Bellman operator convergence enhancements in reinforcement learning algorithms

David Krame Kadurha*
AIMS Cameroon,
African Institute for Mathematical Sciences

Yaé Ulrich Gaba†
Department of Mathematics and Applied Mathematics
Sefako Makgatho Health Sciences University (SMU)
Pretoria, South Africa

Domini Jocema Leko Moutouo‡
AIMS Cameroon,
African Institute for Mathematical Sciences

February 20, 2025

Abstract

This paper reviews the topological groundwork for the study of reinforcement learning (RL) by focusing on the structure of state, action, and policy spaces. We begin by recalling key mathematical concepts such as complete metric spaces, which form the foundation for expressing RL problems. By leveraging the Banach contraction principle, we illustrate how the Banach fixed-point theorem explains the convergence of RL algorithms and how Bellman operators, expressed as operators on Banach spaces, ensure this convergence. The work serves as a bridge between theoretical mathematics and practical algorithm design, offering new approaches to enhance the efficiency of RL. In particular, we investigate alternative formulations of Bellman operators and demonstrate their impact on improving convergence rates and performance in standard RL environments such as MountainCar, CartPole, and Acrobot. Our findings highlight how a deeper mathematical understanding of RL can lead to more effective algorithms for decision-making problems.

1 Introduction

Research on the foundational aspects of Reinforcement Learning (RL), particularly from a topological perspective, remains relatively underdeveloped. While RL has seen significant advancements in terms of algorithmic efficiency and practical applications, there is a lack of comprehensive studies that address the underlying mathematical structures of the problem spaces (namely, state, action, and policy spaces). This work aims to consolidate and formalize these fundamental RL concepts by grounding them in a coherent mathematical framework, with a particular focus on topological and geometric perspectives. In existing literature, several contributions have touched on aspects of RL that relate to topology and geometry, but a unified approach is still missing. For instance, [9] explores the geometric and topological properties of value functions within Markov decision processes (MDPs) that have finite states and actions. This work characterizes the value function space as a polytope, elucidating the intricate relationships between policies and value functions. Similarly, [4] highlights the importance of Lipschitz continuity in model-based RL, advocating for the learning of Lipschitz continuous models to improve value function estimation and providing theoretical error bounds. The study in [17] expands the scope of model-free RL algorithms to continuous state-action spaces using a Q-learning-based approach, thereby extending the applicability of RL to more complex domains. Additionally, [12] introduces a unified framework for defining state similarity metrics in RL, addressing the generalization challenges posed by continuous-state systems. This framework offers new insights into how metric spaces can be leveraged to reason about the learning process in RL. Despite these contributions,

*david.krame@aims-cameroon.org
†yaeulrich.gaba@gmail.com
‡domini@aims.ac.za

the lack of a cohesive foundation to interpret these insights and relate them back to a topological understanding of RL limits the impact of such works. Our primary goal is to address this gap by establishing a solid foundation that unifies and organizes the key concepts in RL, particularly in relation to the topology of state, action, and policy spaces. To this end, we build upon mathematical constructs such as metric spaces, normed spaces, and Banach spaces, and we explore how the Banach fixed-point theorem, specifically through the Banach contraction principle, can be applied to explain the convergence of RL algorithms. Furthermore, we propose alternative formulations of Bellman operators, expressed as operators on Banach spaces, to demonstrate how such theoretical insights can lead to improved convergence rates and more efficient algorithmic performance. These contributions are validated through experimental results on standard RL environments such as MountainCar, CartPole, and Acrobot, showing that our approach not only strengthens the theoretical understanding of RL but also offers practical benefits. By laying the groundwork for studying the topology of RL problem spaces, this work aims to serve as a foundational reference for future researchers. We anticipate that the mathematical insights and frameworks presented will aid in the development of more effective algorithms, ultimately advancing the field of reinforcement learning by bridging theory and practice.

The remainder of this paper is structured as follows, and represents a polished and refined version of the earlier work presented by us in [11]. In Section 2, we begin by introducing the foundational mathematical concepts needed for our analysis. Specifically, in Section 2.1, we review contraction mappings and fixed-point theorems, including the Banach contraction principle, which underpins much of reinforcement learning's convergence theory. In Section 2.2, we provide an overview of reinforcement learning, framing it within the context of Markov Decision Processes and setting the stage for the subsequent mathematical discussions. In Section 3, we delve deeper into the theoretical aspects of reinforcement learning, starting with a refinement of the Banach contraction principle in Section 3.1, followed by an examination of Bellman optimality operators in Section 3.2. Section 3.3 explores policy evaluation and iteration through the lens of operator theory, emphasizing the importance of these mathematical structures in understanding RL algorithms. Section 4 introduces alternative formulations to the classical Bellman operator. Section 4.1 discusses the motivation behind seeking such alternatives, while Sections 4.2 and 4.3 present the Consistent Bellman Operator and the Modified Robust Stochastic Operator, respectively. These alternatives are proposed to address some of the limitations of the classical operator, offering improved convergence properties and robustness in various settings. Finally, in Section 5, we provide detailed implementations of the proposed concepts and analyze their performance through experimental results on standard RL environments. This section demonstrates the practical impact of our theoretical findings and validates the proposed approaches.

2 Preliminaries

2.1 Contraction mappings and fixed points

In this section, we revisit the concept of contraction mappings and the Banach fixed-point theorem, which is foundational for understanding the convergence properties of many algorithms, including those in reinforcement learning (RL). We begin by recalling the notion of a contraction mapping and then provide the Banach fixed-point theorem along with its proof. We also discuss the relevance of these mathematical results to RL, particularly in the context of value iteration and Bellman operators.

Definition 2.2. *Let X be a nonempty set and $f : X \to X$ be a mapping on that set.*

- *A point x is said to be a* ***fixed point*** *of f if $f(x) = x$.*
- *We will write $Fix(f) = \{x \in X : f(x) = x\}$, the set of fixed points of f on X.*

Proposition 2.3. *Let X be a nonempty set and $f : X \to X$ a mapping defined on it. If $x \in X$ is a unique fixed point of f^n with $f^n = \underbrace{f \circ f \circ \cdots \circ f}_{n-times}$ for any $n > 1$, then it is the unique fixed point of f and vice versa:*

$$Fix(f^n) = \{x\} \iff Fix(f) = \{x\}.$$

Contraction mappings

Let (X, d) be a *metric space*, where $d : X \times X \to \mathbb{R}$ is a distance function that satisfies the usual properties of a metric: non-negativity, identity of indiscernibles, symmetry, and the triangle inequality. A mapping $T : X \to X$ is called a *contraction mapping* if there exists a constant $\alpha \in [0, 1)$ such that, for all $x, y \in X$,

$$d(T(x), T(y)) \leq \alpha d(x, y).$$

The constant α is called the *contraction constant.* The key idea is that a contraction mapping brings points closer together, ensuring that successive applications of T shrink distances between any two points.

The first interesting result in this context is the following:

Proposition 2.4. *Let (X, d) be a metric space and $f : X \to X$ a contraction mapping with $\alpha \in (0, 1)$. If f has a fixed point, that point is unique.*

Proof. Suppose we have two fixed points x and y for f with $x \neq y$. Because f is a contraction, we can write:

$$0 \neq d(x, y) = d(f(x), f(y)) \leq \alpha \cdot d(x, y),$$

which is a contradiction. Thus, the fixed point is unique. □

The Banach contraction principle : BCP

A powerful result concerning contraction mappings on complete metric spaces is the *Banach fixed-point theorem* (also known as the contraction mapping theorem), which guarantees the existence and uniqueness of a fixed point for such mappings. A *fixed point* $x^* \in X$ is a point that satisfies $T(x^*) = x^*$.

Theorem 2.5 (See [20])**.** *Let (X, d) be a complete metric space and let $T : X \to X$ be a contraction mapping with contraction constant $\alpha \in [0, 1)$. Then, the following holds:*

1. *T has a unique fixed point $x^* \in X$, i.e., there exists a unique $x^* \in X$ such that $T(x^*) = x^*$.*
2. *For any initial point $x_0 \in X$, the sequence $\{x_n\}$ defined by $x_{n+1} = T(x_n)$ converges to x^* as $n \to \infty$. Moreover, the convergence is geometric, i.e.,*
$$d(x_n, x^*) \leq \frac{\alpha^n}{1-\alpha} d(x_0, x_1).$$

The proof of the Banach fixed-point theorem is constructive and proceeds as follows:

Proof. Let $x_0 \in X$ be an arbitrary initial point. Define a sequence $\{x_n\}$ by $x_{n+1} = T(x_n)$ for all $n \geq 0$. We will first show that $\{x_n\}$ is a Cauchy sequence and then that it converges to a unique limit.

Step 1: Show that $\{x_n\}$ is Cauchy. For $n \geq 0$, we have:

$$d(x_{n+1}, x_n) = d(T(x_n), T(x_{n-1})) \leq \alpha d(x_n, x_{n-1}),$$

where the inequality follows from the contraction property of T. By applying this recursively, we get:

$$d(x_{n+1}, x_n) \leq \alpha^n d(x_1, x_0).$$

Summing this geometric series, we find that for $m > n$,

$$d(x_m, x_n) \leq \sum_{k=n}^{m-1} d(x_{k+1}, x_k) \leq \sum_{k=n}^{m-1} \alpha^k d(x_1, x_0) \leq \frac{\alpha^n}{1-\alpha} d(x_1, x_0).$$

Since $\alpha \in [0, 1)$, this bound tends to zero as $n \to \infty$, which shows that $\{x_n\}$ is a Cauchy sequence.

Step 2: Show that $\{x_n\}$ converges. Since X is complete, every Cauchy sequence in X converges. Therefore, there exists a point $x^* \in X$ such that $x_n \to x^*$ as $n \to \infty$.

Step 3: Show that x^* is a fixed point. We now show that $T(x^*) = x^*$. Since T is continuous and $x_n \to x^*$, we have:

$$T(x_n) \to T(x^*) \quad \text{as} \quad n \to \infty.$$

However, by construction, $T(x_n) = x_{n+1}$, and since $x_n \to x^*$, it follows that $x_{n+1} \to x^*$. Therefore, $T(x^*) = x^*$.

Step 4: Show uniqueness. The uniqueness follows from Proposition 2.4. □

The Banach fixed-point theorem is central to the analysis of many reinforcement learning algorithms, particularly those involving value function approximation. In the context of reinforcement learning, the Bellman operator, often used in value iteration, is a contraction under the supremum norm. Specifically, for a Markov Decision Process (MDP), the Bellman optimality operator T^* satisfies the contraction property with a contraction constant γ, where γ is the discount factor. Thus, by applying the Banach fixed-point theorem, we can guarantee the existence and uniqueness of an optimal value function V^*, and the iterative application of the Bellman operator ensures convergence to this fixed point. This theoretical result forms the basis for the convergence of foundational RL algorithms, such as value iteration and policy iteration.

2.6 Overview on Reinforcement Learning

In this part, we will formally present the framework of a Markov Decision Process (MDP) and its relation to Reinforcement Learning (RL). Following this, we will discuss the concept of optimality in RL, and finally, outline key methods for achieving optimality in decision-making tasks.

Definition 2.7 (Markov Decision Process [13, 16])**.** *A Markov Decision Process (MDP) is defined as a 5-tuple* $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$*, where:*

- $\mathcal{S}$*:* ***State space****, the set of all possible states.*
- $\mathcal{A}$*:* ***Action space****, the set of all possible actions the agent can take.*
- $p(s, a, s')$*:* ***Transition probability****, which describes the probability of moving from state* s *to state* s' *given action* a*:*
$$p(s'|s, a) = Pr(S_{t+1} = s'|S_t = s, A_t = a).$$
- $r(s, a, s')$*:* ***Reward function****, which defines the immediate reward received after transitioning from state* s *to state* s' *via action* a*:*
$$r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R} \subset \mathbb{R}.$$
We should also mention that sometimes it is convenient to simply use $r(s, a)$*, defined as follows:*
$$r(s, a) = \sum_{r \in \mathcal{R}} r \cdot \sum_{s'} p(s', r|s, a) \quad \textit{where} \quad p(s', r|s, a) \equiv Pr(S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a)$$
- $\gamma \in [0, 1)$*:* ***Discount factor****, which determines the present value of future rewards, with smaller* γ *values giving more emphasis to immediate rewards.*

Definition 2.8 (Policy or Decision Rule)**.** *A policy* $\pi : \mathcal{S} \to \Delta\mathcal{A}$ *(precisely from the set of states to the probability simplex under actions) defines a strategy that the agent uses to select actions. It can be either:*

- ***Deterministic****, where a specific action is chosen in each state, or*
- ***Stochastic****, where actions are chosen according to a probability distribution over actions in each state.*

The agent interacts with the environment by observing sequences of states, taking actions, and receiving rewards. This interaction can be described as:
$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$$
The goal of the agent is to maximize the cumulative reward over time, referred to as the *return*, which is formalized as:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \quad \gamma \in [0, 1). \tag{2.1}$$

The value function $v_\pi(s)$, representing the expected return starting from state s and following policy π, is defined as:

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s \right]. \tag{2.2}$$

Similarly, the action-value function $q_\pi(s, a)$, representing the expected return starting from state s, taking action a, and then following policy π, is defined as:

$$q_\pi(s,a) = \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]. \tag{2.3}$$

The optimal value function, $v_*(s)$, and the optimal action-value function, $q_*(s,a)$, are defined as the maximum value that can be obtained by any policy π:

$$\begin{aligned} v_*(s) &= \max_\pi v_\pi(s), \\ q_*(s,a) &= \max_\pi q_\pi(s,a). \end{aligned} \tag{2.4}$$

Bellman Optimality Equations

The Bellman optimality equations describe the recursive relationship for the optimal value functions. For the state-value function $v_*(s)$, the Bellman equation is:

$$v_*(s) = \max_a \sum_{s',r} p(s' \mid s,a)\left[r + \gamma v_*(s')\right]. \tag{2.5}$$

For the action-value function $q_*(s,a)$, the Bellman equation is:

$$q_*(s,a) = r(s,a) + \gamma \sum_{s'} p(s' \mid s,a) v_*(s'). \tag{2.6}$$

Solving an MDP means finding the optimal value functions $v_*(s)$ or $q_*(s,a)$, which leads to the determination of the optimal policy π^*.

Solution Methods

There are two main types of algorithms used to solve MDPs and achieve optimality:

- **Model-based algorithms**: These algorithms rely on a known model of the environment, including the transition probabilities and reward function, to compute value functions and derive optimal policies.
- **Model-free algorithms**: These algorithms do not assume knowledge of the environment's model. Instead, they interact with the environment to estimate value functions and improve the policy through exploration and learning.

In model-free algorithms, a central challenge is balancing the exploration-exploitation trade-off: the agent must explore the environment sufficiently to discover the best actions, while also exploiting the knowledge gained to maximize rewards. Many techniques have been developed to tackle this challenge. But it remains an open question [19].

3 Bellman Operators and convergence of RL algorithms

In this section, we frame reinforcement learning in terms of operators, which offers a more structured understanding of why RL algorithms efficiently converge to the optimal policy. We will refine the Banach contraction principle to suit our needs, define the Bellman operators, and introduce key methods for achieving optimality using this operator-based framework.

3.1 Rephrasing of the Banach contraction principle

Let $(X, ||\cdot||)$ be a normed space. A mapping $\mathcal{T} : X \to X$ is called a γ-contraction mapping if there exists a constant $\gamma \in [0,1)$ such that for any $x_1, x_2 \in X$:

$$||\mathcal{T}x_1 - \mathcal{T}x_2|| \leq \gamma \cdot ||x_1 - x_2||.$$

This inequality implies that the operator $\mathcal{T}$ shrinks distances between points by a factor of at least γ. Furthermore, if a sequence $\{x_n\} \subset X$ converges in norm to some $x \in X$, i.e.,

$$x_n \underset{||\cdot||}{\longrightarrow} x,$$

then the sequence $\{\mathcal{T}x_n\}$ will also converge to $\mathcal{T}x$. That is:

$$\mathcal{T}x_n \underset{||\cdot||}{\longrightarrow} \mathcal{T}x.$$

An element $x^* \in X$ is called a fixed point of $\mathcal{T}$ if:

$$\mathcal{T}x^* = x^*.$$

This fixed point is particularly important, as it represents the point where repeated applications of the operator leave the system unchanged, which aligns with the concept of reaching an optimal solution in reinforcement learning.

Proposition 3.2 (Refined Banach Contraction Principle)**.** *Let $(X, ||\cdot||)$ be a Banach space (a complete normed space), and let $\mathcal{T}: X \to X$ be a γ-contraction mapping. Then the following properties hold:*

1. *$\mathcal{T}$ has a unique fixed point $x^* \in X$.*
2. *For any initial point $x_0 \in X$, the sequence defined by the iterative process $x_{n+1} = \mathcal{T}x_n$ converges to x^* geometrically. Specifically, for all $n \geq 0$, we have:*

$$||x_n - x^*|| \leq \gamma^n ||x_0 - x^*||.$$

This guarantees that the convergence rate is proportional to the contraction factor γ, meaning that the distance between the iterates and the fixed point decreases exponentially.

Proof. The existence and uniqueness of the fixed point follow directly from the Banach fixed-point theorem. To prove geometric convergence, we observe that since $\mathcal{T}$ is a contraction mapping, we have for all $n \geq 0$:

$$||x_{n+1} - x^*|| = ||\mathcal{T}x_n - \mathcal{T}x^*|| \leq \gamma \cdot ||x_n - x^*||.$$

Applying this inequality recursively, we obtain:

$$||x_n - x^*|| \leq \gamma^n ||x_0 - x^*||,$$

which tends to zero as $n \to \infty$, implying that x_n converges to x^* geometrically. □

3.3 Bellman Optimality Operators

In reinforcement learning, operators are mappings in function spaces, and they provide a systematic approach to solving for the optimal value functions and policies. Let $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$ be a Markov Decision Process (MDP), where:

- $\mathcal{S}$ is the state space,
- $\mathcal{A}$ is the action space,
- $p(s'|s, a)$ is the transition probability,
- $r(s, a, s')$ is the reward function, and
- $\gamma \in [0, 1)$ is the discount factor.

Let $\mathcal{V}$ be the space of bounded real-valued functions over $\mathcal{S}$, representing state-value functions, and let $\mathcal{Q}$ be the space of bounded real-valued functions over $\mathcal{S} \times \mathcal{A}$, representing action-value functions. We define the following operators:

- $\mathcal{T}_v^* : \mathcal{V} \to \mathcal{V}$: the Bellman Optimality Operator for state-value functions,
- $\mathcal{T}_Q^* : \mathcal{Q} \to \mathcal{Q}$: the Bellman Optimality Operator for action-value functions.

Definition 3.4. *The Bellman Optimality equation for the state-value function is given by:*

$$v_*(s) = \max_a \left(r(s, a) + \gamma \cdot \sum_{s'} p(s'|s, a) \cdot v_*(s') \right).$$

The Bellman Optimality Operator for state-value functions, denoted by $\mathcal{T}_v^*$, is defined as:

$$(\mathcal{T}_v^* f)(s) \equiv \max_a \left[r(s, a) + \gamma \cdot \sum_{s'} p(s'|s, a) \cdot f(s') \right], \quad \forall f \in \mathcal{V}. \tag{3.1}$$

Properties of the Bellman Optimality Operator The Bellman Optimality Operator $\mathcal{T}_v^*$ has the following key properties [2]:

- **Contraction**: $\mathcal{T}_v^*$ is a γ-contraction, meaning:

$$\|\mathcal{T}_v^* u - \mathcal{T}_v^* v\|_\infty \leq \gamma \cdot \|u - v\|_\infty, \quad \forall u, v \in \mathcal{V}.$$

- **Monotonicity**: $\mathcal{T}_v^*$ is monotonic, i.e.,

$$u \leq v \implies \mathcal{T}_v^* u \leq \mathcal{T}_v^* v, \quad \forall u, v \in \mathcal{V}.$$

Proof. Instead of proving this, we are going to sketch a proof for the action-value expectation operator for coherence with the remainder of this document, but the proofs are almost the same and use the same arguments (see the proof of Proposition 3.6) □

By the Banach Contraction Principle, we can conclude:

- The operator $\mathcal{T}_v^*$ has a unique fixed point $f^* \in \mathcal{V}$.
- For any starting point (function) f_0, the sequence defined by $f_{n+1} = \mathcal{T}_v^* f_n$ converges to f^*, which is then the optimal value function by monotonicity.

The Bellman Optimality Operator $\mathcal{T}_v^*$, as defined, allows us to compute the optimal value function, which corresponds to the optimal policy π^*.

Definition 3.5. *Similarly, the Bellman Optimality equation for action-value functions is given by:*

$$q_*(s, a) = r(s, a) + \gamma \cdot \sum_{s'} p(s'|s, a) \cdot \max_{a'} q_*(s', a').$$

The Bellman Optimality Operator for action-value functions, denoted by $\mathcal{T}_Q^*$, is then defined as:

$$(\mathcal{T}_Q^* f)(s, a) \equiv r(s, a) + \gamma \cdot \sum_{s'} p(s'|s, a) \cdot \max_{a'} f(s', a'). \tag{3.2}$$

The contraction and monotonicity properties that hold for $\mathcal{T}_v$ also apply to $\mathcal{T}_Q$ using similar arguments. These properties extend to the Expectation Operator, whether it relates to the action-value function or the state-value function. Below is the Expectation Operator for the action-value function, which will be used later in this work :

$$\left(\mathcal{T}_Q^\pi f\right)(s, a) = r(s, a) + \gamma \cdot \sum_{s'} p(s'|s, a) \left(\sum_{a'} \pi(a'|s') \cdot f(s', a') \right) \tag{3.3}$$

Proposition 3.6. *We claim that the operator $\mathcal{T}_Q^\pi$, as defined in Equation 3.3, is:*

1. *A γ-contraction mapping.*
2. *A monotonic mapping.*

Proof. Let us consider two functions u and v in the space of action-value functions $\mathcal{Q}$.

1. **Contraction mapping:**

   Let us compute the absolute value of the difference between the transformations of u and v under $\mathcal{T}_Q^\pi$:

$$\begin{aligned}
\left|\mathcal{T}_Q^\pi u(s, a) - \mathcal{T}_Q^\pi v(s, a)\right| &= \gamma \cdot \left| \sum_{s'} p(s'|s, a) \left(\sum_{a'} \pi(a'|s') u(s', a') - \sum_{a'} \pi(a'|s') v(s', a') \right) \right| \\
&\leq \gamma \cdot \max_{s', a'} |u(s', a') - v(s', a')| \\
\Rightarrow \max_{s,a} \left|\mathcal{T}_Q^\pi u(s, a) - \mathcal{T}_Q^\pi v(s, a)\right| &\leq \gamma \cdot \max_{s,a} |u(s, a) - v(s, a)| \\
\Rightarrow \|\mathcal{T}_Q^\pi u - \mathcal{T}_Q^\pi v\|_\infty &\leq \gamma \cdot \|u - v\|_\infty
\end{aligned} \tag{3.4}$$

   This proves that $\mathcal{T}_Q^\pi$ is a contraction mapping with a factor of γ.

2. **Monotonicity:**
   Let us assume that $u(s,a) \le v(s,a)$ for all (s,a). Then:

$$\begin{aligned}\mathcal{T}_Q^\pi u(s,a) - \mathcal{T}_Q^\pi v(s,a) &= \gamma \cdot \sum_{s'} p(s'|s,a)\left(\mathbb{E}_{a'|s'} u(s',a') - \mathbb{E}_{a'|s'} v(s',a')\right) \\ \Rightarrow \mathcal{T}_Q^\pi u(s,a) - \mathcal{T}_Q^\pi v(s,a) &\le 0 \\ \Rightarrow \mathcal{T}_Q^\pi u(s,a) &\le \mathcal{T}_Q^\pi v(s,a)\end{aligned} \tag{3.5}$$

   This proves that $\mathcal{T}_Q^\pi$ is a monotonic mapping.

$\square$

Therefore, $\mathcal{T}_Q^\pi$, as defined, allows us to compute the action-value function associated with the policy π, and the solution is unique.

3.7 Policy Evaluation and Iteration in Operators Setting

For simplicity, we will now present the components of the policy iteration algorithm using these operators. Despite its simplicity, this algorithm is the core of value-based methods.

1. **Policy Evaluation**: Given a fixed policy π, we iteratively update the value function v by applying the Bellman operator for the policy π:

$$v_{k+1} \leftarrow \mathcal{T}^\pi v_k.$$

   By the Banach Fixed-Point Principle, this sequence converges to the value function v_π as $k \to \infty$.

2. **Policy Iteration**: Start with an initial policy π_0 and alternate between two steps:
   - **Policy Evaluation**: Just as defined above. Compute the value function for the current policy:

$$v_{k+1} \leftarrow \mathcal{T}^{\pi_i} v_k.$$

   - **Policy Improvement**: Update the policy by choosing actions that maximize the value:

$$\pi_{i+1}(s) = \operatorname*{argmax}_{a \in \mathcal{A}(s)} q_{\pi_i}(s,a).$$

   This iterative process converges to the optimal policy π^* and the corresponding value function v_{π^*} as $i \to \infty$.

4 Alternatives Bellman Operator

In this section, we discuss the limitations of the classical Bellman operators as introduced in the previous sections. We explore the inherent trade-off between achieving optimality and maintaining efficiency, and we present experimental results that highlight the need for refinements, either to the Bellman operators themselves or to the associated value functions. As we have seen, value-based reinforcement learning algorithms solve decision-making problems through the iterative application of a convergent operator, which recursively improves an initial value function.

While the classical Bellman operator has been widely used in reinforcement learning, numerous studies have proposed alternatives to address its limitations [3, 5, 7, 8, 14, 22]. Among these alternatives, two approaches have shown particularly promising results: the **consistent Bellman Operator** [7] and the family of **Robust Stochastic Operators** [14]. These operators offer improvements in different aspects compared to the standard Bellman operator, which motivates us to examine them more closely. The first alternative, the **consistent Bellman Operator**, introduces a modification that better aligns the learned value function with the underlying policy, thus improving the performance of value-based methods in practice. The second alternative, the **Robust Stochastic Operators**, generalizes the Bellman operator to provide robustness against uncertainty and variability in the environment, offering enhanced stability during learning. Although these operators, especially the Robust Stochastic Operator, show significant promise, we suggest a non-stochastic and refined version that could potentially improve its performance. This refinement aims to increase stability without relying on the stochastic nature of the operator, making it more broadly applicable and effective. In the following sections, we will delve deeper into the mathematical formulation of these alternative operators and present the results of our experiments. These experiments demonstrate that both the consistent Bellman operator and our proposed refinement of the Robust Stochastic Operator yield improvements in stability and convergence speed. These findings suggest that further exploration of these alternative operators, and possibly others, is a fruitful direction for enhancing the performance of reinforcement learning algorithms.

4.1 Motivation [7, 14]

The motivation for exploring new formulations of the Bellman Operator is clear. While Q-learning and other value-based methods have been successfully applied in reinforcement learning (RL) to find optimal policies, there remains a constant need to improve their convergence speed, accuracy, and robustness. A critical factor in this regard is the presence of intrinsic approximation errors, which arise frequently in real-world scenarios. For instance, when using a discrete Markov Decision Process (MDP) to approximate a continuous system, the value function obtained through the Bellman operator may not accurately represent the value of stationary policies. More importantly, when the differences between the optimal state-action value function and suboptimal value functions are small, these minor discrepancies can lead to errors in identifying the truly optimal actions. This issue becomes even more pronounced in environments where approximations are necessary, as is often the case when continuous-time systems are discretized. In such situations, the Bellman operator may not generalize well, and errors in value estimation can propagate through the learning process, leading to suboptimal performance. Thus, while classical Bellman Operators perform well in perfectly discrete settings, we must refine them to be more generalizable to practical, real-world problems where intrinsic errors are unavoidable. For these types of problems, which often arise from discretizing continuous systems, there is always an inherent approximation error. To address this, it is essential to integrate a corrector mechanism into the operator to account for these discrepancies and improve its performance across different settings. In the following, we will explore the effectiveness of the two alternative operators mentioned earlier: the consistent Bellman Operator and our modified version of the Robust Stochastic Operator. By analyzing their performance, we aim to gain further insights into how these refinements can help in the general approach to determining optimal policies in reinforcement learning.

4.2 The Consistent Bellman Operator

The consistent Bellman operator was mentioned for the first time in [7] for the action-value function. It's defined as follows:

$$\mathcal{T}_c f(s,a) = r(s,a) + \gamma \cdot \sum_{s'} p(s'|s,a) \cdot \left[\mathbb{I}_{\{s \neq s'\}} \max_{a'} f(s',a') + \mathbb{I}_{\{s=s'\}} f(s,a)\right], \quad \text{with } f \in \mathcal{Q}, \tag{4.1}$$

where $\mathbb{I}$ denotes the indicator function. And from section 3.3 remember the $\mathcal{Q}$ is the space of action-value functions (space of bounded real-valued functions over $\mathcal{S} \times \mathcal{A}$).

We claim that the consistent Bellman operator given by Equation 4.1 satisfies the following important properties:

1. $\mathcal{T}_c$ is a contraction mapping.
2. $\mathcal{T}_c$ is monotonic.

Proof. To streamline the proof, we first make some refinements:

- For simplicity, we rewrite the expectation over the transition probabilities:

$$\sum_{s'} p(s'|s,a) f \quad \text{as} \quad \mathbb{E}_{\mathbb{P}}(f).$$

- We also rewrite the action-value function as follows:

$$f_s(s',a') = \begin{cases} f(s',a'), & \text{if } s \neq s', \\ f(s,a), & \text{if } s = s', \end{cases} \quad \text{for } f \in \mathcal{Q}. \tag{4.2}$$

With those refinements, we can rewrite the consistent Bellman operator from Equation 4.1 like this:

$$\mathcal{T}_c f(s,a) = r(s,a) + \gamma \cdot \mathbb{E}_{\mathbb{P}}\left[\max_{a'} f_s(s',a')\right], \quad \text{with } f \in \mathcal{Q}. \tag{4.3}$$

Now, we proceed with the proofs:

1. **Contraction:** Let $u, v \in \mathcal{Q}$. We need to show that $\mathcal{T}_c$ is a contraction:

$$\begin{aligned} |\mathcal{T}_c u(s,a) - \mathcal{T}_c v(s,a)| &= \left|\gamma \cdot \mathbb{E}_{\mathbb{P}}\left(\max_{a'} u_s(s',a')\right) - \gamma \cdot \mathbb{E}_{\mathbb{P}}\left(\max_{a'} v_s(s',a')\right)\right| \\ &\leq \gamma \cdot \left|\max_{a'} \mathbb{E}_{\mathbb{P}}\Big(u_s(s',a') - v_s(s',a')\Big)\right| \\ &\leq \gamma \cdot \max_{s',a'} |u_s(s',a') - v_s(s',a')| \\ &= \gamma \cdot \max_{s,a} |u_s(s,a) - v_s(s,a)| \\ &= \gamma \cdot ||u_s(s,a) - v_s(s,a)||_\infty, \end{aligned}$$

which shows that:

$$||\mathcal{T}_c u - \mathcal{T}_c v||_\infty \leq \gamma \cdot ||u - v||_\infty.$$

Hence, $\mathcal{T}_c$ is a contraction.

2. **Monotonicity:** Consider two state-action value functions u and v such that $u(s,a) \leq v(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. From this, we have $u_s(s,a) \leq v_s(s,a)$. Now, we show that:

$$\begin{aligned}\mathcal{T}_c u(s,a) - \mathcal{T}_c v(s,a) &\leq \gamma \cdot \max_{a'} \left(\mathbb{E}_{\mathbb{P}}\left[u_s(s',a') - v_s(s',a')\right]\right) \\ &\leq 0,\end{aligned}$$

which implies:

$$\mathcal{T}_c u(s,a) \leq \mathcal{T}_c v(s,a), \quad \forall u, v \in \mathcal{Q}.$$

Thus, $\mathcal{T}_c$ is monotonic.

□

From this proof, we can conclude that $\mathcal{T}_c$ has a unique fixed point, and this fixed point corresponds to the optimal value function associated to the consistent Bellman equation instead of the classical one.
The key question that remains is how this fixed point relates to the one obtained using the classical Bellman operator. While we are confident that both operators lead to unique fixed point, it is not immediately clear how they compare. At this stage, without the appropriate mathematical tools to analyze the relationship between the two fixed points, we will rely on empirical results to observe how the consistent Bellman operator behaves in practice compared to the classical Bellman operator.

4.3 Modified Robust Stochastic Operator

Our proposed operator takes inspiration from both [14] and [7], with a greater emphasis on the approach taken in the first. Our refinement is more general (an expectation operator, with refined concepts) and differs from the Robust Stochastic Operator suggested in [14], showing the irrelevance of stochasticity if the concepts are defined accordingly.

Naturally, we can express $v_\pi(s)$ as:

$$v_\pi(s) = \sum_a \pi(a|s) \cdot q_\pi(s,a),$$

and the difference between the action value function and the state value funCtion, known as **advantage learning**, is given by:

$$A(s,a) = q_\pi(s,a) - v_\pi(s),$$

which provides insights into the quality of the policy, as well as the value function. During the learning process of the optimal policy, as long as the chosen algorithm improves the policy, the quantity $|A(s,a)|$ should decrease, indicating better action selection.

We propose modifying the **Bellman Expectation Operator for the action-value function**, as defined in Equation 3.3, by directly integrating the concept of **advantage learning** into the operator, instead of applying it later in the learning process, as is sometimes implicitly done in implementations of policy gradient methods [10]. Let this new operator be denoted as $\mathcal{T}_a$, defined for all $f \in \mathcal{Q}$ as:

$$\left(\mathcal{T}_a f\right)(s,a) = r(s,a) + \gamma \cdot \sum_{s'} p(s'|s,a) \left(\sum_{a'} \pi(a'|s') \cdot f(s',a')\right) + \beta \cdot \underbrace{\left[f(s,a) - \sum_a \pi(a|s) f(s,a)\right]}_{\text{advantage learning}}. \tag{4.4}$$

We will now examine the properties of the operator defined by Equation 4.4. The coefficient β is currently any real number, but we will define it appropriately by the end of this theoretical discussion.

Proposition 4.4. *The operator $\mathcal{T}_a$, as defined in Equation 4.4, is not a contraction mapping.*

Proof. Let $u(s,a) \equiv u$ and $v(s,a) \equiv v$ be two elements of $\mathcal{Q}$:

$$\begin{aligned}
|\mathcal{T}_a u(s,a) - \mathcal{T}_a v(s,a)| &= \Bigg|\gamma \cdot \mathbb{E}_{\mathbb{P}}\left(\sum_{a'} \pi(a'|s')\left(u(s',a') - v(s',a')\right)\right) \\
&\qquad + \beta \cdot \left([u(s,a) - v(s,a)] - \sum_{a} \pi(a|s)\left[u(s,a) - v(s,a)\right]\right)\Bigg| \\
&> \beta \cdot |u(s,a) - v(s,a)| \quad \text{for certain } (u,v) \text{ and } \beta \\
\Rightarrow ||\mathcal{T}_a u - \mathcal{T}_a v||_\infty &> \beta \cdot ||u - v||_\infty.
\end{aligned} \tag{4.5}$$

This simplified proof shows that the operator $\mathcal{T}_a$ is usually not a contraction mapping. □

Since $\mathcal{T}_a$ is not a contraction, we cannot guarantee convergence. However, despite this, we can still examine the operator's behavior. Drawing from [14] and [7], we introduce two key properties by which we will qualify a value-based operator as **well-behaving operator**: *optimality preservation* and *gap increasing.*

Note: In violation of our notation conventions, for simplification, below we are going to write value functions using capital letters, which will also help distinguishing operators written as indices.

Definition 4.5 (Optimality Preservation)**.** *Let $\mathcal{T}_a$ be an alternative operator to the Bellman operator $\mathcal{T}_b$. We say that $\mathcal{T}_a$ preserves optimality if:*

$$Q_{k,\mathcal{T}_b} < V_{k,\mathcal{T}_b} \implies Q_{k,\mathcal{T}_a} < V_{k,\mathcal{T}_a} \quad \text{as } k \to \infty,$$

where $Q_{k,\mathcal{T}}$ represents the action-value function at iteration k using operator $\mathcal{T}$, and $V_{k,\mathcal{T}}$ is the associated state-value function.

Definition 4.6 (Gap Increasing)**.** *Using the same notation as before, we say that an alternative operator $\mathcal{T}_a$ induces the gap increasing property if, for every state $s \in \mathcal{S}$ and each feasible action $a \in \mathcal{A}(s)$:*

$$\left|\lim_{k\to\infty} (Q_{k,\mathcal{T}_b} - V_{k,\mathcal{T}_b})\right| \leq \left|\lim_{k\to\infty} (Q_{k,\mathcal{T}_a} - V_{k,\mathcal{T}_a})\right|.$$

The **optimality preservation** property indicates how well the operator preserves the search for the optimal fixed point, while the **gap increasing** property reflects the ability of the operator to distinguish between the values of suboptimal and optimal actions.

Proposition 4.7. *We claim that the operator defined in Equation 4.4 is a well-behaving operator, even though it is not a contraction.*

Proof. We prove these properties in sequence:

1. **Optimality Preservation**: From Definition 4.5, we assume that $Q_{k,\mathcal{T}_b} < V_{k,\mathcal{T}_b}$, and aim to show $Q_{k,\mathcal{T}_a} < V_{k,\mathcal{T}_a}$ as $k \to \infty$. Note that for better flow, when applying the Bellman operator to a quantity, we will sometimes write $\text{Bell}(Q_{k-1})$ instead of $\mathcal{T}_b Q_{k-1}$. We start with:

$$\begin{aligned}
Q_{k,\mathcal{T}_b} < V_{k,\mathcal{T}_b} &\Rightarrow \mathcal{T}_b Q_{k-1} < V_{k,\mathcal{T}_b} \\
&\Rightarrow Bell(Q_{k-1}) < \sum_a \pi(a|s) Bell(Q_{k-1}).
\end{aligned}$$

   Without loss of generality, let $K = \min_a \Big(Q_{k-1}(s,a) - V_{k-1}(s)\Big)$. After including the advantage learning term, we find:

$$\begin{aligned}
Bell(Q_{k-1}) + \beta \cdot K &< \sum_a \pi(a|s)\left(Bell(Q_{k-1}) + \beta \cdot K\right) \\
&\Rightarrow Q_{k,\mathcal{T}_a} < V_{k,\mathcal{T}_a}, \quad \text{as } k \to \infty.
\end{aligned} \tag{4.6}$$

2. **Gap Increasing**: Using Definition 4.6, we show that:

$$\left|\lim_{k\to\infty} (Q_{k,\mathcal{T}_b} - V_{k,\mathcal{T}_b})\right| \leq \left|\lim_{k\to\infty} (Q_{k,\mathcal{T}_a} - V_{k,\mathcal{T}_a})\right|.$$

For optimal actions, this inequality holds because $Q_{k,\mathcal{T}_b}$ and $V_{k,\mathcal{T}_b}$ converge to the same value, resulting in:

$$\lim_{k\to\infty} (Q_{k,\mathcal{T}_b} - V_{k,\mathcal{T}_b}) = 0.$$

Thus, for any state s and action a, the operator $\mathcal{T}_a$ maintains or increases the gap compared to $\mathcal{T}_b$, satisfying the gap increasing property.

□

The operator defined in Equation 4.4 is a **well-behaving operator**, even though we assume convergence under specific conditions for β.

Now, we discuss the possibility of finding a fixed point for this operator. Before doing so, we will first examine the continuity and boundedness of the operator, as these are prerequisites for analyzing fixed points.

Proposition 4.8. *Let the reward function $r(s,a)$ be bounded and continuous. The operator $\mathcal{T}_a$, as defined in Equation 4.4, is also* ***bounded*** *and* ***continuous****.*

Proof. We can break the operator into three components:

$$(\mathcal{T}_a f)(s,a) = \underbrace{r(s,a)}_{\text{Part 1}} + \underbrace{\gamma \cdot \sum_{s'} p(s'|s,a) \left(\sum_{a'} \pi(a'|s') \cdot f(s',a') \right)}_{\text{Part 2}} + \underbrace{\beta \cdot \left[f(s,a) - \sum_{a} \pi(a|s) f(s,a) \right]}_{\text{Part 3}}.$$

1. **Part 1**: The reward function is bounded and continuous by assumption.
2. **Part 2**: This part is a contraction mapping, hence it is both bounded and continuous.
3. **Part 3**: If f is bounded and continuous, then the term $f(s,a) - \sum_a \pi(a|s) f(s,a)$ is also bounded and continuous, provided β has appropriate properties.

Thus, $\mathcal{T}_a$ is continuous and bounded, assuming the behavior of β ensures convergence. □

Given the boundedness and continuity of $\mathcal{T}_a$, we expect that the operator will not diverge if β is chosen carefully. The choice of β must balance between improving speed and optimality while maintaining proximity to the classical Bellman Operator. We therefore propose conditions for β to ensure convergence in a family of operators based on $\mathcal{T}_a$, where β varies across iterations j with i refering to the index of a specific operator within this family. For convergence, the sequence $\beta_{i,j}$ must satisfy the following two conditions:

$$\sum_{j=1}^{\infty} \beta_{i,j} < \infty \,, \text{and } \{\beta_{i,j}\} \to 0, \text{ as } j \to \infty. \tag{4.7}$$

The first condition ensures that the total sum of $\beta_{i,j}$ across iterations is finite, and the second condition guarantees that $\beta_{i,j}$ approaches zero as the iteration index j grows. Together, these conditions allow for the construction of a sequence of operators that converges to the classical Bellman operator.

5 Implementations and analysis

To prove the effectiveness of the suggested *Modified Robust Stochastic Operator*, we have conducted experiments on three groups of classical problems in reinforcement learning using the **Q-Learning** algorithm in **OpenAI Gymnasium environments**. The implementation of Q-Learning we used is inspired by [21], and our own implementation with all the parameters is in our Github Repository [1].

Here are the necessary details on the environments we used:

5.1 Environments and results

1. Mountain Car environment: The theory about this environment is presented in [15]. The state vector is 2-dimensional, continuous with a total of three possible actions. As long as the goal is not yet reached, depending on the action, a negative reward is given to the agent until it reaches the goal. Following [14], we have discretized the state space into a 40×40 grid, but differently, we did 10,000 training steps, with 10,000 episodes each. The following Figure 1 shows the averages across episodes.

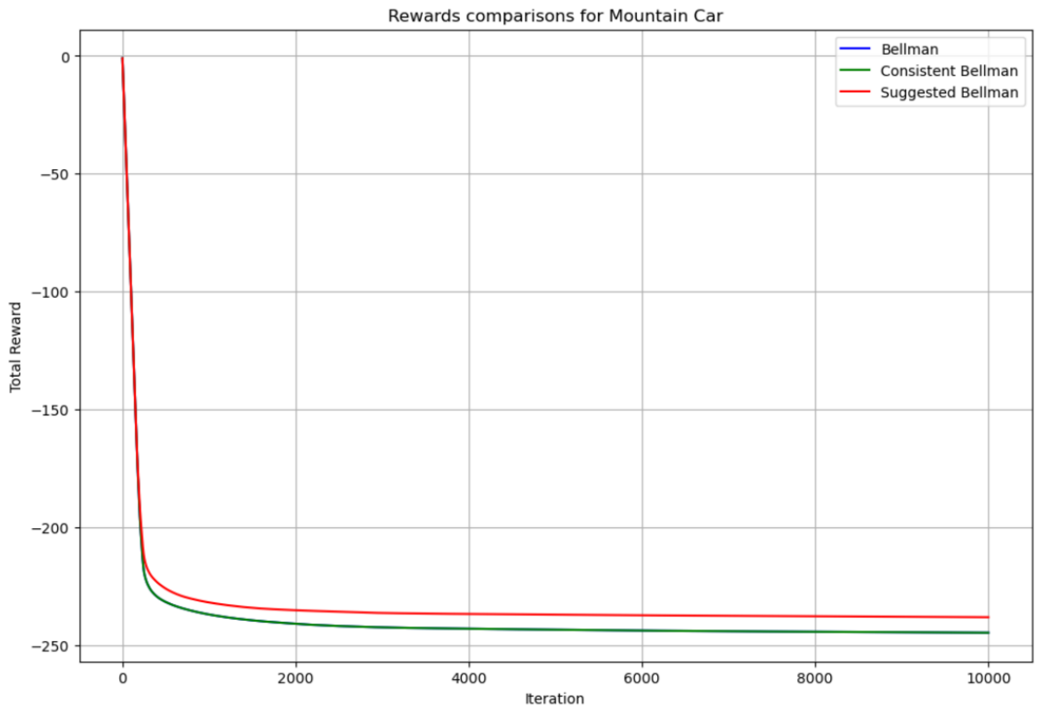

Figure 1: Convergence comparison in the MountainCar environment.

2. Cart Pole environment: The theory about Cart Pole is presented in [6]. The state vector is 4-dimensional and continuous with a total of two possible actions. The aim is to keep the pole upright for as long as possible, with a reward of +1 for each step up to the failure, including the final step. So, the reward is positive at the end. Here, we have discretized the state space into a $150 \times 150 \times 150 \times 150$ grid and again we did 10,000 training steps, with 10,000 episodes each. The following Figure 2 shows the averages across episodes.

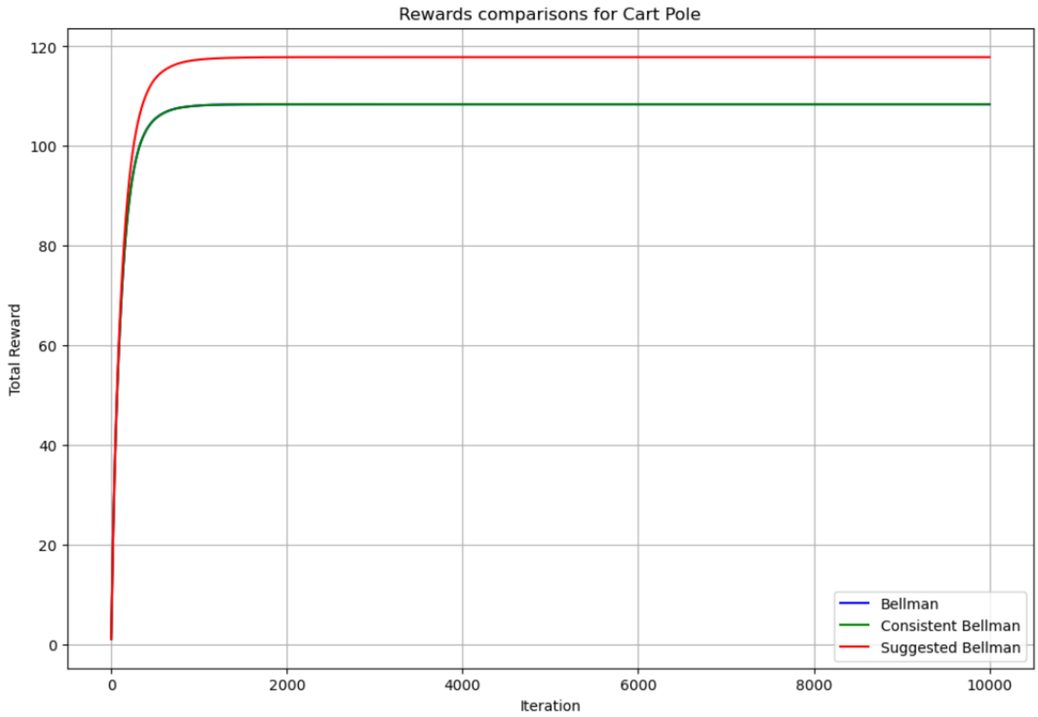

Figure 2: Performance in the CartPole environment.

3. Acrobot environment: The theory about Acrobot is presented in [18]. The state vector is 6-dimensional and continuous with a total of three possible actions. The goal is to have the free end of the Acrobot reach the target height (represented by a horizontal line) in as few steps as possible, with each step not reaching the target being rewarded with -1. So, the reward is negative again as for Mountain Car. Here, we have discretized the state space into a $30 \times 30 \times 30 \times 30 \times 30 \times 30$ grid, due to the limitations in memory allocation of the computer we were using, and again we did 10,000 training steps, with 10,000 episodes each. The following Figure 3 shows the averages across episodes.

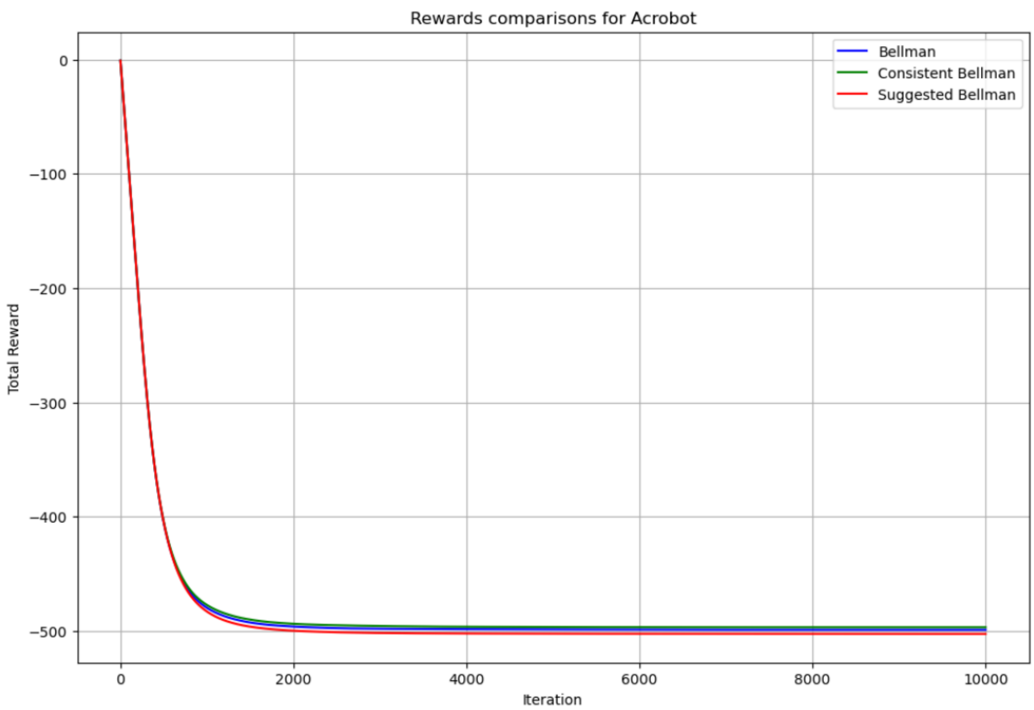

Figure 3: Comparison of learning curves in the Acrobot environment.

5.2 Interpretation and discussion

1. Mountain Car: From the problem presented above, we know that the reward for mountain car can be negative because it depends on how long the agent takes to reach the goal or to be stopped. Now, looking at the Figure 1, we can directly see how the average reward is better using the *Modified Robust Stochastic Operator* than using the Bellman Operator. Also, the classical and the consistent operators perform exactly the same for this specific case. So, overall, the suggested Operator finishes the episodes with a higher reward than for the other two operators (the classical and the consistent).

2. Cart Pole: For Cart Pole, we can see in Figure 2 that the higher reward is reached again while using the Modified Bellman (suggested Bellman) compared to the other operators. And the difference in terms of performance is really clear.

3. Acrobot: Acrobot was more challenging and as we can see in Figure 3, the results using those 3 operators are about the same. We think this is due to the fact that we were not able to make a finer discretization for the Acrobot environment. In fact, this was due to the memory limitations. So, this experiment needs further investigations to establish clearly what is the great attainable difference between those operators.

So, generally we can say that the Consistent Bellman Operator gives about the same result as the classical one, but our modified version of the Robust Stochastic Operator gives most of the time better results compared to the other previous two operators.

6 Conclusion

This work delves into the topological foundations of Reinforcement Learning, providing a rigorous mathematical framework for studying its core concepts. By focusing on the relationship between topology and Reinforcement Learning principles, we aim to pave the way for breakthroughs that can facilitate the contributions of mathematicians toward improving Reinforcement Learning algorithms. We introduced the Consistent Bellman Operator as an alternative to the classical Bellman Operator, demonstrating that it retains critical properties, such as the uniqueness of the optimal fixed point. However, this raised important questions about the relationship between the fixed points of these operators and how they might affect algorithm performance in practice. Additionally, we presented a deterministic variation of the Robust Stochastic Operator, highlighting that stochasticity is not essential for achieving superior results compared to the classical Bellman Operator. Our implementation, using Python and OpenAI Gymnasium environments, showed that the proposed operator outperforms classical methods across a variety of tasks, validating its practical effectiveness. Looking ahead, future research could expand upon our findings by further investigating the state space, action space, and policy space, as well as optimizing the efficiency of the proposed operator. We conjecture that any monotonic contraction mapping incorporating a notion of policy could serve as a viable operator in Reinforcement Learning, provided that the definitions of the associated value functions are adequately refined for this context. This work aims to empower researchers, even those who may not be familiar with the technicalities of the field, to engage with Reinforcement Learning through a mathematically rigorous lens, making the fundamental concepts more accessible and understandable.

Acknowledgments

We would like to express our heartfelt gratitude to all the esteemed individuals at the African Institute for Mathematical Sciences (AIMS), both in Cameroon and globally, who continue to make it a journey that is both beautiful and transformative. Especially **Professor Neil Turok**, **Professor Mama Foupouagnigni**, and **Dr. Daniel Duviol**. Their efforts have been instrumental in making AIMS a vibrant and inspiring institution.

References

[1] . https://github.com/DavidKrame/rl-essay-aims-cameroon.

[2] DeepMind x UCL — Deep Learning Lecture Series 2021, 2021. Accessed on 13/03/2024.

[3] Kavosh Asadi and Michael L Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pages 243–252. PMLR, 2017.

[4] Kavosh Asadi, Dipendra Misra, and Michael L. Littman. Lipschitz continuity in model-based reinforcement learning. *CoRR*, abs/1804.07193, 2018.

[5] Mohammad Gheshlaghi Azar, Remi Munos, Mohammad Ghavamzadeh, and Hilbert Kappen. Speedy q-learning. In *Advances in neural information processing systems*, 2011.

[6] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

[7] Marc G Bellemare, Georg Ostrovski, Arthur Guez, Philip Thomas, and Rémi Munos. Increasing the action gap: New operators for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[8] Dimitri P Bertsekas and Huizhen Yu. Q-learning and enhanced policy iteration in discounted dynamic programming. *Mathematics of Operations Research*, 37(1):66–94, 2012.

[9] Robert Dadashi, Adrien Ali Taïga, Nicolas Le Roux, Dale Schuurmans, and Marc G. Bellemare. The value function polytope in reinforcement learning. *CoRR*, abs/1901.11524, 2019.

[10] Laura Graesser and Wah Loon Keng. *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional, 2019.

[11] David Krame Kadurha. Topological foundations of reinforcement learning. *arXiv preprint arXiv:2410.03706*, 2024.

[12] Charline Le Lan, Marc G. Bellemare, and Pablo Samuel Castro. Metrics and continuity in reinforcement learning. *CoRR*, abs/2102.01514, 2021.

[13] A Lazaric. Markov decision processes and dynamic programming, 2013.

[14] Yingdong Lu, Mark S Squillante, and Chai Wah Wu. A general family of robust stochastic operators for reinforcement learning. *arXiv preprint arXiv:1805.08122*, 2018.

[15] Andrew William Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, Computer Laboratory, 1990.

[16] Olivier Sigaud and Olivier Buffet. *Markov decision processes in artificial intelligence*. John Wiley & Sons, 2013.

[17] Zhao Song and Wen Sun. Efficient model-free reinforcement learning in metric spaces. *CoRR*, abs/1905.00475, 2019.

[18] Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, 8, 1995.

[19] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[20] Anita Tomar and M.C Joshi. *Fixed Point Theory and its Applications to Real World Problems*. Nova Science Publishers, New York, 2021.

[21] vmayoral. Basic Reinforcement Learning. https://github.com/vmayoral/basic_reinforcement_learning/tree/master, 2024.

[22] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.