A General Family of Robust Stochastic Operators for Reinforcement Learning

Yingdong Lu  
IBM Research AI  
T.J. Watson Research Center  
Yorktown Heights, NY 10598  
yingdong@us.ibm.com

Mark S. Squillante  
IBM Research AI  
T.J. Watson Research Center  
Yorktown Heights, NY 10598  
mss@us.ibm.com

Chai Wah Wu  
IBM Research AI  
T.J. Watson Research Center  
Yorktown Heights, NY 10598  
cwwu@us.ibm.com

Abstract

We consider a new family of operators for reinforcement learning with the goal of alleviating the negative effects and becoming more robust to approximation or estimation errors. Various theoretical results are established, which include showing on a sample path basis that our family of operators preserve optimality and increase the action gap. Our empirical results illustrate the strong benefits of our family of operators, significantly outperforming the classical Bellman operator and recently proposed operators.

1 Introduction

Reinforcement learning has a rich history within the machine learning community to solve a wide variety of decision making problems in environments with unknown and unstructured dynamics. Through iterative application of a convergent operator, value-based reinforcement learning generates successive refinements of an initial value function. Q-learning [14] is a particular reinforcement learning technique in which the value iteration computations consist of evaluating the corresponding Bellman equation without a model of the environment.

While Q-learning continues to be broadly and successfully used to determine the optimal actions of an agent in reinforcement learning, the development of new Q-learning approaches that improve convergence speed, accuracy and robustness remains of great interest. One approach might be based on having the agent learn optimal actions through the use of optimality conditions which are weaker than the Bellman equation such that value iteration continues to converge to an action-value function $Q$ associated with an optimal policy, while at the same time increasing the separation between the value function and $Q$-function limits. Exploiting these weaker conditions for optimality could lead to alternatives to the classical Bellman operator that improve convergence speed, accuracy and robustness in reinforcement learning, especially in the company of estimation or approximation errors.

A recent study by Bellemare et al. [3] considers the problem of identifying new alternatives to the Bellman operator based on having the new operators satisfy the properties of optimality-preserving, namely convergence to an optimal action policy, and gap-increasing, namely convergence to a larger deviation between the $Q$-values of optimal actions and suboptimal...
actions. Here the former ensures optimality while the latter can help the learning algorithm determine the optimal actions faster, more easily, and with less errors of mislabeling suboptimal actions. The authors propose a family of operators based on two inequalities between the proposed operator and the Bellman operator, and they show that the proposed family satisfies the properties of optimality-preserving and gap-increasing. Then, after empirically demonstrating the benefits of the proposed operator, the authors [3] raise open fundamental questions with respect to the possibility of weaker conditions for optimality, the statistical efficiency of the proposed operator, and the possibility of a maximally efficient operator.

At the heart of the problem is a fundamental tradeoff between violating the preservation of optimality and increasing the action gap. Although the benefits of increasing the action gap in the presence of approximation or estimation errors are well known [8], increasing the action gap beyond a certain region in a deterministic sense can lead to violations of optimality preservation, thus resulting in value iterations that may not converge to optimal solutions. Our approach is intuitively based on the idea that the action gap can be increased beyond this region for individual value iterations as long as the overall value iterations are controlled in a probabilistic manner that ensures the preservation of optimality in a stochastic sense. In devising a family of operator endowed with these properties, we provide a more general approach that yields greater robustness to approximation or estimation errors.

In this paper we propose a general family of robust stochastic operators, which subsumes the family of operators in [3] as a strict subset, and we address many of the open fundamental questions raised in [3]. Our approach is applicable to arbitrary Q-value approximation schemes and is based on support to devalue suboptimal actions while preserving the set of optimal policies in a stochastic sense. This makes it possible to increase the action gap between the Q-values of optimal and suboptimal actions to a greater extent beyond the aforementioned deterministic region, which can be critically important in practice because of the advantages of increasing the action gap in the company of approximation or estimation errors [8]. Since the value-iteration sequence generated under our family of stochastic operators will be based on realizations of random variables, our theoretical results include establishing the fact that the random variables converge almost surely [5] to the same limit that produces the optimal actions. In addition, our theoretical results include showing that our robust stochastic operators are optimality-preserving and gap-increasing on a sample-path basis [5], establishing that our family of operators significantly broadens the set of weaker conditions for optimality over those in [3], and showing that a statistical ordering of the key components of our operators leads to a corresponding ordering of the action gaps. Key implications of these results include: the search space for the maximally efficient operator should be an infinite dimensional space of random variables, instead of the finite space alluded to in [3]; our statistical ordering results can lead to order relationships among the operators in terms of action gaps, which can in turn lead to maximally efficient operators.

We subsequently apply our robust stochastic operators to obtain empirical results for a wide variety of problems in the OpenAI Gym framework [6], and then compare these empirical results against those under both the classical Bellman operator and the consistent Bellman operators in terms of action gaps, which can in turn lead to maximally efficient operators.

2 Preliminaries

We consider a standard reinforcement learning (RL) framework (see, e.g., [4]) in which a learning agent interacts with a stochastic environment. This interaction is modeled as a discrete-time discounted Markov Decision Process (MDP) given by \((\mathcal{X}, \mathcal{A}, \mathbb{P}, R, \gamma)\), where \(\mathcal{X}\) is the set of states, \(\mathcal{A}\) is the set of actions, \(\mathbb{P}\) is the transition probability kernel, \(R\) is the reward function mapping state-action pairs to a bounded subset of \(\mathbb{R}\), and \(\gamma \in [0, 1)\) is the discount factor. Let \(\mathbb{Q}\) and \(\mathbb{V}\) denote the set of bounded real-valued functions over \(\mathcal{X} \times \mathcal{A}\) and \(\mathcal{X}\), respectively. For \(Q \in \mathbb{Q}\), we define \(V(x) := \max_a Q(x, a)\) and use the same definition for \(\tilde{Q} \in \mathbb{Q}\) and \(\tilde{V}(x)\), and so on. Let \(x'\) always denotes the next state random variable. For the current state \(x\) in which action \(a\) is taken, i.e., \((x, a) \in \mathcal{X} \times \mathcal{A}\), we denote by \(\mathbb{P}(\cdot|x, a)\) the conditional transition probability for the next state \(x'\) and we define \(E_\mathbb{P} := \mathbb{E}_{x' \sim \mathbb{P}(\cdot|x, a)}\) to be the expectation with respect to \(\mathbb{P}(\cdot|x, a)\).
A stationary policy \( \pi(\cdot|x) : \mathcal{X} \to \mathcal{A} \) defines the distribution of control actions given the current state \( x \), which reduces to a deterministic policy when the conditional distribution renders a constant action for state \( x \); with slight abuse of notation, we always write policy \( \pi(x) \). The stationary policy \( \pi \) induces a value function \( V^\pi : \mathcal{X} \to \mathbb{R} \) and an action-value function \( Q^\pi : \mathcal{X} \times \mathcal{A} \to \mathbb{R} \) where \( V^\pi(x) := Q^\pi(x, \pi(x)) \) defines the expected discounted cumulative reward under policy \( \pi \) starting in state \( x \), and \( Q^\pi \) satisfies the Bellman equation

\[
Q^\pi(x, a) = R(x, a) + \gamma \mathbb{E}_p Q^\pi(x', \pi(x')).
\]  

Our goal is to determine a policy \( \pi^* \) that achieves the optimal value function \( V^*(x) := \sup_\pi V^\pi(x), \forall x \in \mathcal{X} \), which also produces the optimal action-value function \( Q^*(x, a) := \sup_\pi Q^\pi(x, a), \forall (x, a) \in \mathcal{X} \times \mathcal{A} \). Define the Bellman operator \( T_B : \mathcal{Q} \to \mathcal{Q} \) pointwise as

\[
T_B Q(x, a) := R(x, a) + \gamma \mathbb{E}_p \max_{b \in \mathcal{A}} Q(x', b),
\]

or equivalently \( T_B Q(x, a) = R(x, a) + \gamma \mathbb{E}_p V(x') \). The Bellman operator \( T_B \) is known (see, e.g., [3]) to be a contraction mapping in supremum norm whose unique fixed point coincides with the optimal action-value function, namely

\[
Q^*(x, a) = R(x, a) + \gamma \mathbb{E}_p \max_{b \in \mathcal{A}} Q^*(x', b),
\]

or equivalently \( Q^*(x, a) = R(x, a) + \gamma \mathbb{E}_p V^*(x') \). This in turn indicates that the optimal policy \( \pi^* \) can be obtained by

\[
\pi^*(x) = \arg \max_{a \in \mathcal{A}} Q^*(x, a), \quad \forall x \in \mathcal{X}.
\]

As noted by Bellemare et al. [3] and illustrated through a simple example, the optimal state-action value function obtained through the Bellman operator does not always describe the value of stationary policies. Although these nonstationary effects cause no problems when the MDP can be solved exactly, such nonstationary effects in the presence of estimation or approximation errors, which may lead to small differences between the optimal state-action value function and the suboptimal ones, can result in errors in identifying the optimal actions. To address issues of nonstationarity of this and related forms arising in practice, Bellemare et al. [3] propose the so-called consistent Bellman operator defined as

\[
T_C Q(x, a) := R(x, a) + \gamma \mathbb{E}_p \left[ \mathbb{1}_{\{x \neq x'\}} \max_{b \in \mathcal{A}} Q(x', b) + \mathbb{1}_{\{x = x'\}} Q(x, a) \right],
\]

where \( \mathbb{1}_{\{\cdot\}} \) denotes the indicator function. The consistent Bellman operator \( T_C \) preserves a local form of stationarity by redefining the action-value function \( Q \) such that, if an action \( a \in \mathcal{A} \) is taken from the state \( x \in \mathcal{X} \) and the next state \( x' = x \), then action \( a \) is taken again. Bellemare et al. [3] proceed to show that the consistent Bellman operator yields the optimal policy \( \pi^* \), and in particular \( T_C \) is both optimality-preserving and gap-increasing, each of which is defined as follows.

**Definition 2.1.** An operator \( \mathcal{T} \) for \( \mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathbb{P}, R, \gamma) \) is optimality-preserving if for any \( Q_0 \in \mathcal{Q} \) and \( x \in \mathcal{X} \) with \( Q_{k+1} := TQ_k \), then \( \hat{V} := \lim_{k \to \infty} \max_{a \in \mathcal{A}} Q_k(x, a) \) exists, is unique, \( \hat{V}(x) = V^*(x) \), and

\[
Q^*(x, a) < V^*(x) \Rightarrow \limsup_{k \to \infty} Q_k(x, a) < V^*(x), \quad \forall a \in \mathcal{A}.
\]

Moreover, an operator \( \mathcal{T} \) for \( \mathcal{M} \) is gap-increasing if for all \( Q_0 \in \mathcal{Q} \), \( x \in \mathcal{X} \) and \( a \in \mathcal{A} \), with \( Q_{k+1} := TQ_k \) and \( V_k(x) := \max_{a \in \mathcal{A}} Q_k(x, a) \), then

\[
\liminf_{k \to \infty} [V_k(x) - Q_k(x, a)] \geq V^*(x) - Q^*(x, a).
\]

The operator property of optimality-preserving is important because it ensures that at least one optimal action remains optimal and that suboptimal actions remain suboptimal. As suggested above, the operator property of gap-increasing is important from the perspective of robustness when the inequality (5) is strict for at least one \( (x, a) \in \mathcal{X} \times \mathcal{A} \). In particular, as the action gap of an operator increases while remaining optimality-preserving, the end result is greater robustness to approximation or estimation errors [3].

3
3 Robust Stochastic Operator

In this section we propose a general family of robust stochastic operators and then establish that this general family of operators are optimality-preserving and gap-increasing. We further show that our family of robust stochastic operators are strictly broader and more general than the family of consistent Bellman operators, while also addressing some of the fundamental open questions raised in [3] concerning weaker conditions for optimality-preserving, statistical efficiency of new operators, and maximally efficient operators. It is important to note, as also emphasized in [3], that our approach can be extended to variants of the Bellman operator such as SARSA [10], policy evaluation [11] and fitted Q-iteration [7].

For all \( Q_0 \in \mathcal{Q}, \ x \in \mathcal{X}, \ a \in \mathcal{A} \) and \( \{\beta_k : k \in \mathbb{N}\} \), a sequence of independent nonnegative random variables with finite support and expectation \( \overline{\beta} := \mathbb{E}_\beta[\beta_k] \in [0,1) \), we define

\[
T_{\beta_k} Q(x,a) := R(x,a) + \gamma \mathbb{E}_b \max_{b \in \mathcal{A}} Q(x',b) - \beta_k V_k(x) - Q_k(x,a),
\]  

(6)

or equivalently \( T_{\beta_k} Q(x,a) := R(x,a) + \gamma \mathbb{E}_b V(x') - \beta_k (V_k(x) - Q_k(x,a)) \).

Then the members of the general family of robust stochastic operators include the \( T_{\beta_k} \) defined over all probability distributions for the sequences \( \{\beta_k\} \) with \( \overline{\beta} \in [0,1) \). Furthermore, we define \( T_{\overline{\beta}} \) to be the general family of robust stochastic operators comprising all \( T \) such that there exists a sequence of \( \{\beta_k\} \) and the following inequalities hold

\[
T_{\overline{\beta}} Q(x,a) - \beta_k (V_k(x) - Q_k(x,a)) \leq TQ(x,a) \leq T_{\overline{\beta}} Q(x,a), \quad \forall x \in \mathcal{X}, a \in \mathcal{A}.
\]  

(7)

It is obvious that these are strictly weaker conditions than those identified in [3]; and since realizations of \( \beta_k \) can clearly take on values outside of \([0,1)\), the family of operators \( T_{\overline{\beta}} \) subsumes the family of operators identified in [3]. Our theorem establishes that the general family of robust stochastic operators are also optimality-preserving and gap-increasing.

**Theorem 3.1.** Let \( T_{\overline{\beta}} \) be the Bellman operator defined in [2] and \( T_{\beta_k} \) the robust stochastic operator defined in [3]. Considering the sequence \( Q_{k+1} := T_{\beta_k} Q_k \) with \( Q_0 \in \mathcal{Q} \), the conditions of optimality-preserving and gap-increasing hold almost surely and therefore the operator \( T_{\overline{\beta}} \) is both optimality-preserving and gap-increasing almost surely. Moreover, all operators in the family \( T_{\overline{\beta}} \) are optimality-preserving and gap-increasing almost surely.

**Proof.** We first consider aspects of the optimality-preserving properties of the family of robust stochastic operators. Since the conditions of Lemma [A.1] due to Bellemare et al. [3] hold, it follows that the sequence \( \{V_k(x) : k \in \mathbb{N}\} \) converges asymptotically on a sample path basis [3] such that \( \lim_{k \to \infty} V_k(x) = \hat{V}(x) \leq V^*(x) \), for all \( x \in \mathcal{X} \). Defining \( \hat{Q}(x,a) := \limsup_{k \to \infty} Q_k(x,a) = \limsup_{k \to \infty} T_{\beta_k} Q_k(x,a) \) and applying a derivation from the proof of Theorem 2 in [3] for each sample path, we can conclude that

\[
\hat{Q}(x,a) \leq \limsup_{k \to \infty} T_{\overline{\beta}} Q_k(x,a) = \limsup_{k \to \infty} \left[ R(x,a) + \gamma \mathbb{E}_b \max_{b \in \mathcal{A}} Q_k(x',b) \right]
\]

\[
\leq R(x,a) + \gamma \mathbb{E}_b \left[ \max_{b \in \mathcal{A}} \limsup_{k \to \infty} Q_k(x',b) \right] = T_{\overline{\beta}} \hat{Q}(x,a).
\]

(8)

holds almost surely [3].

Meanwhile, we have

\[
Q_{k+1}(x,a) = T_{\beta_k} Q_k(x,a) = T_{\overline{\beta}} Q_k - \beta_k (V_k(x) - Q_k(x,a)).
\]

Taking conditional expectation with respect to the filtration \( \mathcal{F}_k = \sigma(\beta_1, \beta_2, \ldots, \beta_k) \), renders

\[
\mathbb{E}[Q_{k+1}(x,a) | \mathcal{F}_k] = \mathbb{E}[T_{\beta_k} Q_k(x,a) | \mathcal{F}_k] = T_{\overline{\beta}} Q_k - \beta_k (V_k(x) - Q_k(x,a)),
\]

or equivalently

\[
Q_k(x,a) + \mathbb{E}[Q_{k+1}(x,a) - Q_k(x,a) | \mathcal{F}_k] = \mathbb{E}[T_{\beta_k} Q_k(x,a) | \mathcal{F}_k] = T_{\overline{\beta}} Q_k - \beta_k (V_k(x) - Q_k(x,a)).
\]

Since \( \overline{\beta} = \mathbb{E}[\beta_k] \in [0,1) \), we know from Lemma [A.1] that \( Q_k(x,a) \) converges on each sample path. Hence, for any \( \epsilon > 0 \) on each sample path, we observe that \( Q_{k+1}(x,a) - Q_k(x,a) < \epsilon \) when \( k \) is sufficiently large. We therefore have, for sufficiently large \( k \),

\[
Q_k(x,a) + \epsilon \geq \mathbb{E}[T_{\beta_k} Q_k(x,a) | \mathcal{F}_k] = T_{\overline{\beta}} Q_k - \beta_k (V_k(x) - Q_k(x,a)).
\]
Because $\epsilon$ is arbitrary, we can conclude that
\[ Q_k(x, a) \geq \mathbb{E}[\mathcal{T}_{\hat{\beta}_k} Q_k(x, a) | F_k] = \mathcal{T}_B Q_k - \beta \mathbb{E}[V_k(x) - Q_k(x, a)] \]
holds for large $k$. Taking the limit superior on both sides, we obtain
\[ \hat{Q}(x, a) \geq \mathcal{T}_B \hat{Q}(x, a) - \beta \hat{V}(x) + \beta \hat{Q}(x, a), \]
which in combination with (8) leads to the conclusion that $\hat{V}(x) = V^*(x)$ almost surely.

Next, to prove that $\mathcal{T}_{\hat{\beta}_k}$ is gap-increasing, the above arguments render $\lim_{k \to \infty} V_k(x) = V^*(x)$ on a sample path basis, and thus (5) is equivalent to
\[ \limsup_{k \to \infty} Q_k(x, a) \leq Q^*(x, a) \]
almost surely. This inequality follows on a sample path basis from $\mathcal{T}_{\beta_k} Q(x, a) \leq \mathcal{T}_B Q(x, a)$ by definition and Lemma A.1 and therefore we have the desired result for the operators $\mathcal{T}_{\beta_k}$. Furthermore, it can be readily verified that the above arguments can be similarly applied to cover all of the operators in $\mathcal{T}^F_{\beta}$.

Lastly, from the above results of (5) and $\hat{V}(x) = V^*(x)$ almost surely, it follows that (4) also holds almost surely for $\mathcal{T}_{\hat{\beta}_k}$ as well as all operators in $\mathcal{T}^F_{\beta}$, thus completing the proof.

The definition of $\mathcal{T}^F_{\beta}$ and Theorem 3.1 significantly enlarges the set of optimality-preserving and gap-increasing operators identified in [3]. In particular, our new sufficient conditions for optimality-preserving operators implies that significant deviation from the Bellman operator is possible without loss of optimality. More importantly, the definition of $\mathcal{T}^F_{\beta}$ and Theorem 3.1 implies that the search space for maximally efficient operators should be an infinite dimensional space of random variables, instead of the finite dimensional space that is alluded to in [3]. We now establish results on certain statistical properties for the sequences $\{\beta_k\}$ within our general family of robust stochastic operators, which offer key relational insights into important orderings of different operators in $\mathcal{T}^F_{\beta}$ in terms of their action gaps. This can then be exploited in searching for and attempting to find maximally efficient operators in practice.

**Theorem 3.2.** Suppose $Q_{k+1}$ and $\hat{Q}_{k+1}$ are respectively updated with two different robust stochastic operators $\mathcal{T}_{\beta_k}$ and $\mathcal{T}_{\hat{\beta}_k}$ that are distinguished by $\beta_k$ and $\hat{\beta}_k$ satisfying $\mathbb{E}[\beta_k] = \mathbb{E}[\hat{\beta}_k]$ and $\text{Var}[\beta_k] \leq \text{Var}[\hat{\beta}_k]$; namely $Q_{k+1} = \mathcal{T}_{\beta_k} Q_k$ and $\hat{Q}_{k+1} = \mathcal{T}_{\hat{\beta}_k} Q_k$. Then we have $\text{Var}[Q_{k+1}] \leq \text{Var}[\hat{Q}_{k+1}]$.

**Proof.** The desired result can be readily seen from
\[
\text{Var}[Q_{k+1}] = \mathbb{E}[\text{Var}[Q_{k+1} | Q_k]] + \text{Var}[\mathbb{E}[Q_{k+1} | Q_k]] = \text{Var}[\beta_k] \mathbb{E}[(V_k(x) - Q_k(x, a))^2] + \text{Var}[\mathbb{E}[Q_{k+1} | Q_k]]
\]
and
\[
\text{Var}[\hat{Q}_{k+1}] = \mathbb{E}[\text{Var}[\hat{Q}_{k+1} | Q_k]] + \text{Var}[\mathbb{E}[\hat{Q}_{k+1} | Q_k]] = \text{Var}[\hat{\beta}_k] \mathbb{E}[(V_k(x) - Q_k(x, a))^2] + \text{Var}[\mathbb{E}[\hat{Q}_{k+1} | Q_k]].
\]

The theorem concludes that a larger variance for $\beta_k$ in fact leads to a larger variance for $Q_k(x, a)$. We know that, in the limit, the optimal action will maintain its state-action value function. Then, when $k$ is sufficiently large, we can expect that the state-value function for the optimal action will be very close to the optimal value. In this case, a larger variance implies that the smaller sub-optimal values will have larger probability, and thus they can be understood to have a larger action gap.

The results of Theorem 3.2 are consistent with our observations from the numerical experiments in Section 4, where the operator $\mathcal{T}_{\beta_k}$ associated with the sequence $\{\beta_k\}$ drawn from a uniform distribution outperforms the operator $\mathcal{T}_{\hat{\beta}_k}$ associated with the constant sequence $\{\beta\}$.
4 Experimental Results

Within the general RL framework of interest, we consider a standard, yet generic, form for Q-learning so as to cover the various experimental programs examined in this section. Specifically, for all $Q_k \in \mathbb{Q}$, $x \in \mathbb{X}$, $a \in \mathbb{A}$ and an operator of interest $\mathcal{T}$, we consider the sequence of action-value $Q$-functions based on the following generic update rule:

$$Q_{k+1}(x, a) = (1 - \alpha_k)Q_k(x, a) + \alpha_k \mathcal{T}Q_k(x, a)$$  \hspace{1cm} (10)

where $\alpha_k$ is the learning rate for iteration $k$. Our empirical comparisons comprise the Bellman operator $\mathcal{T}_B$, the consistent Bellman operator $\mathcal{T}_C$, and instances of our family of robust stochastic operators $\mathcal{T}_{\beta_k}$, denoted hereafter as RSO.

We conduct numerous experiments on several well-known problems using the OpenAI Gym framework [6], namely Mountain Car, Acrobot, Cart Pole and Lunar Lander. This collection of problems span a wide variety of RL examples with different characteristics, dimensions, parameters, and so on; in each case, the state space is continuous and discretized to a finite set of states. For every problem, the specific Q-learning algorithms considered are defined as in [11], where the appropriate operator of interest $\mathcal{T}_B$, $\mathcal{T}_C$ or $\mathcal{T}_{\beta_k}$ is substituted for $\mathcal{T}$; at each timestep, (10) is applied to a single point of the Q-function at the current state and action. The experiments for every problem from the OpenAI Gym were run using the existing code found at [13, 1], exactly as is with the sole change comprising the replacement of the Bellman operator in the code with corresponding implementations of either the consistent Bellman operator or RSO; see Appendix B for the corresponding python code. Multiple experimental trials are run for each problem, where we ensured the setting of the random starting state to be the same in each experimental trial for all three types of operators by initializing them with the same random seed. We observe that for different problems and different variants of the Q-learning algorithm, simply replacing the Bellman operator or the consistent Bellman operator with the Robust Stochastic operator generally results in improved performance.

4.1 Mountain Car

This problem is first discussed in [9]. The state vector is 2-dimensional with a total of three possible actions, and the score represents the number of timesteps needed to solve the problem. We ran 20 experimental trials over 10,000 episodes for training, each of which consists of up to 200 steps; then the problem is solved for 1000 episodes using the policy obtained from the Q-function training. In both cases, the goal is to minimize the score. The RSO considered in each experimental trial consists of $\beta_k$ uniformly distributed over $[0, 2)$.

For the training phase, Figure 1a plots the score, averaged over moving windows of 500 episodes across the 20 trials, as a function of the number of episodes. We observe that the average score under the RSO exhibits much better performance than under the Bellman operator or the consistent Bellman operator. Moreover, as can be seen from the smoothness of the curves in Figure 1a, the standard deviation is relatively small for all three operators. For the testing phase, the average score and the standard deviation of the score over the 20 experimental trials, each comprising 1000 episodes, are respectively given by: 129.93 and 32.68 for the Bellman operator; 127.58 and 30.90 for the consistent Bellman operator; and 122.70 and 7.25 for the RSO. Here we observe that both the average score and its standard deviation under the RSO exhibit better performance than under the Bellman operator or the consistent Bellman operator.

4.2 Acrobot

This problem is first discussed in [12]. The state vector is 6-dimensional with three actions possible in each state, and the score represents the number of timesteps needed to solve the problem. We ran 20 experimental trials over many episodes, with a goal of minimizing the score. The RSO considered in each experimental trial consists of $\beta_k \sim U[0, 2)$.

Figure 1b plots the score, averaged over moving windows of 1000 episodes across the 20 trials, as a function of the number of episodes. We observe that the average score under the RSO exhibits much better performance than under the Bellman operator or the consistent Bellman operator.
operator. Furthermore, as can be seen from the smoothness of the curves in Figure 1b, the standard deviation is relatively small for all three operators.

4.3 Cart Pole

This problem is first discussed in [2]. The state vector is 4-dimensional with two actions possible in each state, and the score represents the number of steps where the cart pole stays upright before either falling over or going out of bounds. We ran 20 experimental trials over many episodes, each of which consists of up to 200 steps with a goal of maximizing the score. When the score is above 195, the problem is considered solved. Two RSOs are considered for each experimental trial, namely one in which \( \beta_k \) is uniformly distributed over \([0, 2)\) and another in which \( \beta_k \) is fixed to be 1.0.

Figure 2a plots the score, averaged over moving windows of 1000 episodes across the 20 trials, as a function of the number of episodes. A plot of the corresponding standard deviation, taken over the same number of score values, is presented in Figure 2b. We observe that both the average score and its standard deviation under the RSOs exhibit better performance than under the Bellman operator or the consistent Bellman operator. In particular, the average score over the last 100 episodes across the 20 trials is 190.92 under the RSO with \( \beta_k \sim U[0, 2) \) in comparison with 183.67 and 184.07 under the Bellman and consistent Bellman operators, respectively; the corresponding standard deviations of the scores are 19.44, 28.87 and 27.82 for the RSO, Bellman operator and consistent operator, respectively. We also observe that the RSO with \( \beta_k \sim U[0, 2) \) tends to perform better than the RSO with fixed \( \beta_k = 1.0 \) over the sequences of episodes.

4.4 Lunar Lander

This problem is discussed in [6]. The state vector is 8-dimensional with a total of four possible actions, and the physics of the problem is known to be more difficult than the
foregoing problems. The score represents the cumulative reward comprising positive points for successful degrees of landing and negative points for fuel usage and crashing. We ran 20 experimental trials over many episodes, each of which consists of up to 200 steps with a goal of maximizing the score. The RSO considered in each experimental trial consists of $\beta_k$ uniformly distributed over $[0, 2)$.

For the training phase, Figure 3a plots the score, averaged over moving windows of 1000 episodes across the 20 trials, as a function of the number of episodes. We observe that the average score under the RSO exhibits better performance than under the Bellman operator or the consistent Bellman operator. Moreover, as can be seen from the smoothness of the curves in Figure 3a, the standard deviation is relatively small for all three operators. For the testing phase, the average score over the 20 experimental trials, each comprising 1000 episodes, is respectively given by: $-241.94$ for the Bellman operator; $-188.44$ for the consistent Bellman operator; and $-167.51$ for the RSO. (Once again, the standard deviation is comparable across all three operators.) Here we observe once again that the average score under the RSO exhibits better performance than under the Bellman operator or the consistent Bellman operator. The improved performance under the RSO can be explained by Figure 3b that shows the distribution of scores for both the RSO and the consistent Bellman operator. Here we observe that the distribution for the RSO is shifted further to the right.

![Figure 3a](image1.png)  ![Figure 3b](image2.png)

(a) Average score in training phase  (b) Distribution of scores in testing phase

Figure 3: Lunar Lander: Statistics on score in solving maximization problem.

5 Conclusions

We proposed and analyzed a new general family of robust stochastic operators for reinforcement learning, which subsumes the classical Bellman operator and a recently proposed family of operators. Our goal was to provably preserve optimality while significantly increasing the action gap, thus providing robustness with respect to approximation or estimation errors. We establish and discuss fundamental theoretical results for our general family of robust stochastic operators. In addition, our collection of empirical results – based on several well-known problems within the OpenAI Gym framework spanning a wide variety of reinforcement learning examples with diverse characteristics – consistently demonstrates and quantifies the significant performance improvements obtained with our operators over existing operators. We believe our work can lead to opportunities to find maximally efficient operators in practice.

References

[1] M. Alzantot. Solution of mountaincar OpenAI Gym problem using Q-learning. URL https://gist.github.com/malzantot/9d1d3fa4f4dc4a101bc48a135d8f9a289

[2] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):854–846, Sept. 1983.
[3] M. G. Bellemare, G. Ostrovski, A. Guez, P. S. Thomas, and R. Munos. Increasing the action gap: New operators for reinforcement learning. In Proc. Thirtieth AAAI Conference on Artificial Intelligence, AAAI’16, pages 1476–1483. AAAI Press, 2016.

[4] D. Bertsekas and J. Tsitsiklis. Neuro-Dynamic Programming. Athena Scientific, 1996.

[5] P. Billingsley. Convergence of Probability Measures. Wiley, New York, Second edition, 1999.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. CoRR, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

[7] D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. Journal of Machine Learning Research, 6:503–556, 2005.

[8] A. Farahmand. Action-gap phenomenon in reinforcement learning. Advances in Neural Information Processing Systems, 24, 2011.

[9] A. Moore. Efficient Memory-Based Learning for Robot Control. PhD thesis, University of Cambridge, Cambridge, U.K., 1990.

[10] G. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical report, Cambridge University, 1994.

[11] R. Sutton. Learning to predict by the methods of temporal differences. Machine Learning, 3(1):9–44, 1988.

[12] R. S. Sutton. Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding. Advances in Neural Information Processing Systems, 8:1038–1044, 1996.

[13] V. M. Vilches. Basic reinforcement learning tutorial 4: Q-learning in OpenAI Gym. URL https://github.com/vmayoral/basic_reinforcement_learning/blob/master/tutorial4/README.md.

[14] C. Watkins. Learning from Delayed Rewards. PhD thesis, University of Cambridge, Cambridge, U.K., 1989.

A Theoretical Results

Lemma A.1 (Bellemare et al. [3]). Let \( Q \in \mathbb{Q} \) and \( \pi^Q \) be the policy greedy with respect to \( Q \). Let \( T \) be an operator with the properties that, for all \( x \in \mathbb{X} \) and \( a \in \mathcal{A} \),

1. \( T Q(x, a) \leq T_B Q(x, a) \), and
2. \( T Q(x, \pi^Q(x)) = T_B Q(x, \pi^Q(x)) \).

Consider the sequence \( Q_{k+1} := T Q_k \) with \( Q_0 \in \mathbb{Q} \) and let \( V_k(x) := \max_a Q_k(x, a) \). Then the sequence \( \{V_k \colon k \in \mathbb{N}\} \) converges, and furthermore, for all \( x \in \mathbb{X} \),

\[
\lim_{k \to \infty} V_k(x) \leq V^*(x).
\]

B Python Code

We tested the various operators of interest on several RL problems and algorithms. For our empirical comparisons, the existing code that updates the Q-learning value based on the Bellman operator \( T_B \) is replaced with the corresponding code for the \( T_C \) and \( T_{\beta_k} \) operators. In particular, the following snippets of code describe how this is generically implemented for the original \( T_B \) operator together with the added \( T_C \) and \( T_{\beta_k} \) operators, respectively.
def UpdateQBellman(self, currentState, action, nextState, reward, alpha, gamma):
    Qvalue = self.Q[currentState, action]
    rvalue = reward + gamma * max([self.Q[nextState, a] for a in self.actionsSet])
    self.Q[currentState, action] += alpha * (rvalue - Qvalue)

def UpdateQConsistent(self, currentState, action, nextState, reward, alpha, gamma):
    Qvalue = self.Q[currentState, action]
    rvalue = reward + gamma * (max([self.Q[nextState, a] for a in self.actionsSet])
        if currentState != nextState else Qvalue)
    self.Q[currentState, action] += alpha * (rvalue - Qvalue)

def UpdateQRSO(self, currentState, action, nextState, reward, alpha, gamma, beta):
    Qvalue = self.Q[currentState, action]
    rvalue = reward + (gamma * (max([self.Q[nextState, a] for a in self.actionsSet])))
        - beta * (max([self.Q[currentState, a] for a in self.actionsSet]) - Qvalue)
    self.Q[currentState, action] += alpha * (rvalue - Qvalue)