Sample-efficient reinforcement learning for environments with rare high-reward states

Daniel Mastropietro, Urtzi Ayesta, Matthieu Jonckheere

Sample-efficient reinforcement learning for environments with rare high-reward states

Daniel Mastropietro*
CNRS-IRIT
Université de Toulouse
31071 Toulouse, France
daniel.mastropietro@irit.fr

Urtzi Ayesta
CNRS-IRIT
31071 Toulouse, France
Ikerbasque - Basque Foundation for Science
48009 Bilbao, Spain
UPV/EHU
University of the Basque Country
20018 Donostia, Spain
urtzi.ayesta@irit.fr

Matthieu Jonckheere
CNRS-LAAS
31071 Toulouse, France
mjonckheer@laas.fr

Abstract

We introduce FVAC (Fleming-Viot Actor-Critic), an algorithm for efficient learning of optimal policies in reinforcement learning problems with rare, high-reward states. FVAC uses Actor-Critic policy gradient, with the critic estimated via the so-called Fleming-Viot particle system, a stochastic process used to model population evolution which is able to boost the visit frequency of the rare states. This frequency boosting is obtained by *forcing* exploration outside a set of states identified as highly visited during an initial exploration of the environment. The only requirements of the method are that learning must be set under the average reward criterion, and that a black-box simulator or emulator can be run on the environment. We showcase the method's performance in windy grid worlds, where a non-zero reward is only observed at a terminal cell, which is difficult to reach due to the wind. Our results show that FVAC learns significantly faster than standard reinforcement learning algorithms based on Monte-Carlo exploration with temporal difference learning.

1 Introduction

Reinforcement learning (RL) excels in solving complex stochastic optimisation problems by making minimal assumptions about the underlying model, treating systems as Markov decision processes (MDPs) without the need to know their transition probabilities. Through interactions between an agent and the environment, RL methods learn optimal policies through trial and error, adjusting

*Alternative e-mail: daniel.mastropietro@gmail.com

17th European Workshop on Reinforcement Learning (EWRL 2024).

decisions based on rewards or costs received. This model-free paradigm is particularly powerful for addressing intricate and nonlinear optimisation challenges, but can be computationally demanding. This is especially true in vast environments where many states and actions are under-explored, yet potentially highly informative.

This paper focuses on environments characterised by infrequently observed high-reward states, which complicate the optimisation of objectives heavily dependent on such rare occurrences. Our broad objective is to develop methods that optimize long-run expectations of rewards, significant in settings where rare but critical states dictate long-term performance.

Traditional strategies for addressing the challenges of rare rewards in RL include importance sampling [11] and reward shaping [5]. Importance sampling modifies the exploration policy to increase the occurrence of rare events, but it provides limited improvement in environments where no policy can effectively achieve a sufficient coverage of crucial states. Reward shaping and curiosity-driven methods [2, 7], which adjust rewards or add incentives to explore under-visited states, often require extensive domain knowledge and might lead to narrow or ineffective policies.

Our approach is based on a novel exploration mechanism introduced in [4, 3] that not only encourages but enforces exploration outside of frequently visited states. It utilizes the notion of Fleming-Viot particle system, a stochastic process used to model population evolution, to ensure a more balanced exploration of the space, targeting states that are rarely visited. This facilitates a more effective estimation of long-term expected values, critical for policy optimisation in complex stochastic environments.

Contribution We extend the FVRL method (Fleming-Viot Reinforcement Learning) presented in [3, 4] as a mechanism that leverages Fleming-Viot particle systems and policy gradient to efficiently solve RL control problems with sparse and rare rewards, where sparse means that non-zero rewards are accrued in very few states. Those works develop the method for continuous-time systems in continuing learning scenarios, and showcase it on the design of threshold-type admission control policies for queues and stochastic networks, a very specific class of policy. The threshold-type policies allow for an ad-hoc policy parameterisation that makes the policy gradient also sparse, which considerably facilitates the estimation of the objective functions's gradient.

This paper extends FVRL in three ways: (i) to discrete-time episodic learning tasks (e.g. games), (ii) to scenarios where rewards need not be sparse, and (iii) to general parameterised policies. We call the extended method FVAC (Fleming-Viot Actor-Critic), in order to emphasize its applicability to optimisation problems where a critic needs to be learned for all states in the environment, not only those with non-zero gradient.

FVAC also relaxes the need for prior knowledge of the underlying Markov process about a subset $\mathcal{A}$ of states with zero reward, a hyperparameter that is at the core of the method. The concept of $\mathcal{A}$ has now been replaced with the set of states that are frequently visited, above a pre-specified threshold. Thus, set $\mathcal{A}$ is now defined from an initial exploration of the environment, where both state visit frequency and rewards are collected. Not only does this provide an automatic definition of $\mathcal{A}$, without user intervention or knowledge about the problem, but also eliminates the sparsity requirement in [3, 4] (embedded in the condition that no state in $\mathcal{A}$ should yield a reward). All rewards possibly coming from visits to states in $\mathcal{A}$ are now integrated into the long-run expected reward estimation through the initial exploration of the environment. This approach also makes the set of the method's hyperparameters easier to define, as the $\mathcal{A}$ hyperparameter has been replaced with a visit frequency threshold, a much more intuitive concept.

The only knowledge still required about the environment is that a black-box simulator or emulator can be run on it. Relying on the convergence guarantees in [4], we demonstrate the method application in windy grid worlds where a non-zero reward is observed at just one terminal cell, which is very difficult to reach due to the wind. Here, the objective is to find the optimal path amidst obstacles, which can be likened to navigating through rare rewarding states. We employ a neural network model as a function approximator for the policy, which allows the RL agent to generalize from limited data and infer the shortest path to the terminal state. This application illustrates the method potential in environments where paths to rewards are not only rare but obstructed by challenges, making traditional exploration methods less adequate.

Through experiments run on a small 4x5 grid world test bench, with a few specifically placed obstacles and a "strong" wind, and on a larger 6x8 grid world with obstacles placed at random and a milder wind, we numerically show the significant advantage of FVAC in learning an optimal policy over a benchmark method based on Monte-Carlo exploration of the grid world. It is important to note that, for a fair comparison, both compared methods use the same exploration and learning budget over the course of each experiment.

The rest of the paper is organized as follows: in Section 2 we describe the stochastic optimisation problem and our proposal to solve it efficiently in the presence of rare high-reward states. In Section 3 we present the FVAC methodology in detail, in Section 4 we summarize the experimental results, and we finalize with conclusions.

2 Problem description

We consider stochastic optimisation problems that can be modelled by a discrete-time MDP $(\mathcal{S}, \mathcal{U}, \mathcal{P}, \mathcal{R})$ with finite state space $\mathcal{S}$, finite action space $\mathcal{U}$, transition probability matrix $\mathcal{P} = \{P(x, a, x')\}_{x,x' \in \mathcal{S}, a \in \mathcal{U}}$, and $\mathcal{R}$ the space of reward functions r of the current state x, the action taken a, and the next state x', namely $r : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}$.

We denote by $\pi : \mathcal{S} \to \mathcal{U}$ the policy applied by an agent interacting with the environment, and by $\{X_n\}_{n \in \mathbb{N}}$ the discrete-time Markov chain that follows the dynamics of the MDP. For compactness of notation, we may also write X_n^π to denote the Markov chain defined by policy π. This Markov chain has a transition probability matrix $\mathcal{P}^\pi = \{P^\pi(x, x')\}_{x,x' \in \mathcal{S}}$, with $P^\pi(x, x') = \sum_{a \in \mathcal{U}} P(x, a, x')\pi(a|x)$. As in [10, Chapter 10.3], we assume that the policy π applied to the MDP results in an erdogic Markov chain, meaning that a unique stationary probability, $p^\pi(x)$, exists.

Our goal is the design of an RL algorithm that is able to efficiently find optimal policies when the reward landscape of the MDP contains large rewards rarely observed. The proposed method, called Fleming-Viot Actor Critic (FVAC), requires that the objective function of the optimisation problem can be expressed as a long-run expectation, due to the characteristics of the Fleming-Viot process behind FVAC which is used to provide a reliable estimation of the stationary distribution. The most direct example is the long-run expected reward in RL continuing tasks, a.k.a. RL under the average reward criterion. If the problem is not in this form, it must be cast into it before applying FVAC. A common such scenario is when the optimisation problem naturally calls for an RL episodic task, such as games. In order to easily cast the episodic problem into a continuing learning task setting, the RL approach should consider the expected total reward as optimisation objective, i.e. no discount should be used. The RL problem can then be cast into the average reward criterion setting by simply restarting the MDP at an environment's start state whenever an episode finishes.

To develop our method, we therefore consider as objective function the long-run expected reward observed by the MDP under policy π, defined as:

$$\overline{r}(\pi) \doteq \lim_{n \to \infty} \mathbb{E}^\pi(\overline{r}(X_n, \pi)) = \sum_{x \in \mathcal{S}} \overline{r}(x, \pi) p^\pi(x), \tag{1}$$

where $\overline{r}(x, \pi)$ is the average reward observed at state x under policy π, i.e. $\overline{r}(x, \pi) = \sum_{a \in \mathcal{U}} \overline{r}(x, a)\pi(a|x)$, with $\overline{r}(x, a) = \sum_{x' \in \mathcal{S}} r(x, a, x')P(x, a, x')$.

Before delving in Section 3 into the method's details, we provide notation and relevant definitions.

Notation Lowercase letters are used for true quantities, e.g. $v^\pi(x)$, and uppercase letters for random and/or estimated quantities, e.g. $V^\pi(x)$. Exceptions are probabilities, expectations, and quantities indicated in greek symbols for which a hat is used to indicate estimates, e.g. $\hat{\mathbb{P}}, \hat{\mathbb{E}}, \hat{\phi}$.

Initial conditions of the MDP in expectation and probability computations are indicated as subscripts. Unless otherwise indicated, the initial conditions correspond to the stationary regime. E.g. $\mathbb{E}^\pi_{\overleftarrow{\partial \mathcal{A}}}(.)$ means that the expectation is computed when the underlying Markov process X_n^π starts at the entrance boundary of $\mathcal{A}$ (see definition below) following the entrance state distribution under stationarity.

The policy π that drives the Markov process underlying the computation of expectations, probabilities, and estimators, is indicated as a superscript, e.g. $V^\pi(x)$. However, to avoid overloaded expressions, the superscript might be sometimes omitted when obvious.

Definitions The set $\mathcal{A}$ mentioned in the Introduction is called the absorption set because the underlying Markov process used to construct the Fleming-Viot particle system is absorbed when it touches $\mathcal{A}$ [3]. We define the entrance boundary of $\mathcal{A}$ as the subset of $\mathcal{A}$ that can be reached in one step from outside $\mathcal{A}$, and the exit boundary of $\mathcal{A}$ as all the states *outside* $\mathcal{A}$ that can be reached in one step from $\mathcal{A}$[2]. These boundaries are denoted as $\overleftarrow{\partial\mathcal{A}}$ and $\overrightarrow{\partial\mathcal{A}}$, respectively. Similarly its entrance and exit state stationary distributions under π are respectively denoted by $p^\pi_{\overleftarrow{\partial\mathcal{A}}}$ and $p^\pi_{\overrightarrow{\partial\mathcal{A}}}$.

We also define the killing time $T_\mathcal{K}$ as the hitting time of $\mathcal{A}$, and $T_\mathcal{A}$ as the cycle reabsorption time to $\mathcal{A}$, i.e. the time elapsed between two consecutive entries to $\mathcal{A}$. For their formal definition see [3].

Value functions: In the RL average reward criterion context, the state and action value functions take a differential form [10, Chapter 10.3], i.e. they are defined in terms of the difference between the observed reward and the long-run expected reward, as:

$$\begin{aligned} v^\pi(x) &\doteq \mathbb{E}^\pi[\sum_{n=1}^{\infty} \overline{r}(X_n,\pi) - \overline{r}(\pi)|X_0 = x] \\ q^\pi(x,a) &\doteq \mathbb{E}^\pi[\sum_{n=1}^{\infty} \overline{r}(X_n,\pi) - \overline{r}(\pi)|X_0 = x, A_0 = a]. \end{aligned}$$

The advantage function $h^\pi(x,a)$ is defined as the difference between the action and the state value functions,

$$h^\pi(x,a) = q^\pi(x,a) - v^\pi(x),$$

and is useful to easily compare the advantage of taking a particular action instead of another one, in terms of maximizing the long-run expected reward.

3 Methodology

In this section we describe the FVAC method (Fleming-Viot Actor-Critic), which integrates a Fleming-Viot estimator of the critic into the Actor-Critic policy gradient algorithm for continuing learning tasks.

The Actor-Critic policy gradient algorithm optimizes a parameterised policy maximizing the performance measure of the problem [10, Chapters 13.5, 13.6]. In this algorithm, the critic is the one-step bootstrapped temporal difference (TD) error, which is an estimate of the advantage function, $h^\pi(x,a)$.

For each parameterised policy π_θ, the advantage function in tabular form is first estimated by a Fleming-Viot particle system, which is then used as critic in the policy gradient step. We remark that, contrary to the original Actor-Critic algorithm presented in [10, Chapter 13.6], FVAC, by construction, does not allow learning the critic during the policy learning excursion, as Fleming-Viot modifies the original MDP to boost the visit frequency of the states that are rarely observed under the original MDP.

Because Fleming-Viot is a continuous-time stochastic process, the discrete-time MDP first needs to be cast into a continuous-time MDP, as explained in the next section. A schematic diagram of the FVAC method is presented in Figure 1 and is given in detail in Algorithm 1.

For simplicity of exposition, we assume that we have direct access to the reward function r, or otherwise that the function can be easily learned (which is commonly the case).

3.1 Estimating the long-run expected reward with Fleming-Viot particle systems

The Fleming-Viot particle system (FV) is defined in [4] for continuous-time MDPs in continuing learning tasks as a sample-efficient mechanism to estimate the long-run expected reward by boosting

[2]Formally, the entrance boundary is the set of states $y \in \mathcal{A}$ s.t. $P(x,a,y) > 0$ for some $a \in \mathcal{U}, x \in \mathcal{S} \setminus \mathcal{A}$, and the exit boundary is the set of states $y \in \mathcal{S} \setminus \mathcal{A}$ s.t. $P(x,a,y) > 0$ for some $a \in \mathcal{U}, x \in \mathcal{A}$.

the visit frequency of rarely observed states. Convergence guarantees of the FV estimator are given when the state and action spaces are finite. The reader is invited to consult the given reference for further details.

Here we limit ourselves to explaining how the continuous-time FV estimation of the long-run expected reward can be extended to discrete-time MDPs, which are either continuing or episodic.

We first recall that at the core of the FV estimator of the long-run expected reward presented in [3, 4] is the estimation of the state stationary distribution, which is required to compute the long-run expected reward using (1). The FV estimator of the stationary probability relies on the FV particle system which is designed to boost the exploration of rarely visited states. This boost is obtained by defining a set $\mathcal{A}$ of frequently visited states where the MDP is killed or absorbed, hence $\mathcal{A}$ is called the absorption set. An FV particle system containing N particles works as follows: all N particles independently follow the dynamics of the MDP until one of them is absorbed in $\mathcal{A}$. At this moment, the particle is immediately reactivated to the position (state) of one of the other $N-1$ particles selected uniformly at random. The independent evolution resumes until the next absorption is observed that leads to the next reactivation, and so forth.

Having collected trajectories through a simulation of the FV particle system, the FV estimator of the stationary probability is constructed on the basis of the following characterisation of $p^\pi(x)$, derived from renewal theory [3, 4]:

$$p^\pi(x) = \frac{\int_0^\infty \mathbb{P}^\pi_{\underset{\partial\mathcal{A}}{\rightarrow}}(T_\mathcal{K} > t)\phi^\pi_{\underset{\partial\mathcal{A}}{\rightarrow}}(t,x)dt}{\mathbb{E}^\pi_{\underset{\partial\mathcal{A}}{\leftarrow}}(T_\mathcal{A})}, \forall x \notin \mathcal{A}, \tag{2}$$

where $\phi^\pi_{\underset{\partial\mathcal{A}}{\rightarrow}}(t,x) = \mathbb{P}^\pi_{\underset{\partial\mathcal{A}}{\rightarrow}}(X_t = x | T_\mathcal{K} > t)$ is the occupation probability of state x conditional to not being absorbed, when the process started at the exit boundary of $\mathcal{A}$. All subscripts denote that probabilities and expectations are to be computed when the MDP starts following the stationary distribution of the indicated set[3]. The FV estimator of $p^\pi(x)$ is then constructed by estimating each of the three components of (2) as explained in [3].

We now describe how this continuous-time estimation approach can be adapted to discrete-time settings, using Figure 1 as support diagram. Of the four steps given below, only the fourth affects implementation; the first three are methodological. Given the original discrete-time MDP, either continuing or episodic (box (1) in the diagram), the adaptation steps are the following:

i) Cast the process to a continuing MDP if originally episodic.
   If the MDP is episodic, convert it to continuing by restarting the MDP at an environment's start state whenever a terminal state is reached.

ii) Cast to a continuous-time MDP, which leads to box (2) of the diagram.
   The discrete-time MDP is cast to a continuous-time MDP with homogeneous jump rate equal to 1, which has the same stationary distribution as the discrete-time MDP [1, Chapter 13].

   Steps (1) and (2) above do not change the set of optimal policies [8] and allow moving to the average reward context of the Fleming-Viot particle system.

iii) Use uniformization [9] to create two discrete-time MDPs that allow carrying out the simulation and defining the FV estimator of the stationary probability in (2):
   (a) the original discrete-time MDP using uniformization constant equal to 1. This leads to box (3).
   (b) the FV particle system with N particles with uniformization constant equal to N, the highest (and constant) jump rate of the FV system. This leads to box (4). We note that the obtained discrete-time MDP is measured at discrete times $\{n/N\}_{n\geq 1}$, which is important for the construction of the FV estimator given below.

iv) Simulate system (a) to estimate the expected reabsorption time in the denominator of (2), $\hat{\mathbb{E}}^\pi_{\underset{\partial\mathcal{A}}{\leftarrow}}(T_\mathcal{A})$. Simulate system (b) to estimate the FV integral in the numerator of (2). Thanks

[3]The probability of a set B when the MDP starts with a given distribution μ is defined as $\mathbb{P}^\pi_\mu(B) = \sum_{x\in\mathcal{S}} \mathbb{P}^\pi_x(B)\mu(x)$.

to uniformization, the estimates of the two probability functions in the FV integral are piecewise constant with pieces of sizes that are integer multiples of $1/N$. The FV estimator of the stationary distribution can then be written as:

$$\hat{p}^{\pi}_{FV}(x) = \frac{\frac{1}{N}\sum_{n=1}^{N} \hat{\mathbb{P}}^{\pi}_{\underset{\partial\mathcal{A}}{\rightarrow}}(T_{\mathcal{K}} \geq n/N)\hat{\phi}^{\pi}_{\underset{\partial\mathcal{A}}{\rightarrow}}(n/N, x)}{\hat{\mathbb{E}}^{\pi}_{\underset{\partial\mathcal{A}}{\leftarrow}}(T_{\mathcal{A}})}, \quad \forall x \notin \mathcal{A}, \tag{3}$$

where $\hat{\mathbb{P}}^{\pi}_{\underset{\partial\mathcal{A}}{\rightarrow}}(T_{\mathcal{K}} \geq n/N)$ is a regular moment estimator of the survival probability of the N particles, and $\hat{\phi}^{\pi}_{\underset{\partial\mathcal{A}}{\rightarrow}}(n/N, x)$ is the empirical occupation proportion at state x of the N particles at the given times n/N [3]. The estimation details of the three quantities involved in (3) are given in Algorithm 1.

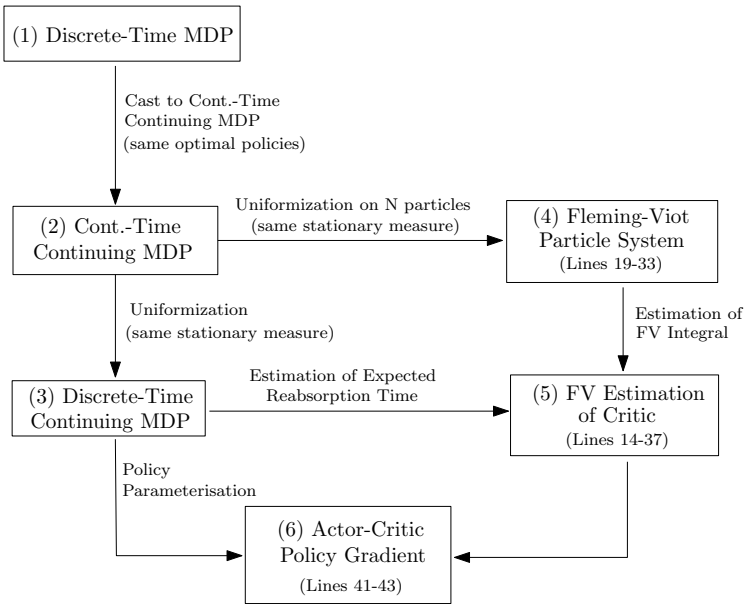


Figure 1: Diagram giving a schematic description of the FVAC method for optimal policy learning, including the methodological adaptation of the discrete-time MDP that allows applying it. Line numbers correspond to the section in the FVAC Algorithm 1 that is relevant to the respective box.

The estimated stationary distribution is instrumental in estimating the long-run expected reward using (1), which in turn is needed to estimate the advantage function $h^{\pi}(x, a)$, as explained in Section 3.3.

3.2 Learning the actor with Actor-Critic policy gradient

The optimal policy $\pi_\theta(a|x)$ is learned by an Actor-Critic policy gradient approach where θ is the parameter vector of a model (e.g. neural network) with state x as input, and the probability of taking action a given state x, as output.

We aim at finding an optimal θ that maximizes the performance $j(\theta)$ defined as the long-run expected reward, $j(\theta) \doteq \overline{r}(\pi_\theta)$, by stochastic gradient ascent, i.e. by iteratively updating θ with a stochastic estimate of the gradient.

By the policy gradient theorem applied to continuing learning tasks [10, Section 13.6], the gradient of $j(\theta)$ only depends on the gradient of the policy, and can be written as:

$$\nabla j(\theta) = \mathbb{E}_{X_n \sim p^{\pi_\theta}, A_n \sim \pi_\theta}\left[h^{\pi_\theta}(X_n, A_n)\nabla \log \pi_\theta(A_n|X_n)\right]. \tag{4}$$

Thus, the stochastic gradient ascent algorithm consists of the following update formula for θ at each learning step $t \geq 0$:

$$\theta_{t+1} = \theta_t + \eta_t \nabla J(\theta_t) \tag{5}$$

where $\eta_t > 0$ is the learning rate at step t and $\nabla J(\theta_t)$ is a moment estimate of the gradient in (4) obtained from a trajectory of the MDP under π_{θ_t}, as:

$$\nabla J(\theta_t) = \frac{1}{M} \sum_{n=1}^{M} H^{\pi_{\theta_t}}(X_n, A_n) \nabla \log \pi_{\theta_t}(A_n|X_n), \tag{6}$$

where M is a predefined fixed number of steps taken by the MDP under policy π_{θ_t}, and $H^{\pi_{\theta_t}}(X_n, A_n)$ is an estimate of the advantage function when following policy π_{θ_t}, obtained separately from the generated trajectory and used as critic.

Our proposed FVAC method consists of using a Fleming-Viot particle system to estimate the critic for each value of θ_t –the advantage function in (6)– and then using (5) as the stochastic gradient ascent update formula to learn an optimal θ. This integration is described in the next section.

3.3 FVAC: Learning the optimal policy with Fleming-Viot Actor-Critic

We now give the details of the FV estimation of the critic, the advantage function, and its integration with the Actor-Critic policy gradient algorithm.

Given a policy π and following Algorithm 2 proposed in [12], the advantage function under the average reward criterion is estimated as the one-step bootstrapped error of the target at each discrete-time n, in other words as the TD(0) error corrected by the long-run average reward, i.e.:

$$H^{\pi}(X_n, A_n) = R_n - \overline{R}_n(\pi) + V^{\pi}(X_{n+1}) - V^{\pi}(X_n),$$

where $R_n \doteq r(X_n, A_n, X_{n+1})$ and X_{n+1} are respectively the reward and the next state observed after taking action $A_n \sim \pi$, and $\overline{R}_n(\pi)$ and $V^{\pi}(\cdot)$ are the estimates at time n of the long-run expected reward and the state value function, respectively.

The above one-step bootstrapped estimation of the advantage function using a Fleming-Viot particle system is obtained from the following general estimation scheme: the state value function $v^{\pi}(x)$ is learned using TD(0) updates at every transition of the FV particles that is consistent with the original MDP (i.e. particle reactivations are excluded), and the long-run expected reward $\overline{r}(\pi)$ is estimated using the procedure described in Section 3.1 on the data collected by the FV learner trajectories. Details are given in the FVAC Algorithm 1.

The learned advantage function is then used as critic to perform one learning step of an Actor-Critic policy learner. The Actor-Critic learner updates the θ parameter of the parameterised policy π_θ using update formula (5), which uses an estimate of the gradient of the performance measure computed by (6) from a pre-defined number M of simulation steps of the Markov chain under the current policy π_θ[4].

The process of learning the advantage function by Fleming-Viot and feeding it into the Actor-Critic policy gradient step is repeated until a pre-specified given condition is reached, such as a maximum number of policy learning steps or obtaining sufficiently small variations in consecutive θ estimates.

4 Experimental results

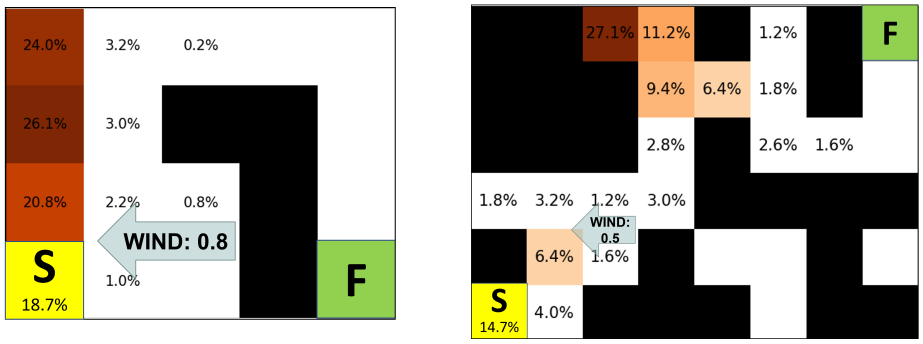
To showcase the method, we consider the problem of learning the shortest path of simple 2D mazes with a single exit, defined as grid worlds with a fixed start location and one terminal cell, the only state with positive reward. The state space is made up of all cells of the grid world and the action space is the set {0, 1, 2, 3} representing all possible movements of the agent, respectively {UP, RIGHT, DOWN, LEFT}.

In order to easily parameterise the rareness of the single reward state, we add a homogeneous wind of varying intensity defining the probability with which the agent is deviated one cell in the direction of the wind after having moved, obstacle permitting, in the chosen direction.

[4]No convergence guarantees to an optimal policy exist when the policy optimisation problem is non-convex, such as a neural network.

The policy π_θ is parameterised by a three-layer neural network with $|\mathcal{S}|$ input neurons representing each cell on the grid world and 4 output neurons representing each of the four possible actions at each cell. The hidden layer contains 12 neurons.

Experiments were run using simulation of the MDP on the two mazes depicted in Figure 2 with wind blowing uniformly in the left direction: a smaller 4×5 maze with obstacles placed in four specifically chosen cells and intensity 0.8 of the wind, and a larger 6×8 maze with randomly placed obstacles in approximately 50% of the cells and intensity 0.5 of the wind. Also shown is the relative visit frequency of the cells during the initial exploration of the environment under the initial random policy[5], from which the absorption set $\mathcal{A}$ is defined following the procedure described in Algorithm 1, whose cells are highlighted in different shades of orange[6]. Although not shaded, the start state is also part of the absorption set in both cases.



(a) 4×5 maze

(b) 6×8 maze

Figure 2: The two windy mazes used in the policy learning experiments. S: start cell, F: finish cell, black cell: obstacle location that the agent cannot visit, orange cells: locations belonging to the identified absorption set $\mathcal{A}$ as described in Algorithm 1, presenting a relative visit frequency of at least 5% from the initial exploration of the environment (darker colors indicate higher frequencies and in both cases, the start state is also part of the absorption set). Percent values represent the relative cell visit frequency observed during the initial exploration of the environment under the random policy for $T = 500$ steps. Rewards are zero everywhere except at F where it is +1. The blue arrow indicates a homogeneous wind to the left, which is 0.8 in the smaller 4×5 maze and 0.5 in the larger 6×8 maze.

Each experiment consists of running the FVAC method and a benchmark method on a certain number of replications, as follows:

1. The FVAC algorithm is run first on a predefined number of policy learning steps (150 in this case). The number of steps used by the Fleming-Viot estimation of the advantage function at each policy learning step, on each replication, is taken note of.
2. The benchmark method is then run on as many replications and for the same number of policy learning steps, using, at each policy learning iteration, as many steps as the average number observed by FVAC along the policy learning process at the respective replication.

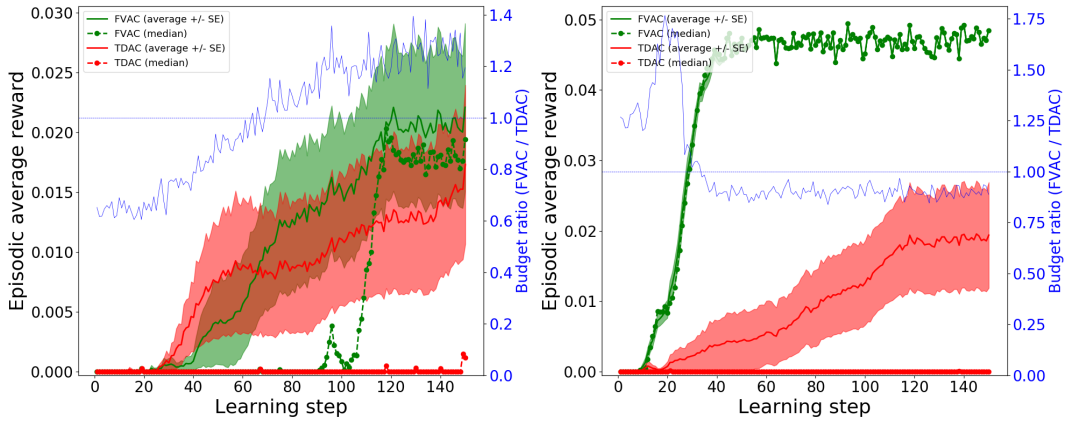
The above setup allows a fair comparison between the FVAC and the benchmark method, as it guarantees that both use the same total number of simulation steps over all policy learning steps.

We use the so-called TDAC (Temporal Difference Actor-Critic) as benchmark method, in which TD(0) learns the advantage function at each policy learning step. The hyperparameteter setup of the FVAC algorithm corresponds to the default values presented in Algorithm 1.

[5]Considering the policy parameter θ is given by the parameters of the neural network used to model the policy, all weights and biases are initialized to random values sampled from a zero-mean Gaussian distribution with 0.01 standard deviation, except the biases of the 4 output neurons which are initialized to 1. This setup guarantees that the policy is approximately uniformly random. Weights are not exactly set to zero, as otherwise no learning ever takes place, due to the characteristics of the back-propagation algorithm.

[6]The threshold for the minimum relative visit frequency is the default one: $\alpha_{\mathcal{A}} = 5\%$.

The learning curves of FVAC and TDAC over 10 replications are presented in Figure 3. We observe that FVAC outperforms TDAC, both on average and median learning. For the 4×5 maze (Figure 3(a)), the average and median learning curves of FVAC reach a stable value around 0.020 (representing 50 steps from start to finish), which is consistent with the shortest path of the problem given the strong wind, whereas TDAC doesn't reach this level and remains practically stuck at 0 in median. For the 6×8 maze (Figure 3(b)), TDAC also remains at 0 in median compared to a very fast learning curve by FVAC, while on average TDAC reaches half of the value reached by FVAC. Additionally, the variability observed in the FVAC learning curves is negligible compared to that of the TDAC learning curves, thanks mostly to a very appropriately identified absorption set $\mathcal{A}$ which keeps exploration away from the uninteresting set of cells and close in part to the finish state, as shown in Figure 2(b).



(a) 4×5 maze (b) 6×8 maze

Figure 3: Learning curves for the windy maze environments depicted in Figure 2: comparison between FVAC and TDAC over 10 replications of learning a near-optimal policy leading to the shortest path from S to F. Both the average ($\pm$ its standard error) and the median learning curves are plotted for each method. The main FVAC hyperparameters are the default ones: number of FV particles $N = 20$, initial exploration steps $T = 500$, and threshold for absorption set $\alpha_{\mathcal{A}} = 5\%$. The remaining hyperparameters take the default values given in Algorithm 1. The blue line plotted on the right axis corresponds to the budget ratio described in the text whose average is 1 by design, which guarantees a fair comparison between FVAC and TDAC. Experiments were run using Python-3.6 on a Windows 10 (64-bit) computer, with execution times by replication varying in the range 13-19 minutes for FVAC and 7-20 minutes for TDAC in the 4×5 maze, and in the range 14-25 minutes for FVAC and 20-80 minutes for TDAC in the 6×8 maze.

5 Conclusions

The FVAC (Fleming-Viot Actor-Critic) method, which uses Fleming-Viot particle systems to learn the critic of Actor-Critic policy gradient methods, was presented as a way to solve stochastic optimisation problems under the average reward criterion, in particular when their performance measure is strongly affected by high-reward rarely observed states.

The method generalizes the FVRL algorithm presented in [3, 4] in that a general parameterised policy can be used, and no sparsity assumption is imposed on the reward landscape. This makes the method applicable to very general RL problems, either continuing or episodic, with the restriction that episodic learning tasks must be defined under the average reward criterion, i.e. without discount. The only requirement of the RL environment is that a black-box simulator or emulator can be run on it. That is, at the present moment, FVAC cannot be applied on previously collected data.

Using windy grid worlds as test bench, the FVAC method was seen to clearly outperform, in terms of learning curves, the so-called TDAC benchmark method which uses TD(0) to learn the critic during Monte-Carlo excursions. We observed that, if the absorption set $\mathcal{A}$ is favorably identified to keep exploration away from uninteresting states and close to the high-reward states, the improvement of FVAC over TDAC can be dramatic, as was the case for the 6×8 labyrinth with random obstacles. The

fact that the same number of samples is used in each experiment across the two compared methods, makes FVAC a more sample-efficient approach than TDAC for learning near-optimal policies, while maintaining comparable execution times.

Paths for future work include (i) applying FVAC to other discrete-time RL problems, such as the mountain car problem [6] or card games, (ii) devising strategies to appropriately define the method's hyperparameters, particularly the number of particles N, and (iii) extending the applicability of FVAC to discounted reward learning settings.

Acknowledgements

This work was partially supported by the French National Research Agency under the program "Investments for the Future" with reference code ANR-11-LABX-0040.

References

[1] Pierre Brémaud. Markov chains: Gibbs fields, Monte Carlo simulation, and queues, volume 31. Springer Science & Business Media, 2013.

[2] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. arXiv preprint arXiv:1808.04355, 2018.

[3] Daniel Mastropietro, Urtzi Ayesta, Matthieu Jonckheere, and Szymon Majewski. Efficient reinforcement learning with Fleming-Viot particle systems: application to stochastic networks with rarely observed rewards. working paper or preprint, May 2023.

[4] Daniel Mastropietro, Szymon Majewski, Urtzi Ayesta, and Matthieu Jonckheere. Boosting reinforcement learning with sparse and rare rewards using Fleming-Viot particle systems. In 15th European Workshop on Reinforcement Learning (EWRL 2022), Milano, Italy, September 2022.

[5] Maja J Mataric. Reward functions for accelerated learning. In Machine learning proceedings 1994, pages 181–189. Elsevier, 1994.

[6] Andrew William Moore. Efficient memory-based learning for robot control. Technical Report UCAM-CL-TR-209, University of Cambridge, Computer Laboratory, November 1990.

[7] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 2778–2787. PMLR, 06–11 Aug 2017.

[8] M. L. Puterman. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, 1994.

[9] S.M. Ross. Simulation. Elsevier Science, 2022.

[10] Richard Sutton and Andrew Barto. Reinforcement Learning, an introduction. MIT Press, 2018.

[11] Andrea Tirinzoni, Andrea Sessa, Matteo Pirotta, and Marcello Restelli. Importance weighted transfer of samples in reinforcement learning. In International Conference on Machine Learning, pages 4936–4945. PMLR, 2018.

[12] Yiming Zhang and Keith W Ross. On-policy deep reinforcement learning for the average-reward criterion. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 12535–12545. PMLR, 18–24 Jul 2021.

A Fleming-Viot Actor Critic algorithm (FVAC)

Algorithm 1 Tabular FVAC: Fleming-Viot Actor Critic policy learning under the average reward criterion with tabular critic (required definitions are given in Section 2)

Input: [Problem specific]

p_S: the initial state law of the environment.

r: the reward function defined in Section 2.

Input: [Optimizer]

I: the number of optimisation iterations (policy learning steps). **Default:** 150.

η: the initial learning rate used by the optimizer responsible for learning the policy (e.g. Adam, BFGS, SGD, etc.). **Default:** 0.05 for Adam optimizer.

Input: [Critic learning by FV]

N: the number of FV particles used to learn the expected reward and the value functions at each policy learning step. Guideline: inverse function of the smallest probability among the states expected to yield large rewards. **Default:** 20.

T: the number of steps used to estimate the expected reabsorption time $\mathbb{E}(T_{\mathcal{A}})$ used in the FV estimator of the stationary distribution. Guideline: proportional to $\mathbb{E}(T_{\mathcal{A}})$ which sometimes can be inferred in order of magnitude. **Default:** 500.

$\alpha_{\mathcal{A}}$: the minimum relative visit frequency threshold that should be observed during an initial exploration for a state to be classified as part of the absorption set $\mathcal{A}$. **Default:** 5%.

L (stopping threshold): the simulation stops when either L simulation steps over all particles has been reached or when at least $100 * \beta\%$ (defined below) of the FV particles have been absorbed at least once. Guideline: proportional to N. **Default:** $60N$

β: the proportion of absorbed FV particles at least needed to stop the FV simulation when the number of simulation steps is smaller than L. **Default:** 100%.

ϵ: probability of a random action to ensure ergodic Markov chains under all policies parameterised by model $\mathcal{M}$. We denote by ϵ-π the policy π that is affected by this random action selection at every step. **Default:** 10%.

Input: [Actor learning with policy gradient]

M: the number of steps used at each policy learning step to estimate the gradient in (6). Guideline: proportional to the number of states in the environment. **Default:** $5|\mathcal{S}|$.

$\mathcal{M}$: the structure and characteristics of the model used to learn the parameterised policy, π_θ. **Default:** a 3-layer neural network model with $|\mathcal{S}|$ dummy input neurons representing the states, 12 hidden neurons, and $|\mathcal{U}|$ output neurons giving the policy obtained from the softmax function applied to the output neurons.

Output: Estimate of a parameter vector θ^* for which the parameterised π_{θ^*} policy near-maximizes the long-run expected reward of a reinforcement learning continuning task.

1: Initializations:
- Policy parameter θ_0 s.t. $\pi_{\theta_0}(a|x)$ is (approximately) uniformly distributed $\forall x \in \mathcal{S}$.
- Exit state distribution $\hat{p}^{\epsilon\text{-}\pi_{\theta_{-1}}}_{\underset{\partial\mathcal{A}}{\rightarrow}}(x)$ is uniformly distributed $\forall x \in \mathcal{S}$.
- Average reward estimate: $\overline{R}^{\epsilon\text{-}\pi_{\theta_{-1}}} = 0$.
- State value function estimates: $V^{\epsilon\text{-}\pi_{\theta_{-1}}}(x) = 0, \forall x \in \mathcal{S}$.
- Advantage function estimate: $H^{\epsilon\text{-}\pi_{\theta_{-1}}}(x, a) = 0, \forall x \in \mathcal{S}, \forall a \in \mathcal{U}$

2: $\mathcal{A} \leftarrow$ EstimateAbsorptionSet(T, p_S)

3: **for** $t = 0$ to $I - 1$ **do**

4: $\quad H^{\epsilon\text{-}\pi_{\theta_t}} \leftarrow$ EstimateAdvantageFunctionUsingFV$(\mathcal{A}, t, \epsilon\text{-}\pi_{\theta_t})$

5: $\quad \nabla J(\theta_t) \leftarrow$ ComputePerformanceGradient$(M, H^{\epsilon\text{-}\pi_{\theta_t}}, \pi_{\theta_t})$

6: $\quad$ Update parameter estimate: $\theta_{t+1} \leftarrow \theta_t + \eta_t \nabla J(\theta_t)$

7: **end for**

8: Estimate optimum parameter $\theta^* \leftarrow \theta_I$.

9: **function** ESTIMATEABSORPTIONSET(T, p_S, π, α_A)

10: Simulate T time steps of the Markov chain $\{X_n^\pi\}_{n\in\mathbb{N}}$ initialized at $x_0 \sim p_S$, the environment's start state law.

11: Compute the state visit count, $f(x), \forall x \in \mathcal{S}$.

12: $\mathcal{A} \leftarrow \{x \in \mathcal{S} : f(x)/T > \alpha_A\}$
return $\mathcal{A}$

13: **end function**

14: **function** ESTIMATEADVANTAGEFUNCTIONUSINGFV($\mathcal{A}$, t, θ_t)

15: Initialize, using estimates from the previous policy learning step:
- $\overline{R}_0^{\epsilon\text{-}\pi_{\theta_t}} \leftarrow \overline{R}^{\epsilon\text{-}\pi_{\theta_{t-1}}}$
- $V^{\epsilon\text{-}\pi_{\theta_t}}(x) \leftarrow V^{\epsilon\text{-}\pi_{\theta_{t-1}}}(x), \forall x \in \mathcal{S}$
- $H^{\epsilon\text{-}\pi_{\theta_t}}(x,a) \leftarrow H^{\epsilon\text{-}\pi_{\theta_{t-1}}}(x,a), \forall x \in \mathcal{S}, \forall a \in \mathcal{U}$

16: Estimate, from an initial exploration of the environment:
$\hat{\mathbb{E}}_{\underset{\partial\mathcal{A}}{\leftarrow}}^{\epsilon\text{-}\pi_{\theta_t}}(T_{\mathcal{A}}), \hat{p}_{\underset{\partial\mathcal{A}}{\rightarrow}}^{\epsilon\text{-}\pi_{\theta_t}}, \overline{R}_0^{\epsilon\text{-}\pi_{\theta_t}} \leftarrow$ ESTIMATESFROMINITIALEXPLORATION($T, \epsilon\text{-}\pi_{\theta_t}, \hat{p}_{\underset{\partial\mathcal{A}}{\rightarrow}}^{\epsilon\text{-}\pi_{\theta_t}}, \mathcal{A}, \overline{R}_0^{\epsilon\text{-}\pi_{\theta_t}}$)

17: Initialize the FV N-particle system, using the exit distribution $\hat{p}_{\underset{\partial\mathcal{A}}{\rightarrow}}^{\epsilon\text{-}\pi_{\theta_t}}$ just estimated to select the start state of each particle.

18: Initialize, done $\leftarrow$ False, $n \leftarrow 0$, and:
- $N_{abs} \leftarrow 0$ ($N_{abs} \leq N$ is the number of FV particles absorbed at least once so far)
- $t_n \leftarrow 0$ (t_n is the time of the n-th first absorption time of the FV particles; by simulation construction, it always increases as n increases, which permits the iterative estimation of the FV integral done in function UPDATEREWARDWEIGHTEDFVINTEGRAL())
- $\mathcal{I} \leftarrow 0$ ($\mathcal{I}$ is a vector indexed by the states outside $\mathcal{A}$ where the expected reward under the current policy is non-zero, used to store the reward-weighted FV integral appearing in the numerator of the stationary distribution estimate in (3))

19: **while** not done **do**

20: Choose uniformly at random an FV particle (i) and update its state following the dynamics of the MDP. Record the following transition tuple $(X_n^{(i)}, A_n^{(i)}, X_{n+1}^{(i)})$.

21: Update $H^{\epsilon\text{-}\pi_{\theta_t}}(x,a)$ and $V^{\epsilon\text{-}\pi_{\theta_t}}(x)$ (in this order) by TD(0), i.e.:

$$\begin{aligned} H^{\epsilon\text{-}\pi_{\theta_t}}(X_n^{(i)}, A_n^{(i)}) &\leftarrow R_n^{(i)} - \overline{R}^{\epsilon\text{-}\pi_{\theta_t}} + V^{\epsilon\text{-}\pi_{\theta_t}}(X_{n+1}^{(i)}) - V^{\epsilon\text{-}\pi_{\theta_t}}(X_n^{(i)}) \\ V^{\epsilon\text{-}\pi_{\theta_t}}(X_n^{(i)}) &\leftarrow V^{\epsilon\text{-}\pi_{\theta_t}}(X_n^{(i)}) + \gamma_n H^{\epsilon\text{-}\pi_{\theta_t}}(X_n^{(i)}, A_n^{(i)}), \end{aligned}$$

where $R_n^{(i)} = r(X_n^{(i)}, A_n^{(i)}, X_{n+1}^{(i)})$ and γ_n is an appropriately adjusted learning rate required to guarantee the convergence of stochastic approximation algorithms [10], e.g. $\gamma_n = 1/f_n(X_n^{(i)})$, where $f_n(x)$ is the visit count function of state x at the current time step n, and $f_0(X_0^{(i)}) = 1$.

22: $n \leftarrow n + 1$

23: **if** chosen particle i experiences an absorption **then**

24: **if** this is the first absorption event for particle i **then**

25: $N_{abs} \leftarrow N_{abs} + 1$

26: $\mathcal{I}, t_n \leftarrow$ UPDATEREWARDWEIGHTEDFVINTEGRAL($n, N_{abs}, N, \mathcal{S}, \mathcal{A}, r, \epsilon\text{-}\pi_{\theta_t}, \hat{\phi}, \mathcal{I}, t_n$)

27: Update the average reward with the FV contribution:
$\overline{R}^{\epsilon\text{-}\pi_{\theta_t}} \leftarrow \overline{R}_0^{\epsilon\text{-}\pi_{\theta_t}} + \mathcal{I}/\hat{\mathbb{E}}_{\underset{\partial\mathcal{A}}{\leftarrow}}^{\epsilon\text{-}\pi_{\theta_t}}(T_{\mathcal{A}})\ \ N_{abs}/(N + N_{abs})$

28: **end if**

29: Reactivate the particle to the position of a randomly chosen particle among the other $N-1$ particles.

30: **end if**

31: $\hat{\phi} \leftarrow$ UPDATEPHI($n, N, \mathcal{S}, \mathcal{A}, r, \epsilon\text{-}\pi, \hat{\phi}$)

32: done $\leftarrow n = L$ or $N_{abs} > \beta N$.

33: **end while**

34: **if** $N_{abs} < N$ **then**

35: Consider all non-absorbed particles as censored observations at $t_n = n/N$:
$\mathcal{I}, t_n \leftarrow$ UPDATEREWARDWEIGHTEDFVINTEGRAL($n, N_{abs}+1, N, \mathcal{S}, \mathcal{A}, r, \epsilon\text{-}\pi_{\theta_t}, \hat{\phi}, t_n, \mathcal{I}$)

36: **end if**
return $H^{\epsilon\text{-}\pi_{\theta_t}}$

37: **end function**

function ESTIMATESFROMINITIALEXPLORATION($T, \pi, \hat{p}_{\overrightarrow{\partial\mathcal{A}}}, \mathcal{A}, \overline{R}_0^\pi$)

Simulate T time steps of the Markov chain $\{X_n^\pi\}_{n\in\mathbb{N}}$ initialized at $x_0 \sim \hat{p}_{\overrightarrow{\partial\mathcal{A}}}$, an exit state distribution, and compute:

- the estimated expected reabsorption time to $\mathcal{A}$ as $\hat{\mathbb{E}}^\pi_{\overleftarrow{\partial\mathcal{A}}}(T_\mathcal{A}) = \frac{\tilde{T}}{C}$, where $\tilde{T}$ is the number of steps over all observed full return cycles to $\mathcal{A}$ and C is the number of full return cycles;
- the exit state distribution $\hat{p}^\pi_{\overrightarrow{\partial\mathcal{A}}}$ as the relative frequency of each observed exit state over all exit events;
- the updated average reward, $\overline{R}_0^\pi \leftarrow \overline{R}_0^\pi + (\overline{R}_T^\pi - \overline{R}_0^\pi)/2$, where $\overline{R}_T^\pi$ is the average reward observed during the current T-step excursion, and the factor $1/2$ reflects the fact that the average reward estimate before this update, $\overline{R}_0^\pi$, is also based on T observations.

return $\hat{\mathbb{E}}^\pi_{\overleftarrow{\partial\mathcal{A}}}(T_\mathcal{A}), \hat{p}^\pi_{\overrightarrow{\partial\mathcal{A}}}, \overline{R}_0^\pi$

end function

function COMPUTEPERFORMANCEGRADIENT(M, H, π_{θ_t})

Simulate M time steps of the Markov chain $\{X_n^{\pi_{\theta_t}}\}_{n\in\mathbb{N}}$ initialized at $x_0 \sim p_S$, the environment's start state law, and compute the performance gradient $\nabla J(\theta_t)$ using (6):

$$\nabla J(\theta_t) = \frac{1}{M}\sum_{n=1}^{M} H(X_n, A_n)\nabla \log \pi_{\theta_t}(A_n|X_n),$$

return $\nabla J(\theta_t)$

end function

function UPDATEREWARDWEIGHTEDFVINTEGRAL($n, N_{abs}, N, \mathcal{S}, \mathcal{A}, r, \pi, \hat{\phi}, \mathcal{I}, t_{n-1}$)

$t_n \leftarrow n/N$

Let $\mathcal{A}^c = \mathcal{S} \setminus \mathcal{A}$

Let $\mathcal{Y} = \{x \in \mathcal{S} : \overline{r}(x,\pi) \neq 0\} \bigcap \mathcal{A}^c$ the set of states outside $\mathcal{A}$ with non-zero reward under policy π.

for all $y \in \mathcal{Y}$ **do**

Compute the contribution from the piecewise constant $\hat{\phi}(.,y)$ function in the latest inter-absorption interval $(t_{n-1}, t_n]$:

$\Phi_n(y) = \sum_{i:t_{n-1}<s_i\leq t_n}(s_i - s_{i-1})\hat{\phi}(s_{i-1}, y)$,

where the s_i's are the times at which $\hat{\phi}(.,y)$ changes.

Update the FV integral at y, weighted by the expected reward observed at y, $\overline{r}(y,\pi)$:

$$\mathcal{I}(y) \leftarrow \mathcal{I}(y) + \frac{1}{N}(1 - \frac{N_{abs}-1}{N})\Phi_n(y)\overline{r}(y,\pi),$$

where $1/N$ is the discrete step size multiplying the sum in (3), and $1-(N_{abs}-1)/N$ is the contribution from the estimated survival probability function, $\hat{\mathbb{P}}(T_\mathcal{K} \geq t)$, which is constant in $(t_{n-1}, t_n]$.

end for

return $\mathcal{I}, t_n$

end function

function UPDATEPHI($n, N, \mathcal{S}, \mathcal{A}, r, \pi, \hat{\phi}$)

Let $\mathcal{A}^c = \mathcal{S} \setminus \mathcal{A}$

Let $\mathcal{Y} = \{x \in \mathcal{S} : \overline{r}(x,\pi) \neq 0\} \bigcap \mathcal{A}^c$ the set of states outside $\mathcal{A}$ with non-zero reward under policy π.

for all $y \in \mathcal{Y}$ **do**

$\hat{\phi}(n/N, y) \leftarrow \sum_{i=1}^{N} \mathbf{1}\{X_n^{(i)} = y\}/N$ (proportion of N particles at y).

end for

end function