Renewal Monte Carlo: Renewal theory based reinforcement learning

Jayakumar Subramanian and Aditya Mahajan

***Abstract*— In this paper, we present an online reinforcement learning algorithm, called Renewal Monte Carlo (RMC), for infinite horizon Markov decision processes with a designated start state. RMC is a Monte Carlo algorithm and retains the advantages of Monte Carlo methods including low bias, simplicity and ease of implementation while, at the same time, circumvents their key drawbacks of high variance and delayed (end of episode) updates. The key ideas behind RMC are as follows. First, under any reasonable policy, the reward process is ergodic. So, by renewal theory, the performance of a policy is equal to the ratio of expected discounted reward to the expected discounted time over a regenerative cycle. Second, by carefully examining the expression for performance gradient, we propose a stochastic approximation algorithm that only requires estimates of the expected discounted reward and discounted time over a regenerative cycle and their gradients. We propose two unbiased estimators for evaluating performance gradients—a likelihood ratio based estimator and a simultaneous perturbation based estimator—and show that for both estimators, RMC converges to a locally optimal policy. We also generalize the RMC algorithm to post-decision state models. We conclude by presenting numerical experiments on a randomly generated MDP and event driven communication.**

I. Introduction

In recent years, reinforcement learning [1]–[4] has emerged as a leading framework to learn how to act optimally in unknown environments. Policy gradient methods [5]–[10] have played a prominent role in the success of reinforcement learning. Such methods have two critical components: policy evaluation and policy improvement. In the policy evaluation step, the performance of a parameterized policy is evaluated while in the policy improvement step, the policy parameters are updated using stochastic gradient ascent.

Policy gradient methods may be broadly classified as Monte Carlo methods and temporal difference methods. In Monte Carlo methods, performance of a policy is estimated using the discounted return of a single sample path; in temporal difference methods, the value(-action) function is guessed and this guess is iteratively improved using temporal differences. Monte Carlo methods are attractive because they have zero bias, are simple and easy to implement, and work for both discounted and average reward setups as well as for models with continuous state and action spaces. However, they suffer from various drawbacks. First, they have high variance because a single sample path is used to estimate performance. Second, they are not asymptotically optimal for infinite horizon models because it is effectively assumed that the model is episodic; in infinite horizon models, the trajectory is arbitrarily truncated to treat the model as an episodic model. Third, the policy improvement step cannot be carried out in tandem with policy evaluation. One must wait until the end of the episode to estimate the performance and only then can the policy parameters be updated. It is for these reasons that Monte Carlo methods are largely ignored in the literature on policy gradient methods, which almost exclusively focuses on temporal difference methods such as actor-critic with eligibility traces [3].

In this paper, we propose a Monte Carlo method—which we call *Renewal Monte Carlo* (RMC)—for infinite horizon Markov decision processes with designated start state. Like Monte Carlo, RMC has low bias, is simple and easy to implement, and works for models with continuous state and action spaces. At the same time, it does not suffer from the drawbacks of typical Monte Carlo methods. RMC is a low-variance online algorithm that works for infinite horizon discounted and average reward setups. One doesn't have to wait until the end of the episode to carry out the policy improvement step; it can be carried out whenever the system visits a designated reference state.

Although renewal theory is commonly used to estimate performance of stochastic systems in the simulation optimization community [11], [12], those methods assume that the probability law of the primitive random variables and its weak derivate are known, which is not the case in reinforcement learning. Renewal theory is also commonly used in the engineering literature on queuing theory and systems and control for Markov decision processes (MDPs) with average reward criteria and a known system model. There is some prior work on using renewal theory for reinforcement learning [13], [14], where renewal theory based estimators for the average return and differential value function for average reward MDPs is developed. In RMC, renewal theory is used in a different manner for discounted reward MDPs (and the results generalize to average cost MDPs).

II. RMC Algorithm

Consider a Markov decision process (MDP) with state $S_t \in \mathcal{S}$ and action $A_t \in \mathcal{A}$. The system starts in an initial state $s_0 \in \mathcal{S}$ and at time t:

1) there is a controlled transition from S_t to S_{t+1} according to a transition kernel $P(A_t)$;
2) a per-step reward $R_t = r(S_t, A_t, S_{t+1})$ is received.

This work was supported by the Natural Sciences and Engineering Research Council of Canada through NSERC Discovery Accelerator RGPAS 493011-16.

The authors are with the Electrical and Computer Engineering Department, McGill University, Montreal, QC H3A 0E9, Canada. Emails: jayakumar.subramanian@mail.mcgill.ca, aditya.mahajan@mcgill.ca

Future is discounted at a rate $\gamma \in (0,1)$.

A (time-homogeneous and Markov) policy π maps the current state to a distribution on actions, i.e., $A_t \sim \pi(S_t)$. We use $\pi(a|s)$ to denote $\mathbb{P}(A_t = a|S_t = s)$. The performance of a policy π is given by

$$J_\pi = \mathbb{E}_{A_t\sim\pi(S_t)}\left[\sum_{t=0}^{\infty}\gamma^t R_t \;\middle|\; S_0 = s_0\right]. \tag{1}$$

A policy that maximizes the performance is called an optimal policy. In the sequel, we present a sample path based online learning algorithm, which we call Renewal Monte Carlo (RMC), which identifies a locally optimal policy within the class of parameterized policies.

Suppose policies are parameterized by a closed and convex subset Θ of the Euclidean space. For example, Θ could be the weight vector in a Gibbs soft-max policy, or the weights of a deep neural network, or the thresholds in a control limit policy, and so on. Given $\theta \in \Theta$, we use π_θ to denote the policy parameterized by θ and J_θ to denote J_{π_θ}. We assume that for all policies π_θ, $\theta \in \Theta$, the designated start state s_0 is positive recurrent.

The typical approach for policy gradient based reinforcement learning is to start with an initial guess $\theta_0 \in \Theta$ and iteratively update it using stochastic gradient ascent. In particular, let $\widehat{\nabla} J_{\theta_m}$ be an unbiased estimator of $\nabla_\theta J_\theta\big|_{\theta=\theta_m}$, then update

$$\theta_{m+1} = \left[\theta_m + \alpha_m \widehat{\nabla} J_{\theta_m}\right]_\Theta \tag{2}$$

where $[\theta]_\Theta$ denotes the projection of θ onto Θ and $\{\alpha_m\}_{m\geq 1}$ is a sequence of learning rates that satisfies the standard assumptions of

$$\sum_{m=1}^{\infty}\alpha_m = \infty \quad \text{and} \quad \sum_{m=1}^{\infty}\alpha_m^2 < \infty. \tag{3}$$

Under mild technical conditions [15], the above iteration converges to a θ^* that is locally optimal, i.e., $\nabla_\theta J_\theta\big|_{\theta=\theta^*} = 0$. In RMC, we approximate $\nabla_\theta J_\theta$ by a Renewal theory based estimator as explained below.

Let $\tau^{(n)}$ denote the stopping time when the system returns to the start state s_0 for the n-th time. In particular, let $\tau^{(0)} = 0$ and for $n \geq 1$ define

$$\tau^{(n)} = \inf\{t > \tau^{(n-1)} : s_t = s_0\}.$$

We call the sequence of (S_t, A_t, R_t) from $\tau^{(n-1)}$ to $\tau^{(n)} - 1$ as the *n-th regenerative cycle*. Let $\mathsf{R}^{(n)}$ and $\mathsf{T}^{(n)}$ denote the total discounted reward and total discounted time of the n-th regenerative cycle, i.e.,

$$\mathsf{R}^{(n)} = \Gamma^{(n)}\sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1}\gamma^t R_t \quad \text{and} \quad \mathsf{T}^{(n)} = \Gamma^{(n)}\sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1}\gamma^t, \tag{4}$$

where $\Gamma^{(n)} = \gamma^{-\tau^{(n-1)}}$. By the strong Markov property, $\{\mathsf{R}^{(n)}\}_{n\geq 1}$ and $\{\mathsf{T}^{(n)}\}_{n\geq 1}$ are i.i.d. sequences. Let R_θ and T_θ denote $\mathbb{E}[\mathsf{R}^{(n)}]$ and $\mathbb{E}[\mathsf{T}^{(n)}]$, respectively. Define

$$\widehat{\mathsf{R}} = \frac{1}{N}\sum_{n=1}^{N}\mathsf{R}^{(n)} \quad \text{and} \quad \widehat{\mathsf{T}} = \frac{1}{N}\sum_{n=1}^{N}\mathsf{T}^{(n)}, \tag{5}$$

where N is arbitrarily chosen number of cycles. Then, $\widehat{\mathsf{R}}$ and $\widehat{\mathsf{T}}$ are unbiased and asymptotically consistent estimators of R_θ and T_θ.

From ideas similar to standard Renewal theory [16], we have the following.

Proposition 1 (Renewal Relationship) *The performance of policy π_θ is given by:*

$$J_\theta = \frac{\mathsf{R}_\theta}{(1-\gamma)\mathsf{T}_\theta}. \tag{6}$$

PROOF For ease of notation, define

$$\overline{\mathsf{T}}_\theta = \mathbb{E}_{A_t\sim\pi_\theta(S_t)}\left[\gamma^{\tau^{(n)}-\tau^{(n-1)}}\right]$$

Using the formula for geometric series, we get that $\mathsf{T}_\theta = (1-\overline{\mathsf{T}}_\theta)/(1-\gamma)$. Hence,

$$\overline{\mathsf{T}}_\theta = 1 - (1-\gamma)\mathsf{T}_\theta. \tag{7}$$

Now, consider the performance:

$$\begin{aligned} J_\theta &= \mathbb{E}_{A_t\sim\pi_\theta(S_t)}\Bigg[\sum_{t=0}^{\tau^{(1)}-1}\gamma^t R_t \\ &\qquad + \gamma^{\tau^{(1)}}\sum_{t=\tau^{(1)}}^{\infty}\gamma^{t-\tau^{(1)}}R_t \;\Bigg|\; S_0 = s_0\Bigg] \\ &\overset{(a)}{=} \mathsf{R}_\theta + \mathbb{E}_{A_t\sim\pi_\theta(S_t)}[\gamma^{\tau^{(1)}}]\, J_\theta \\ &= \mathsf{R}_\theta + \overline{\mathsf{T}}_\theta J_\theta, \end{aligned} \tag{8}$$

where the second expression in (a) uses the independence of random variables from $(0, \tau^{(1)}-1)$ to those from $\tau^{(1)}$ onwards due to the strong Markov property. Substituting (7) in (8) and rearranging terms, we get the result of the proposition.■

Differentiating both sides of (6) with respect to θ, we get that

$$\nabla_\theta J_\theta = \frac{H_\theta}{\mathsf{T}_\theta^2(1-\gamma)}, \;\text{ where } H_\theta = \mathsf{T}_\theta\nabla_\theta\mathsf{R}_\theta - \mathsf{R}_\theta\nabla_\theta\mathsf{T}_\theta. \tag{9}$$

Therefore, instead of using stochastic gradient ascent to find the maximum of J_θ, we can use stochastic approximation to find the root of H_θ. In particular, let $\widehat{H}_m$ be an unbiased estimator of H_{θ_m}. We then use the update

$$\theta_{m+1} = \left[\theta_m + \alpha_m \widehat{H}_m\right]_\Theta \tag{10}$$

where $\{\alpha_m\}_{m\geq 1}$ satisfies the standard conditions on learning rates (3). The above iteration converges to a locally optimal policy. Specifically, we have the following.

Theorem 1 *Let $\widehat{\mathsf{R}}_m$, $\widehat{\mathsf{T}}_m$, $\widehat{\nabla}\mathsf{R}_m$ and $\widehat{\nabla}\mathsf{T}_m$ be unbiased estimators of R_{θ_m}, T_{θ_m}, $\nabla_\theta\mathsf{R}_{\theta_m}$, and $\nabla_\theta\mathsf{R}_{\theta_m}$, respectively such that $\widehat{\mathsf{T}}_m \perp \widehat{\nabla}\mathsf{R}_m$ and $\widehat{\mathsf{R}}_m \perp \widehat{\nabla}\mathsf{T}_m$.*[1] *Then,*

$$\widehat{H}_m = \widehat{\mathsf{T}}_m\widehat{\nabla}\mathsf{R}_m - \widehat{\mathsf{R}}_m\widehat{\nabla}\mathsf{T}_m \tag{11}$$

is an unbiased estimator of H_θ. Furthermore, assume that

[1]The notation $X \perp Y$ means that the random variables X and Y are independent.

1) H_θ *is continuous,*
2) *the estimate* $\widehat{H}_m$ *has bounded variance,*
3) *The ODE* $d\theta/dt = H_\theta$ *has isolated limit points that are locally asymptotically stable.*

Then, the sequence $\{\theta_m\}_{m\geq 1}$ *generated by* (10) *converges almost surely and*

$$\lim_{m\to\infty} \nabla_\theta J_\theta\big|_{\theta_m} = 0.$$

PROOF The unbiasedness of $\widehat{H}_m$ follows immediately from the independence assumption. The convergence of the $\{\theta_m\}_{m\geq 1}$ follows from [17, Theorem 2.1, page 127], the fact that the model satisfies conditions (A2.1)–(A2.6) of [17, pg 126] and the convergence to local optima is as per the discussion in [17, Sec 5.8, page 157]. ■

In the remainder of this section, we present two methods for estimating the gradients of R_θ and T_θ. The first is a likelihood ratio based gradient estimator which works when the policy is differentiable with respect to the policy parameters. The second is a simultaneous perturbation based gradient estimator that uses finite differences, which is useful when the policy is not differentiable with respect to the policy parameters.

A. Likelihood ratio based gradient based estimator

One approach to estimate the performance gradient is to use likelihood ratio based estimates [12], [18], [19]. Suppose the policy $\pi_\theta(a|s)$ is differentiable with respect to θ. For any time t, define the likelihood function

$$L_t = \nabla_\theta \log[\pi_\theta(A_t \mid S_t)], \tag{12}$$

and for $\sigma \in \{\tau^{(n-1)}, \ldots, \tau^{(n)} - 1\}$, define

$$\mathsf{R}_\sigma^{(n)} = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^t R_t, \;\; \mathsf{T}_\sigma^{(n)} = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^t. \tag{13}$$

In this notation $\mathsf{R}^{(n)} = \mathsf{R}^{(n)}_{\tau^{(n-1)}}$ and $\mathsf{T}^{(n)} = \mathsf{T}^{(n)}_{\tau^{(n-1)}}$. Then, define the following estimators for $\nabla_\theta \mathsf{R}_\theta$ and $\nabla_\theta \mathsf{T}_\theta$:

$$\widehat{\nabla}\mathsf{R} = \frac{1}{N} \sum_{n=1}^{N} \sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} \mathsf{R}_\sigma^{(n)} L_\sigma^{(n)}, \tag{14}$$

$$\widehat{\nabla}\mathsf{T} = \frac{1}{N} \sum_{n=1}^{N} \sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} \mathsf{T}_\sigma^{(n)} L_\sigma^{(n)}, \tag{15}$$

where N is arbitrary.

Proposition 2 $\widehat{\nabla}\mathsf{R}$ *and* $\widehat{\nabla}\mathsf{T}$ *defined above are unbiased and asymptotically consistent estimators of* $\nabla_\theta \mathsf{R}_\theta$ *and* $\nabla_\theta \mathsf{T}_\theta$.

PROOF Let P_θ denote the probability induced on the sample paths when the system is following policy π_θ. For $t \in \{\tau^{(n-1)}, \ldots, \tau^{(n)} - 1\}$, let $D_t^{(n)}$ denote the sample path $(S_s, A_s, S_{s+1})_{s=\tau^{(n-1)}}^{t}$ for the n-th regenerative cycle until time t. Then,

$$P_\theta(D_t^{(n)}) = \prod_{s=\tau^{(n-1)}}^{t} \pi_\theta(A_s|S_s) \mathbb{P}(S_{s+1}|S_s, A_s)$$

Algorithm 1: RMC Algorithm with likelihood ratio based gradient estimates.

input : Intial policy θ_0, discount factor γ, initial state s_0, number of regenerative cycles N

for *iteration* $m = 0, 1, \ldots$ **do**

  for *regenerative cycle* $n_1 = 1$ *to* N **do**

    Generate n_1-th regenerative cycle using π_{θ_m}.

    Compute $\mathsf{R}^{(n_1)}$ and $\mathsf{T}^{(n_1)}$ using (4).

  Set $\widehat{\mathsf{R}}_m = \texttt{average}(\mathsf{R}^{(n_1)} : n_1 \in \{1, \ldots, N\})$.

  Set $\widehat{\mathsf{T}}_m = \texttt{average}(\mathsf{T}^{(n_1)} : n_1 \in \{1, \ldots, N\})$.

  for *regenerative cycle* $n_2 = 1$ *to* N **do**

    Generate n_2-th regenerative cycle using π_{θ_m}.

    Compute $\mathsf{R}_\sigma^{(n_2)}$, $\mathsf{T}_\sigma^{(n_2)}$ and $L_\sigma^{(n_2)}$ for all σ.

  Compute $\widehat{\nabla}\mathsf{R}_m$ and $\widehat{\nabla}\mathsf{T}_m$ using (14) and (15).

  Set $\widehat{H}_m = \widehat{\mathsf{T}}_m \widehat{\nabla}\mathsf{R}_m - \widehat{\mathsf{R}}_m \widehat{\nabla}\mathsf{T}_m$.

  Update $\theta_{m+1} = \big[\theta_m + \alpha_m \widehat{H}_m\big]_\Theta$.

Therefore,

$$\nabla_\theta \log P_\theta(D_t^{(n)}) = \sum_{s=\tau^{(n-1)}}^{t} \nabla_\theta \log \pi_\theta(A_s|S_s) = \sum_{s=\tau^{(n-1)}}^{t} L_s. \tag{16}$$

Note that R_θ can be written as:

$$\mathsf{R}_\theta = \Gamma^{(n)} \sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1} \gamma^t \mathbb{E}_{A_t\sim\pi_\theta(S_t)}[R_t].$$

Using the log derivative trick,[2] we get

$$\begin{aligned}
\nabla_\theta \mathsf{R}_\theta &= \Gamma^{(n)} \sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1} \gamma^t \, \mathbb{E}_{A_t\sim\pi_\theta(S_t)}[R_t \nabla_\theta \log P_\theta(D_t^{(n)})] \\
&\overset{(a)}{=} \Gamma^{(n)} \mathbb{E}_{A_t\sim\pi_\theta(S_t)}\left[\sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1} \left[\gamma^t R_t \sum_{\sigma=\tau^{(n-1)}}^{t} L_\sigma\right]\right] \\
&\overset{(b)}{=} \mathbb{E}_{A_t\sim\pi_\theta(S_t)}\left[\sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} L_\sigma \left[\Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^t R_t\right]\right] \\
&\overset{(c)}{=} \mathbb{E}_{A_t\sim\pi_\theta(S_t)}\left[\sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} \mathsf{R}_\sigma^{(n)} L_\sigma\right]
\end{aligned} \tag{17}$$

where (a) follows from (16), (b) follows from changing the order of summations, and (c) follows from the definition of $\mathsf{R}_\sigma^{(n)}$ in (13). $\widehat{\nabla}\mathsf{R}$ is an unbiased and asymptotically consistent estimator of the right hand side of the first equation in (17). The result for $\widehat{\nabla}\mathsf{T}$ follows from a similar argument. ■

To satisfy the independence condition of Theorem 1, we use two independent sample paths: one to estimate $\widehat{\mathsf{R}}$ and $\widehat{\mathsf{T}}$ and the other to estimate $\widehat{\nabla}\mathsf{R}$ and $\widehat{\nabla}\mathsf{T}$. The complete algorithm

[2]Log-derivative trick: For any distribution $p(x|\theta)$ and any function f,

$$\nabla_\theta \mathbb{E}_{X\sim p(X|\theta)}[f(X)] = \mathbb{E}_{X\sim p(X|\theta)}[f(X) \nabla_\theta \log p(X|\theta)].$$

in shown in Algorithm 1. An immediate consequence of Theorem 1 is the following.

Corollary 1 *Under the conditions of Theorem 1, the sequence $\{\theta_m\}_{m\geq 1}$ generated by Algorithm 1 converges to a locally optimal solution.* □

Remark 1 Algorithm 1 is presented in its simplest form. It is possible to use standard variance reduction techniques such as subtracting a baseline [19]–[21] to reduce variance. □

Remark 2 In Algorithm 1, we use two separate runs to compute $(\widehat{\mathsf{R}}_m, \widehat{\mathsf{T}}_m)$ and $(\nabla\widehat{\mathsf{R}}_m, \nabla\widehat{\mathsf{T}}_m)$ to ensure that the independence conditions of Proposition 2 are satisfied. In practice, we found that using a single run to compute both $(\widehat{\mathsf{R}}_m, \widehat{\mathsf{T}}_m)$ and $(\nabla\widehat{\mathsf{R}}_m, \nabla\widehat{\mathsf{T}}_m)$ has negligible effect on the accuracy of convergence (but speeds up convergence by a factor of two). □

Remark 3 It has been reported in the literature [22] that using a biased estimate of the gradient where $R^{(n)}_\sigma$ is given by:

$$\mathsf{R}^{(n)}_\sigma = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^{t-\sigma} R_t, \tag{18}$$

(and a similar expression for $T^{(n)}_\sigma$) leads to faster convergence. We call this variant *RMC with biased gradients* and, in our experiments, found that it does converge faster than RMC.□

B. Simultaneous perturbation based gradient estimator

Another approach to estimate performance gradient is to use simultaneous perturbation based estimates [23]–[26]. The general one-sided form of such estimates is

$$\widehat{\nabla}\mathsf{R}_\theta = \delta(\widehat{\mathsf{R}}_{\theta+c\delta} - \widehat{\mathsf{R}}_\theta)/c$$

where δ is a random variable with the same dimension as θ and c is a small constant. The expression for $\widehat{\nabla}\mathsf{T}_\theta$ is similar. When $\delta_i \sim$ Rademacher(± 1), the above method corresponds to simultaneous perturbation stochastic approximation (SPSA) [23], [24]; when $\delta \sim$ Normal$(0, I)$, the above method corresponds to smoothed function stochastic approximation (SFSA) [25], [26].

Substituting the above estimates in (11) and simplifying, we get

$$\widehat{H}_\theta = \delta(\widehat{\mathsf{T}}_\theta\widehat{\mathsf{R}}_{\theta+c\delta} - \widehat{\mathsf{R}}_\theta\widehat{\mathsf{T}}_{\theta+c\delta})/c.$$

The complete algorithm is shown in Algorithm 2. Since $(\widehat{\mathsf{R}}_\theta, \widehat{\mathsf{T}}_\theta)$ and $(\widehat{\mathsf{R}}_{\theta+c\delta}, \widehat{\mathsf{T}}_{\theta+c\delta})$ are estimated from separate sample paths, $\widehat{H}_\theta$ defined above is an unbiased estimator of H_θ. Then, an immediate consequence of Theorem 1 is the following.

Corollary 2 *The sequence $\{\theta_m\}_{m\geq 1}$ generated by Algorithm 2 converges to a locally optimal solution.* □

Algorithm 2: RMC Algorithm with simultaneous perturbation based gradient estimates.

input : Intial policy θ_0, discount factor γ, initial state s_0, number of regenerative cycles N, constant c, perturbation distribution Δ

for *iteration* $m = 0, 1, \ldots$ **do**

  for *regenerative cycle* $n_1 = 1$ *to* N **do**

    Generate n_1-th regenerative cycle using π_{θ_m}.

    Compute $\mathsf{R}^{(n_1)}$ and $\mathsf{T}^{(n_1)}$ using (4).

  Set $\widehat{\mathsf{R}}_m = \texttt{average}(\mathsf{R}^{(n_1)} : n_1 \in \{1, \ldots, N\})$.

  Set $\widehat{\mathsf{T}}_m = \texttt{average}(\mathsf{T}^{(n_1)} : n_1 \in \{1, \ldots, N\})$.

  Sample $\delta \sim \Delta$.

  Set $\theta'_m = \theta_m + c\delta$.

  for *regenerative cycle* $n_2 = 1$ *to* N **do**

    Generate n_2-th regenerative cycle using π_{θ_m}.

    Compute $\mathsf{R}^{(n_2)}$ and $\mathsf{T}^{(n_2)}$ using (4).

  Set $\widehat{\mathsf{R}}'_m = \texttt{average}(\mathsf{R}^{(n_2)} : n_2 \in \{1, \ldots, N\})$.

  Set $\widehat{\mathsf{T}}'_m = \texttt{average}(\mathsf{T}^{(n_2)} : n_2 \in \{1, \ldots, N\})$.

  Set $\widehat{H}_m = \delta(\widehat{\mathsf{T}}_m\widehat{\mathsf{R}}'_m - \widehat{\mathsf{R}}_m\widehat{\mathsf{T}}'_m)/c$.

  Update $\theta_{m+1} = \left[\theta_m + \alpha_m\widehat{H}_m\right]_\Theta$.

III. RMC for Post-Decision State Model

In many models, the state dynamics can be split into two parts: a controlled evolution followed by an uncontrolled evolution. For example, many continuous state models have dynamics of the form

$$S_{t+1} = f(S_t, A_t) + N_t,$$

where $\{N_t\}_{t\geq 0}$ is an independent noise process. For another example, see the event driven communication model in Sec IV. Such models can be written in terms of a post-decision state model described below. Note that the results of this section apply to continuous state models as long as the model satisfies the standard conditions under which the Bellman equation has a solution [27].

Consider a post-decision state MDP with pre-decision state $S^-_t \in \mathcal{S}^-$, post-decision state $S^+_t \in \mathcal{S}^+$, action $A_t \in \mathcal{A}$. The system starts at an initial state $s^+_0 \in \mathcal{S}^+$ and at time t:

1) there is a controlled transition from S^-_t to S^+_t according to a transition kernel $P^-(A_t)$;
2) there is an uncontrolled transition from S^+_t to S^-_{t+1} according to a transition kernel P^+;
3) a per-step reward $R_t = r(S^-_t, A_t, S^+_t)$ is received.

Future is discounted at a rate $\gamma \in (0, 1)$.

Remark 4 When $\mathcal{S}^+ = \mathcal{S}^-$ and P^+ is identity, then the above model reduces to the standard MDP model, considered in Sec II. When P^+ is a deterministic transition, the model reduces to a standard MDP model with post decision states [28], [29]. □

As in Sec II, we choose a (time-homogeneous and Markov) policy π that maps the current pre-decision state $\mathcal{S}^-$ to a distribution on actions, i.e., $A_t \sim \pi(S^-_t)$. We use $\pi(a|s^-)$ to denote $\mathbb{P}(A_t = a|S^-_t = s^-)$.

The performance when the system starts in post-decision

state $s_0^+ \in \mathcal{S}^+$ and follows policy π is given by

$$J_\pi = \mathbb{E}_{A_t \sim \pi(S_t)} \left[\sum_{t=0}^{\infty} \gamma^t R_t \;\middle|\; S_0^+ = s_0^+ \right]. \tag{19}$$

Let $\tau^{(n)}$ denote the stopping times such that $\tau^{(0)} = 0$ and for $n \geq 1$,

$$\tau^{(n)} = \inf\{t > \tau^{(n-1)} : s_{t-1}^+ = s_0^+\}.$$

The slightly unusual definition (using $s_{t-1}^+ = s_0^+$ rather than the more natural $s_t^+ = s_0^+$) is to ensure that the formulas for $\mathsf{R}^{(n)}$ and $\mathsf{T}^{(n)}$ used in Sec. II remain valid for the post-decision state model as well. Thus, both variants of RMC presented in Sec. II converge to a locally optimal parameter θ for the post-decision state model as well.

IV. Numerical Experiments

We conduct two experiments to evaluate the performance of RMC: a randomly generated MDP and event driven communication.

A. Randomized MDP (GARNET)

In this experiment, we study a randomly generated GARNET$(100, 10, 50)$ model [30], which is an MDP with 100 states, 10 actions, and a branching factor of 50 (which means that each row of all transition matrices has 50 non-zero elements, chosen Unif$[0, 1]$ and normalized to add to 1). For each state-action pair, with probability $p = 0.05$, the reward is chosen Unif$[10, 100]$, and with probability $1 - p$, the reward is 0. Future is discounted by a factor of $\gamma = 0.9$. The first state is chosen as start state. The policy is a Gibbs soft-max distribution parameterized by 100×10 (states $\times$ actions) parameters, where each parameter belongs to the interval $[-30, 30]$. The temperature of the Gibbs distribution is kept constant and equal to 1.

We compare the performance of RMC, RMC with biased gradient (denoted by RMC-B, see Remark 2), and actor critic with eligibility traces for the critic [3] (which we refer to as SARSA-λ and abbreviate as S-λ in the plots), with $\lambda \in \{0, 0.25, 0.5, 0.75, 1\}$. For both the RMC algorithms, we use the same runs to estimate the gradients (see Remark 2 in Sec. II). Each algorithm[3] is run 100 times and the mean and standard deviation of the performance (as estimated by the algorithms themselves) is shown in Fig. 1a. The performance of the corresponding policy evaluated by Monte-Carlo evaluation over a horizon of 250 steps and averaged over 100 runs is shown in Fig. 1b. The optimal performance computed using value iteration is also shown.

The results show that SARSA-λ learns faster (this is expected because the critic is keeping track of the entire value function) but has higher variance and gets stuck in local minima. On the other hand, RMC and RMC-B learn slower but have a low bias and do not get stuck in local minima. The same qualitative behavior was observed for other randomly generated models although we are not sure why RMC and SARSA differ in which local minima they converge to. Also, it was observed that RMC-B (RMC with biased evaluation of the gradient) learns faster than RMC.

B. Event Driven Communication

In this experiment, we study an event driven communication problem that arises in networked control systems [32], [33]. A transmitter observes a first-order autoregressive process $\{X_t\}_{t\geq 1}$, i.e., $X_{t+1} = \alpha X_t + W_t$, where $\alpha, X_t, W_t \in \mathbb{R}$, and $\{W_t\}_{t\geq 1}$ is an i.i.d. process. At each time, the transmitter uses an event-triggered policy (explained below) to determine whether to transmit or not (denoted by $A_t = 1$ and $A_t = 0$, respectively). Transmission takes place over an i.i.d. erasure channel with erasure probability p_d. Let S_t^- and S_t^+ denote the "error" between the source realization and it's reconstruction at a receiver. It can be shown that S_t^- and S_t^+ evolve as follows [32], [33]: when $A_t = 0$, $S_t^+ = S_t^-$; when $A_t = 1$, $S_t^+ = 0$ if the transmission is successful (w.p. $(1 - p_d)$) and $S_t^+ = S_t^-$ if the transmission is not successful (w.p. p_d); and $S_{t+1}^- = \alpha S_t^+ + W_t$. Note that the post-decision state resets to zero after every successful transmission.[4]

The per-step cost has two components: a communication cost of λA_t, where $\lambda \in \mathbb{R}_{>0}$ and an estimation error $(S_t^+)^2$. The objective is to minimize the expected discounted cost.

An event-triggered policy is a threshold policy that chooses $A_t = 1$ whenever $|S_t^-| \geq \theta$, where θ is a design choice. Under certain conditions, such an event-triggered policy is known to be optimal [32], [33]. When the system model is known, algorithms to compute the optimal θ are presented in [34], [35]. In this section, we use RMC to identify the optimal policy when the model parameters are not known.

In our experiment we consider an event-triggered model with $\alpha = 1$, $\lambda = 500$, $p_d \in \{0, 0.1, 0.2\}$, $W_t \sim \mathcal{N}(0, 1)$, $\gamma = 0.9$, and use simultaneous perturbation variant of RMC[5] to identify θ. We run the algorithm 100 times and the result for different choices of p_d are shown in Fig. 1c.[6] For $p_d = 0$, the optimal threshold computed using [35] is also shown. The results show that RMC converges relatively quickly and has low bias across multiple runs.

V. Conclusions

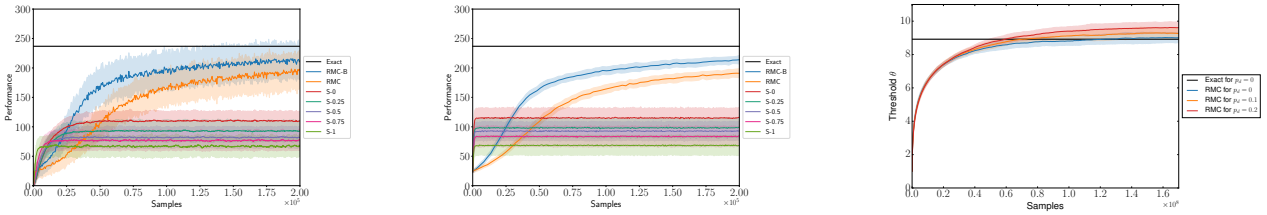
We present a renewal theory based reinforcement learning algorithm called Renewal Monte Carlo. RMC retains the key advantages of Monte Carlo methods and has low bias, is simple and easy to implement, and works for models with continuous state and action spaces. In addition, due to the averaging over multiple renewals, RMC has low variance. We also generalized RMC to post-decision state models.

[3]For all algorithms, the learning rate is chosen using ADAM [31] with default hyper-parameters and the α parameter of ADAM equal to 0.05 for RMC, RMC-B, and the actor in SARSA-λ and the learning rate is equal to 0.1 for the critic in SARSA-λ. For RMC and RMC-B, the policy parameters are updated after $N = 5$ renewals.

[4]Had we used the standard MDP model instead of the model of Sec. II, this restart would not have always resulted in a renewal.

[5]An event-triggered policy is a parametric policy but $\pi_\theta(a|s^-)$ is not differentiable in θ. Therefore, the likelihood ratio method cannot be used to estimate performance gradient.

[6]We choose the learning rate using ADAM with default hyper-parameters and the α parameter of ADAM equal to 0.01. We choose $c = 0.3$, $N = 100$ and $\Delta = \mathcal{N}(0, 1)$ in Algorithm 2.



(a) GARNET
Online performance estimates.

(b) GARNET
Mean of 100 Monte Carlo evaluations.

(c) Event-driven communication
Policy parameters averaged over 100 runs.

Fig. 1: Performance of different learning algorithms on GARNET$(100, 10, 50)$ with $p = 0.05$ and $\gamma = 0.9$ for a rollout horizon of 250 and event-driven communication using RMC for different values of p_d. The solid lines show the mean value and the shaded region shows the $\pm$ one standard deviation region.

Although we restricted attention to discounted reward model, all the results immediately extend to the average reward model as well. To simplify the discussion, we assumed that the reference state is the same as the start state. Even if that is not the case, the arguments presented in this paper go through with slight modification.

Finally, we only presented the simplest form of the RMC algorithm. It is possible to obtain an "every step" variant of RMC that can be used to estimate the entire value function (or its approximation).

REFERENCES

[1] D. Bertsekas and J. Tsitsiklis, *Neuro-dynamic Programming*, ser. Anthropological Field Studies. Athena Scientific, 1996.

[2] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[3] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 1998.

[4] C. Szepesvári, *Algorithms for reinforcement learning*. Morgan & Claypool Publishers, 2010.

[5] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, Nov. 2000, pp. 1057–1063.

[6] S. M. Kakade, "A natural policy gradient," in *Advances in Neural Information Processing Systems*, Dec. 2002, pp. 1531–1538.

[7] V. R. Konda and J. N. Tsitsiklis, "On actor-critic algorithms," *SIAM Journal on Control and Optimization*, vol. 42, no. 4, pp. 1143–1166, 2003.

[8] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Intl. Conference on Machine Learning*, June 2015, pp. 1889–1897.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[10] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. others Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[11] P. Glynn, "Optimization of stochastic systems," in *Proc. Winter Simulation Conference*, Dec. 1986, pp. 52–59.

[12] ——, "Likelihood ratio gradient estimation for stochastic systems," *Communications of the ACM*, vol. 33, pp. 75–84, 1990.

[13] P. Marbach and J. N. Tsitsiklis, "Simulation-based optimization of Markov reward processes," *IEEE Trans. Autom. Control*, vol. 46, no. 2, pp. 191–209, Feb 2001.

[14] ——, "Approximate gradient methods in policy-space optimization of Markov reward processes,," *Discrete Event Dynamical Systems*, vol. 13, no. 2, pp. 111–148, 2003.

[15] V. Borkar, *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008.

[16] W. Feller, *An Introduction to Probability Theory and its Applications*. John Wiley and Sons, 1966, vol. 1.

[17] H. Kushner and G. G. Yin, *Stochastic approximation and recursive algorithms and applications*. Springer, 2003.

[18] R. Y. Rubinstein, "Sensitivity analysis and performance extrapolation for computer simulation models," *Operations Research*, vol. 37, no. 1, pp. 72–81, 1989.

[19] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[20] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," *Journal of Machine Learning Research*, vol. 5, no. Nov, pp. 1471–1530, 2004.

[21] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *Intl. Conf. on Intelligent Robots and Systems, 2006 IEEE/RSJ*. IEEE, Oct. 2006, pp. 2219–2225.

[22] P. Thomas, "Bias in natural actor-critic algorithms," in *Intl. Conference on Machine Learning*, June 2014, pp. 441–448.

[23] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Trans. Autom. Control*, vol. 37, no. 3, pp. 332–341, 1992.

[24] J. L. Maryak and D. C. Chin, "Global random optimization by simultaneous perturbation stochastic approximation," *IEEE Trans. Autom. Control*, vol. 53, no. 3, pp. 780–783, Apr. 2008.

[25] V. Katkovnik and Y. Kulchitsky, "Convergence of a class of random search algorithms." *Automation and Remote Control*, vol. 33, no. 8, pp. 1321–1326, 1972.

[26] S. Bhatnagar, H. Prasad, and L. Prashanth, *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*. Springer, 2013.

[27] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov Control Processes: Basic Optimality Criteria*. Springer, 1996.

[28] B. Van Roy, D. P. Bertsekas, Y. Lee, and J. N. Tsitsiklis, "A neuro-dynamic programming approach to retailer inventory management," in *Proc. Conference on Decision and Control*, vol. 4, Dec. 1997, pp. 4052–4057.

[29] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. John Wiley & Sons, 2011.

[30] S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee, "Natural actor-critic algorithms," Department of Computing Science, University of Alberta, Canada, Tech. Rep., 2009.

[31] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[32] G. M. Lipsa and N. Martins, "Remote state estimation with communication costs for first-order LTI systems," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2013–2025, Sep. 2011.

[33] J. Chakravorty, J. Subramanian, and A. Mahajan, "Stochastic approximation based methods for computing the optimal thresholds in remote-state estimation with packet drops," in *Proc. American Control Conference*, Seattle, WA, May 2017, pp. 462–467.

[34] Y. Xu and J. P. Hespanha, "Optimal communication logics in networked control systems," in *Proc. Conference on Decision and Control*, Dec. 2004, pp. 3527–3532.

[35] J. Chakravorty and A. Mahajan, "Fundamental limits of remote estimation of Markov processes under communication constraints," *IEEE Trans. Autom. Control*, vol. 62, no. 3, pp. 1109–1124, Mar. 2017.