Renewal Monte Carlo: Renewal theory based reinforcement learning

Jayakumar Subramanian and Aditya Mahajan

***Abstract*—An online reinforcement learning algorithm called Renewal Monte Carlo (RMC) is presented. RMC works for infinite horizon Markov decision processes with a designated start state. RMC is a Monte Carlo algorithm that retains the key advantages of Monte Carlo—viz., simplicity, ease of implementation, and low bias—while circumventing the main drawbacks of Monte Carlo—viz., high variance and delayed updates. Given a parameterized policy π_θ, the algorithm consists of three parts: estimating the expected discounted reward R_θ and the expected discounted time T_θ over a regenerative cycle; estimating the derivatives $\nabla_\theta \mathsf{R}_\theta$ and $\nabla_\theta \mathsf{T}_\theta$, and updating the policy parameters using stochastic approximation to find the roots of $\mathsf{R}_\theta \nabla_\theta \mathsf{T}_\theta - \mathsf{T}_\theta \nabla_\theta \mathsf{R}_\theta$. It is shown that under mild technical conditions, RMC converges to a locally optimal policy. It is also shown that RMC works for post-decision state models as well. An approximate version of RMC is proposed where a regenerative cycle is defined as successive visits to a pre-specified "renewal set". It is shown that if the value function of the system is locally Lipschitz on the renewal set, then RMC converges to an approximate locally optimal policy. Three numerical experiments are presented to illustrate RMC and compare it with other state-of-the-art reinforcement learning algorithms.**

***Index Terms*—Reinforcement learning, Markov decision processes, renewal theory, Monte Carlo methods, policy gradient, stochastic approximation**

I. Introduction

In recent years, reinforcement learning [1]–[4] has emerged as an effective framework for learning 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 policy evaluation, the performance of a parameterized policy is evaluated while in policy improvement, 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 one or more sample paths; in temporal difference methods, an initial estimate for the (action-) value function is chosen arbitrarily and then improved iteratively 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 a high variance because a single sample path is used to estimate performance. Second, in Monte Carlo methods it is implicitly assumed that the model is episodic (i.e., there is an end state and the system stops when it reaches the end state). To use these methods for infinite horizon models, the trajectory is arbitrarily truncated to treat the model as an episodic model. For that reason, the resultant policy is not asymptotically optimal. 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. For these reasons the literature on policy gradient methods largely ignores Monte Carlo methods and almost exclusively focuses on temporal difference methods such as actor-critic with eligibility traces [3].

In this paper an online reinforcement learning algorithm called Renewal Monte Carlo (RMC) is presented. RMC works for infinite horizon Markov decision processes with a designated start state. RMC is a Monte Carlo algorithm that retains the key advantages of Monte Carlo—viz., simplicity, ease of implementation, and low bias—while circumventing the main drawbacks of Monte Carlo—viz., high variance and delayed updates. The key intuition behind RMC is that, under any reasonable policy, the reward process is ergodic. Therefore, using ideas from renewal theory, it can be shown that the performance of any parameterized policy π_θ is proportional to R_θ / T_θ, where R_θ and T_θ are the expected discounted reward and the expected discounted time of the reward process over a regenerative cycle. Hence, the performance gradient is proportional to $H_\theta = \nabla R_\theta T_\theta - R_\theta \nabla T_\theta$. Hence, any policy for which H_θ is zero is locally optimal.

In RMC, R_θ and T_θ are estimated from Monte Carlo evaluations over multiple regenerative cycles; ∇R_θ and ∇T_θ are estimated using either likelihood ratio or simultaneous perturbation based estimators; and the root of H_θ is obtained using stochastic approximation. We show that under mild technical conditions, RMC converges to a locally optimal policy.

The RMC algorithm is generalized to post-decision state models, where regenerative cycle is defined as successive visits to an initial post-decision state.

An approximate RMC algorithm is proposed where successive visits to a pre-specified "renewal set" is viewed as a regenerative cycle. We show that if the value function for the system is locally Lipschitz continuous on the renewal set, then RMC converges to approximate locally optimal policy.

The effectiveness of RMC is illustrated on three examples:

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

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

Preliminary version of this paper was presented at the 57th IEEE Conference on Decision and Control.

randomly generated Markov decision processes, event-driven communication, and inventory control. The last two examples have continuous state space and show that RMC works well for continuous state models as well.

Although renewal theory is commonly used to estimate performance of stochastic systems [11], [12], those methods assume that the probability law of the primitive random variables and its weak derivative are known, which is not the case in reinforcement learning. Renewal theory is also commonly used in queuing theory and 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 are 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 each time t there is a controlled transition from S_t to S_{t+1} according to a transition kernel $P(A_t)$. At each time t, a per-step reward $R_t = r(S_t, A_t, S_{t+1})$ is received.

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

$$J_\pi = \mathbb{E}_{A_t \sim \pi(S_t)}\biggl[\sum_{t=0}^{\infty} \gamma^t R_t \biggm| S_0 = s_0\biggr], \tag{1}$$

where $\gamma \in (0,1)$ is the discount factor. We are interested in identifying an optimal policy, i.e., a policy that maximizes the performance. When $\mathcal{S}$ and $\mathcal{A}$ are Borel spaces, we assume that the model satisfies the standard regularity conditions under which time-homogeneous Markov policies are optimal [15].

Suppose policies are parameterized by a closed and convex subset Θ of the Euclidean space.[1] 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 choice $\theta_0 \in \Theta$ 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

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

where $[\theta]_\Theta$ denotes the projection of θ onto Θ and $\{\alpha_m\}_{m \ge 1}$ are learning rates that satisfy the standard assumptions:

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

Under mild technical conditions [16], 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 $n \ge 1$ define $\tau^{(n)} = \min\{t > \tau^{(n-1)} : s_t = s_0\}$. 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 [17], $\{\mathsf{R}^{(n)}\}_{n \ge 1}$ and $\{\mathsf{T}^{(n)}\}_{n \ge 1}$ 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 an 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 of renewal theory [18], 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 Consider the performance:

$$J_\theta = \mathbb{E}_{A_t \sim \pi_\theta(S_t)}\biggl[\sum_{t=0}^{\tau^{(1)}-1} \gamma^t R_t + \gamma^{\tau^{(1)}} \sum_{t=\tau^{(1)}}^{\infty} \gamma^{t-\tau^{(1)}} R_t \biggm| S_0 = s_0\biggr]$$

$$\overset{(a)}{=} \mathsf{R}_\theta + \mathbb{E}_{A_t \sim \pi_\theta(S_t)}[\gamma^{\tau^{(1)}}]\, J_\theta \tag{7}$$

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 [17].

Now, by definition, $\mathsf{T}_\theta = (1 - \mathbb{E}_{A_t \sim \pi_\theta(S_t)}[\gamma^{\tau^{(1)}}])/(1-\gamma)$. Rearranging terms, we get $\mathbb{E}_{A_t \sim \pi_\theta(S_t)}[\gamma^{\tau^{(1)}}] = 1 - (1-\gamma)\mathsf{T}_\theta$. Substituting this in (7), we get the result of the proposition.■

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

$$\nabla_\theta J_\theta = \frac{H_\theta}{\mathsf{T}_\theta^2(1-\gamma)}, \quad \text{where } H_\theta = \mathsf{T}_\theta \nabla_\theta \mathsf{R}_\theta - \mathsf{R}_\theta \nabla_\theta \mathsf{T}_\theta. \tag{8}$$

Therefore, instead of using stochastic gradient ascent to find a local maximum of J_θ, we can use stochastic approximation to find a root of H_θ.

Theorem 1 *Consider the sequence $\{\theta_m\}_{m \ge 1}$ where the initial $\theta_0 \in \Theta$ is chosen arbitrarily, and for $m > 0$,*

$$\theta_{m+1} = \bigl[\theta_m + \alpha_m \widehat{H}_m\bigr]_\Theta, \tag{9}$$

where $\{\alpha_m\}_{m \ge 1}$ satisfies (3) *and $\widehat{H}_m$ is an unbiased estimator of H_{θ_m}. Then, the sequence $\{\theta_m\}_{m \ge 1}$ converges almost surely and*

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

[1] Examples of such parametized policies include the weights of a Gibbs soft-max policy, the weights of a deep neural network, or the thresholds in a control limit policy, and so on.

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

```
input    : Intial policy $\theta_0$, discount factor $\gamma$, initial
           state $s_0$, number of regenerative cycles $N$
for iteration $m = 0, 1, \dots$ do
    for regenerative cycle $n_1 = 1$ to $N$ do
        Generate $n_1$-th regenerative cycle using policy $\pi_{\theta_m}$.
        Compute $\mathsf{R}^{(n_1)}$ and $\mathsf{T}^{(n_1)}$ using (4).
    Set $\widehat{\mathsf{R}}_m = \texttt{mean}(\mathsf{R}^{(n_1)} : n_1 \in \{1, \dots, N\})$.
    Set $\widehat{\mathsf{T}}_m = \texttt{mean}(\mathsf{T}^{(n_1)} : n_1 \in \{1, \dots, N\})$.
    for regenerative cycle $n_2 = N + 1$ to $2N$ do
        Generate $n_2$-th regenerative cycle using policy $\pi_{\theta_m}$.
        Compute $\mathsf{R}_\sigma^{(n_2)}$, $\mathsf{T}_\sigma^{(n_2)}$ and $\Lambda_\sigma$ for all $\sigma$ using (12).
        Set $\widehat{\mathsf{R}}^{(n_2)} = \sum_{\sigma=\tau^{n_2-1}}^{\tau^{n_2}-1} \mathsf{R}_\sigma^{(n_2)} \Lambda_\sigma$.
        Set $\widehat{\mathsf{T}}^{(n_2)} = \sum_{\sigma=\tau^{n_2-1}}^{\tau^{n_2}-1} \mathsf{T}_\sigma^{(n_2)} \Lambda_\sigma$.
    Set $\widehat{\nabla}\mathsf{R}_m = \texttt{mean}(\widehat{\mathsf{R}}^{(n_2)} : n_2 \in \{N+1, \dots, 2N\})$
    Set $\widehat{\nabla}\mathsf{T}_m = \texttt{mean}(\widehat{\mathsf{T}}^{(n_2)} : n_2 \in \{N+1, \dots, 2N\})$
    Set $\widehat{H}_m = \widehat{\mathsf{T}}_m \widehat{\nabla}\mathsf{R}_m - \widehat{\mathsf{R}}_m \widehat{\nabla}\mathsf{T}_m$.
    Update $\theta_{m+1} = \big[\theta_m + \alpha_m \widehat{H}_m\big]_\Theta$.
```

PROOF The convergence of the $\{\theta_m\}_{m \geq 1}$ follows from [16, Theorem 2.2] and the fact that the model satisfies conditions (A1)–(A4) of [16, pg 10–11]. ■

Proposition 2 *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 $\nabla_\theta \mathsf{T}_{\theta_m}$, respectively such that $\widehat{\mathsf{T}}_m \perp \widehat{\nabla}\mathsf{R}_m$ and $\widehat{\mathsf{R}}_m \perp \widehat{\nabla}\mathsf{T}_m$.*[2] *Then,*

$$\widehat{H}_m = \widehat{\mathsf{T}}_m \widehat{\nabla}\mathsf{R}_m - \widehat{\mathsf{R}}_m \widehat{\nabla}\mathsf{T}_m \tag{10}$$

is an unbiased estimator of H_{θ_m}. Furthermore, assume that

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

Then, the sequence $\{\theta_m\}_{m \geq 1}$ generated by (9) *converges almost surely and*

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

PROOF The independence assumption implies that $\widehat{H}_m$ is unbiased. The model satisfies conditions (A2.1)–(A2.6) of [19, pg. 126], so [19, Thm 2.1] implies that $\{\theta_m\}_{m \geq 1}$ converges. The convergence to a local maximum follows from the discussion in [19, Sec. 5.8]. ■

We can estimate R_θ and T_θ using (5). We present two methods to estimate the gradients of R_θ and T_θ: (i) a likelihood ratio based gradient estimator which works when the policy is differentiable with respect to the policy parameters; and (ii) 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 estimator

One approach to estimate the performance gradient is to use likelihood radio based estimates [12], [20], [21]. Suppose the policy $\pi_\theta(a|s)$ is differentiable with respect to θ. For any time t, define the likelihood function

$$\Lambda_t = \nabla_\theta \log[\pi_\theta(A_t \mid S_t)], \tag{11}$$

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

$$\mathsf{R}_\sigma^{(n)} = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^t R_t \quad \text{and} \quad \mathsf{T}_\sigma^{(n)} = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^t. \tag{12}$$

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 $\nabla_\theta \mathsf{T}_\theta$:

$$\widehat{\nabla}\mathsf{R} = \frac{1}{N} \sum_{n=1}^{N} \sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} \mathsf{R}_\sigma^{(n)} \Lambda_\sigma, \tag{13}$$

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

where N is an arbitrarily chosen number.

Proposition 3 *$\widehat{\nabla}\mathsf{R}$ 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 $t \in \{\tau^{(n-1)}, \dots, \tau^{(n)} - 1\}$, 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,

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

Therefore,

$$\nabla_\theta \log P_\theta(D_t^{(n)}) = \sum_{s=\tau^{(n-1)}}^{t} \nabla_\theta \log \pi_\theta(A_s|S_s) = \sum_{s=\tau^{(n-1)}}^{t} \Lambda_s. \tag{15}$$

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,[3] we get

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

[2] $X \perp Y$ denotes that random variables X and Y are independent.

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

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

$$\overset{(c)}{=} \mathbb{E}_{A_t \sim \pi_\theta(S_t)} \Bigg[\sum_{\sigma=\tau^{(n-1)}}^{\tau^{(n)}-1} \mathsf{R}_\sigma^{(n)} \Lambda_\sigma \Bigg] \tag{16}$$

where (a) follows from (15), (b) follows from changing the order of summations, and (c) follows from the definition of $\mathsf{R}_\sigma^{(n)}$ in (12). $\widehat{\nabla}\mathsf{R}$ is an unbiased and asymptotically consistent estimator of the right hand side of the last equation in (16). The result for $\widehat{\nabla}\mathsf{T}$ follows from a similar argument. ■

Algorithm 1 combines the above estimates with the stochastic gradient ascent iteration of Theorem 1. An immediate consequence of Proposition 2 and Theorem 1 is the following.

Corollary 1 *The sequence $\{\theta_m\}_{m \ge 1}$ generated by Algorithm 1 converges to a local maximum.* □

Remark 1 Algorithm 1 is presented in its simplest form. It is possible to use standard variance reduction techniques such as subtracting a baseline [21]–[23] 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 condition of Proposition 2 is 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 [24] that using a biased estimate of the gradient given by:

$$\mathsf{R}_\sigma^{(n)} = \Gamma^{(n)} \sum_{t=\sigma}^{\tau^{(n)}-1} \gamma^{t-\sigma} R_t, \tag{17}$$

(and a similar expression for $T_\sigma^{(n)}$) 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 [25]–[28]. 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 $\delta_i \sim \text{Rademacher}(\pm 1)$, the above method corresponds to simultaneous perturbation stochastic approximation (SPSA) [25], [26]; when $\delta \sim \text{Normal}(0, I)$, it corresponds to smoothed function stochastic approximation (SFSA) [27], [28].

Substituting these estimates in (10) 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 in 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 Proposition 2 and Theorem 1 is the following.

Corollary 2 *The sequence $\{\theta_m\}_{m \ge 1}$ generated by Algorithm 2 converges to a local maximum.* □

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

```
input  : Intial policy $\theta_0$, discount factor $\gamma$, initial
         state $s_0$, number of regenerative cycles $N$,
         constant $c$, perturbation distribution $\Delta$
for iteration $m = 0, 1, \dots$ do
    for regenerative cycle $n_1 = 1$ to $N$ do
        Generate $n_1$-th regenerative cycle using policy $\pi_{\theta_m}$.
        Compute $\mathsf{R}^{(n_1)}$ and $\mathsf{T}^{(n_1)}$ using (4).
    Set $\widehat{\mathsf{R}}_m = \mathtt{mean}(\mathsf{R}^{(n_1)} : n_1 \in \{1, \dots, N\})$.
    Set $\widehat{\mathsf{T}}_m = \mathtt{mean}(\mathsf{T}^{(n_1)} : n_1 \in \{1, \dots, N\})$.
    Sample $\delta \sim \Delta$.
    Set $\theta'_m = \theta_m + c\delta$.
    for regenerative cycle $n_2 = N + 1$ to $2N$ do
        Generate $n_2$-th regenerative cycle using policy $\pi_{\theta_m}$.
        Compute $\mathsf{R}^{(n_2)}$ and $\mathsf{T}^{(n_2)}$ using (4).
    Set $\widehat{\mathsf{R}}'_m = \mathtt{mean}(\mathsf{R}^{(n_2)} : n_2 \in \{N + 1, \dots, 2N\})$.
    Set $\widehat{\mathsf{T}}'_m = \mathtt{mean}(\mathsf{T}^{(n_2)} : n_2 \in \{N + 1, \dots, 2N\})$.
    Set $\widehat{H}_m = \delta(\widehat{\mathsf{T}}_m \widehat{\mathsf{R}}'_m - \widehat{\mathsf{R}}_m \widehat{\mathsf{T}}'_m)/c$.
    Update $\theta_{m+1} = \big[\theta_m + \alpha_m \widehat{H}_m\big]_\Theta$.
```

C. Remark on average reward setup

The results presented above also apply to average reward models where the objective is to maximize

$$J_\pi = \lim_{t_h \to \infty} \frac{1}{t_h} \mathbb{E}_{A_t \sim \pi(S_t)} \Bigg[\sum_{t=0}^{t_h-1} R_t \;\Bigg|\; S_0 = s_0 \Bigg]. \tag{18}$$

Let the stopping times $\tau^{(n)}$ be defined as before. Define the total reward $\mathsf{R}^{(n)}$ and duration $\mathsf{T}^{(n)}$ of the n-th regenerative cycle as

$$\mathsf{R}^{(n)} = \sum_{t=\tau^{(n-1)}}^{\tau^{(n)}-1} R_t \quad \text{and} \quad \mathsf{T}^{(n)} = \tau^{(n)} - \tau^{(n-1)}.$$

Let R_θ and T_θ denote the expected values of $\mathsf{R}^{(n)}$ and $\mathsf{T}^{(n)}$ under policy π_θ. Then from standard renewal theory we have that the performance J_θ is equal to $\mathsf{R}_\theta / \mathsf{T}_\theta$ and, therefore $\nabla_\theta J_\theta = H_\theta / T_\theta^2$, where H_θ is defined as in (8). We can use both variants of RMC presented above to obtain estimates of H_θ and use these to update the policy parameters using (9).

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 $\{N_t\}_{t \ge 0}$ is an independent noise process. For other examples, see the inventory control and event-triggered communication models in Sec. V. Such models can be written in terms of a post-decision state model described below.

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.

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 [29], [30]. □

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

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

where $\gamma \in (0,1)$ is the discount factor. As before, we are interested in identifying an optimal policy, i.e., a policy that maximizes the performance. When $\mathcal{S}$ and $\mathcal{A}$ are Borel spaces, we assume that the model satisfies the standard conditions under which time-homogeneous Markov policies are optimal [15]. Let $\tau^{(n)}$ denote the stopping times such that $\tau^{(0)} = 0$ and for $n \ge 1$,

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

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, using arguments similar to Sec. II, we can show that both variants of RMC presented in Sec. II converge to a locally optimal parameter θ for the post-decision state model as well.

IV. Approximate RMC

In this section, we present a variant of RMC that trades off accuracy with the speed of convergence. One potential limitation of RMC is that the system may take a long time to revisit the initial state. We can circumvent this limitation by considering a "renewal set" B around the start state and pretending that a renewal takes place whenever the state enters B. Doing so, results in a loss in accuracy. Since each regenerative cycles does not start in the same state, the renewal relationship of Proposition 1 is no longer valid. Nonetheless, in this section, we show that if the model has sufficient regularity so that the value function is locally Lipschitz in the renewal set, the error due to this approximation is bounded.

Suppose that the state and action spaces $\mathcal{S}$ and $\mathcal{A}$ are separable metric spaces (with metrics d_S and d_A). Given a "renewal set" B containing the start state s_0 and let $\rho^B = \sup_{s \in B} d_S(s, s_0)$ denote the radius of B with respect to s_0. Given a policy π, let $\tau^{(n)}$ denote the stopping times for successive visits to B, i.e., $\tau^{(0)} = 0$ and for $n \ge 1$,

$$\tau^{(n)} = \min\{t > \tau^{(n-1)} : s_t \in B\}.$$

Define $\mathsf{R}^{(n)}$ and $\mathsf{T}^{(n)}$ as in (4) and let R_θ^B and T_θ^B denote the expected values of $\mathsf{R}^{(n)}$ and $\mathsf{T}^{(n)}$, respectively. Define

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

Theorem 2 *Given a policy π_θ, let V_θ denote the value function and $\overline{\mathsf{T}}_\theta^B = \mathbb{E}_{A_t \sim \pi_\theta(S_t)}[\gamma^{\tau^{(1)}} | S_0 = s_0]$ (which is always less than γ). Suppose the following condition is satisfied:*

(C) *The value function V_θ is locally Lipschitz in B, i.e., there exists a L_θ such that for any $s, s' \in B$,*

$$|V_\theta(s) - V_\theta(s')| \le L_\theta d_S(s, s').$$

Then

$$\big|J_\theta - J_\theta^B\big| \le \frac{L_\theta \overline{\mathsf{T}}_\theta^B}{(1-\gamma)\mathsf{T}_\theta^B} \rho^B \le \frac{\gamma}{(1-\gamma)} L_\theta \rho^B. \tag{20}$$

Proof We follow an argument similar to Proposition 1.

$$\begin{aligned} J_\theta = V_\theta(s_0) = \mathbb{E}_{A_t \sim \pi_\theta(S_t)} \Bigg[& \sum_{t=0}^{\tau^{(1)}-1} \gamma^t R_t \\ & + \gamma^{\tau^{(1)}} \sum_{t=\tau^{(1)}}^{\infty} \gamma^{t-\tau^{(1)}} R_t \,\Bigg|\, S_0 = s_{\tau^{(1)}} \Bigg] \\ \overset{(a)}{=} \mathsf{R}_\theta^B + \mathbb{E}_{A_t \sim \pi_\theta(S_t)}&[\gamma^{\tau^{(1)}} | S_0 = s_0]\, V_\theta(s_{\tau^{(1)}}) \end{aligned} \tag{21}$$

where (a) uses the strong Markov property [17]. Since V_θ is locally Lipschitz with constant L_θ and $s_{\tau^{(1)}} \in B$, we have that

$$|J_\theta - V_\theta(s_{\tau^{(1)}})| = |V_\theta(s_0) - V_\theta(s_{\tau^{(1)}})| \le L_\theta \rho^B.$$

Substituting the above in (21) gives

$$J_\theta \le \mathsf{R}_\theta^B + \overline{\mathsf{T}}_\theta^B (J_\theta + L_\theta \rho^B).$$

Substituting $\mathsf{T}_\theta^B = (1 - \overline{\mathsf{T}}_\theta^B)/(1-\gamma)$ and rearranging the terms, we get

$$J_\theta \le J_\theta^B + \frac{L_\theta \overline{\mathsf{T}}_\theta^B}{(1-\gamma)\mathsf{T}_\theta^B} \rho^B.$$

The proof for the other direction is similar. The second inequality in (20) follows from $\overline{\mathsf{T}}_\theta^B \le \gamma$ and $\mathsf{T}_\theta^B \ge 1$. ■

Based on Theorem 2, a policy that minimizes J_θ^B is approximately optimal. Such a policy can be identified by modifying both variants of RMC to declare a renewal whenever the state lies in B.

Local Lipschitz continuity of value functions can be verified for specific models (e.g., the model presented in Sec. V-C). Sufficient conditions for *global* Lipschitz continuity have been identified in [31, Theorem 4.1], [32, Lemma 1, Theorem 1], and [33, Lemma 1]). We state these conditions below.

Proposition 4 *Let V_θ denote the value function for any policy π_θ. Suppose the model satisfies the following conditions:*

1) *The transition kernel P is Lipschitz, i.e., there exists a constant L_P such that for all $s, s' \in \mathcal{S}$ and $a, a' \in \mathcal{A}$,*

$$\mathcal{K}(P(\cdot|s,a), P(\cdot|s',a')) \le L_P \big[d_S(s,s') + d_A(a,a') \big],$$

where $\mathcal{K}$ is the Kantorovich metric (also called the Wasserstein distance) between probability measures.

2) *The per-step reward r is Lipschitz, i.e., there exists a constant L_r such that for all $s, s', s_+ \in \mathcal{S}$ and $a, a' \in \mathcal{A}$,*

$$|r(s,a,s_+) - r(s',a',s_+)| \le L_r \big[d_S(s,s') + d_A(a,a')\big].$$

In addition, suppose the policy satisfies the following:

3) *The policy π_θ is Lipschitz, i.e., there exists a constant L_{π_θ} such that for any $s, s' \in \mathcal{S}$,*

$$\mathcal{K}(\pi_\theta(\cdot|s), \pi_\theta(\cdot|s')) \le L_{\pi_\theta}\, d_S(s,s').$$

4) $\gamma L_P(1 + L_{\pi_\theta}) < 1.$
5) *The value function V_θ exists and is finite.*

Then, V_θ is globally Lipschitz. In particular, for any $s, s' \in \mathcal{S}$,

$$|V_\theta(s) - V_\theta(s')| \le L_\theta d_S(s,s'),$$

where $L_\theta = L_r(1 + L_{\pi_\theta})/\big(1 - \gamma L_P(1 + L_{\pi_\theta})\big)$.

V. Numerical Experiments

We present three experiments to evaluate the performance of RMC: a randomly generated MDP, event-triggered communication, and inventory management. The code for all the experiments is available at [34].

A. Randomized MDP (GARNET)

In this experiment, we study a randomly generated GARNET$(100, 10, 50)$ model [35], 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. The discount factor $\gamma = 0.9$. The first state is chosen as start state. The policy is parameterized by a Gibbs soft-max distribution (which has states × actions = 100×10 parameters) where each parameter belongs to the interval $[-10, 10]$ and the temperature is kept constant and equal to 1.

We compare the performance of the following algorithms:

1) RMC with likelihood ratio based gradient estimator (see Sec. II-A) where the gradient is estimated using a single run (see Remark 2 in Sec. II). The policy parameters are updated after $N = 4$ renewals and the learning is adapted using ADAM(0.05)[4] [36].
2) RMC with biased gradient denoted by RMC-B (see Remark 2) where all parameters are same as in RMC.
3) Actor critic with eligibility traces for the critic [3], which we refer to as AC-λ with $\lambda \in \{0, 0.5, 1\}$, where the learning rate for the actor is adapted using ADAM(0.1) [36].
4) TPRO [8] and PPO [9], which are two state of the art policy gradient based RL algorithms for models with discrete action spaces, where we use the default architecture and parameters from ChainerRL [37].

We run each algorithm for 2×10^5 samples and repeat this experiment 100 times. To compare the performance of these algorithms, we periodically evaluate the performance of π_{θ_m} for each trajectory using Monte Carlo evaluation (over 200 samples averaged over 10 independent runs). The median, first quartile, and third quartile across 100 runs are shown in Fig. 1a. The optimal performance (which is computed using value iteration and the knowledge of the model) is also shown.

We observe that AC-λ, TRPO, and PPO learn faster (which is expected because the critic is keeping track of the entire value function) but have higher variance. AC-λ gets stuck in a local minimum while RMC, RMC-B, TRPO, and PPO do not. Policy gradient algorithms only guarantee convergence to a local optimum. We are not sure why AC-λ converges to a different local maximum from RMC, RMC-B, TRPO and PPO. We also observe that RMC-B (which is RMC with biased evaluation of the gradient) learns faster than RMC.

It is worth highlighting that although TRPO/PPO converge in fewer number of samples compared to RMC/RMC-B, they require significantly more computational resources. In our experiments, each run of TRPO took ≈ 10 minutes (wall clock time), PPO took ≈ 16 minutes, AC-λ took ≈ 1 minute, whereas RMC/RMC-B took ≈ 40 seconds.

B. Event-Triggered Communication

In this experiment, we study an event-triggered communication problem that arises in networked control systems [38], [39]. A transmitter observes a first-order autoregressive process $\{X_t\}_{t\ge1}$, i.e., $X_{t+1} = \alpha X_t + W_t$, where $\alpha, X_t, W_t \in \mathbb{R}$, and $\{W_t\}_{t\ge1}$ 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 its reconstruction at a receiver. It can be shown that S_t^- and S_t^+ evolve as follows [38], [39]: 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 this is a post-decision state model, where the post-decision state resets to zero after every successful transmission.[5]

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.

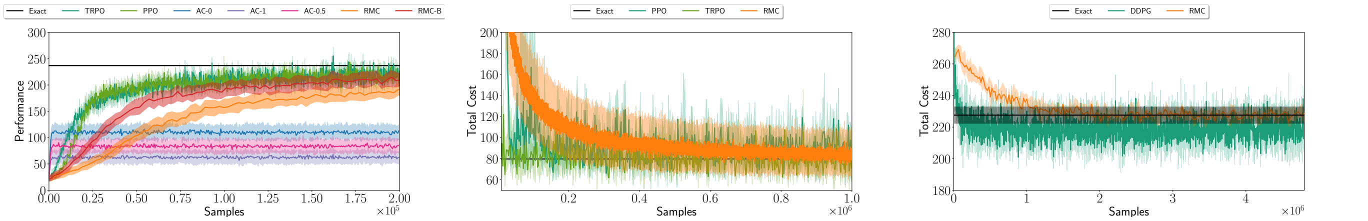
An event-triggered policy is a threshold policy that chooses $A_t = 1$ whenever $|S_t^-| \ge \theta$, where θ is a design choice. Under certain conditions, such an event-triggered policy is known to be optimal [38], [39]. When the system model is known, algorithms to compute the optimal θ are presented in [40], [41]. 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 $\alpha = 1$, $\lambda = 500$, $p_d = 0.0$, $W_t \sim \mathcal{N}(0,1)$, $\gamma = 0.9$.

We compare the performance for the following algorithms:

[4] We use ADAM(α) to denote the choice of the α parameter of ADAM. All other parameters have their default value.

[5] Had we used the standard MDP model instead of the post-decision state model, this restart would not have always resulted in a renewal.



(a) GARNET (b) Event-Triggered communication (c) Inventory control

Fig. 1: Comparison of RMC with other state of the art algorithms for the three benchmark environments. The solid lines show the median values and the shaded area shows the region between the first and third quartiles.

1) RMC with simultaneous perturbation based gradient estimate (see Sec. II-B)[6], where the policy is parameterized by the threshold θ. We choose $c = 0.3$, $N = 1$ and $\Delta = \mathcal{N}(0, 1)$ in Algorithm 2. The learning rate is adapted using ADAM(0.01) [36].
2) TPRO [8] and PPO [9], which are two state of the art policy gradient based RL algorithms for models with discrete action spaces, where we use the default architecture and parameters from ChainerRL [37].

We run each algorithm for 2×10^6 samples and repeat this experiment 100 times for RMC and 10 times for TRPO and PPO. To compare the performance of these algorithms, we periodically evaluate the performance of π_{θ_m} for each trajectory using Monte Carlo evaluation (over 200 samples averaged over 10 independent runs). The median, first quartile, and third quartile across the runs are shown in Fig. 1b. The optimal total cost computed using [41] and the knowledge of the model is also shown in Fig. 1b.

We observe that all three algorithms converge to the optimal values. TRPO and PPO converge in fewer number of samples (which is expected because the critic is keeping track of the entire value function), but require significantly more computational resources. In our experiments, each run of TRPO took ≈ 1.4 hours (wall clock time), PPO took ≈ 2.7 hours whereas RMC took ≈ 0.5 seconds.

C. Inventory Control

In this experiment, we study an inventory management problem that arises in operations research [42], [43]. Let $S_t \in \mathbb{R}$ denote the volume of goods stored in a warehouse, $A_t \in \mathbb{R}_{\geq 0}$ denote the amount of goods ordered, and D_t denotes the demand. The state evolves according to $S_{t+1} = S_t + A_t - D_{t+1}$.

We work with the normalized cost function:

$$C(s) = a_p s(1-\gamma)/\gamma + a_h s \mathbb{1}_{\{s \geq 0\}} - a_b s \mathbb{1}_{\{s<0\}},$$

where a_p is the procurement cost, a_h is the holding cost, and a_b is the backlog cost (see [44, Chapter 13] for details).

It is known that there exists a threshold θ such that the optimal policy is a base stock policy with threshold θ (i.e., whenever the current stock level falls below θ, one orders up to θ). Furthermore, for $s \leq \theta$, we have that [44, Sec. 13.2]

$$V_\theta(s) = C(s) + \frac{\gamma}{(1-\gamma)} \mathbb{E}[C(\theta - D)]. \tag{22}$$

So for $B \subset (0, \theta)$, the value function is locally Lipschitz in B with

$$L_\theta = \left(a_h + \frac{1-\gamma}{\gamma} a_p \right).$$

So, we can use approximate RMC to learn the optimal policy.

In our experiments, we consider an inventory management model with $a_h = 1$, $a_b = 1$, $a_p = 1.5$, $D_t \sim \text{Exp}(\lambda)$ with $\lambda = 0.025$, start state $s_0 = 1$, discount factor $\gamma = 0.9$.

We compare the performance for the following algorithms:

1) RMC with simultaneous perturbation based gradient (see Sec. II-B), where the policy is parameterized by the threshold θ. We choose $c = 3.0$, $N = 100$, and $\Delta = \mathcal{N}(0, 1)$ in Algorithm 2 and choose $B = (0, 1)$ for approximate RMC. The learning rate is adapted using ADAM(0.25) [36].
2) DDPG [45], which is of one of state of the art RL algorithms for models with continuous action spaces, where we use the default architecture and implementation from ChainerRL [37].

We run each algorithm for $\approx 5 \times 10^6$ samples and repeat this experiment 100 times for RMC and 10 times for DDPG. To compare the performance of these algorithms, we use Monte Carlo evaluation (over 200 samples averaged over 100 independent runs for RMC and 10 independent runs for DDPG) periodically to evaluate the performance of π_{θ_m} for each trajectory. The median, first quartile and third quartile across the runs are shown in Fig. 1c. The optimal performance computed using [44, Sec. 13.2][7] is also shown.

We observe that DDPG learns in fewer number of samples but it takes more time. In our experiments each run of DDPG took ≈ 10 hours (wall clock time) whereas RMC took ≈ 30 seconds. In addition, RMC converges smoothly to an approximately optimal parameter value with total cost within the bound predicted in Theorem 2. The grey rectangular region in Fig. 1c shows this bound.

[6] 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.

[7] For Exp(λ) demand, the optimal threshold is (see [44, Sec. 13.2])

$$\theta^* = \frac{1}{\lambda} \log \left(\frac{a_h + a_b}{a_h + a_p(1-\gamma)/\gamma} \right).$$

VI. Conclusions

We present a renewal theory based reinforcement learning algorithm called Renewal Monte Carlo (RMC). 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 generalize the RMC algorithm to post-decision state models and present a variant that converges faster to an approximately optimal policy, where the renewal state is replaced by a renewal set. The error in using such an approximation is bounded by the size of the renewal set.

In certain models, one is interested in the performance at a reference state that is not the start state. In such models, we can start with an arbitrary policy and ignore the trajectory until the reference state is visited for the first time and use RMC from that time onwards (assuming that the reference state is the new start state).

Acknowledgment

The authors are grateful to Joelle Pineau for useful feedback and for suggesting the idea of approximate RMC. The authors are also grateful to the anonymous reviewers for suggestions that led to an improved exposition and more detailed numerical experiments.

References

[1] D. Bertsekas and J. Tsitsiklis, *Neuro-dynamic Programming*. 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, 2018.

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

[5] R. S. Sutton, D. A. McAllester *et al.*, “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 J. Control Optim.*, vol. 42, no. 4, pp. 1143–1166, 2003.

[8] J. Schulman, S. Levine *et al.*, “Trust region policy optimization,” in *International Conference on Machine Learning*, June 2015.

[9] J. Schulman, F. Wolski *et al.*, “Proximal policy optimization algorithms,” *arXiv:1707.06347*, 2017.

[10] D. Silver, J. Schrittwieser *et al.*, “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] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov Control Processes: Basic Optimality Criteria*. Springer, 1996, vol. 30.

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

[17] S. P. Meyn and R. L. Tweedie, *Markov chains and stochastic stability*. Springer, 2012.

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

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

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

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

[22] 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.

[23] J. Peters and S. Schaal, “Policy gradient methods for robotics,” in *International Conference on Intelligent Robots and Systems*, Oct. 2006.

[24] P. Thomas, “Bias in natural actor-critic algorithms,” in *International Conference on Machine Learning*, June 2014, pp. 441–448.

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

[26] 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.

[27] 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.

[28] S. Bhatnagar, H. Prasad, and L. Prashanth, *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*. Springer, 2013, vol. 434.

[29] B. Van Roy, D. P. Bertsekas *et al.*, “A neuro-dynamic programming approach to retailer inventory management,” in *36th IEEE Conference on Decision and Control, 1997*, vol. 4, Dec. 1997, pp. 4052–4057.

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

[31] K. Hinderer, “Lipschitz continuity of value functions in Markovian decision processes,” *Mathematical Methods of Operations Research*, vol. 62, no. 1, pp. 3–22, Sep 2005.

[32] E. Rachelson and M. G. Lagoudakis, “On the locality of action domination in sequential decision making,” in *International Symposium on Artificial Intelligence and Mathematics*, Fort Lauderdale, US, Jan. 2010.

[33] M. Pirotta, M. Restelli, and L. Bascetta, “Policy gradient in Lipschitz Markov decision processes,” *Machine Learning*, vol. 100, no. 2, pp. 255–283, Sep 2015.

[34] J. Subramanian and A. Mahajan, “Renewal Monte Carlo,” https://codeocean.com/capsule/027c3bab-27cf-4f47-8153-6533c2bfc1e5, Aug. 2019.

[35] S. Bhatnagar, R. Sutton *et al.*, “Natural actor-critic algorithms,” Dept. of Computing Science, University of Alberta, Canada, Tech. Rep., 2009.

[36] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[37] Preferred Networks Inc. “ChainerRL, A deep reinforcement learning library built on top of Chainer”. [Online]. Available: https://github.com/chainer/chainerrl

[38] 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.

[39] 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.

[40] Y. Xu and J. P. Hespanha, “Optimal communication logics in networked control systems,” in *43rd IEEE Conference on Decision and Control*, Dec. 2004, pp. 3527–3532.

[41] 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.

[42] K. J. Arrow, T. Harris, and J. Marschak, “Optimal inventory policy,” *Econometrica: Journal of the Econometric Society*, pp. 250–272, 1951.

[43] R. Bellman, I. Glicksberg, and O. Gross, “On the optimal inventory equation,” *Management Science*, vol. 2, no. 1, pp. 83–104, 1955.

[44] P. Whittle, *Optimization Over Time: Dynamic Programming and Optimal Control*. John Wiley and Sons, Ltd., 1982.

[45] T. P. Lillicrap, J. J. Hunt *et al.*, “Continuous control with deep reinforcement learning,” in *International Conference on Learning Representations, San Juan, Puerto Rico, May 2-4*, 2016.