The three non-significant pairs after FDR correction are all borderline ($q < 0.08$)

Table 3: **Phase 2: Welch's t-test.** $\alpha = 0$ vs. $\alpha = 0.9$ reward, 10 seeds. Thirteen of 16 pairs show significant degradation after Benjamini–Hochberg FDR correction ($q < 0.05$). Rewards rounded to integers.

Environment	Algorithm	$\alpha=0$ Reward	$\alpha=0.9$ Reward	t-stat	Significant
CartPole-v1	PPO	500	53	94.3	Yes
Pendulum-v1	PPO	−166	−1191	57.4	Yes
Pendulum-v1	SAC	−133	−1134	48.2	Yes
HalfCheetah-v4	SAC	8920	−42	22.1	Yes
Walker2d-v4	SAC	4244	318	21.3	Yes
Walker2d-v4	PPO	2587	227	10.2	Yes
HalfCheetah-v4	PPO	1654	336	9.25	Yes
CartPole-v1	A2C	408	50	9.24	Yes
Hopper-v4	SAC	2574	432	7.53	Yes
Hopper-v4	PPO	2487	418	7.24	Yes
Hopper-v4	A2C	169	67	5.22	Yes
HalfCheetah-v4	A2C	893	193	4.70	Yes
Pendulum-v1	A2C	−901	−1292	3.41	Yes
Acrobot-v1	PPO	−88	−217	2.31	No ($q = 0.053$)
Walker2d-v4	A2C	186	112	1.95	No ($q = 0.08$)
Acrobot-v1	A2C	−286	−150	−2.09	No ($q = 0.07$)

and involve either A2C in environments where it already performs modestly, or Acrobot where the absolute reward scale is small.

A caveat: AR(1) noise with larger α has higher marginal variance, so part of the degradation may reflect noise power rather than temporal correlation per se. Matching marginal variance across α while varying only autocorrelation would isolate the non-Markov contribution; this more controlled design is left to future work.

5.5 Combined Analysis

Phase 1 and Phase 2 independently establish two facts: MVS tracks violation severity in high-dimensional environments (Spearman ρ up to 0.78, 60 observations per pair), and those same violations degrade reward (13 of 16 pairs significant after FDR correction). Figure 3 puts the two together by plotting Phase 1 MVS against Phase 2 reward ratio for each condition.

The scatter reveals a clear pattern: in environments where MVS works correctly—primarily high-dimensional ones like HalfCheetah—points fan out to the right with increasing noise, and higher MVS corresponds to worse reward. In inverted environments, MVS stays near zero regardless of noise level, so reward degrades without a corresponding MVS signal. This is consistent with the inversion mechanism from Section 5.3. The qualitative takeaway is that MVS is most informative in environments where the random forest cannot easily memorize the noise—precisely the higher-dimensional settings where diagnosis is most needed.

5.6 Practical Utility: MVS-Guided Architecture Selection

The results so far show that MVS detects violations and that violations hurt reward. But can a practitioner actually *use* the score? This is tested with a simple architecture-selection experiment: choose between a standard memoryless policy and one that receives a window of recent observations.

Setup. CartPole-v1 is used under two observation conditions. In the *full* condition the agent sees all four state variables (position, velocity, angle, angular velocity). In the *masked* condition the two velocities are zeroed out, leaving only positions—a clean partial-observability setting where the current observation alone cannot determine the next state. For each condition, both a standard

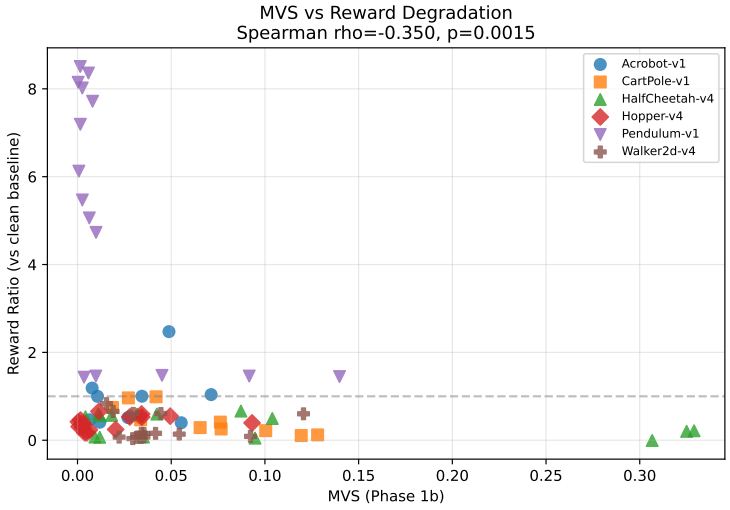

Figure 3: **Combined: MVS vs. reward ratio.** Each point is one environment–algorithm–noise-level condition. In environments where MVS correctly tracks violations (e.g., HalfCheetah, CartPole), higher MVS corresponds to lower reward. Inverted pairs cluster near MVS ≈ 0 regardless of reward loss.

MLP policy and a history-augmented policy (current plus two prior observations concatenated) are trained. All runs use PPO, 100k steps, 5 seeds, 20 evaluation episodes.

MVS detection. Table 4 shows MVS computed on trajectories from both random and trained policies.

Table 4: **MVS detection of partial observability.** Masking velocities creates non-Markov observations. MVS is near zero for full observations and rises to 0.42 under masking with a trained policy, correctly flagging the violation.

Observation	Random Policy MVS	Trained Policy MVS
Full (4D)	0.000	0.000
Masked (positions only)	0.002	0.421

MVS correctly flags the masked condition (0.42) and confirms that full observations are Markov (0.00). Interestingly, the signal is much stronger under a trained policy than a random one (0.42 vs. 0.002), likely because a trained policy concentrates on a narrow region of state space where the missing velocity information matters more.

Architecture selection. Table 5 shows what happens when the MVS signal is acted upon.

Table 5: **Architecture selection results.** Under full observations both policies hit 500. Under masking the standard policy collapses while the history-augmented policy fully recovers. PPO, 100k steps, 5 seeds, 20 evaluation episodes.

Observation	Policy	Mean Reward	Std
Full	Standard	500.0	0.0
Full	History-augmented	500.0	0.0
Masked	Standard	43.1	1.2
Masked	History-augmented	500.0	0.0

The takeaway is clear. When observations are Markov ($\mathrm{MVS} = 0$), a standard policy suffices and adding history buys nothing. When observations are non-Markov ($\mathrm{MVS} = 0.42$), the standard policy collapses to reward 43—a 91% drop—while the history-augmented policy recovers fully to 500. A simple rule—use the standard architecture when MVS is near zero, switch to history augmentation otherwise—gets optimal performance in both cases without adding unnecessary complexity.

5.7 Comparison with PCMCI-Based Detection

Prediction-based MVS is compared against the PCMCI-based approach of Mysore (2025) and a linear Granger baseline on identical trajectory data (same protocol as Phase 1: 16 pairs, 10 seeds, 6 α levels). Following Mysore (2025), PCMCI uses the ParCorr conditional-independence test in tigramite (Runge et al., 2019) ($\tau_{\max} = 3$, $p < 0.05$); the PCMCI score is the fraction of significant lag$\geq$1 links. The Granger score is the fractional reduction in linear prediction error from adding lagged observations. All scores in Table 6 are computed on the same trajectories from an independent replication; Pred-MVS values differ slightly from Table 2 due to different random seeds but match in direction and qualitative grouping for all 16 pairs.

Table 6: **Method comparison.** Spearman ρ vs. α ($n = 60$ per pair). Bold: uncorrected $p < 0.05$ (positive **+** or negative $-$).

Environment	Algo	Pred-MVS	PCMCI	Granger
Positive monotonicity (Pred-MVS):				
HalfCheetah-v4	PPO	**+0.81**	+0.05	**+0.56**
HalfCheetah-v4	SAC	**+0.78**	0.00	**+0.46**
HalfCheetah-v4	A2C	**+0.75**	+0.03	**+0.55**
CartPole-v1	A2C	**+0.57**	−0.19	+0.19
Walker2d-v4	A2C	**+0.40**	**−0.70**	−0.11
Hopper-v4	A2C	**+0.33**	**−0.91**	**−0.60**
CartPole-v1	PPO	**+0.30**	−0.05	**−0.55**
Inverted or not significant (Pred-MVS):				
Acrobot-v1	A2C	−0.19	**−0.40**	**−0.46**
Pendulum-v1	A2C	**−0.33**	**−0.83**	**−0.87**
Walker2d-v4	SAC	**−0.43**	**−0.93**	−0.24
Acrobot-v1	PPO	**−0.56**	**−0.28**	**−0.63**
Hopper-v4	PPO	**−0.61**	**−0.98**	**−0.63**
Pendulum-v1	SAC	**−0.66**	**−0.71**	**−0.94**
Hopper-v4	SAC	**−0.79**	**−0.97**	**−0.72**
Pendulum-v1	PPO	**−0.82**	**−0.74**	**−0.94**
Walker2d-v4	PPO	+0.08	**−0.85**	−0.06
Positive count (of 16):		**7**	**0**	**3**

Prediction-based MVS achieves positive monotonicity in 7 of 16 pairs, compared to 0 for PCMCI-ParCorr and 3 for Granger. In the inverted regime, all three methods show negative ρ. These results compare to the specific PCMCI-ParCorr baseline; nonlinear conditional-independence tests may yield different outcomes.

6 Limitations and Future Directions

MVS inversion in low-dimensional environments. The most significant limitation is the inversion phenomenon documented in Section 5.3: in 8 of 16 environment–algorithm pairs, MVS *decreases* as the true violation grows stronger. Because the random forest in Stage 1 is flexible enough to absorb the noise signal itself, the residuals end up cleaner than they should be—and Stage 2 finds nothing. This is not an implementation bug but a fundamental tension in any two-stage approach with a powerful first stage. Several directions seem worth exploring: restricting forest capacity

(fewer trees, shallower depth) so it cannot latch onto temporal noise patterns; falling back to a linear first stage in low-dimensional settings where Markov dynamics are roughly linear anyway; ensembling across Stage 1 models of varying capacity; or collapsing to a single-stage direct comparison between Markov and history models, accepting higher false-positive rates in nonlinear-but-Markov systems.

AR(1) noise as the sole violation type. MVS has been tested exclusively against AR(1) observation noise, which provides a clean, parameterized violation. Real-world non-Markov structure comes from many sources—sensor latency, frame stacking artifacts, communication delays, and missing state dimensions—and whether MVS generalizes to these settings is an open question.

From diagnosis to remedy. Section 5.6 shows that MVS can guide architecture selection in a controlled setting, but fully closing the loop remains open. MVS could potentially run as an online diagnostic during training, triggering adaptive responses when the score crosses a threshold. The inversion problem must be resolved before MVS can serve as a reliable online signal in all environments.

Scalability and algorithm coverage. All steps scale linearly in trajectory length and polynomially in observation dimensionality; for the environments tested (up to 17 dimensions), computation is negligible next to RL training time. Scaling to image observations would require a representation learning step. PPO, A2C, and SAC are evaluated here; how Markov violations interact with model-based RL, offline RL, or multi-agent settings remains unexplored.

Cross-validation, noise design, and theoretical guarantees. Stage 2 selects ridge regularization via leave-one-out cross-validation. Because adjacent time points share lagged features, LOO can leak temporal information; blocked or rolling CV would provide a stricter protocol. Additionally, AR(1) noise with larger α has higher marginal variance, so Phase 2 reward degradation conflates temporal correlation with noise power. MVS currently lacks formal statistical guarantees; establishing Type I and Type II error rates would put the method on firmer theoretical ground.

7 Conclusion

A prediction-based Markov Violation Score is introduced that quantifies non-Markovian structure in RL observation trajectories. The two-stage approach—random forest residualization followed by ridge regression comparison—yields a bounded, interpretable scalar: zero when the process is Markov, increasing with the severity of history dependence.

Across six environments, three algorithms, and controlled AR(1) noise, MVS successfully detects violations in 7 of 16 environment–algorithm pairs, with Spearman correlations up to $\rho = 0.78$ between noise intensity and the score. In these environments, higher MVS corresponds to worse policy performance. Phase 2 results confirm that these violations have real consequences: 13 of 16 pairs show statistically significant reward degradation under noise. In a practical utility experiment, MVS is shown to guide architecture selection—when partial observability renders observations non-Markovian, the score flags the problem and a history-augmented policy chosen accordingly recovers the lost performance entirely. At the same time, an inversion phenomenon in low-dimensional environments limits detection in nearly half the tested conditions, pointing to a fundamental tension between flexible first-stage modeling and preserving the signal that the second stage needs.

The candid analysis of when MVS works and when it does not is a contribution in its own right. Reliable detection of history dependence is a prerequisite for principled mitigation, and understanding why prediction-based approaches fail in certain regimes should inform the design of more robust diagnostics going forward.

References

Cameron Allen, Aaron Kirtland, Ruo Yu Tao, Sam Lobel, Daniel Scott, Nicholas Petrocelli, Omer Gottesman, Ronald Parr, Michael L. Littman, and George Konidaris. Mitigating partial observability in sequential decision processes via the lambda discrepancy. In *Advances in Neural Information Processing Systems*, 2024.

C. W. J. Granger. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica*, 37(3):424–438, 1969. DOI: 10.2307/1912791.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, August 2018. URL http://arxiv.org/abs/1801.01290. arXiv:1801.01290 [cs].

Jakob Hollenstein, Sayantan Auddy, Matteo Saveriano, Erwan Renaudo, and Justus Piater. Action Noise in Off-Policy Deep Reinforcement Learning: Impact on Exploration and Performance. *Transactions on Machine Learning Research*, June 2022. DOI: 10.48550/arXiv.2206.03787. URL https://arxiv.org/abs/2206.03787. arXiv:2206.03787 [cs.LG].

Jakob Hollenstein, Georg Martius, and Justus Piater. Colored Noise in PPO: Improved Exploration and Performance through Correlated Action Sampling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(11):12466–12472, March 2024. ISSN 2374-3468, 2159-5399. DOI: 10.1609/aaai.v38i11.29139. URL http://arxiv.org/abs/2312.11091. arXiv:2312.11091 [cs].

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in Reinforcement Learning with Selective Noise Injection and Information Bottleneck. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. DOI: 10.48550/arXiv.1910.12911. URL https://proceedings.neurips.cc/paper/2019/hash/e2ccf95a7f2e1878fcafc8376649b6e8-Abstract.html. arXiv:1910.12911 [cs.LG].

Mikko Lauri, David Hsu, and Joni Pajarinen. Partially Observable Markov Decision Processes in Robotics: A Survey. *IEEE Transactions on Robotics*, 39(1):21–40, February 2023. ISSN 1552-3098, 1941-0468. DOI: 10.1109/TRO.2022.3200138. URL https://ieeexplore.ieee.org/document/9899480/.

Qinghua Liu, Alan Chung, Csaba Szepesvari, and Chi Jin. When Is Partially Observable Reinforcement Learning Not Scary? In *Proceedings of Thirty Fifth Conference on Learning Theory*, pp. 5175–5220. PMLR, June 2022a. URL https://proceedings.mlr.press/v178/liu22f.html. ISSN: 2640-3498.

Zijian Liu, Qinxun Bai, Jose Blanchet, Perry Dong, Wei Xu, Zhengqing Zhou, and Zhengyuan Zhou. Distributionally Robust Q-Learning. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 13623–13643. PMLR, June 2022b. URL https://proceedings.mlr.press/v162/liu22a.html. ISSN: 2640-3498.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1928–1937. PMLR, 2016.

Naveen Mysore. Quantifying first-order markov violations in noisy reinforcement learning: A causal discovery approach. *arXiv preprint*, arXiv:2503.00206, February 2025. DOI: 10.48550/arXiv.2503.00206. URL https://arxiv.org/abs/2503.00206.

Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, and Mohammad Ghavamzadeh. Robust Reinforcement Learning using Offline Data, October 2022. URL http://arxiv.org/abs/2208.05129. arXiv:2208.05129 [cs].

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust Adversarial Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 2817–2826. PMLR, July 2017. URL https://proceedings.mlr.press/v70/pinto17a.html. ISSN: 2640-3498.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dorber. Stable-Baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5 (11):eaau4996, 2019. DOI: 10.1126/sciadv.aau4996.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017. URL http://arxiv.org/abs/1707.06347. arXiv:1707.06347 [cs].

Chengchun Shi, Runzhe Wan, Rui Song, Wenbin Lu, and Ling Leng. Does the Markov Decision Process Fit the Data: Testing for the Markov Property in Sequential Decision Making. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 8807–8817. PMLR, November 2020. URL https://proceedings.mlr.press/v119/shi20c.html. ISSN: 2640-3498.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.

Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily B. Fox. Neural Granger Causality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4267–4279, August 2022. DOI: 10.1109/TPAMI.2021.3065601. Epub 2022 Jul 1. PMID: 33705309; PMCID: PMC9739174.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *arXiv preprint arXiv:2407.17032*, July 2024. DOI: 10.48550/arXiv.2407.17032. URL https://arxiv.org/abs/2407.17032. arXiv:2407.17032 [cs.LG].

Yuhui Wang, Hao He, and Xiaoyang Tan. Robust Reinforcement Learning in POMDPs with Incomplete and Noisy Observations, February 2019. URL http://arxiv.org/abs/1902.05795. arXiv:1902.05795 [cs].

Mariusz Wisniewski, Paraskevas Chatzithanos, Weisi Guo, and Antonios Tsourdos. Benchmarking Deep Reinforcement Learning for Navigation in Denied Sensor Environments, October 2024. URL http://arxiv.org/abs/2410.14616. arXiv:2410.14616 [cs].

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based Conditional Independence Test and Application in Causal Discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 804–813, February 2011. DOI: 10.48550/arXiv.1202.3775. URL https://arxiv.org/abs/1202.3775. arXiv:1202.3775 [cs.LG].

Supplementary Materials

The following content was not necessarily subject to peer review.

8 Declaration of LLM Usage

Large language models (GPT-4, Claude) were used during manuscript preparation for grammar correction and revising passive voice constructions. All scientific content, experimental design, implementation, and analysis are the sole work of the author.

9 Implementation and Reproducibility Details

All RL agents are trained using Stable-Baselines3 (Raffin et al., 2021) with default hyperparameters for each algorithm (PPO, A2C, SAC). The AR(1) noise wrapper, MVS computation pipeline, and analysis scripts are included in the supplementary source code. Random seeds are fixed for reproducibility; each condition is evaluated over 10 independent seeds.