LEARNING ACROSS THE NOISE SPECTRUM: AN APPROACH TO REINFORCEMENT LEARNING WITH ASYNCHRONOUS SIGNALS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) frameworks assume agents receive complete observation vectors at each timestep. However, real-world robotic systems typically operate in environments with asynchronous signals, i.e. sensors that update at different frequencies. We model **asynchronous environments** as an instance of a noise-parameterized family of partially observable Markov decision processes (POMDPs). Our primary contribution, **Learning Across the Noise Spectrum (LANS)**, is a novel strategy that exposes the agent to multiple simulated noise regimes during training, implemented using Soft Actor-Critic (SAC) with recurrent neural networks (RNNs). By sampling different asynchronicity rates, we encourage the development of robust estimators. We prove that LANS acts a *time-aware* regularization term, equivalent to a Jacobian penalty along time-sensitive directions. Experiments on MuJoCo environments with simulated asynchronicity demonstrate that LANS outperforms alternative methods on a variety of tasks—up to a factor of $> 1.5\times$ in some instances—offering a solution for robotic systems that must operate with imperfect sensory information.

1 INTRODUCTION

Reinforcement learning (RL) (Sutton & Barto, 1998; Mnih et al., 2015) is an established technology for training agents in environments such as competitive games (Silver et al., 2016; Holcomb et al., 2018; Vinyals et al., 2019), conversational language models (Ouyang et al., 2022; Zhu et al., 2023), and industrial manufacturing (Johannink et al., 2018; Zhang et al., 2022). RL research is grounded on the theory of Markov decision processes (MDPs) (Feinberg & Shwartz, 2012), a formalization in which agents receive complete observation vectors at each timestep—an assumption that does not always hold in real-world deployments. Physical systems, particularly robots, must often operate with sensors that update at different and irregular frequencies, creating what we term **asynchronous environments** (Nebot et al., 1999).

In asynchronous settings, observations are composed of signals from multiple sensors (e.g., cameras, thermometers, accelerometers), each with its own renewal rate. At each timestep, agents receive only a subset of signals, based on which sensors have provided new readings. This poses a challenge for RL methods that assume synchronized observations. Partially observable Markov decision processes (POMDPs) (Krishnamurthy, 2016) provide the framework for addressing incomplete information in RL. While general POMDP techniques are applicable to asynchronous environments, the setting has so far eluded dedicated and systematic treatment.

We formalize asynchronous environments as **asynchronous Markov decision processes** (**AMDPs**). Asymptotically, AMDPs behave as POMDPs with a ground-truth state space $S = S_1 \times \cdots \times S_n$ consisting of signals from n sensors. Our key insight is that these environments form a smooth, parameterized family of POMDPs. A noise parameter ω represents the expected ratio of received signals, ranging from $\omega = 0$ (no signal updates) to $\omega = 1$ (complete readings).

The analysis informs our primary contribution: **Learning Across the Noise Spectrum (LANS)**, an RL learning strategy that exposes the agent to varied noise regimes during training. LANS is a regularization technique that builds on the literature of noise regularization (Bishop, 1995; Sajjadi et al., 2016). Our core novelty is a noising process that acts along the *time axis* and works by

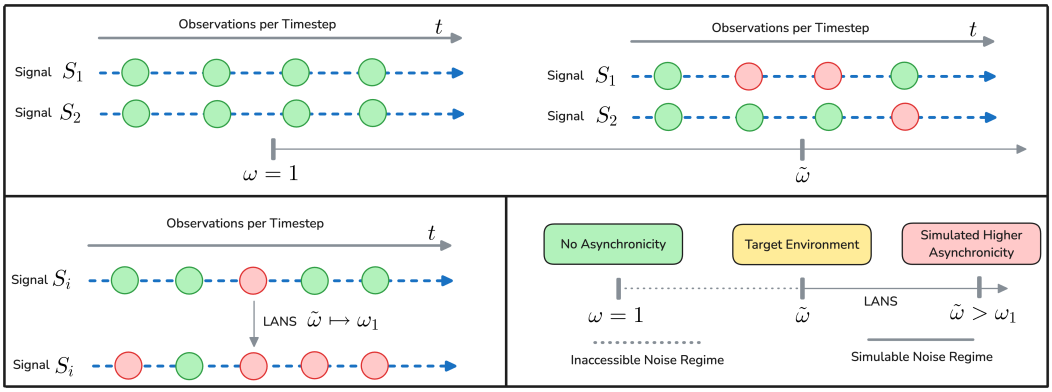

Figure 1: *Top* Parametrization of asynchronous environments by noise ω; lower ω imply sparser sensor updates. *Bottom-Left* LANS mechanism of noise simulation, randomly masking observed signal during training. *Bottom-Right* High-level illustration of LANS, showing the ability to simulate environments with higher noise $\omega \leq \tilde{\omega}$ than the target one, leading to more robust policies.

simulating environments with higher asynchronicity than that of the target task ($\omega \leq \tilde{\omega}$), as depicted in the *Bottom-Right* of Figure 1. LANS promotes robustness, acting as a Jacobian penalty. We prove that this follows from the loss objective implied by LANS and support it with empirical evidence.

We build a new benchmark suite and framework for defining and evaluating AMDPs. Experiments on asynchronous MuJoCo environments (Tassa et al., 2018) show that LANS significantly outperforms baseline methods, by a factor of $1.5\times$ in some cases. The approach provides a bridge between theory and practice for deploying RL in real-world systems.

Our contributions are threefold:

1. We provide a formalization of **asynchronous Markov decision processes** (**AMDPs**) and prove that they admit an approximation as POMDPs. The characterization enables tractable study of the problem and informs our method's design.

2. We develop a benchmark suite of asynchronous MuJoCo tasks for evaluation of RL in asynchronous regimes. The framework is general and can be readily applied to define AMDPs in new environments, facilitating future research and development.

3. We introduce **LANS**, a training strategy that regularizes policies by exposing them to a spectrum of noise regimes, leveraging *time-based* noising processes. We share both theoretical and empirical evidence to support our claims.

2 ASYNCHRONOUS ENVIRONMENTS

We establish the framework for RL in asynchronous environments, assuming familiarity with MDPs and POMDPs. We refer readers to Feinberg & Shwartz (2012); Krishnamurthy (2016) for comprehensive introductions. Our formalization proceeds in three steps: first, we define the concept of a *signal*; second, we extend MDPs to include asynchronicity; and third, we demonstrate the approximation of asynchronous MDPs as POMDPs.

Throughout this paper, we denote MDPs as tuples (S, A, P_a, R_a), where S and A represent the state and action spaces, while P_a and R_a define the transition probabilities and reward functions conditioned on action $a \in A$. POMDPs also include an observation space Ω and conditional observation probabilities $O(\cdot \mid s \in S)$ that govern how states $s \in S$ result in partial observations $\Omega \ni o \sim O(o \mid s \in S)$.

2.1 Asynchronous Markov decision processes

In real-world applications, particularly in physical systems, observations often arrive from multiple sensors, providing the state space with a distinctive structure.

Definition 1. A **signaled state space** is a space S together with subspaces $S_1, \ldots, S_n$, called **channels**, such that $S = \bigoplus_{i \leq n} S_i$.

Each channel S_i corresponds to a distinct signal source (e.g., camera, thermometer, accelerometer) that reads information on the environment. In MDPs, we assume agents observe to the complete state $s_t \in S$ at each timestep $t \in \mathbb{N}$. The assumption fails in asynchronous environments, as sensors update at varying and possibly non-deterministic rates.

To represent missing signals, we extend each signal space S_i with a null element denoted by $\perp$ that indicates an unreceived signal. We define $S_i^* = S_i \cup \perp$ as the extended signal space for channel i.

Definition 2. An **asynchronous Markov decision process** (**AMDP**) extends an MDP $\mathcal{M} = (S, A, P_a, R_a)$ with a signaled space $S = S_1 \times \cdots \times S_n$ by introducing:

- An observation space $\Omega = (\prod_{i=1}^n S_i^*) \times \mathbb{N}^n \times A^k$, where an observation $o \in \Omega$ comprises:
  - A vector of potentially null signals $(s_1^*, \ldots, s_n^*)$, where $s_i^* \in S_i^*$.
  - A vector of timestamps $(\delta^1, \ldots, \delta^n)$ indicating the elapsed time since each sensor's last reading.
  - A history of the k most recent actions $(a_{t-k}, \ldots, a_{t-1})$.
- A signal renewal process for each channel i, described by renewal probabilities $p_i : \mathbb{N} \to [0, 1]$. Each value $p_i(\delta^i)$ is the probability that the i-th sensor provides a new reading after δ^i time steps since its last update.

The **observation component** for the i-th sensor at time $t \in \mathbb{N}$ is the vector $o_{(t,i)} = \left(s^*_{(t,i)}, \delta^i_t\right)$, which includes its value and time elapsed since its last non-null reading.

In an AMDP, the dynamics of the environment are still governed by the transition probabilities P_a, but the agent's observations are subject to asynchronous updates through the channel-specific renewal processes. For each channel i, the probability of receiving a new signal at time t depends on δ_t^i, the time elapsed since its last reading.

In an AMDP, we assume that the transition dynamics $\tilde{P}_a$ in the observation space Ω are separable by channel:

$$\tilde{P}_{a_t}(o_{t+1} \mid o_t) = \prod_{i=1}^{n} \tilde{P}_{(a_t,i)}(o_{(t+1,i)} \mid o_{(t,i)}). \tag{1}$$

In the following, we simplify the notation by removing the channel index i unless necessary, and instead assume the existence of only one signal. Equation 1 guarantees that future developments about individual signals readily apply to general AMDPs.

The dynamics of signal updates are the described by their renewal process through the following equation:

$$\tilde{P}_{a_t}(o_{t+1} \mid o_t) = \begin{cases} p(\delta_t + 1) \cdot P_{a_t}(s_{t+1} \mid s_t) & \text{if } o_{t+1} = (s_{t+1}, 0) \\ 1 - p(\delta_t + 1) & \text{if } o_{t+1} = (\perp, \delta_t + 1) \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

At each time step $t \in \mathbb{N}$, a new reading s_{t+1} is recorded with probability $p\left(\delta_t + 1\right)$ and the elapsed time value δ_{t+1} is reset. Alternatively, a null signal is received and the elapsed time value is increased by one $\delta_t + 1$.

2.2 Relationship between AMDPs and POMDPs

AMDPs are not a special case of POMDPs. To relate the two, one must define conditional probabilities $O\left(o \mid s\right)$ that describe the distribution of observations $o \in \Omega$ w.r.t. to the complete state $s \in S$.

The obstacle is the time-dependent dynamics of signal renewals. In an AMDP, the elapsed time value δ is necessary to derive the observation probabilities, the complete state alone is not sufficient. Instead, we state an approximation that treats AMDPs as POMDPs in the asymptotic regime.

Proposition 1. *Let $\mathcal{A}$ be an AMDP with stationary signal renewal process and well-defined rate $\lambda = \frac{1}{\mu}$. For $t \to \infty$, the process $\mathcal{A}$ can be approximated by a POMDP with conditional observation probabilities:*

$$O\left(o_t \mid s_t\right) \overset{t\to\infty}{\to} \begin{cases} \lambda & \text{if } o = s \\ 1 - \lambda & \text{if } o = \perp. \end{cases} \tag{3}$$

Proof. Appendix A.1, leveraging the elementary renewal theorem (Ross, 2010). □

For sufficiently long-running processes, the renewal probability becomes stationary and equals the inverse of the average inter-arrival period λ. An AMDP is then characterized by the vector $\boldsymbol{\lambda} = (\lambda_i)_{i \leq n} \in \mathbb{R}^n$. We use the notation $\mathcal{A}\left[\boldsymbol{\lambda}\right]$ to denote the POMDP approximating an AMDP $\mathcal{A}$ with converging sensor rates $\lambda_1, \ldots, \lambda_n$.

3 LANS

Learning Across the Noise Spectrum (**LANS**) is an off-policy training strategy for RL agents operating in asynchronous environments. It relies on the insight that we can view AMDPs as instances of a parameterized family of POMDPs, in which a parameter $\omega \in [0, 1]$ controls the degree of observability. By training across differentiated noise regimes—from highly impaired (low ω) to the maximum available in the target environment ($\tilde{\omega}$)—we encourage the development of more robust policies. Algorithm 1 details LANS implementation.

Remarkably, LANS is not restricted to AMDPs and is virtually applicable to any class of parametrized POMDPs. The approach exploits the ability to simulate processes with higher noise ($\omega \leq \tilde{\omega}$) than that of the target environment. This requires knowledge of the noising process. In asynchronous environments, it can be simulated by appropriately masking observations, making AMDPs an ideal test case. Exploration of this idea in other categories of POMDPs is beyond the scope of our work, but we believe it is an exciting direction for future research.

3.1 NOISE PARAMETERIZATION OF AMDPS

In Section 2.2, we prove that an AMDP $\mathcal{A}$ can be approximated as a POMDP $\mathcal{A}\left[\boldsymbol{\lambda}\right]$ with stationary signal rates $\boldsymbol{\lambda} = (\lambda_i)_{i \leq n}$. The vector $\boldsymbol{\lambda}$ is a measure of the environment's stochasticity. A more compact representation is given by the **expected ratio of received signals**, the noise parameter:

$$\omega = \frac{1}{n} \sum_{i=1}^{n} \lambda_i. \tag{4}$$

The case $\omega = 1$ corresponds to the fully observable environment. As $\omega \to 0$, renewals become increasingly rare.

We employ an abuse of notation and use $\mathcal{A}\left[\omega\right]$ to denote the POMDP approximation of $\mathcal{A}$ with noise parameter ω. Formally, $\mathcal{A}\left[\omega\right]$ comprises a collection of POMDPs, one for each $\boldsymbol{\lambda}$ that satisfies Equation 4. In practical implementations, given environment rates $\tilde{\boldsymbol{\lambda}} = (\tilde{\lambda}_i)_{i \leq n}$, we define $\mathcal{A}\left[\omega\right]$ by scaling $\tilde{\boldsymbol{\lambda}}$ by a factor of $c \in [0, 1]$:

$$\mathcal{A}\left[\omega\right] = \mathcal{A}\left[c\tilde{\boldsymbol{\lambda}}\right] \text{ with } \frac{1}{n} \sum_{i \leq n} c\tilde{\lambda}_i = \omega. \tag{5}$$

The map $\omega \mapsto \mathcal{A}\left[\omega\right]$ describes a family of noise-parametrized POMDPs, in which the target environment is realized at $\mathcal{A}\left[\tilde{\omega}\right]$ (corresponding to $c = 1$). The parametrization is smooth, see Appendix A.2 for details.

Algorithm 1 LANS: Learning Across the Noise Spectrum

Input: Parametric networks θ, ϕ_1, ϕ_2 for actor and Q-value functions.
$c \in (0, 1)$ defining minimum noise simulation ($\omega_{\min} = c \cdot \tilde{\omega}$).
K, B, L: optimization steps per rollout, batch size, and trajectory length.
R, T: number of rollouts, max rollout length.
for each rollout $r = 1 \dots R$ **do**
  Collect a rollout $\tau \sim \mathcal{A}[\tilde{\omega}]$ with policy $a_t \sim \pi(\cdot \mid \mu_\theta, \sigma_\theta)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\tau_t\}_{t \le T}$
  for K steps **do**
    $D \leftarrow (\tau^b_{[l_b:l_b+L]}) \sim \mathcal{D}$
    NOISESIMULATION(D, c) ▷ Core LANS mechanism
    if $s^*_{(t,i)} = \perp$ **then** $s^*_{(t,i)} \leftarrow s^*_{(t-\delta^i_t,i)}$ **end if** ▷ Do this across all signals
  end for
  UPDATECRITIC(D)
  UPDATEACTOR(D)
end for
procedure NOISESIMULATION(D, c)
  $(c_b)_{b \le B} \sim \mathcal{U}_{[c,1]}$ ▷ Each c_b is drawn randomly from a uniform distribution
  for each trajectory τ^b in D **do** ▷ Can be performed concurrently
    if Bernoulli $(c_b) = 1$ **then** $s^*_{(t,i)} \leftarrow \perp$ **end if** ▷ Do this across all signals
  end for
end procedure

3.2 METHOD

LANS extends the training domain of an RL algorithm from the target AMDP $\mathcal{A}[\tilde{\omega}]$ to a collection of asynchronous environments $\mathcal{A}[\omega]$ drawn from a noise range $\omega \in [\omega_{\min}, \tilde{\omega}]$.

Algorithm and architecture We use SAC (Haarnoja et al., 2018) with two Q-value networks (Dankwa & Zheng, 2020) as the actor-critic loss. We employ RNNs for the architectural backbone of our models, particularly gated recurrent units (GRU) (Bahdanau et al., 2014). The actor θ and Q-value networks ϕ_1, ϕ_2 use separate RNNs and do not share weights. Mathematically, $\phi_i\left(o_t, a_t, h^{\phi_i}_t\right) = \left(Q_{\phi_i}\left(o_t, a_t, h^{\phi_i}_t\right), h^{\phi_i}_{t+1}\right)$ and $\theta\left(o_t, h^\theta_t\right) = \left(\mu_t, \sigma_t, h^\theta_{t+1}\right)$, i.e. the three functions take the last hidden state $h^{\cdot}_t$ as one of their inputs and compute the next hidden state $h^{\cdot}_{t+1}$ together with Q-values and policy distribution parameters.

Data Training alternates between data collection and optimization (Williams, 1992). During collection, we sample a full rollout $\tau = (o_t, a_t, r_t)_{t \le T}$ by running the SAC stochastic policy on the target environment $\mathcal{A}[\tilde{\omega}]$ and store it on a replay buffer $\mathcal{D}$ (Lin, 1992). After each sampled rollout, we perform K optimization steps on batches drawn from $\mathcal{D}$. A batch $D = \left(\tau^b_{[l_b:l_b+L]}\right) \in \mathbb{R}^{B \times L \times (\dim O + \dim A + 1)}$ is a tensor storing observations, actions, and rewards for B trajectories of length L, padded if necessary.

Noise simulation Rollouts are drawn from the target environment, yielding batches $D \sim \mathcal{A}[\tilde{\omega}]$. Before each optimization step, we simulate noisier samples $D^* \sim \mathcal{A}[\omega_{\min} : \tilde{\omega}]$ by uniformly drawing $\omega_b \in [\omega_{\min}, \tilde{\omega}]$ for each trajectory τ^b and randomly masking a percentage $\frac{\tilde{\omega} - \omega_b}{\tilde{\omega}}$ of signal readings:

$$s^*_t \mapsto \begin{cases} \perp & \text{if Bernoulli}\left(\frac{\tilde{\omega} - \omega_b}{\tilde{\omega}}\right) = 1 \text{ or } s^*_t = \perp \\ s^*_t & \text{otherwise.} \end{cases} \tag{6}$$

The elapsed time values δ_t are also updated accordingly. In Algorithm 1, this step corresponds to the NOISESIMULATION procedure.

As of Definition 2, observation components $o_t = \left[s^*_t, \delta_t, a_{[t-k:t]}\right]$ include a history $a_{[t-k:t]}$ of the k most recent actions and times δ elapsed since each last renewal. Before being provided as inputs to

the models, we replace each unreceived signal $s_t^* = \perp$ with its most recently observed value $s_{t-\delta_t}^*$. The decision provides more meaningful inputs to the networks and is crucial for the developments of Section 3.3.

Hyperparameters *Rollout and data* parameters include the rollout horizon T, the number of optimization steps per rollout K, batch size B (measured in number of sub-trajectories), and the sub-trajectory length L. *Noise* parameters comprise the minimum noise factor c, which defines the lower bound $\omega_{\min} = c \cdot \tilde{\omega}$ of the simulated noise spectrum. *Optimization* parameters include the optimizer's learning rate and the entropy regularization coefficient in SAC. Appendix B reports the values used for our experiments.

3.3 LANS AND REGULARIZATION

LANS encourages learning invariant representations of observations with respect to time-dependent noising, which implies more robust modeling (Zhang et al., 2021a). Here, we elaborate on the regularization mechanism rigorously and prove our core result.

RL algorithms often learn parametric policies $\pi_\theta(a \mid o) = \mathcal{P}_\rho(a, \rho = \theta(o))$ (Bishop, 2006), which are functions $\theta : \Omega \to \mathbb{R}^m$ outputting the parameters ρ of a finite-dimensional distribution $\mathcal{P}_\rho$ on A. Therefore, we assume $\pi : \Omega \to \mathbb{R}^m$ to be a deterministic function.

Given a sequence of observations $(o_t)_{t \leq T}$, the NOISESIMULATION procedure of Algorithm 1 applies a random transformation $G : (o_t)_{t \leq T} \mapsto (o_t^\star)_{t \leq T}$ *projecting* the sequence into a lower dimensional subspace. LANS can be interpreted as implicitly minimizing the following loss function:

$$\mathcal{L}_{\text{LANS}}(o) = \mathbb{V}_{o^\star \sim G(o)}(\pi(o^\star)). \tag{7}$$

Policy predictions must have contained variance w.r.t the random transformations G that project observation sequences into less informative subspaces.

In the primary result of our work, we prove that LANS acts a regularization term:

Proposition 2. *If the policy π is C^1, up to second order in $(o^\star - o)$,*

$$\mathcal{L}_{\text{LANS}}(\pi) = \mathbb{E}_{o \sim \mathcal{D}}\left[\text{tr}\left(J_\pi(o)\,\Sigma_G(o)\,J_\pi(o)^\top\right)\right] + o\left(\|o^\star - o\|^2\right) \tag{8a}$$

$$\Sigma_G(o) \approx \text{diag}\left(p(1-p)\,\Delta_i(o)^2\right) \text{ with } \Delta_i(o) = o_i^\star - o_i. \tag{8b}$$

Proof. Proof provided in Appendix A.3. □

Expanding Equation 8a, we obtain:

$$\mathcal{L}_{\text{LANS}}(\pi) \approx \mathbb{E}_{o \sim \mathcal{D}}\left[\sum_i p(1-p)\,\Delta_i(o)\,\|\partial_{o_i}\pi(o)\|_2^2\right]. \tag{9}$$

The loss $\mathcal{L}_{\text{LANS}}$ acts as a minimization term on the policy's derivatives—i.e., a Jacobian penalty (Rifai et al., 2011; Sokolić et al., 2017; Novak et al., 2018). The effect is proportional on the value of $\Delta_i(o)$, which quantifies the shift between two consecutive readings. The regularization penalty is *strongest in the directions most sensitive to the dynamics of renewal*.

4 RELATED WORK

Partially observable Markov decision processes Extensive research addresses POMDPs in RL. It includes learning the dynamics of systems from individual frames (Hausknecht & Stone, 2015; Payne et al., 2024), path-planning (Xie et al., 2021), and multi-agent cooperation (Oroojlooyjadid & Hajinezhad, 2019; Papoudakis et al., 2020). Other works leverage POMDPs to investigate traditional RL settings such as Meta-RL (Schmidhuber, 1987; Soorki et al., 2023; Parisotto et al., 2020) and generalization performance (Rajeswaran et al., 2017; Agarwal et al., 2020). Memory architectures, especially RNNs, have quickly become popular as the standard approach in partially

observable environments (Schmidhuber, 1990; Lu et al., 2023). Model-based approaches learn to recover ground-truths states from partial observations (Han et al., 2020; Ren et al., 2023; Lee et al., 2020). On the opposite, model-free architectures have objectives that focus solely on reward maximization (Meng et al., 2021; Mirowski et al., 2017). Ni et al. (2022) demonstrate that model-free approaches are sufficient to tackle POMDPs in most scenarios, provided careful design of the architectural backbone.

Noise-based regularization Regularization (Hastie et al., 2009) aims at constraining the complexity of machine learning models with positive effects on robustness (Jeong & Shin, 2020; Li & Zhang, 2021) and generalization (Loshchilov & Hutter, 2017; Wan et al., 2013; Li et al., 2018) (but see Zhang et al. (2017; 2021b) for counterarguments). Among regularization techniques, noise-based (Sajjadi et al., 2016) ones work by introducing noise on the inputs (An, 1996; Bishop, 1995), weights (Fortunato et al., 2018; Rakin et al., 2018), or outputs (Zhang & Sabuncu, 2018). Dropout (Srivastava et al., 2014; Gal & Ghahramani, 2015) is a popular regularization strategy that relies on random masking of intermediate network's reprsentations. In Section 3.3, we show that LANS works as a noise-based regularization approach. Denoising auto-encoders (DAE) (Vincent et al., 2008) are especially relevant to our work, due to their noise-injection mechanism based on masking of random coordinates. In the experiments, we show that for AMDPs, LANS is a more effective strategy than DAEs.

Data augmentation LANS' core mechanism is the NOISESIMULATION procedure of Algorithm 1, which applies transformations to the input data before performing a gradient step. Data augmentation (Shorten & Khoshgoftaar, 2019; Park et al., 2019) refers to the class of strategies that exploit symmetries (Chen et al., 2020; Bjerrum, 2017) in the data distribution to simulate larger datasets. Data augmentation is prevalent in RL (Sun et al., 2024), with most works focusing on visual observations (Yarats et al., 2021b;a; Zhang et al., 2021a). Despite the similarity, data augmentation differs from LANS because of its focus on transformations that preserve the semantics of samples (Trabucco et al., 2024). Noising is occasionally treated as a data augmentation strategy (Iglesias et al., 2023); Xie et al. (2020) discuss the advantages in deploying realistic noising processes, which is what LANS achieves w.r.t AMDPs.

5 EXPERIMENTS

We share a framework and benchamrk suite for AMDPs definition, and we conduct experiments on standard RL environments modified to incorporate asynchronicity. We compare LANS against alternative regularization techniques—DAEs and Gaussian additive noise—and a no-regularization baseline, proving that our strategy (i) improves learning performance and/or variance in asynchronous environments and (ii) it is more effective than other forms of regularization.

5.1 ASYNCHRONOUS RL AND MUJOCO ENVIRONMENTS

We develop a dedicated codebase for asynchronous RL environments. The library builds on top of PyTorch (Ansel et al., 2024), Gymnasium (Towers et al., 2024), and TorchRL (Bou et al., 2023), and provides a modular implementation of AMDPs. It allows users to specify processes for signals, simulate asynchronicity, and interface with RL pipelines. We include detailed documentation to facilitate reproducibility and adoption.

For evaluation, we adapt the widely used MuJoCo continuous control benchmarks Todorov et al. (2012) from the DeepMind Control Suite (Tassa et al., 2018). In our formulation, each joint state—comprising position, velocity, and rotational coordinates—is modeled as a signal. Asynchronicity is introduced by assigning each channel a renewal process. Concretely, we sample renewal intervals from a Gamma distribution $\delta^i \sim \text{Gamma}\,(1, 1)$, truncated to $\delta \leq 4$ steps. This results in an average signal ratio of $\tilde{\omega} \approx 0.39$, i.e., agents observe $\sim 40\%$ of the signals, on average.

We construct a suite of asynchronous MuJoCo benchmarks—including *Async-HalfCheetah*, *Async-Hopper*, *Async-Ant*, *Async-Walker2d*, and *Async-Reacher*. By grounding our experiments in these standard tasks (Duan et al., 2016), we retain comparability with prior work while introducing a controlled source of asynchronicity that stresses the ability of algorithms to operate on AMDPs.

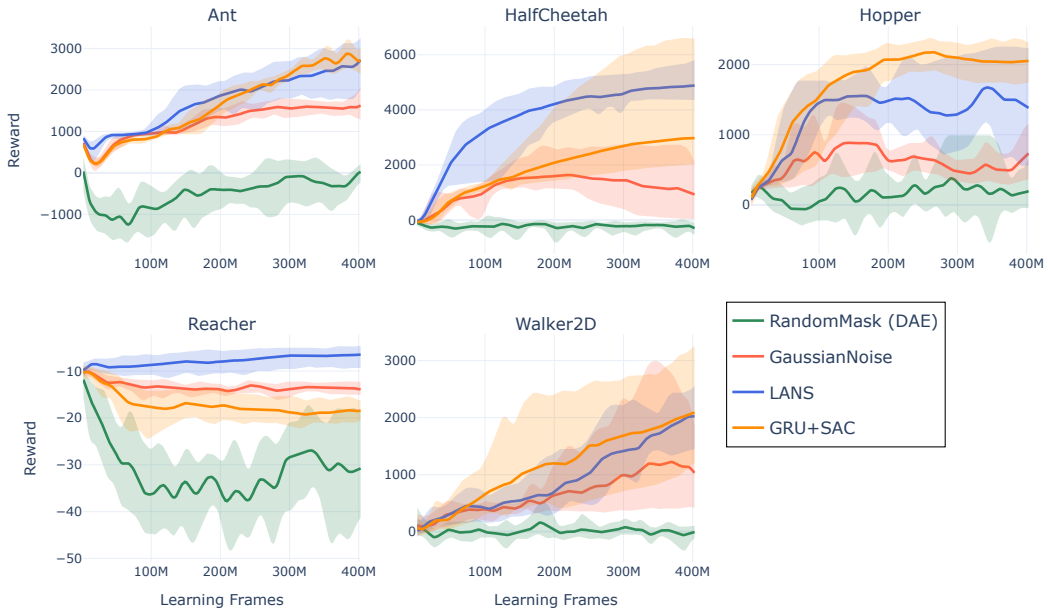

Figure 2: Training curves for LANS, GRU+SAC, DAE, and GAUSSNOISE, averaged across five runs per environment. For all environments except Async-Hopper, LANS either outperforms other methods after $400M$ learning frames or exhibits more controlled variance.

Table 1: Average reward per episode after $400M$ training frames, averaged across 5 runs per task, variance in parentheses. Bold values indicate best within the environment. LANS achieves either best returns on Async-HalfCheetah and Async-Reacher, or controlled variance for other environments, except Async-Hopper. Alternative regularization algorithms do not display competitive results.

ALGORITHM	Ant (K)	HalfCheetah (K)	Hopper (K)	Walker2D (K)	Reacher
LANS	2.6 (0.8)	**4.9** (0.6)	1.4 (0.8)	**2.0** (0.5)	**−6.7** (2.9)
GRU+SAC	**2.8** (0.6)	3.0 (2.1)	**2.1** (0.2)	**2.0** (0.9)	−18 (2.2)
DAE	0.0 (0.3)	−0.2 (0.05)	0.2 (0.4)	0.0 (0.1)	0.0 (0.1)
GAUSSNOISE	1.6 (0.5)	1.0 (0.6)	0.7 (0.5)	1.1 (0.4)	1.1 (0.4)

5.2 ANALYSIS ACROSS ENVIRONMENTS

We compare LANS with three baselines across five environments, while retaining the same architectural backbone: a SAC loss function with GRU networks modeling actor and critic functions. **GRU+SAC** is the standard baseline, trained without modifications to the original algorithm. Each experiment is run five times to control for random oscillations and to estimate variance.

LANS Our approach, detailed in Algorithm 1. Compared to the standard baseline, it requires setting the additional hyperparameter $c \in [0, 1]$. We choose $c = 0.5$ for all experiments except Hopper, where a value of $c = 0.1$ shows minor performance improvements.

DAE We borrow a technique from denoising autoencoder (Vincent et al., 2008), consisting of training the network while masking a fixed percentage $\nu \in [0, 1]$ of input coordinates at every gradient step. The decision is informed by the method's similarity to LANS: it is obtained by replacing the NOISESIMULATION procedure in Algorithm 1 with one that masks input coordinates in a *time-unaware* fashion. We adopt $\nu = 0.5$ for all our experiments.

GAUSSNOISE Gaussian additive noise applied to inputs underlies an established regularization technique (Bishop, 1995). GAUSSNOISE modifies the NOISESIMULATION procedure of Algorithm 1, replacing random signal masking with Gaussian shifts.

Figure 2 shows training curves for LANS and other algorithms, displaying the evolution of average episode return per frame. Table 1 reports average episode return of each model after $400M$ learning frames. In most tasks, LANS achieves either the highest returns or the most controlled variance. In the case of Async-HalfCheetah, the performance is better for a factor of $> 1.5\times$ than the one of the standard baseline. Async-Hopper is the only environment in which LANS' underperfoms; it the easiest task among the five and therefore might not provide sufficient complexity to detect regularization impact. For Async-Ant and Async-Walker2d, there are no visible performance gains, but significantly lower variance. Across all experiments, competing regularization algorithms fail to learn meaningful policies, demonstrating their differences with LANS to be crucial.

5.3 CURVATURE ANALYSIS

In Section 3.3, we establish LANS as a regularization mechanism. To validate this claim, we estimate the curvature of policies and compare them with GRU+SAC. We approximate the Hessian of the policy output with respect to observations using finite differences, and report its Frobenius norm as a curvature measure. Estimates are computed over $1K$ observations sampled from the evaluation environment.

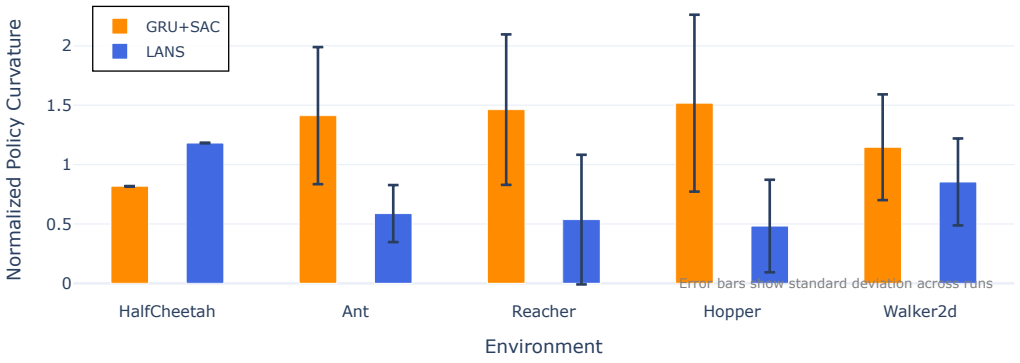

Figure 3: Estimated policy curvature across tasks. Bars show the Frobenius norm of the Hessian of the learned policy, averaged over $1,000$ observations and five random seeds, and normalized. Error bars denote standard deviation across seeds. LANS produces policies with lower curvature, except for Async-HalfCheetah.

Figure 3 shows the average curvature values for policies trained with and without LANS, which have been normalized to preserve scale. Results are aggregated across five random seeds, with error bars representing standard deviation. With the exception of Async-HalfCheetah, LANS yields policies with lower curvature, sometimes by a factor of < 0.5. By exposing agents to a spectrum of noise regimes, LANS acts as a regularizer that penalizes excessive curvature of the policy function.

6 CONCLUSION

We introduce **Learning Across the Noise Spectrum (LANS)**, an approach to reinforcement learning in asynchronous environments. We formalize asynchronous environments as noise-parameterized POMDPs, and develop a theoretical foundation linking LANS to regularization. We demonstrate performance improvements across modified MuJoCo benchmarks. Our research establishes LANS as an effective solution for real-world robotic systems operating with imperfect sensory information, bridging a gap between theoretical RL frameworks and practical deployment in industrial settings.

REFERENCES

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2020.

Luigi Ambrosio, Nicola Giglio, and Giuseppe Savaré. *The Wasserstein Distance and its Behaviour along Geodesics*, pp. 151–165. Birkhäuser Basel, Basel, 2008. ISBN 978-3-7643-8722-8. doi: 10.1007/978-3-7643-8722-8_9. URL https://doi.org/10.1007/978-3-7643-8722-8_9.

Guozhong An. The effects of adding noise during backpropagation training on a generalization performance. *Neural Computation*, 8(3):643–674, 1996. doi: 10.1162/neco.1996.8.3.643.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ASPLOS '24, pp. 929–947, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703850. doi: 10.1145/3620665.3640366. URL https://doi.org/10.1145/3620665.3640366.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL https://api.semanticscholar.org/CorpusID:11212020.

Chris M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7(1):108–116, 01 1995. ISSN 0899-7667. doi: 10.1162/neco.1995.7.1.108. URL https://doi.org/10.1162/neco.1995.7.1.108.

Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, January 2006. URL https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

Esben Jannik Bjerrum. SMILES enumeration as data augmentation for neural network modeling of molecules. *CoRR*, abs/1703.07076, 2017. URL http://arxiv.org/abs/1703.07076.

Albert Bou, Matteo Bettini, Sebastian Dittert, Vikash Kumar, Shagun Sodhani, Xiaomeng Yang, Gianni De Fabritiis, and Vincent Moens. Torchrl: A data-driven decision-making library for pytorch, 2023.

Shuxiao Chen, Edgar Dobriban, and Jane H. Lee. A group-theoretic framework for data augmentation. *J. Mach. Learn. Res.*, 21(1), January 2020. ISSN 1532-4435.

Stephen Dankwa and Wenfeng Zheng. Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent. In *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, ICVISP 2019, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450376259. doi: 10.1145/3387168.3387199. URL https://doi.org/10.1145/3387168.3387199.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1329–1338, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/duan16.html.

E.A. Feinberg and A. Shwartz. *Handbook of Markov Decision Processes: Methods and Applications*. International Series in Operations Research & Management Science. Springer US, 2012. ISBN 9781461508052. URL https://books.google.com/books?id=TpwKCAAAQBAJ.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alexander Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. In *Proceedings of the International Conference on Representation Learning (ICLR 2018)*, Vancouver (Canada), 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2015. URL https://api.semanticscholar.org/CorpusID:160705.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Dongqi Han, Kenji Doya, and Jun Tani. Variational recurrent models for solving partially observable control tasks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1lL4a4tDB.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Basis Expansions and Regularization*, pp. 139–189. Springer New York, New York, NY, 2009. ISBN 978-0-387-84858-7. doi: 10.1007/978-0-387-84858-7_5. URL https://doi.org/10.1007/978-0-387-84858-7_5.

Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents (AAAI-SDMIA15)*, Arlington, Virginia, USA, November 2015. URL http://www.cs.utexas.edu/users/ai-lab?hausknecht:sdmia15.

Sean D. Holcomb, William K. Porter, Shaun Van Ault, Guifen Mao, and Jin Wang. Overview on deepmind and its alphago zero ai. *Proceedings of the 2018 International Conference on Big Data and Education*, 2018. URL https://api.semanticscholar.org/CorpusID:44110863.

Guillermo Iglesias, Edgar Talavera, Ángel González-Prieto, Alberto Mozo, and Sandra Gómez-Canaval. Data augmentation techniques in time series domain: a survey and taxonomy. *Neural Computing and Applications*, 35(14):10123–10145, May 2023. ISSN 1433-3058. doi: 10.1007/s00521-023-08459-3. URL https://doi.org/10.1007/s00521-023-08459-3.

Jongheon Jeong and Jinwoo Shin. Consistency regularization for certified robustness of smoothed classifiers. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 10558–10570. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/77330e1330ae2b086e5bfcae50d9ffae-Paper.pdf.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6023–6029, 2018. URL https://api.semanticscholar.org/CorpusID:54464184.

Vikram Krishnamurthy. *Partially Observed Markov Decision Processes: From Filtering to Controlled Sensing*. Cambridge University Press, 2016.

Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: deep reinforcement learning with a latent variable model. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Dongyue Li and Hongyang Zhang. Improved regularization and robustness for fine-tuning in neural networks. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:243847344.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex Chichung Kot. Domain generalization with adversarial feature learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018. URL https://api.semanticscholar.org/CorpusID:52833113.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992. doi: 10.1007/BF00992699. URL https://doi.org/10.1007/BF00992699.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017. URL https://api.semanticscholar.org/CorpusID:53592270.

Chris Lu, Yannick Schroecker, Albert Gu, Emilio Parisotto, Jakob Foerster, Satinder Singh, and Feryal Behbahani. Structured state space models for in-context reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 47016–47031. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/92d3d2a9801211ca3693ccb2faa1316f-Paper-Conference.pdf.

Lingheng Meng, Rob Gorbet, and Dana Kulić. Memory-based deep reinforcement learning for pomdps. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5619–5626. IEEE Press, 2021. doi: 10.1109/IROS51168.2021.9636140. URL https://doi.org/10.1109/IROS51168.2021.9636140.

Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andy Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to navigate in complex environments. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJMGPrcle.

Kosto V. Mitov and Edward Omey. *Discrete Time Renewal Processes*, pp. 53–65. Springer International Publishing, Cham, 2014. ISBN 978-3-319-05855-9. doi: 10.1007/978-3-319-05855-9_2. URL https://doi.org/10.1007/978-3-319-05855-9_2.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015. URL https://api.semanticscholar.org/CorpusID:205242740.

Eduardo Mario Nebot, Mohammad Bozorg, and Hugh F. Durrant-Whyte. Decentralized architecture for asynchronous sensors. *Autonomous Robots*, 6:147–164, 1999. URL https://api.semanticscholar.org/CorpusID:35608625.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *International Conference on Machine Learning*, pp. 16691–16723. PMLR, 2022.

Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJC2SzZCW.

Afshin Oroojlooyjadid and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53:13677–13722, 2019. URL https://api.semanticscholar.org/CorpusID:199543559.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS Datasets and Benchmarks*, 2020. URL https://api.semanticscholar.org/CorpusID:235417602.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7487–7498. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/parisotto20a.html.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. Specaugment: A simple data augmentation method for automatic speech recognition. In *Interspeech*, 2019. URL https://api.semanticscholar.org/CorpusID:121321299.

John Payne, Aishwaryaprajna, and Peter R. Lewis. Online learning of temporal dependencies for the sustainable foraging problem. *2024 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pp. 67–72, 2024. URL https://api.semanticscholar.org/CorpusID:270870515.

Aravind Rajeswaran, Kendall Lowrey, Emanuel V. Todorov, and Sham M Kakade. Towards generalization and simplicity in continuous control. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/9ddb9dd5d8aee9a76bf217a2a3c54833-Paper.pdf.

Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 588–597, 2018. URL https://api.semanticscholar.org/CorpusID:53716271.

Tongzheng Ren, Chenjun Xiao, Tianjun Zhang, Na Li, Zhaoran Wang, sujay sanghavi, Dale Schuurmans, and Bo Dai. Latent variable representation for reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=mQpmZVzXK1h.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 833–840, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Sheldon M. Ross. Chapter 7 - renewal theory and its applications. In Sheldon M. Ross (ed.), *Introduction to Probability Models (Tenth Edition)*, pp. 421–495. Academic Press, Boston, tenth edition edition, 2010. ISBN 978-0-12-375686-2. doi: https://doi.org/10.1016/B978-0-12-375686-2.00003-0. URL https://www.sciencedirect.com/science/article/pii/B9780123756862000030.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 1171–1179, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Jurgen Schmidhuber. Evolutionary principles in self-referential learning. *On learning how to learn: The meta-meta-... hook.) Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1(2):48, 1987.

Jürgen Schmidhuber. Reinforcement learning in markovian and non-markovian environments. In *Proceedings of the 4th International Conference on Neural Information Processing Systems*, NIPS'90, pp. 500–506, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc. ISBN 1558601848.

Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48, 2019. URL https://api.semanticscholar.org/CorpusID:195811894.

David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel R. D. Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280, 2017. doi: 10.1109/TSP.2017.2708039.

Mehdi Naderi Soorki, Hossein Aghajari, Sajad Ahmadinabi, Hamed Bakhtiari Babadegani, Christina Chaccour, and Walid Saad. Catch me if you can: Deep meta-rl for search-and-rescue using lora uav networks. *IEEE Transactions on Mobile Computing*, 24:763–778, 2023. URL https://api.semanticscholar.org/CorpusID:259075687.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15: 1929–1958, 2014. URL https://api.semanticscholar.org/CorpusID:6844431.

Yuewen Sun, Erli Wang, Biwei Huang, Chaochao Lu, Lu Feng, Changyin Sun, and Kun Zhang. Acamda: Improving data efficiency in reinforcement learning through guided counterfactual data augmentation. In *AAAI Conference on Artificial Intelligence*, 2024. URL https://api.semanticscholar.org/CorpusID:268712346.

Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction. *IEEE Trans. Neural Networks*, 9:1054–1054, 1998. URL https://api.semanticscholar.org/CorpusID:60035920.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy P. Lillicrap, and Martin A. Riedmiller. Deepmind control suite. *ArXiv*, abs/1801.00690, 2018. URL https://api.semanticscholar.org/CorpusID:6315299.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. URL https://api.semanticscholar.org/CorpusID:5230692.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

Brandon Trabucco, Kyle Doherty, Max A Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ZWzUA9zeAg.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 1096–1103, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390294. URL https://doi.org/10.1145/1390156.1390294.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350 – 354, 2019. URL https://api.semanticscholar.org/CorpusID:204972004.

Li Wan, Matthew D. Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, 2013. URL https://api.semanticscholar.org/CorpusID:2936324.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992. ISSN 1573-0565. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6256–6268. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/44feb0096faa8326192570788b38c1d1-Paper.pdf.

Ronglei Xie, Zhi Jun Meng, Lifeng Wang, Haochen Li, Kaipeng Wang, and Zhenping Wu. Unmanned aerial vehicle path planning algorithm based on deep reinforcement learning in large-scale and dynamic environments. *IEEE Access*, 9:24884–24900, 2021. URL https://api.semanticscholar.org/CorpusID:231920142.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *Deep RL Workshop NeurIPS 2021*, 2021a. URL https://openreview.net/forum?id=L5HKN-IsdSE.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=GY6-6sTvGaf.

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=-2FCwDKRREu.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM*, 64(3):107–115, February 2021b. ISSN 0001-0782. doi: 10.1145/3446776. URL https://doi.org/10.1145/3446776.

Yi Zhang, Haihua Zhu, Dunbing Tang, Tong Zhou, and Yong Gui. Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems. *Robotics Comput. Integr. Manuf.*, 78:102412, 2022. URL https://api.semanticscholar.org/CorpusID:250362179.

Zhilu Zhang and Mert Rory Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 32:8792–8802, 2018. URL https://api.semanticscholar.org/CorpusID:29164161.

Banghua Zhu, Michael Jordan, and Jiantao Jiao. Principled reinforcement learning with human feedback from pairwise or k-wise comparisons. In Andreas Krause, Emma Brunskill, Kyunghyun

Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 43037–43067. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/zhu23f.html.

A PROOFS

In this appendix, we provide proofs to the propositions and claims contained in the main paper. Throughout this section, we restrict the discussion to the case where an AMDP is described by only one signal. This way, we omit the index i (e.g. $o_{(t,i)} = \left(s^*_{(t,i)}, \delta^i_t\right)$ becomes $o_t = (s^*_t, \delta_t)$) referring to channels in our notation, easing readability. Equation 1 ensures that the multi-signals case can be reduced to the single one.

A.1 APPROXIMATION OF AMDPS AS POMDPS

We prove Proposition 1, stating that AMPDs—as defined in Definition 2—can be approximated by POMDPs in the asymptotic regime. The proof relies on the theory of renewal processes and especially on the Erdös-Feller-Pollard theorem. Before proceeding, we provide the essential definitions and assumptions that underlie the next developments. We refer to Mitov & Omey (2014) for a source on the material exposed here.

Definition 3 (Renewal Process)**.** Let $X_n \in \mathbb{N}^+$ be a sequence of i.i.d. positive random variables. The increasing sequence of partial sums variables S_n defined as

$$S_n = \sum_{i=0}^{n} X_i \tag{10}$$

is called a **renewal process** and the S_n **renewal times**.

A renewal process describes the dynamics of events that occur at possibly non-deterministic frequency, with independent and identically distributed renewal intervals.

Crucially, renewal processes describe the asynchronicity of signals in an AMDP. Given a sequence of observations $(o_t = (s^*_t, \delta_t))_{t\geq 0}$, we can restrict to the sub-sequence of observed readings $(o_{t_n})_{n\geq 0}$ such that $\delta_{t_n} = 0$ and $\delta_u \neq 0, \forall u \neq t_n$. The sequence $\Delta_n = t_{n+1} - t_n$ is that of the renewal intervals between readings. The variables Δ_n are i.i.d. as a consequence of Equation 2:

$$P(\Delta_n = k) = p(k+1) \sum_{j=1}^{k} (1 - p(j)). \tag{11}$$

In particular, $P(\Delta_n = k)$ does not depend on n nor the value of Δ_m for $m < n$. Therefore, the sequence of reading times $\Sigma_n = \sum_{i=1}^{n} \Delta_i$ is a renewal process.

Let $r_t = P(\delta_t = 0)$ denote the probability that there is an observed signal at time $t \geq 0$. We have

$$r_t = \sum_{n=1}^{\infty} P(\Sigma_n = t). \tag{12}$$

To prove Proposition 1, we must prove that $\lim_{t\to\infty} r_t = \lambda$ where $\lambda = \frac{1}{\mathbb{E}[\Delta_n]}$ indicates the signal renewal rate, which is the reciprocal of the average renewal time.

The Erdös-Feller-Pollard theorem directly proves Equation 12, but it relies on two assumptions about the renewal process induced by Δ_n.

Assumption 1. *The average inter-arrival renewal time $\mu = \mathbb{E}[\Delta_n]$ is finite.*

Assumption 2. *For each arithmetic progression $k\mathbb{N} = \{kn \mid n \in \mathbb{N}\}$ with $k \geq 2$, we have*

$$\sum_{i \in k\mathbb{Z}} P(\Delta_n = i) < 1. \tag{13}$$

The first assumption poses a probabilistic bound on renewal times. Assumption 2 states that the renewal process is **aperiodic**, meaning there is no $k \geq 2$ such that renewals only occur at times $t = k \cdot u \in k\mathbb{N}$.

We remark that for the purpose of our work, it is safe to assume that AMDPs satisfy Assumptions 1 and 2. Sensors with infinite average renewal times would provide unreliable information that can potentially be disrupted indefinitely. For sensors with renewal times concentrated on a periodic subsequence $t \in k\mathbb{N}$, we can express the problem in terms of observations drawn at time-steps in $k\mathbb{N}$ and Assumption 2 would hold.

Finally, we state Erdös-Feller-Pollard theorem.

Theorem 1 (Erdös-Feller-Pollard)**.** *Let $S_n = \sum_{i=1}^{n} X_n$ be a renewal process satisfying Assumptions 1 and 2 and let $\mu = \mathbb{E}[X_n]$. Then*

$$r_t = \sum_{n=1}^{\infty} P(S_n = t) \overset{t\to\infty}{\to} \frac{1}{\mu}. \tag{14}$$

Proposition 1 follows as a corollary.

Proposition 3. *Let $\mathcal{A}$ be an AMDP with stationary signal renewal processes and well-defined rates $\lambda_i = \frac{1}{\mu_i}$. The process $\mathcal{A}$ can be approximated by a POMDP with conditional observation probabilities:*

$$O(o \mid s) = \prod_{i=1}^{n} O_i(o_i \mid s_i) \tag{15}$$

where for each channel i:

$$O_i(o_i \mid s_i) = \begin{cases} \lambda_i & \text{if } o_i = s_i \\ 1 - \lambda_i & \text{if } o_i = \perp. \end{cases} \tag{16}$$

Proof. Since Equation 2 separates transition probabilities by signals, we will restrict our proof to the one-signal case, omitting indexes indicating channels. The full statement of the proposition is obtained by application across signals.

Equation 16 states that the variable $R_t = \mathbf{1}_{\{o_t = s_t\}} \sim \text{Bernoulli}(\lambda)$. For an AMPD, the variable $Q_t = \mathbf{1}_{\{\delta_t = 0\}} \sim \text{Bernoulli}(r_t)$. Theorem 1 implies $r_t \overset{t\to\infty}{\to} \lambda = \frac{1}{\mu}$. □

A.2 Smoothness of noise-parametrization of AMPDs

In Section 3.1, we state that the AMDPs $\mathcal{A}[\omega]$ form a parametrization of POMDPs that is **smooth** for $\omega \in [0, 1]$. We formalize and prove this notion.

The AMDPs $\mathcal{A}[\omega]$ are characterized by their conditional probabilities $O_\omega : \Omega \times S \to \mathbb{R}$, $(o, s) \overset{\omega}{\mapsto} O_\omega(o \mid s)$. The parametrization is smooth if the $O_\cdot : [0, 1] \to (\Omega \times S \to \mathbb{R})$ is smooth w.r.t. $\omega \in [0, 1]$ in the Wassertein space $\mathcal{P}_1(\Omega \times S)$ of probability measures Ambrosio et al. (2008).

Definition 4 (Wassertein metric)**.** Let (M, d) be a metric space and P, Q two probability distributions on M with finite expected value. The **Wassertein distance** $W_1(P, Q)$ between P and Q is defined as

$$W_1(P, Q) = \inf_{\gamma \in \Gamma(P,Q)} \mathbb{E}_{(x,y)\sim\gamma}[d(x, y)] \tag{17}$$

where $\Gamma(P, Q)$ denotes the set of **couplings** between P and Q, i.e. a coupling γ is a distribution whose marginal probabilities equal P and Q respectively. The metric space $(\mathcal{P}_1(M), W_1)$ is called the **Wassertein space**.

Before proceeding to verify the smoothness of $\mathcal{A}[\omega]$, we must specify the structure of a metric space for the observation space Ω. While in general $S \subseteq \mathbb{R}^m$ for some $m \in \mathbb{N}$, the existence of $\perp \in \Omega$ implies that Ω does not naturally embed into a real space. In practice, however, we can choose $\perp = \mathbf{0}$ and equate the lack of a signal with the null vector. More than just a theoretical trick, this choice describes the concrete implementation of LANS as described in Section 3.2, since unobserved

signals are replaced with their most recent reading, or masked to the null vector if no most recent observation is available.

Proposition 4. *For each* $\omega \in [0,1]$

$$\lim_{h\to 0} \frac{W_1\left(O_{\omega+h}\left(\cdot \mid s\right), O_\omega\left(\cdot \mid s\right)\right)}{|h|} = d\left(s, \perp\right). \tag{18}$$

Proof. We observe that $O_{\omega+h}\left(\cdot \mid s\right)$ and $O_\omega\left(\cdot \mid s\right)$ have both support on the two-elements set $\{s, \perp\}$. Therefore, any coupling $\gamma_{\omega,h}\left(s\right)$ is identified by a probability matrix

$$\gamma_{\omega,h}\left(s\right) \sim \begin{pmatrix} B_s = P\left(s,s\right) & F_s = P\left(s,\perp\right) \\ F_\perp = P\left(\perp,s\right) & B_\perp = P\left(\perp,\perp\right) \end{pmatrix}. \tag{19}$$

We have

$$B_s + F_s = O_\omega\left(s \mid s\right) \tag{20a}$$
$$B_s + F_\perp = O_{\omega+h}\left(s \mid s\right) \tag{20b}$$
$$F_s + F_\perp = P\left(O_\omega\left(\cdot \mid s\right) \neq O_{\omega+h}\left(\cdot \mid s\right)\right). \tag{20c}$$

From Equations 20a and 20b it follows

$$|h| = \left|O_\omega\left(s \mid s\right) - O_{\omega+h}\left(s \mid s\right)\right| = \tag{21a}$$
$$= \left|B_s + F_s - B_s - F_\perp\right| = \tag{21b}$$
$$= \left|F_s - F_\perp\right|. \tag{21c}$$

Because F_s and $F_\perp$ are both non-negative, it must hold $F_s + F_\perp \geq |h|$.

We now observe

$$\mathbb{E}_{(x,y)\sim\gamma_{\omega,h}(s)}\left[d\left(x,y\right)\right] = P_{(x,y)\sim\gamma_{\omega,h}(s)}\left(x \neq y\right) d\left(s,\perp\right) = \tag{22a}$$
$$= \left(F_s + F_\perp\right) d\left(s,\perp\right) \geq \tag{22b}$$
$$\geq |h|\, d\left(s,\perp\right), \tag{22c}$$

which implies

$$W_1\left(O_{\omega+h}\left(\cdot \mid s\right), O_\omega\left(\cdot \mid s\right)\right) \geq |h|\, d\left(s,\perp\right). \tag{23}$$

For $h > 0$, a possible coupling $\tilde{\gamma}_{\omega,h}\left(s\right)$ is given by

$$\tilde{\gamma}_{\omega,h}\left(s\right) = \begin{pmatrix} \omega & 0 \\ |h| & 1-\omega-h \end{pmatrix}, \tag{24}$$

and for $h < 0$

$$\tilde{\gamma}_{\omega,h}\left(s\right) = \begin{pmatrix} \omega+h & |h| \\ 0 & 1-\omega \end{pmatrix}, \tag{25}$$

which together imply

$$\mathbb{E}_{(x,y)\sim\tilde{\gamma}_{\omega,h}(s)}\left[d\left(x,y\right)\right] = |h|\, d\left(s,\perp\right) \implies \tag{26a}$$
$$\implies W_1\left(O_{\omega+h}\left(\cdot \mid s\right), O_\omega\left(\cdot \mid s\right)\right) \leq |h|\, d\left(s,\perp\right). \tag{26b}$$

Combining Equations 23 and 26b we obtain

$$W_1\left(O_{\omega+h}\left(\cdot \mid s\right), O_\omega\left(\cdot \mid s\right)\right) = |h|\, d\left(s,\perp\right) \tag{27}$$

and finally

$$\lim_{h\to 0} \frac{W_1\left(O_{\omega+h}\left(\cdot \mid s\right), O_\omega\left(\cdot \mid s\right)\right)}{|h|} = \lim_{h\to 0} \frac{|h|}{|h|} d\left(s,\perp\right) = d\left(s,\perp\right). \tag{28}$$

□

A.3 LANS AND REGULARIZATION

We continue the discussion in Section 3.3 and prove Proposition 2. We recall the implicit LANS loss at an input $o \in \Omega$:

$$\mathcal{L}_{\text{LANS}}(o) \doteq \operatorname{Var}_{o^\star \sim G(o)}\big(\pi(o^\star)\big) = \mathbb{E}_{o^\star \sim G(o)}\left[\left\|\pi(o^\star) - \mathbb{E}_{u^\star \sim G(o)}[\pi(u^\star)]\right\|_2^2\right]. \tag{29}$$

Proposition 5. *Suppose π is continuously differentiable in a neighborhood of o, and let $\Delta \doteq o^\star - o$. Then, up to second order in Δ,*

$$\mathcal{L}_{\text{LANS}}(o) = \operatorname{tr}\big(J_\pi(o)\,\Sigma_G(o)\,J_\pi(o)^\top\big) + o(\|\Delta\|^2), \tag{30}$$

where $J_\pi(o) \in \mathbb{R}^{m\times d}$ is the Jacobian of π at o, and $\Sigma_G(o) = \operatorname{Cov}(o^\star - o \mid o)$ is the covariance of the perturbation induced by G.

Proof. By a first-order Taylor expansion of π around o we have

$$\pi(o^\star) = \pi(o) + J_\pi(o)\,\Delta + R(\Delta), \tag{31}$$

where the remainder $R(\Delta)$ satisfies $\|R(\Delta)\| = o(\|\Delta\|)$. Taking expectation under $G(o)$ gives

$$\mathbb{E}[\pi(o^\star) \mid o] = \pi(o) + J_\pi(o)\,\mathbb{E}[\Delta \mid o] + o(\|\Delta\|). \tag{32}$$

Subtracting this mean from the expansion yields

$$\pi(o^\star) - \mathbb{E}[\pi(o^\star) \mid o] = J_\pi(o)\,(\Delta - \mathbb{E}[\Delta \mid o]) + o(\|\Delta\|). \tag{33}$$

Therefore, to second order in Δ,

$$\begin{aligned}
\mathcal{L}_{\text{LANS}}(o) &= \mathbb{E}\Big[\big\|\pi(o^\star) - \mathbb{E}[\pi(o^\star) \mid o]\big\|_2^2\Big] && (34)\\
&= \mathbb{E}\big[\|J_\pi(o)\,(\Delta - \mathbb{E}[\Delta \mid o])\|_2^2\big] + o(\|\Delta\|^2) && (35)\\
&= \operatorname{tr}\big(J_\pi(o)\,\Sigma_G(o)\,J_\pi(o)^\top\big) + o(\|\Delta\|^2), && (36)
\end{aligned}$$

where the last equality follows since $\mathbb{E}[(\Delta - \mathbb{E}[\Delta])(\Delta - \mathbb{E}[\Delta])^\top] = \Sigma_G(o)$ by definition. □

Under the conditions of Proposition 5,

$$\mathcal{L}_{\text{LANS}}(\pi) \doteq \mathbb{E}_{o\sim\mathcal{D}}\big[\mathcal{L}_{\text{LANS}}(o)\big] \approx \mathbb{E}_{o\sim\mathcal{D}}\big[\operatorname{tr}\big(J_\pi(o)\,\Sigma_G(o)\,J_\pi(o)^\top\big)\big], \tag{37}$$

so training with LANS is equivalent to adding a Jacobian regularizer weighted by the covariance $\Sigma_G(o)$ of the temporal noise process.

LANS For the noising process used by LANS, each coordinate $i \in \{1, \dots, n\}$ is replaced by its most recent past value $\tilde{o}_i$ with probability p_i, and kept unchanged with probability $1 - p_i$. The perturbation along dimension i is therefore

$$\Delta_i = \begin{cases} \tilde{o}_i - o_i, & \text{with probability } p_i,\\ 0, & \text{with probability } 1 - p_i. \end{cases} \tag{38}$$

Assuming independence across coordinates, the covariance matrix $\Sigma_G(o)$ is diagonal with entries

$$[\Sigma_G(o)]_{ii} = p_i(1 - p_i)\,\Delta_i^2. \tag{39}$$

Plugging this expression into Proposition 5 yields

$$\mathcal{L}_{\text{LANS}}(o) \approx \sum_{i=1}^{n} p_i(1 - p_i)\,\Delta_i^2\,\|\partial_{o_i}\pi(o)\|_2^2. \tag{40}$$

B HYPERPARAMETERS AND STATISTICS

B.1 HYPERPARAMETERS

The hyperparameters most relevant to the various methods are reported in the main paper. In this section, we report architecture and loss function parameters omitted from the main work.

Network We adopt the same hyperparameters relative to network size for every experiment. We use a depth of 1 and a hidden size of 128 both for the actor GRU and the Q-value one. Observations are embedded in a latent space of dimensionality 32 both before being provided to the RNN. They are embedded again with the same dimensionality, but different weights, and concatenated to the output of the RNN, which is then processed through an feed-forward network of depth 2 and hidden dimension 256. We adopt ReLU for nonlinearities.

Loss function Rewards are scaled by a factor of 5 during training. We use an interpolation factor τ of 5×10^{-3} and a discount factor γ of 0.99 for SAC. We rely on automatic entropy regularization with the same learning rate as the rest of the network, i.e. 3×10^{-4}.

Training We train each run for $1,536$ rollouts, each with a maximum of $1,024$ frames. We adopt an initial random exploration phase for 16 rollouts. We perform 128 gradient step per trajectory. The batch size of each gradient step is $2,048$ frames, split across sub-trajectories of maximum length 64, implying a sequence batch size of 32.

Evaluation We do not apply reward scaling during evaluation. We perform an evaluation cycle every $1,024$ gradient steps. We run each evaluation cycle by collecting the average total reward of an episodes across 16 instances.

B.2 Training times

We run all our experiments on NVIDIA Tesla P100, on which a single run takes between 44 and 56 hours of processing time.

C Usage of LLMs

Large language models (LLMs) were employed in the preparation of this work exclusively for auxiliary purposes. They were used to refine the exposition by improving grammar, clarity, and stylistic consistency, but not for the generation of substantive scientific content or entire paragraphs. Additionally, LLMs were leveraged to automate repetitive aspects of code development, particularly in the production of scripts for generating plots and figures. All theoretical contributions, derivations, experimental design, and main textual content were authored by the researchers.