Optimal Control of the Future via Prospective Learning with Control

Yuxin Bai[1] YBAI31@JH.EDU
Aranyak Acharyya[2*] AACHARY6@JH.EDU
Ashwin De Silva[1*] LDESILV2@JH.EDU
Zeyu Shen[3] ZSHEN39@JH.EDU
James Hassett[1] JHASSET1@JH.EDU
Joshua T. Vogelstein[1] JOVO@JH.EDU
[1] *Department of Biomedical Engineering, Johns Hopkins University, Baltimore, MD 21218*
[2] *Mathematical Institute for Data Science, Johns Hopkins University, Baltimore, MD 21218*
[3] *Department of Applied Mathematics and Statistics, Johns Hopkins University, Baltimore, MD 21218*

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Optimal control of the future is the next frontier for AI. Current approaches to this problem are typically rooted in reinforcement learning (RL). RL is mathematically distinct from supervised learning, which has been the main workhorse for the recent achievements in AI. Moreover, RL typically operates in a stationary environment with episodic resets, limiting its utility. Here, we extend supervised learning to address learning to *control* in non-stationary, reset-free environments. Using this framework, called "Prospective Learning with Control" (PLuC), we prove that under certain fairly general assumptions, empirical risk minimization (ERM) asymptotically achieves the Bayes optimal policy. We then consider a specific instance of prospective learning with control: foraging, a canonical task relevant to both natural and artificial agents. We illustrate that modern RL algorithms, which assume stationarity, struggle in these non-stationary reset-free environments. Even with time-aware modifications, they converge orders of magnitude slower than our prospective foraging agents on a simple 1-D foraging benchmark. Code is available at: https://github.com/neurodata/procontrol.

Keywords: statistical learning for dynamical and control systems, adaptive control, reset-free single-episode reinforcement learning

1. Introduction

When building artificial intelligence (AI) systems, we often start with some basic operating assumptions about the world. For example, a prevailing, though clearly limited assumption, is that the data-generating process is stationary. This is a core assumption in much of foundational estimation and learning theory dating back nearly a century Glivenko (1933); Cantelli (1933); Vapnik and Chervonenkis (1971); Valiant (1984), all the way through modern machine learning tools including large language models Vaswani et al. (2017). A second prevailing and obviously wildly simplifying assumption is that the learner's decisions have no impact on the data distribution. This assumption leads to developing systems based purely on predictions and forecasts, without concerning oneself

* Equal contribution

with the consequences of one's actions. For example, consider early machine learning successes such as spam detection Sahami et al. (1998), which operated according to both assumptions: no time and no impact. Once it is deployed, spammers change their behavior to subvert the system. This simple example illustrates a few key properties required of deployed machine learning systems.

First, time is real. In particular, the world is dynamic; therefore, for the system to preserve its desirable properties, it must also be dynamic on approximately the same time-scales. Second, decisions have impacts. Therefore, pure inference or forecasting systems are missing a key ingredient. The history of machine learning has traversed a trajectory of starting with the simplest possible assumptions, and slowly relaxing these assumptions to obtain more effective deployed systems. One limitation of relaxing the simplifying assumptions is that as we strive to estimate more complex functions, we require more data. Thus, as our ability to acquire and analyze more high-quality data has improved, so has our toolbox generalized accordingly.

Nearly one hundred years ago, people realized that one could estimate arbitrary properties of data under quite general assumptions, as long as we assumed no time and no impact Glivenko (1933); Cantelli (1933). In the 1950s, decision theory seriously relaxed the assumption of no impact Wald (1949). Markov models and Hidden Markov Models (HMM) relaxed the assumption of no time Baum and Petrie (1966), and Markov decision processes (MDP) and Partially Observable MDPs (POMDPs) further relaxed the assumption of no impact with the Bellman equation Bellman (1958), introducing control theory. Then Kalman filters addressed the fact that time and impact were real, but the parameters of the system were partially unknown Kalman (1960), founding adaptive control Landau (1984). Modern approaches to adaptive control are typically considered reinforcement learning (RL) Sutton and Barto (2018). Modern ML has many successes, though they are typically restricted to settings where the environment is stationary, there are many trials, and the whole world is perfectly observable (e.g., in video games Silver et al. (2016)). Non-stationary adaptive control and RL, and continual RL address the possibility of an environment that changes over time Khetarpal et al. (2022); Abel et al. (2023); Kumar et al. (2025). However, they typically operate under the assumption of many trials, rather than single-life Chen et al. (2022) and reset-free Lu et al. (2020) approaches. As is the norm in RL, these approaches continue to operate under a Markov Decision Process framework.

Our contribution builds on these earlier works, proposing a framework for non-stationary reset-free environments that does not require the standard MDP assumption. In a departure from much of RL, our work builds on estimation theory Glivenko (1933); Cantelli (1933), Probably Approximately Correct learning theory Valiant (1984, 2013), and modern machine learning and supervised learning practice Mohri et al. (2012); Goodfellow et al. (2016). Continual learning Vogelstein et al. (2025) and prospective learning De Silva et al. (2023); Silva et al. (2024); Bai et al. (2026) generalize estimation and learning theory by relaxing the assumption of no time. In this work, we introduce Prospective Learning with Control, which is denoted by **P**rospective **L**earning pl**u**s **C**ontrol (**PLuC**), where we further relax the assumption of no impact.

We formally introduce our framework in Section 2. Algorithms for solving PLuC problems are provided in Section 4. Numerical results are provided in Section 5, demonstrating that leading RL algorithms, even augmented to incorporate non-stationarity, either fail to converge to the Bayes optimal, or converge orders of magnitude more slowly than our algorithms. Theory proving that our algorithms converge to the Bayes optimal solution under suitable assumptions is in Section 3. This initial collection of empirical and theoretical results suggests that the prospective learning

with control is a promising direction worthy of further study, particularly as it is extended to more complex domains.

2. Framework for Prospective Learning with Control (PLuC)

This work builds on prospective learning defined previously De Silva et al. (2023); Silva et al. (2024); Bai et al. (2026), which we refer to hereafter as Prospective Learning without Control (PLiC). See Appendix A for a review of that work with slightly modified notation, and contrast with Prospective Learning with Control (PLuC), as defined below.

Data Let x_t be the position of the agent in an environment at time t. We also record time $t \in \mathcal{T} \subset \mathbb{N} \equiv [1, 2, 3, \dots]$. We refer to the position and time together as the observation state (or simply state), (x_t, t). Let $y_t \in \mathcal{Y}$ be the reward the agent would receive at time t at any of the positions. We define the observed datum at time t as $Z_t = (X_t, Y_t)$, with realization $z_t = (x_t, y_t)$.We denote the observed history up to time t by $z_{\leq t} = (z_1, \dots, z_t)$, and the unobserved future data by $z_{>t}$. We model the sequential interaction between the agent and environment as a stochastic decision process. In particular, the observed sequence $Z = (Z_t)_{t \in \mathbb{N}}$ is a stochastic process defined on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$.

Hypothesis The hypothesis space generalizes the notion from PLiC in Appendix A. Here, $h_t : \mathcal{X} \times \mathcal{T} \to \mathcal{X} \times \mathcal{Y}$ is a hypothesis in $\mathcal{H}_t$, where $\mathcal{H}_t \subseteq \{h : \mathcal{X} \times \mathcal{T} \to \mathcal{X} \times \mathcal{Y}\}$, is a hypothesis class characterizing the environment's dynamics and feedback. Therefore, h_t infers an output (x_{t+1}, y_{t+1}) for a given input x_t at time t.

Loss The instantaneous loss is the negative reward the agent receives at the position the agent is in at that time. Recalling that x_s encodes the position of the agent at time s, and that y_{s+1} encodes the reward at all locations at time $s + 1$, we can write

$$\bar{\ell}(h_s(x_s, s), y_{s+1}) = -y_{s+1}(x_{s+1}). \tag{1}$$

The loss at time t is the weighted cumulative instantaneous loss[1],

$$\ell_t(h, Z) = \sum_{s>t} w_{s-t} \bar{\ell}(h_s(x_s, s), y_{s+1}).$$

Learner The learner $\mathcal{L}$ is a map from the observed data, $\mathcal{D}_t$, to a hypothesis, $\hat{h}_t$. The observed data are $\mathcal{D}_t = (x_s, y_{s+1}(x_{s+1}), s)_{s \leq t-1}$.

Theoretical Goal We define risk as the expected loss:

$$R_t(h) = \mathbb{E}_F \left[\ell_t(h, Z) \mid z_{\leq t} \right] = \int \ell_t(h, Z) \, \mathrm{d}\mathbb{P}_{Z|z_{\leq t}}.$$

The goal is to choose a learner that obtains a hypothesis that minimizes risk.

$$\hat{h}^t = \operatorname*{argmin}_{h \in \mathcal{H}_t} R_t(h) \equiv \operatorname*{argmax}_{h \in \mathcal{H}_t} \int \sum_{s>t} w_{s-t} \cdot y^h_{s+1}(x^h_{s+1}) \, \mathrm{d}\mathbb{P}_{Z|z_{\leq t}}, \tag{2}$$

where y^h_{s+1} and x^h_{s+1} indicate that the subsequent positions and rewards are functions of h.

1. We intentionally do not specify whether the horizon is finite or infinite here.

Empirical Goal We cannot directly solve Eq. (2) because we cannot sample all possible futures, and we do not know the likelihood of each possible future. We therefore replace the integral in Eq. (2) with a sum up to the current time, t:

$$\hat{h}^t \approx \underset{h^t \in \mathcal{H}_t}{\operatorname{argmin}} \sum_{t'>0}^{t} \sum_{s>t'}^{t} -w_{s-t'} \cdot y_{s+1}(x_{s+1}). \tag{3}$$

3. Theory for Prospective Learning with Control

Here we provide proof sketches under asymptotic assumptions; formal proofs are in Appendix B. We note that finite-sample convergence rates remain an open problem.

Lemma 1 *Suppose $\{Z_t\}_{t=1}^{\infty}$ is a stochastic process and let $\{\mathcal{H}_t\}_{t=1}^{\infty}$ be an increasing hypothesis class, such that there exists $h_t \in \mathcal{H}_t$ which satisfies $\lim_{t\to\infty} \mathbb{E}[R_t(h_t) - R_t^*] = 0$. Additionally, suppose $\{u_t\}_{t=1}^{\infty}$ is a sequence such that $u_t \to \infty$ as $t \to \infty$, and $u_t \leq t$ for all t. Define the partial cumulative prospective loss to be $e_m(h) = \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1})$. Then, there exists $\tilde{h}_t \in \mathcal{H}_t$ such that $\lim_{t\to\infty} \mathbb{E}\left[\sup_{u_t \leq m \leq \infty} e_m(\tilde{h}_t) | Z_{\leq t}\right] - R_t^* = 0$.*

Sketch of proof A condition for the above lemma to hold is the existence of a sequence of hypotheses whose risks converge to the Bayes optimal risk. We establish that for this asymptotically optimal sequence of hypotheses, there exists a subsequence of timepoints on which the expected partial cumulative loss is arbitrarily close to the Bayes risk, using Markov's Inequality and Borel-Cantelli Lemma. We use this subsequence of timepoints to establish that for a particular sequence of hypotheses, the expected partial cumulative loss approaches the Bayes optimal risk.

Theorem 2 *Consider a family of stochastic processes $(\mathbf{X}_t, \mathbf{Y}_t)_{t>0}$. Suppose there exists a hypothesis class such that $\mathcal{H}_1 \subseteq \mathcal{H}_2 \ldots$ such that $\lim_{t\to\infty} [\inf_{h\in\mathcal{H}_t} R_t(h) - R_t^*] = 0$. Also, there exist sequences $\{u_t\}_{t=1}^{\infty}$ and $\{\xi_t\}_{t=1}^{\infty}$ satisfying $\lim_{t\to\infty} \xi_t = 0$, and $u_t \leq t$, such that*

$$\mathbb{E}\left[\max_{h\in\mathcal{H}_t} \left| \sum_{s=t+1}^{\infty} w_{s-t} \tilde{y}_{s+1}(x_{s+1}) - \max_{u_t \leq m \leq t} \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1}) \right| \right] \leq \xi_t.$$

Then, there exists a sequence $\{i_t\}_{t=1}^{\infty}$ such that

$\hat{h}^{(t)} = \operatorname{argmin}_{h\in\mathcal{H}_{i_t}} \max_{u_{i_t} \leq m \leq t} \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1})$ reaches the Bayes optimal risk.

Sketch of proof We note that for any sequence of hypotheses, the difference between the prospective loss and the partial cumulative loss is bounded by terms approaching zero. We find a subsequence of timepoints on which this difference is sufficiently small, and using Lemma 1 we establish that the Bayes risk of the minimizer of the partial cumulative loss on that particular subsequence of timepoints approaches zero.

4. Algorithms for Prospective Learning with Control

Algorithm 1 Prospective Learner with Control (PLuC)

Initialize: Instantaneous loss model $g_i(x_t, t\,;\theta)$, Cumulative loss model $g_i(x_t, t\,;\varphi)$, Planning horizon H, Discount factor γ, Replay buffer $\mathcal{D}$, Initial position x_0, sampling parameters $\{\eta_t\}_{t=0}^{\infty}$
▷ Warm-up phase using equation 4
for $t = 0, 1, 2, \ldots, T_{\text{warmup}} - 1$
    Randomly sample and move to the next allowed position x_t
    Observe the instantaneous losses $y_t(x_t)$ and store $(x_t, t, y_t(x_t))$ in $\mathcal{D}$
end for
▷ Online learning and planning phase
for $t = T_{\text{warmup}} \ldots, T_{\text{terminal}}$
    ▷ Update the models
    Calculate cumulative losses $\sum_{k=t+1}^{T} \gamma^{k-t-1} y_k$
    Update g_i to minimize MSE on observed instantaneous losses
    Update g_c to minimize MSE on calculated cumulative losses
    ▷ Selecting next position to move according to equation 6
    Identify all possible position sequences $\boldsymbol{x}_{t:t+H}$
    for each sequence $\boldsymbol{x}_{t:t+H}$
        Calculate the finite horizon loss $Q_{\text{finite}} = \sum_{h=1}^{H} \gamma^{h-1} g_i(x_{t+h}, t+h)$
        Calculate the terminal loss $Q_{\text{terminal}} = \gamma^{H} \cdot g_c(x_{t+H}, t+H)$
        Total sequence loss $Q(\boldsymbol{x}_{t:t+H}) = Q_{\text{finite}} + Q_{\text{terminal}}$
    end for
    ▷ Learner-environment interaction by equation 7
    Construct a sampling distribution over trajectories: $p_t(\boldsymbol{x}_{t:t+H}) \propto \exp\left(\eta_t \cdot Q(\boldsymbol{x}_{t:t+H})\right)$
    Sample the selected trajectory from this distribution: $\boldsymbol{x}^* \sim p_t$
    Move to the first position of the optimal sequence $x_t = x_t^*$
    Observe the instantaneous loss $y_{t+1}(x_t)$ and store $(x_t, t, y_{t+1}(x_t))$ in $\mathcal{D}$
end for

Inspired by online re-planning strategies in RL (Bertsekas, 2023), we adopt an online execution framework. Prior to iterative policy updates, we perform a warm-up phase during $t = 0, \ldots, T_{\text{warmup}}$ where the agent follows a nearly random policy to collect initial interaction data (states and rewards). This pretraining step alleviates the "cold-start" problem and facilitates more effective exploration in the early stages of the sequence. Following this phase, at each online iteration, we estimate both cumulative and instantaneous losses to select the next action based on the current learners.

4.1. Warm Start via Batch Pretraining

Cumulative loss estimation Recall the empirical objective for prospective learning with control from Eq. (3). To operationalize this, we construct an estimator for the prospective loss. We have the states, times, and the corresponding rewards from the warm-start interaction data. We construct a

dataset with inputs $\{(\varphi_x(x_s), \varphi_t(s))\}_{s \le T_{\text{warmup}}}$ and targets consisting of the cumulative rewards:

$$Y_s = \sum_{t'=s+1}^{T_{\text{warmup}}} -w_{t'-s} \cdot y_{t'}(x_{t'}) \tag{4}$$

Here, $\varphi_x(\cdot)$ is an state-encoding (such as one-hot encoding) and $\varphi_t(\cdot)$ is a time-encoding comprising of sinusoids and cosines of different frequencies. We then train a supervised regressor $\hat{g}_c$ —such as a deep neural network or a decision forest—that learns to map these state-time pairs to an estimated cumulative loss. This estimated cumulative loss serves as the prospective loss at state x_s and time s, providing the agent with a heuristic for the long-term consequences of its current position.

Instantaneous loss estimation When evaluating various hypotheses, many will propose trajectories x'_s that deviate from the historical positions x_s selected by the previous hypothesis $\hat{h}^s$. These counterfactual positions lack direct observations of y_s. Consequently, Eq. (3) cannot be solved directly due to missing data. To address this, we introduce an approximation $\tilde{y}_{s+1}$. Let x'_s denote a potential outcome (the position the agent could take) and x_s denote the actual observed position. We define the estimated reward $\tilde{y}_s$ as:

$$\tilde{y}_s(x', s) = \begin{cases} y_s(x_s) & \text{if } x'_s = x_s \\ \hat{y}_s(x'_s) & \text{if } x'_s \neq x_s \qquad \triangleright \text{ these are counterfactuals} \end{cases} \tag{5}$$

where $\hat{y}_s(x'_s)$ is the reward predicted by an estimator for a counterfactual state. Incorporating this into our objective yields:

$$\hat{h}^t \approx \arg\min_{h^t \in \mathcal{H}_t} \sum_{t'>0}^{t} \sum_{s>t'}^{t} -w_{s-t'} \cdot \tilde{y}_{s+1}(x_{s+1}).$$

By utilizing these estimators, we are no longer constrained by the current time t. Given a predefined horizon H, we can estimate the loss over future intervals:

$$\hat{h}^t \approx \arg\min_{h^t \in \mathcal{H}_t} \sum_{t'>0}^{t} \sum_{s>t'}^{t+H} -w_{s-t'} \cdot \tilde{y}_{s+1}(x_{s+1}).$$

To construct the estimator $\hat{y}_s(\cdot)$, we train a supervised regressor $\hat{g}^i$ on inputs $\{(\varphi_x(x_s), \varphi_t(s))\}_{s \le T_{\text{warmup}}}$ and targets $\{y_s(x_s)\}_{s \le T_{\text{warmup}}}$.

4.2. Online estimation and inference

Our framework combines estimated instantaneous losses and cumulative losses to approximate the infinite-horizon prospective loss (recall that our goal is Eq. (3)). This approach provides counterfactual insight while maintaining a tractable approximation of future costs. For a given horizon H, we estimate the hypothesis as:

$$\hat{h}^t \approx \arg\min_{h^t \in \mathcal{H}_t} \sum_{t'>0}^{t} \left\{ -w_{t+H-t'} \cdot \bar{y}_{t+H} + \sum_{s>t'}^{t+H} -w_{s-t'} \cdot \tilde{y}_{s+1}(x_{s+1}) \right\} \tag{6}$$

where $\bar{y}_{t+T}$ serves as the terminal cost estimation (Bertsekas, 2023) at the horizon boundary, and $\tilde{y}_s$ is the instantaneous cost. This structure mirrors "truncated rollout with terminal cost approximation", employing two models: one for short-term look-ahead and another to account for the truncated tail of the horizon.

Inference in Practice We evaluate potential future trajectories $\boldsymbol{x}_{s:s+H}$ by calculating their total prospective loss:

$$\hat{Q}(\boldsymbol{x}_{s:s+H}) = -\left(w_T \cdot \hat{g}^c(x_{s+T}, s+T) + \sum_{k=s}^{s+T} w_{k-s} \cdot \hat{g}^i(x_k, k) \right).$$

For example, with a horizon $T = 6$ and action space $A = 3$, there are $3^6 = 729$ possible trajectories. We select the next state stochastically using a Boltzmann rule with a decaying inverse-temperature parameter η_s:

$$p(\boldsymbol{x}_{s:s+H}) \propto \exp\left(\eta_s \hat{Q}(\boldsymbol{x}_{s:s+H})\right) \tag{7}$$

At each time t, the decision is governed by the learner trained on observations up to $t-1$ (see Algorithm 1). While we use an exhaustive search here, environments with larger state-action spaces can utilize more efficient search heuristics, such as Monte Carlo Tree Search (MCTS) (Brügmann, 1993; Kocsis and Szepesvári, 2006; Coulom, 2007; Silver et al., 2016, 2017).

5. Experiments for Prospective Learning with Control

To validate our framework, we devise a specific prospective learning problem called prospective foraging. Foraging is informally defined as the sequential search for resources (Barack, 2024). It is a central behavior of all agents, be they living organisms, AI bots, or robots. Standard formalizations of foraging assume that the environment is essentially stationary. For example, without the single agent's intervention, resources stay available indefinitely. This is a poor model of real-world foraging because, typically, resources are depleted, rather than persistent. We therefore introduce prospective foraging as a special instance of a prospective learning with control problem.

5.1. Prospective foraging

Informally, we assume an agent is navigating an environment, and resources (also called affordances (Gibson, 2014) or rewards (Sutton and Barto, 2018)) appear and disappear in that environment at various locations. At any given time, the agent may move its position, and the loss is the negative reward associated with the agent's current location.

Data Assume that $\mathcal{X}$ is a $H \times W$ grid, and let $H = 1$, so the environment is a chain (see Figure 1). Thus, $x_s \in \{1, \ldots, W\}$, and $y_s \in \mathbb{R}_+^W$. We encode position via a one-hot encoding, thereby x_s becomes a binary W-dimensional vector. We also embed time into a 50-dimensional vector using the classical 'positional' encoding strategy, as in Silva et al. (2024).

Hypothesis At each time step, the hypothesis can move the agent by one cell in the grid in either direction, assuming it is not at a boundary, or stay put. At a boundary, it can only move nowhere or in the opposite direction. Formally, $h(x_s, s) = x_{s+1} \in \mathcal{X}_s$, where $\mathcal{X}_s$ could be either $\{x_s - 1, x_s, x_s + 1\}$, $\{x_s - 1, x_s\}$, or $\{x_s, x_s + 1\}$.

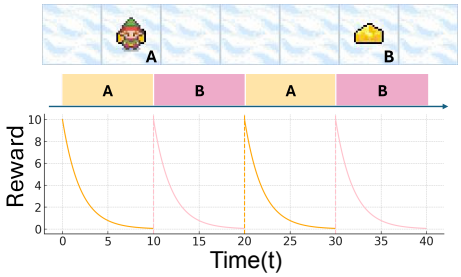

Figure 1: 1-D foraging environment. An agent moves along a 1×7 linear track with two reward patches (A, B). Rewards alternate between the two patches over time, and the currently active patch's reward decays exponentially.

Loss The loss is defined as $-\sum_{k=t+1}^{\infty} \gamma^{k-t-1} y_k$, with the temporal discounting factor of γ.

Assumptions Let $F_{\boldsymbol{Y}}$ denote the prior distribution of $\boldsymbol{Y}$. Assume that $\boldsymbol{Y}$ is independent of $\boldsymbol{X}$, i.e. $F_{\boldsymbol{Y}|\boldsymbol{X}} = F_{\boldsymbol{Y}}$. Let $y_s \in \mathbb{R}_+^W$ be a W-dimensional non-negative vector, where each element encodes the reward that the agent would receive at location x_s at time s. Assume that for two locations, A and B, there are rewards sometimes, but for the other $W-2$ locations, there are never any rewards. For locations A and B, assume that rewards arrive on a schedule. At every r time steps, location A receives a boost of one reward, which then decays exponentially down to zero at a rate of τ. Location B has the same temporal dynamics, but shifted by $r/2$ time steps. Thus, $y_s(A) = e^{-(s \mod r)/\tau}$, $y_s(B) = e^{-((s+r/2) \mod r)/\tau}$, where $s \mod r$ indicates the modulus function. The dynamics of the conditional distribution, $F_{\boldsymbol{X}|\boldsymbol{Y}}$, are a function of the learned hypotheses, $\hat{h}$.

We implement and evaluate the prospective learning controller (PLuC) within the 1-D foraging environment described in Section 5.1 and Bai et al. (2026): the agent forages along a 1×7 linear track containing only two reward patches, A and B, located three grid cells apart, as shown in Figure 1. Every $r = 10$ timestamps, rewards alternatively appear at A and B. The agent's goal is to maximize the total reward in its single lifetime, meaning there are no resets within each run.

We define the prospective regret of a hypothesis h_t as the difference between the prospective risk returned by hypothesis h_t and Bayes optimal risk: $\Delta h_t = R_t(h_t) - R_t^*$. In practice, one cannot evaluate regret over an infinite future. Therefore, we compute the prospective regret rate by evaluating it over a finite horizon of T time steps and taking the average: $(R_{t,T}(h_t) - R_{t,T}^\star)/T$, where $R_{t,T}$ is the finite horizon prospective risk. In addition, since we only have one trajectory after time t, we replace the risk by the prospective loss, so $\Delta h_t \approx \frac{1}{T}\sum_{i=s}^{s+T} w_{i-s}(y_s(x_s^*) - y_s(x_s))$, where $(x_s^*, x_{s+1}^*, ..., x_{s+T}^*)$ are the states returned by the Bayes-optimal hypothesis at the corresponding times.

5.2. Numerical Results

After an initial warm-up phase, we let the PLuC learn in an online manner as it interacts with the environment over time (Figure 2). The procedure follows the steps detailed in Algorithm 1. In the results presented here, however, we use a deterministic state-selection rule, where we choose the next state that yields the highest predicted prospective reward. Additional discussion of the Boltzmann sampling strategy described in the Algorithm 1 is provided in the Appendix D. During the online updating stage, we retrain regressors with all data collected so far for each update. We compute the normalized prospective regret of PLuC over time, and compare it to Fitted Q-Iteration (FQI) (Ernst et al., 2005; Munos and Szepesvari, 2008), Soft Actor-Critic (SAC) (Haarnoja et al., 2018a,b), and Proximal Policy Optimization Algorithm (PPO) (Schulman et al., 2017), three classical reinforcement learning algorithms, and the time-aware variants that we introduce here to adapt

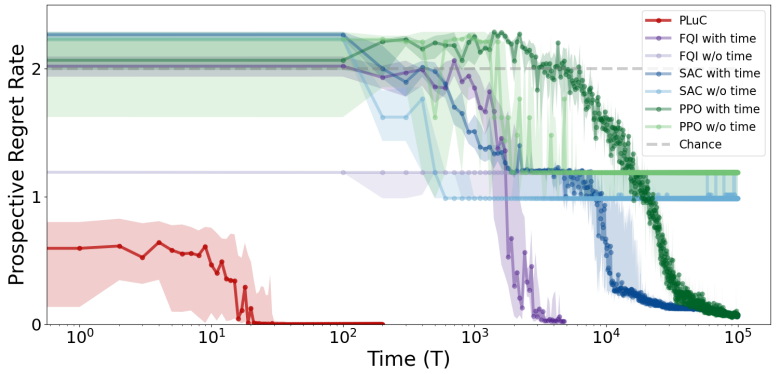

Figure 2: **PLuC converges to Bayes optimal regret more efficiently than RL baselines on the 1-D foraging task.** The prospective regret rate of PLuC (red), time-aware FQI (FQI with time, blue, our invention to improve FQI), time-agnostic FQI(FQI w/o time, light-blue), time-aware SAC (SAC with time, purple-red), time-agnostic SAC(SAC w/o time, lavender), time-aware PPO (PPO with time, dark-green), and time-agnostic PPO (PPO w/o time, light-green). While PLuC, time-aware FQI, time-aware SAC, and time-aware PPO converge to having zero regret, PLuC is orders of magnitude more efficient than any of them. And time-agnostic variants converge to a sub-optimal regret regardless of the time spent in interaction.

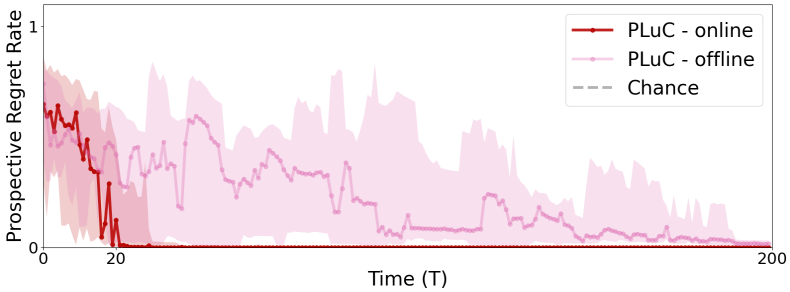

Figure 3: **PLuC online is several fold more efficient than offline.** Prospective regret rate for PLuC for online (red) and offline (pink). After warm-starting with 200 time steps, the online one converges in 20 time steps, whereas the offline one requires about 4× more data to converge.

them to non-stationary environments (see Appendix C and algorithms 2 to 4). In both PLuC and FQI variants, we use random forests as the regressors.

Online or Offline? Building on the online formulation, we compare the online and offline PLuC, under the same environment settings and parameters for training and inference (Figure 3). The only difference is the next state exploration: online-PLuC chooses the next step using its currently learned policy, while offline-PLuC explores the next step uniformly at random.

Instantaneous, Cumulative or Both? PLuC uses two learners during training and inference phases, g^i for instantaneous loss, and g^c for future cumulative loss. During inference, we combine a finite-horizon "lookahead" from g^i with a terminal cost approximation from g^c. We compare offline invariants of Instantaneous-only (PLuC-I), Future-only (PLuC-C), and combined (PLuC)

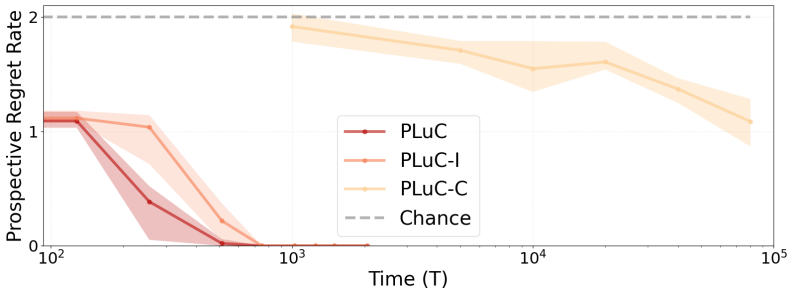

Figure 4: Prospective regret rate for PLuC (red), PLuC-I (orange), and PLuC-C (yellow). Removing either component reduces performance relative to PLuC.

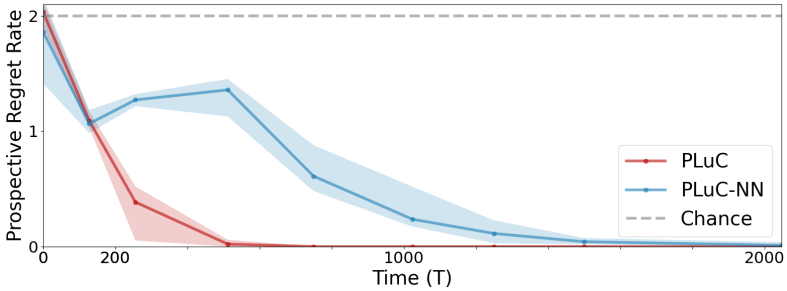

Figure 5: **PLuC with decision forests is 4x more efficient than with neural networks.** Prospective regret rate for PLuC with Gradient-Boosted Trees (red) and MLP Regressor (blue). While PLuC is 4x more efficient, PLuC-NN does converge as well.

(Figure 4). Using the terminal cost alone (PLuC-C) converges inefficiently, but we find adding it to the instantaneous loss accelerates learning versus instantaneous-only.

PLuC with Neural Network In Figure 5, we show that neural networks can also be used as the regressor in PLuC. While the PLuC with a neural network (specifically, a multi-layer perceptron with two hidden layers comprising 128 units) converges eventually, the convergence is about an order of magnitude slower than the PLuC employing a random forest as the regressor.

6. Discussion of Prospective Learning with Control

In this work, we introduce the framework of prospective learning with control and evaluate it on a foraging task, a simple but illustrative non-stationary control problem. Our results show that our algorithm, PLuC, can learn to plan ahead optimally. We also observe that standard RL baselines, when applied without modification, fail to converge to the optimal policy despite the simple nature of the one-dimensional foraging environment, which is an expected outcome given their stationary assumptions. Time-aware modifications improve convergence but at a significant sample-efficiency cost relative to PLuC. Much work remains to be done, as our experiments are currently restricted to a simple environment with deterministic dynamics and periodic reward. We believe prospective learning with control is an initial step towards understanding and utilizing optimal control of the future in both natural and artificial intelligence. We leave generalizing to continuous action, state, and time spaces, and deploying our algorithms with transition dynamics in complex real-world environments, as future work.

References and Notes

David Abel, André Barreto, Benjamin Van Roy, Doina Precup, H V Hasselt, and Satinder Singh. A definition of continual reinforcement learning. *Neural Information Processing Systems*, abs/2307.11046, 2023. doi: 10.48550/arXiv.2307.11046. URL https://openreview.net/pdf?id=ZZS9WEWYbD.

Yuxin Bai, Cecelia Shuai, Ashwin De Silva, Siyu Yu, Pratik Chaudhari, and Joshua T Vogelstein. *Prospective learning in retrospect*, pages 17–29. Lecture notes in computer science. Springer Nature Switzerland, 2026.

David L Barack. What is foraging? *Biology & Philosophy*, 39:3, 2024.

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37:1554–1563, 1966. ISSN 0003-4851,2168-8990. doi: 10.1214/aoms/1177699147. URL http://dx.doi.org/10.1214/aoms/1177699147.

Richard Bellman. Dynamic programming and stochastic control processes. *Information and control*, 1:228–239, 1958. ISSN 0019-9958,1878-2981. doi: 10.1016/s0019-9958(58)80003-0. URL http://dx.doi.org/10.1016/S0019-9958(58)80003-0.

Dimitri Bertsekas. *A course in Reinforcement Learning*. Athena Scientific, 2023.

Bernd Brügmann. Monte carlo go, 1993.

Francesco Paolo Cantelli. Sulla determinazione empirica delle leggi di probabilita. *Giorn. Ist. Ital. Attuari*, 4, 1933.

Nicolò Cesa-Bianchi, Claudio Gentile, Gábor Lugosi, and Gergely Neu. Boltzmann exploration done right. *Advances in neural information processing systems*, 30, 2017.

Annie S Chen, Archit Sharma, S Levine, and Chelsea Finn. You only live once: Single-life reinforcement learning. *Advances in Neural Information Processing Systems*, abs/2210.08863, 2022. ISSN 1049-5258. doi: 10.48550/arXiv.2210.08863. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/5ec4e93f2cec19d47ef852a0e1fb2c48-Paper-Conference.pdf.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In H. Jaap van den Herik, Paolo Ciancarini, and H. H. L. M. Donkers, editors, *Computers and Games, CG 2006, Turin, Italy, May 29–31, 2006, Revised Papers*, Lecture Notes in Computer Science, pages 72–83. Springer, 2007.

Ashwin De Silva, Rahul Ramesh, Lyle Ungar, Marshall Hussain Shuler, Noah J Cowan, Michael Platt, Chen Li, Leyla Isik, Seung-Eon Roh, Adam Charles, Archana Venkataraman, Brian Caffo, Javier J How, Justus M Kebschull, John W Krakauer, Maxim Bichuch, Kaleab Alemayehu Kinfu, Eva Yezerets, Dinesh Jayaraman, Jong M Shin, Soledad Villar, Ian Phillips, Carey E Priebe, Thomas Hartung, Michael I Miller, Jayanta Dey, Ningyuan Huang, Eric Eaton, Ralph Etienne-Cummings, Elizabeth L Ogburn, Randal Burns, Onyema Osuagwu, Brett Mensh, Alysson R

Muotri, Julia Brown, Chris White, Weiwei Yang, Andrei A Rusu Timothy Verstynen, Konrad P Kording, Pratik Chaudhari, and Joshua T Vogelstein. Prospective Learning: Principled Extrapolation to the Future. In Sarath Chandar, Razvan Pascanu, Hanie Sedghi, and Doina Precup, editors, *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232 of *Proceedings of Machine Learning Research*, pages 347–357. PMLR, 2023.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005.

James J Gibson. *The Ecological Approach to Visual Perception: Classic Edition (Psychology Press & Routledge Classic Editions)*. Psychology Press, 1 edition, 2014.

V Glivenko. Sulla determinazione empirica delle leggi di probabilita. *Gion. Ist. Ital. Attauri.*, 4: 92–99, 1933. URL https://ci.nii.ac.jp/naid/10026792179/.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep Learning*, volume 1 of *Adaptive Computation and Machine Learning series*. MIT Press, 2016. ISBN 9780262337434. URL https://www.amazon.com/dp/B01MRVFGX4/ref=dp-kindle-redirect?_encoding=UTF8&btkr=1.

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

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

R E Kalman. A new approach to linear filtering and prediction problems. *International Journal of Engineering, Transactions A: Basics*, 82:35–45, 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. URL http://fluidsengineering.asmedigitalcollection.asme.org/article.aspx?articleid=1430402.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards Continual Reinforcement Learning: A Review and Perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022. ISSN 1076-9757,1076-9757. doi: 10.1613/jair.1.13673. URL https://www.jair.org/index.php/jair/article/view/13673.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18–22, 2006, Proceedings*, volume 4212 of *Lecture Notes in Computer Science*, pages 282–293. Springer, 2006.

Saurabh Kumar, Henrik Marklund, Ashish Rao, Yifan Zhu, Hong Jun Jeon, Yueyang Liu, and Benjamin Van Roy. Continual learning as computationally constrained reinforcement learning. *Foundations and Trends® in Machine Learning*, 18:913–1053, 2025. ISSN 1935-8237,1935-8245. doi: 10.1561/2200000116. URL http://dx.doi.org/10.1561/2200000116.

Yoan D Landau. Adaptive control: The model reference approach. *IEEE transactions on systems, man, and cybernetics*, SMC-14:169–170, 1984. ISSN 0018-9472,2168-2909. doi: 10.1109/tsmc.1984.6313284. URL http://dx.doi.org/10.1109/TSMC.1984.6313284.

Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. Reset-free lifelong learning with skill-space planning. *arXiv [cs.LG]*, 2020. URL https://openreview.net/pdf?id=HIGSa_3kOx3.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, kindle edition, 2012.

Remi Munos and Csaba Szepesvari. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

J Neyman. On the application of probability theory to agricultural experiments: Essay on principles, section 9.(translated in 1990). *Statistical Science*, 5:465–480, 1923.

D Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66:688–701, 1974.

M Sahami, S Dumais, D Heckerman, and E Horvitz. A Bayesian approach to filtering junk E-mail. *AAAI Conference on Artificial Intelligence*, 1998. URL https://cdn.aaai.org/Workshops/1998/WS-98-05/WS98-05-009.pdf.

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

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Ashwin De Silva, Rahul Ramesh, Rubing Yang, Siyu Yu, Joshua T Vogelstein, and Pratik Chaudhari. Prospective learning: Learning for a dynamic future. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

David Silver, Aja Huang, Chris J. Maddison, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017. URL https://arxiv.org/abs/1712.01815.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

L G Valiant. A Theory of the Learnable. *Communications of the ACM*, 27:1134–1142, 1984. ISSN 0001-0782. doi: 10.1145/1968.1972. URL http://doi.acm.org/10.1145/1968.1972.

Leslie Valiant. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books, 2013. ISBN 9780465032716.

V Vapnik and A Chervonenkis. On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971. ISSN 0040-585X. doi: 10.1137/1116025. URL https://doi.org/10.1137/1116025. doi: 10.1137/1116025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Joshua T Vogelstein, Jayanta Dey, Hayden S Helm, Will LeVine, Ronak D Mehta, Tyler M Tomita, Haoyin Xu, Ali Geisa, Qingyang Wang, Gido M van de Ven, Chenyu Gao, Weiwei Yang, Bryan Tower, Jonathan Larson, Christopher M White, and Carey E Priebe. Simple lifelong learning machines. *IEEE transactions on pattern analysis and machine intelligence*, PP:1–15, 2025. ISSN 0162-8828,1939-3539. doi: 10.1109/TPAMI.2025.3595364. URL http://dx.doi.org/10.1109/TPAMI.2025.3595364.

Abraham Wald. Statistical Decision Functions. *Annals of Mathematical Statistics*, 20:165–205, 1949. ISSN 0003-4851,2168-8990. doi: 10.1214/aoms/1177730030. URL https://projecteuclid.org/euclid.aoms/1177730030.

Appendix A. Prospective Learning without control (PLiC)

Here we briefly review the prior work on this topic, which is called "prospective learning" (De Silva et al., 2023; Silva et al., 2024; Bai et al., 2026) (PL), modifying notation slightly for convenience. In retrospect, it would more specifically be called prospective learning without control, or **P**rospective **L**earning m**i**nus **C**ontrol (**PLiC**).

Data Let the input and output be denoted by $x_t \in \mathcal{X}$ and $y_t \in \mathcal{Y}$, respectively. Let $z_t = (x_t, y_t)$. We will denote the observed data, $z_{\leq t} = \{z_1, \dots, z_t\}$, and the unobserved data, $z_{>t}$. In contrast to PAC learning, t is not just a dummy variable, but rather, indexes time. Let $t \in \mathcal{T} \subseteq \mathbb{N}$, where $\mathbb{N} = \{1, 2, \dots\}$ are the counting numbers, and let $|\mathcal{T}| = T \leq \infty$ be the total number of time steps. We therefore define the data triple (x_t, y_t, t).

Hypothesis The hypothesis space generalizes the notion from PAC learning. Here, $h_t : \mathcal{X} \times \mathcal{T} \to \mathcal{Y}$ is a hypothesis in $\mathcal{H}_t \subseteq \{h : \mathcal{X} \times \mathcal{T} \to \mathcal{Y}\}$, so h_t infers an output y_t for a given input x_t at time t.Denote the hypothesis sequence, $h \equiv (h_1, h_2, \dots)$, where each element of this sequence $h_t : \mathcal{X} \mapsto \mathcal{Y}$. At times, we will consider a sequence of nested hypothesis spaces, $\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots \subseteq \mathcal{H}$.

Loss The instantaneous loss, $\bar{\ell} : \mathcal{H} \times \mathbb{Z} \to \mathbb{R}$, is a map from the hypothesis and current data to the reals. In an abuse of notation, we write $\bar{\ell}(h_t, z_t) \equiv \bar{\ell}(h(x_t, t), y_t)$. In all real-world problems, we care about the integrated future loss. Let w_t be a non-increasing non-negative weighting function that sums to one, that is $\sum_t w_t = 1$ and $0 \leq w_t \leq 1 \, \forall t$. We thus define loss at time t as $\ell_t(h, z_{>t}) = \sum_{s>t} w_{s-t} \bar{\ell}(h(x_s, s), y_s)$, which is the weighted cumulative loss over all the future data.[2,3] To fully specify loss, therefore, requires specifying a temporal discounting function, $\boldsymbol{w} \equiv (w_1, w_2, \dots)$. For example, we could let $\boldsymbol{w}$ be an exponentially decaying function, $w_{s-t} = \gamma^{s-t}$, where $\gamma \leq 1$.[4]

Learner Let a learner, $\mathcal{L}$ maps from the data to a hypothesis, $\mathcal{L}(\mathcal{D}_t) = \hat{h}^t$, where $\mathcal{D}_t = (x_s, y_s, s)_{s \leq t}$, and $\hat{h}^t$ is the estimated hypothesis sequence obtained by the learner at time t.[5].

Assumptions Let $\boldsymbol{X} = X_{0<t}$ and $\boldsymbol{Y} = Y_{0<t}$ be stochastic processes, taking values $\boldsymbol{X} = x_{0<t}$ and $\boldsymbol{Y} = y_{0<t}$. Assume $(\boldsymbol{X}, \boldsymbol{Y}) \sim F \in \mathcal{F}$. In general, we place no assumptions on $\mathcal{F}$.

Theoretical Goal Define risk as expected loss, $R_t(h) = \mathbb{E}_F[\ell_t(h, Z) \mid z_{\leq t}] = \int \ell_t(h, Z) \, \mathrm{d}\mathbb{P}_{Z|z_{\leq t}}$.[6] Let $R_t^* = \min_{h \in \mathcal{H}_t} R_t(h)$ be the (Bayes) optimal risk.[7]

We seek a learner, $\mathcal{L}$ with the following property: for any $F \in \mathcal{F}$, and $\epsilon, \delta > 0$, there exists a $t'(\epsilon, \delta)$ such that the learner outputs a hypothesis $\hat{h}^t$ satisfying $\mathbb{P}[R_t(\hat{h}^t) - R_t^* < \epsilon] \geq 1 - \delta$, for any $t > t'(\epsilon, \delta)$.

One implication of the above desiderata is that the estimated hypothesis, $\hat{h}^t$, is a consistent estimator of the risk (Shalev-Shwartz and Ben-David, 2014). In other words, $R(\hat{h}^t) \to R_t^*$ as

2. If T is infinite, we replace the sums with limiting sums, e.g., $\lim_{T \to \infty} \sum_{t=1}^{T} w_t$, to ensure that the sum converges.
3. Note that if we let w_t be a constant function of t, we recover something equivalent to the loss in PAC learning.
4. This is a slight variation of how prospective loss was previously defined in Silva et al. (2024).
5. Formally, we assume h is a random variable and $h \in \sigma(Z_{\leq t})$ where $\sigma(\cdot)$ denotes the filtration (an increasing sequence of sigma-algebras) of the stochastic process Z
6. This is a slight abuse of notation, because previously ℓ mapped from a hypothesis and data corpus, but now we are saying that it maps from a hypothesis and a collection of random variables. We have used the shorthand $\mathbb{E}[Y \mid x]$ for $\mathbb{E}[Y \mid X = x]$. Note that $\mathbb{E}[Z|z_{\leq t}]$ is equivalent to $\mathbb{E}[Z_{>t}|z_{\leq t}]$ because the past z's are given.
7. In a slight abuse of notation, we use $\min$ in place of $\inf$ even if the $\min$ might not exist. Here we use the shorthand $\mathcal{H}_t \equiv \sigma(Z_{\leq t})$.

$t \to \infty$. The above goal, however, is not an explicit objective function. In practice, therefore, we specify an objective function to minimize. The hope is that if we minimize this objective, then eventually, the resulting estimand satisfies the above desiderata. We seek to minimize the weighted sum of instantaneous losses on *future* data:

$$\hat{h}^t = \operatorname*{argmin}_{h \in \mathcal{H}_t} R_t(h) = \operatorname*{argmin}_{h \in \mathcal{H}_t} \int \ell_t(h, Z) \, \mathrm{d}\mathbb{P}_{Z|z_{\leq t}} = \operatorname*{argmin}_{h \in \mathcal{H}_t} \int \sum_{s>t} w_{s-t} \bar{\ell}(h(x_s, s), y_s) \, \mathrm{d}\mathbb{P}_{Z|z_{\leq t}}. \tag{8}$$

If z is a discrete set, the above integral becomes a sum. Let p_z denote the probability of a particular sequence $z \in \mathbb{Z}$. With this notation, Eq. (8) becomes

$$\hat{h}^t = \operatorname*{argmin}_{h \in \mathcal{H}_t} \sum_{z \in \mathbb{Z}: z_{\leq t}} p_z \ell_t(h, Z) = \operatorname*{argmin}_{h \in \mathcal{H}_t} \sum_{z \in \mathbb{Z}: z_{\leq t}} p_z \sum_{s>t} w_{s-t} \bar{\ell}(h(x_s, s), y_s). \tag{9}$$

Empirical Goal In practice, in general, we cannot minimize Eq. (8) because it requires solving an intractable integral, and we typically cannot solve Eq. (9), because it requires sampling too many possible futures (there are exponentially many even if z is binary). Moreover, we cannot minimize Eqs. (8) and (9) because they both require sampling from an unknown distribution, $\mathbb{P}_{Z|z_{\leq t}}$. To address the first issue, we use a Monte Carlo approximate sum, sampling only a subset of possible futures. To address the second issue, rather than sampling from the unobserved future, we sample only from the observed past. With these two approximations in hand, we have

$$\hat{h}^t \approx \operatorname*{argmin}_{h^t \in \mathcal{H}_t} \sum_{t'>0}^{t} \ell_{t'}(h, z_{>t'}) \approx \operatorname*{argmin}_{h^t \in \mathcal{H}_t} \sum_{t'>0}^{t} \sum_{s>t'}^{t} w_{s-t'} \bar{\ell}(h^{t'}(x_s, s), y_s). \tag{10}$$

Main result The main result of Silva et al. (2024) is approximating a variant of Eq. (8), under suitable assumptions, satisfies our theoretical goal. The key assumptions are (a) consistency and (b) uniform concentration. The focus in that work was on settings in which our decisions $\hat{h}^t(x_s, s)$ had no impact on future outcomes or distributions. In such cases, we can ignore the sum in Eq. (8), because all that matters for the current decision is the instantaneous loss. Here, we are interested in general settings where the decisions also impact future distributions and losses.

Relationship to prospective learning without control The observed *data* triples in PLuC are $(x_s, y_{s+1}(x_{s+1}), s)$, rather than (x_s, y_s, s) in PLiC. Note that these data triples differ in two ways. First, the index on y changes from s to $s+1$ as we go from PLiC to PLuC; this is required because the rewards are necessarily a function of the action of the agent. Second, rather than the entire vector of output being observed as in PLiC, in PLuC, we only obtain a partial observation: the reward at the location we happen to currently inhabit. This renders PLuC closely related to causal inference, in which the outcomes are defined as a function of the intervention (or treatment), and therefore only partially observed, leading to the potential outcomes framework Neyman (1923); Rubin (1974). *Hypotheses* are no longer maps from $\mathcal{X} \times \mathcal{T}$ to $\mathcal{Y}$, rather, they map to $\mathcal{X}$ and $\mathbb{R}_+$. *Loss* in PL with or without control is the cumulative instantaneous loss. However, in PLiC, instantaneous loss is typically an error signal, e.g., the difference between the predicted output and the actual output. In PLuC, loss is just a negative reward. Moreover, loss is not observed for any of the decisions that the agent did not make, necessitating the use of some surrogate loss function in practice. *Learners* map from data to hypotheses, though the data are subtly different due to the

time indices and partial observation. *Assumptions* can be the same, though note that the future data are now necessarily a function of the learner and hypothesis in PLuC, whereas they were not necessarily in PLiC. The *theoretical goal* is the same, modulo replacing the loss from PLiC with that from PLuC. The *empirical goal* changes somewhat similarly, insofar as in PLuC we cannot directly solve Eq. (2), so we use a surrogate optimization problem, Eq. (3).

Appendix B. Theoretical Results

Lemma 3 *Suppose $\{Z_t\}_{t=1}^{\infty}$ is a stochastic process and let $\{\mathcal{H}_t\}_{t=1}^{\infty}$ be an increasing hypothesis class, such that there exists $h_t \in \mathcal{H}_t$ which satisfies*

$$\lim_{t\to\infty} \mathbb{E}[R_t(h_t) - R_t^*] = 0. \tag{11}$$

Additionally, suppose $\{u_t\}_{t=1}^{\infty}$ is a sequence such that $u_t \to \infty$ as $t \to \infty$, and $u_t \leq t$ for all t. Define

$$e_m(h) = \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1}).$$

Then, there exists $\tilde{h}_t \in \mathcal{H}_t$ such that

$$\lim_{t\to\infty} \mathbb{E}\left[\sup_{u_t \leq m \leq \infty} e_m(\tilde{h}_t) | Z_{\leq t}\right] - R_t^* = 0.$$

Proof. There exists $h_t \in \mathcal{H}_t$ such that

$$\lim_{t\to\infty} \mathbb{E}[R_t(h_t) - R_t^*] = 0.$$

Note that $R_t(h) \geq R_t^*$ almost surely. Choose a subsequence $\{j_k\}_{k=1}^{\infty}$ such that

$$\mathbb{E}[R_{j_k}(h_{j_k}) - R_t^*] \leq \frac{1}{4^k}.$$

Now,

$$\begin{aligned}
\mathbb{E}[R_t(h_t) - R_t^*] &= \mathbb{E}\left[\mathbb{E}[\lim\sup_{m\to\infty} e_m(h_t)|Z_{\leq t}] - R_t^*\right] \\
&= \mathbb{E}\left[\mathbb{E}[\lim_{i\to\infty} \sup_{u_i \leq m \leq \infty} e_m(h_t)|Z_{\leq t}] - R_t^*\right] \\
&= \lim_{i\to\infty} \mathbb{E}\left[\mathbb{E}[\sup_{u_i \leq m \leq \infty} e_m(h_t)|Z_{\leq t}] - R_t^*\right]
\end{aligned}$$

Thus, for every k, there exists integer i_k such that

$$\mathbb{E}\left[\mathbb{E}[\sup_{u_{i_k} \leq m \leq \infty} e_m(h_{j_k})|Z_{\leq j_k}] - R_{j_k}^*\right] \leq \mathbb{E}[R_{j_k}(h_{j_k}) - R_{j_k}^*] + \frac{1}{4^k} \leq \frac{2}{4^k}.$$

Note that,

$$\mathbb{E}\left[\sup_{u_i \leq m \leq \infty} e_m(h_t)|Z_{\leq t}\right] \geq \mathbb{E}\left[\sum_{s=t+1}^{m} w_{s-t}\tilde{y}_{s+1}(x_{s+1})|Z_{\leq t}\right] \geq R_t^*.$$

Using Markov's Inequality,

$$\begin{aligned}
&\sum_{k=0}^{\infty} \mathbb{P}\left(\mathbb{E}\left[\sup_{u_{i_k} \leq m \leq \infty} e_m(h_{j_k}) | Z_{\leq j_k} \right] - R^*_{j_k} > 2^{\frac{1}{2}-k} \right) \\
&\leq \sum_{k=0}^{\infty} \frac{1}{2^{\frac{1}{2}-k}} \mathbb{E}\left[\mathbb{E}\left[\sup_{u_{i_k} \leq m \leq \infty} e_m(h_{j_k}) | Z_{\leq j_k} \right] - R^*_{j_k} \right] \\
&\leq \sum_{k=0}^{\infty} \frac{1}{2^{\frac{1}{2}-k}} \frac{2}{4^k} \\
&< \infty.
\end{aligned}$$

Thus, by Borel-Cantelli Lemma, there exists an integer k_0 such that for all $k \geq k_0$,

$$\mathbb{E}\left[\sup_{u_{i_k} \leq m \leq \infty} e_m(h_{j_k}) | Z_{\leq j_k} \right] - R^*_{j_k} \leq 2^{\frac{1}{2}-k}.$$

Now, we define $k_t = \max\{k \in \mathbb{N} \cup \{0\} : \max\{i_k, j_k\} \leq t\}$. Note that $k_t \to \infty$ as $t \to \infty$, because $i_k, j_k \leq \infty$. Define $\alpha_t = 2^{\frac{1}{2}-k_t}$, and observe that $\lim_{t\to\infty} \alpha_t = 0$. Let $t_0 \in \mathbb{N}$ be a random variable such that for all $t \geq t_0$, $k_t \geq k_0$, and thus,

$$\mathbb{E}\left[\sup_{u_{i_{k_t}} \leq m \leq \infty} e_m(h_{j_{k_t}}) \middle| Z_{\leq j_{k_t}} \right] - R^*_{j_{k_t}} \leq \alpha_t.$$

Choose $\tilde{h}_t = h_{j_{k_t}}$ for all t. Note that $j_{k_t} \leq t \Rightarrow \mathcal{H}_{j_{k_t}} \subset \mathcal{H}_t$. Hence, for $t \geq t_0$,

$$\mathbb{E}\left[\sup_{u_t \leq m \leq \infty} e_m(\tilde{h}_t) | Z_{\leq j_{k_t}} \right] - R^*_{j_{k_t}} \leq \mathbb{E}\left[\sup_{u_{i_{k_t}} \leq m \leq \infty} e_m(\tilde{h}_t) | Z_{\leq j_{k_t}} \right] - R^*_{j_{k_t}} \leq \alpha_t.$$

Since $\alpha_t \to 0$ as $t \to \infty$, we have,

$$\lim_{t\to\infty} \left(\mathbb{E}\left[\sup_{u_t \leq m \leq \infty} e_m(\tilde{h}_t) | Z_{\leq j_{k_t}} \right] - R^*_{j_{k_t}} \right) = 0.$$

Theorem 4 *Consider a family of stochastic processes $(\mathbf{X}_t, \mathbf{Y}_t)_{t>0}$. Suppose there exists a hypothesis class such that $\mathcal{H}_1 \subseteq \mathcal{H}_2 \ldots$ such that*

$$\lim_{t\to\infty} \left[\inf_{h \in \mathcal{H}_t} R_t(h) - R^*_t \right] = 0.$$

Also, there exist sequences $\{u_t\}_{t=1}^{\infty}$ and $\{\xi_t\}_{t=1}^{\infty}$ satisfying $\lim_{t\to\infty} \xi_t = 0$, and $u_t \leq t$, such that

$$\mathbb{E}\left[\max_{h \in \mathcal{H}_t} \left| \sum_{s=t+1}^{\infty} w_{s-t} \tilde{y}_{s+1}(x_{s+1}) - \max_{u_t \leq m \leq t} \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1}) \right| \right] \leq \xi_t.$$

Then, there exists a sequence $\{i_t\}_{t=1}^{\infty}$ such that

$$\hat{h}^{(t)} = \operatorname*{argmin}_{h \in \mathcal{H}_{i_t}} \max_{u_{i_t} \leq m \leq t} \sum_{s=1}^{m} w_s \tilde{y}_{s+1}(x_{s+1})$$

reaches the Bayes' optimal risk.

Proof. We consider a subsequence $\{i_t\}_{t=1}^{\infty}$ such that $\xi_{i_t} \to 0$ exponentially so that $\sum_{t=1}^{\infty} \xi_{i_t} < \infty$. From Markov's Inequality,

$$\begin{aligned}
&\sum_{t=0}^{\infty} \mathbb{P}\left[\max_{h\in\mathcal{H}_{i_t}} \left|\bar{l}_t(h,Z) - \max_{u_{i_t}\le m\le i_t} e_m(h)\right| > \sqrt{\xi_{i_t}}\right] \\
&\le \sum_{t=0}^{\infty} \frac{1}{\sqrt{\xi_{i_t}}} \mathbb{E}\left[\max_{h\in\mathcal{H}_{i_t}} \left|\bar{l}_t(h,Z) - \max_{u_{i_t}\le m\le i_t} e_m(h)\right|\right] \\
&\le \sum_{t=0}^{\infty} \sqrt{\xi_{i_t}} \\
&< \infty.
\end{aligned}$$

By the Borel-Cantelli Lemma, there exists $t_1 \in \mathbb{N}$, such that for all $t \ge t_1$,

$$\max_{h\in\mathcal{H}_{i_t}} \left|\bar{l}_t(h,Z) - \max_{u_{i_t}\le m\le i_t} e_m(h)\right| \le \sqrt{\xi_{i_t}}.$$

Let $j_t = \max\{t' : i_{t'} \le t\}$. Observe that $i_{j_t} \to \infty$ as $t \to \infty$. Since $i_{j_t} \le t$,

$$\bar{l}_t(h,Z) - \max_{u_{i_{j_t}}\le m\le t} e_m(h) \le \bar{l}_t(h,Z) - \max_{u_{i_{j_t}}\le m\le i_{j_t}} e_m(h).$$

Construct a random variable t_2 such that for all $t \ge t_2$,$j_t \ge t_1$. Thus, for all $t \ge t_2$,

$$\begin{aligned}
&\max_{h\in\mathcal{H}_{i_{j_t}}} \left\{\bar{l}_t(h,Z) - \max_{u_{i_{j_t}}\le m\le t} e_m(h)\right\} \\
&\le \max_{h\in\mathcal{H}_{i_{j_t}}} \left|\bar{l}_t(h,Z) - \max_{u_{i_{j_t}}\le m\le i_{j_t}} e_m(h)\right| \\
&\le \sqrt{\xi_{i_{j_t}}}.
\end{aligned}$$

Choose $i_t = i_{j_t}$ and notice that since $i_t \to \infty$, we have $i_t \le t$. With probability one, for all $t \ge t_2$,

$$\max_{h\in\mathcal{H}_{i_t}} \left\{\bar{l}_t(h,Z) - \max_{u_{i_t}\le m\le t} e_m(h)\right\} \le \sqrt{\xi_{i_t}}.$$

For $t \ge t_2$,

$$\begin{aligned}
\bar{l}_t(\hat{h}^{(t)},Z) &\le \max_{u_{i_t}\le m\le t} e_m(\hat{h}^{(t)}) + \sqrt{\xi_{i_t}} \\
&\le \max_{u_{i_t}\le m\le t} e_m(h_t) + \sqrt{\xi_{i_t}} \\
&\le \sup_{u_{i_t}\le m\le t} e_m(h_t) + \sqrt{\xi_{i_t}}
\end{aligned}$$

Hence,

$$\mathbb{E}[\bar{l}_t(\hat{h}^{(t)},Z)|Z_{\le t}] - \mathbb{E}\left[\sup_{u_{i_t}\le m\le t} e_m(h_t)|Z_{\le t}\right] \to 0$$

almost surely. By Lemma 3, we have,

$$\mathbb{E}\left[\bar{l}_t(\hat{h}^{(t)}, Z)|Z_{\leq t}\right] - R_t^* \to 0.$$

Hence, by the bounded convergence theorem,

$$0 = \mathbb{E}\left[\lim_{t\to\infty}\left(\mathbb{E}\left[\bar{l}_t(\hat{h}^{(t)}, Z)|Z_{\leq t}\right] - R_t^*\right)\right] = \lim_{t\to\infty}\mathbb{E}\left[\mathbb{E}\left[\bar{l}(\hat{h}^{(t)}, Z)|Z_{\leq t}\right] - R_t^*\right].$$

Appendix C. Reinforcement Learning Baselines

Let $s_t \in \mathcal{S}, a_t \in \mathcal{A}$ and $r_t \in \mathbb{R}$ denote the state, action, and reward at time t, respectively.

C.1. Fitted Q-Iteration

Suppose we are given a dataset of interactions $\mathcal{D} = \{(s_t, a_t, r_t)\}_{t=1}^T$ collected using some stochastic behavior policy by interacting with a Markov Decision Process (MDP). The generic Fitted Q-iteration (FQI) (Ernst et al., 2005; Munos and Szepesvari, 2008) algorithm approximates the state-action value function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ by running the following iterative condition. At iteration k, it constructs training targets

$$y_t^{(k)} = r_t + \gamma \max_{a' \in \mathcal{A}} Q_k(s_{t+1}, a'),$$

from the dataset $\mathcal{D}$, and then it updates Q_{k+1} by fitting a regressor from a model class $\mathcal{F}$ onto these targets.

$$Q_{k+1} = \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{t=1}^{T} \left(f(s_t, a_t) - y_t^{(k)}\right)^2$$

Once we have the approximated $\hat{Q}$-function, we use the greedy policy given by,

$$\pi(s) = \operatorname*{argmax}_{a \in \mathcal{A}} \hat{Q}(s, a)$$

In the time-aware variant of FQI, we make the Q-function a function of time t in addition to the state s and action a. We do this by slightly modifying the iterative condition as follows. At iteration k, it constructs training targets

$$y_t^{(k)} = r_t + \gamma \max_{a' \in \mathcal{A}} Q_k(s_{t+1}, a', t),$$

from the dataset $\mathcal{D}$, and then it updates Q_{k+1} by fitting a regressor from a model class $\mathcal{F}$ onto these targets.

$$Q_{k+1} = \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{t=1}^{T} \left(f(s_t, a_t, t) - y_t^{(k)}\right)^2$$

As before, once we have the approximated $\hat{Q}$-function, we construct a greedy policy by,

$$\pi(s, t) = \operatorname*{argmax}_{a \in \mathcal{A}} \hat{Q}(s, a, t)$$

However, unlike the static policy from generic FQI, the time-aware variant leads to a dynamic policy that varies with time. In our experiments, we use a Random Forest with 1000 trees as the regressor of the FQI. We illustrate the pseudocode of online FQI in algorithm 2.

Algorithm 2 Online Fitted Q-Iteration with ϵ-greedy exploration

Initialize Foraging environment `env`, experience buffer $\mathcal{D}$, warmup size W, update interval I, exploration parameter ϵ, action space $\mathcal{A} = \{-1, 0, 1\}$
$\hat{Q} \leftarrow$ None
for $t = 1$ to T
    $s_t \leftarrow$ `env.get_current_state()`
    Sample $u \sim$ Uniform$[0, 1]$
    if $u < \max(0.01, \epsilon \cdot 0.999^t)$ **or** Q is None
        $a_t \leftarrow$ RandomChoice$(\mathcal{A})$ ▷ Explore
    else
        $a_t \leftarrow \text{argmax}_{a' \in \mathcal{A}} \hat{Q}(s, a', t)$ ▷ Exploit
    end if
    $(s_{t+1}, r_t) \leftarrow$ `env.step`(a_t)
    Store (s_t, a_t, r_t, t) in $\mathcal{D}$
    if $|\mathcal{D}| \geq W$ **and** $(t + 1) \bmod I = 0$
        $\hat{Q} =$ `FQI.fit`$(\mathcal{D})$
    end if
end for

C.2. Soft Actor-Critic

Soft Actor-Critic (SAC) (Haarnoja et al., 2018a,b) is an off-policy, model-free algorithm developed to perform entropy-regularized reinforcement learning, where the agent aims to maximize both cumulative return and the entropy of the policy;

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r_t + \alpha H\left(\pi(\cdot \mid s_t)\right)\right)\right]$$

Here, α is the temperature parameter, $H(\cdot)$ is the entropy, and γ is the discount factor. Maximizing the entropy of the policy alongside the cumulative reward encourages exploration and prevents premature convergence to suboptimal policies. Algorithm 3 illustrates the online-SAC algorithm used in this work. We build the time-aware SAC by making the policy, critics, and target critics functions of time t.

C.3. Proximal Policy Optimization

Proximal Policy Optimization (PPO) (Schulman et al., 2017) is an on-policy reinforcement learning algorithm that seeks to improve an agent's policy while ensuring that updates do not deviate too far from the current behavior.

While standard PPO is typically applied to episodic tasks with frequent environment resets, our implementation is adapted for a single-episode, reset-free regime. In this setting, the agent must continuously learn and adapt over a single, indefinite trajectory. We introduce several key modifications to the original framework:

Algorithm 3 Online Soft Actor-Critic

Initialize Foraging environment `env`, experience buffer $\mathcal{D}$, Policy network π_θ, Q-function/critic networks Q_{ϕ_1} and Q_{ϕ_2}, batch size B
Set target critic parameters equal to main critic parameters $\phi_{\text{targ},1} \leftarrow \phi_1$, $\phi_{\text{targ},2} \leftarrow \phi_2$
for $t = 1$ to T
    $s_t \leftarrow$ `env.get_current_state()`
    $a_t \sim \pi_\theta(\cdot \mid s_t, t)$
    $(s_{t+1}, r_t) \leftarrow$ `env.step`(a_t)
    Store (s_t, a_t, r_t, t) in $\mathcal{D}$
    if $|\mathcal{D}| \geq B$
        Randomly sample a batch $\mathcal{B} = \{(s_t, a_t, r_t, t)\}$ from $\mathcal{D}$
        Compute targets for the critic networks:

$$y(r_t, s_{t+1}) = r_t + \gamma \left(\min_{i=1,2} Q_{\phi_{\text{targ},i}}(s_{t+1}, a', t+1) - \alpha \log \pi_\theta(a' \mid s_{t+1}, t+1) \right);\ a' \sim \pi_\theta(\cdot \mid s_{t+1}, t+1)$$

        Update the critic networks by one step of gradient descent using

$$\nabla_{\phi_{\text{targ},i}} \frac{1}{B} \sum_{(s_t, a_t, r_t, t) \in \mathcal{B}} \left(Q_{\phi_{\text{targ},i}}(s_t, a_t, t) - y(r_t, s_{t+1}) \right)^2;\ \text{for } i = 1, 2$$

        Update policy network by one step of gradient ascent using

$$\nabla_\theta \frac{1}{B} \sum_{s_t \in \mathcal{B}} \left(\min_{i=1,2} Q_{\phi_{\text{targ},i}}(s_t, a_\theta(s_t), t) - \alpha \log \pi_\theta(a_\theta(s_t) \mid s_t, t) \right);\ a_\theta(s_t) \sim \pi_\theta(\cdot \mid s_t, t)$$

        Update the target networks with

$$\phi_{\text{targ},i} \leftarrow \rho \phi_{\text{targ},i} + (1 - \rho) \phi_i;\ \text{for } i = 1, 2$$

    end if
end for

Time-aware policy and value networks To handle the non-stationary nature of the foraging task, we make both the policy (π_θ) and critic (V_ϕ) functions of time t.

Infinite-Horizon Bootstrapping: In the absence of terminal states, we modify the Generalized Advantage Estimation (GAE). The advantage $\hat{A}_t$ is calculated by bootstrapping from the critic's value estimate of the next state for every transition, treating the simulation as a continuous stream of experience rather than a series of independent episodes.

Online transition management The Rollout Buffer is utilized to store a fixed window of transitions. Updates are performed at regular intervals (every 512 steps), allowing for online policy refinement without the need for a global reset.

The online training loop of our PPO learner is outlined in algorithm 4.

Algorithm 4 Online PPO for single-episode reset-free tasks

Initialize: Policy π_θ, Value network V_ϕ, Rollout Buffer $\mathcal{B}$
Set parameters: Clip ϵ, GAE parameters γ, λ, update interval N, epochs K
for $t = 0, 1, 2, \ldots, T$
    $s_t \leftarrow$ `env`.get_current_state()
    $a_t \sim \pi_\theta(\cdot \mid s_t, t)$
    $(s_{t+1}, r_t) \leftarrow$ `env`.step(a_t)
    Store $(s_t, a_t, t, r_t, \log \pi_\theta(a_t|s_t, t), V_\phi(s_t, t))$ in $\mathcal{B}$
    if $(t+1) \bmod N = 0$
        Bootstrap next value: $\hat{V}_{next} = V_\phi(s_{t+1}, t)$
        Estimate the advantages $\hat{A}_i$ using GAE
        Compute targets $R_i = \hat{A}_i + V_\phi(s_i, E_i)$ and normalize $\hat{A}$
        for $k = 1, \ldots, K$
            Calculate ratio $r_i(\theta) = \exp(\log \pi_\theta(a_i|s_i, t) - \log \pi_{old}(a_i|s_i, t))$
            $L_{clip} = \mathbb{E}_i[\min(r_i(\theta)\hat{A}_i, \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i)]$
            $L_v = \mathbb{E}_i[(V_\phi(s_i, t) - R_i)^2]$
            $L_{ent} = \mathbb{E}_i[\text{Entropy}(\pi_\theta(\cdot|s_i, t))]$
            Update θ, ϕ via Adam $\nabla_{\theta,\phi}(L^{clip} - c_1 L^v + c_2 L^{ent})$
        end for
        Clear buffer $\mathcal{B}$
    end if
end for

Appendix D. Stochastic Sampling

In Algorithm 1, we show that during the online update stage, the next state is sampled using Boltzmann sampling (Sutton, 1990; Cesa-Bianchi et al., 2017), where the selection probability of each candidate next state is proportional to its estimated prospective reward. Here, we compare two online PLuC variants: one using deterministic argmax style selection, and the other using Boltzmann sampling.

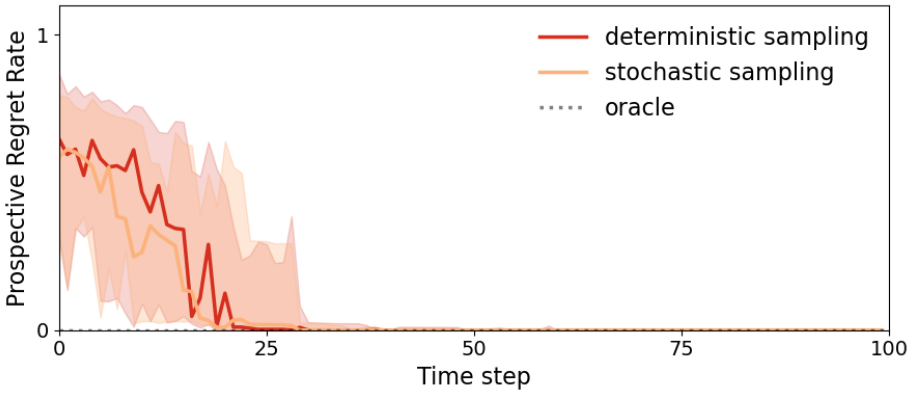

Figure 6: **Online PLuC with stochastic sampling and deterministic sampling both approach oracle level performance.** In this environment, both approaches reduce prospective regret over time and eventually approach oracle-level performance, while stochastic sampling (red) exhibits a slightly faster decrease at the beginning than the deterministic sampling (orange).