Extended version of paper published in Proc. ECAI 2025, pages 3363--3370, IOS Press, 2025.

Efficient Computation of Blackwell Optimal Policies using Rational Functions

Dibyangshu Mukherjee[a] **and Shivaram Kalyanakrishnan**[a]

[a]IIT Bombay, Mumbai, India
{dbnshu, shivaram}@cse.iitb.ac.in

Abstract. Markov Decision Problems (MDPs) provide a foundational framework for modelling sequential decision-making across diverse domains, guided by optimality criteria such as discounted and average rewards. However, these criteria have inherent limitations: discounted optimality may overly prioritise short-term rewards, while average optimality relies on strong structural assumptions. Blackwell optimality addresses these challenges, offering a robust and comprehensive criterion that ensures optimality under both discounted and average reward frameworks. Despite its theoretical appeal, existing algorithms for computing Blackwell Optimal (BO) policies are computationally expensive or hard to implement.

In this paper we describe procedures for computing BO policies using an ordering of rational functions in the vicinity of 1. We adapt state-of-the-art algorithms for deterministic and general MDPs, replacing numerical evaluations with symbolic operations on rational functions to derive bounds independent of bit complexity. For deterministic MDPs, we give the first strongly polynomial-time algorithms for computing BO policies, and for general MDPs we obtain the first subexponential-time algorithm. We further generalise several policy iteration algorithms, extending the best known upper bounds from the discounted to the Blackwell criterion.

1 Introduction

Markov Decision Problems (MDPs) are a widely used mathematical framework for modelling sequential decision-making problems. They form the backbone of reinforcement learning where agents learn to take decisions by interacting with an environment. Applications span diverse fields, including treatment planning in healthcare [2], automated control systems [8], robotics [25], game-solving [42], and financial portfolio management [44]. Their flexibility and rigorous foundation make MDPs a central tool in operations research, artificial intelligence, and economics.

An MDP is characterised by a tuple (S, A, T, R), where S is the set of states and A is the set of actions. In this paper we assume S and A are finite with sizes n and k respectively. When an agent takes an action $a \in A$ from state $s \in S$, it transitions to a new state s' with a probability $T(s, a, s')$ and receives a mean reward $R(s, a)$. The key feature of MDPs is the Markov Property, which asserts that the transition dynamics depend only on the current state and action and not on the sequence of past states.

The objective in an MDP is to determine a policy: a rule that specifies the action to take in each state. Starting from an initial state s_0, if the agent follows a policy $\pi : S \to A$, it encounters a sequence $\left(s^t, \pi(s^t), r^t\right)_{t=0}^{\infty}$. The long-term reward of the agent for a state s is determined by the chosen *optimality criterion*.

Under the **discounted reward** criterion, a policy is evaluated using a value function that quantifies the cumulative discounted sum of rewards obtained by following the policy from state s, expressed as:

$$V_\gamma^\pi(s) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \gamma^t R(s^t, a^t) \middle| s^0 = s \right], \qquad (1)$$

where the discount factor $\gamma \in [0, 1)$ is specified as part of the MDP.

The discounted framework is widely favoured [45, 26] for its mathematical simplicity, notably due to the contraction property. However, it is also subject to limitations, which we discuss later.

Under the **average reward** criterion, the value of a policy π for a state s is characterised by two components: the *gain* and the *bias*. The gain, denoted by $V_{\mathrm{g}}^\pi(s)$, represents the long-term average reward per time step under the policy π from state s. It is defined as:

$$V_{\mathrm{g}}^\pi(s) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\frac{1}{T} \sum_{t=0}^{T-1} R(s^t, a^t) \middle| s^0 = s \right].$$

The bias, denoted by $V_{\mathrm{b}}^\pi(s)$, reflects the transient behaviour of the system, capturing how the reward dynamics evolve before the steady-state is reached. It is defined for each state s as:

$$V_{\mathrm{b}}^\pi(s) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \left(R(s^t, a^t) - V_{\mathrm{g}}^\pi(s) \right) \middle| s^0 = s \right].$$

Gain and bias optimality are specific instances of a broader optimality criterion known as sensitive discount optimality [49], which focuses on the cumulative sum of rewards as the discount factor approaches 1. Each of these criteria defines an optimal policy, $\pi^\star$, such that $V^{\pi^\star}$ *dominates* (as a vector) V^π for all policies π. Here, V^π serves as a placeholder for the value corresponding to the given optimality criterion, such as V_{g}^π or V_γ^π.

MDPs can be solved, or an optimal policy found, using value iteration, policy iteration, or linear programming under both optimality criteria. These algorithms are straightforward and effective in the discounted case. However, in the average-reward case, the structure of the Markov chains induced by stationary policies plays a significant role, making it harder to design a general, simple algorithm.

Blackwell [5] introduced a refined notion of optimality, known as *Blackwell optimality*. A policy π is Blackwell Optimal (BO) if there exists a threshold $\overline{\gamma} \in (0, 1)$ such that π remains optimal under the discounted reward criterion for all discount factors $\gamma \in (\overline{\gamma}, 1)$.

By definition, any BO policy is discount-optimal for all sufficiently large discount factors, and is also average-optimal. Thus Blackwell optimality serves as a conceptual bridge between average and discounted optimality. While a BO policy is guaranteed to exist for every finite MDP [5], existing methods for computing BO policies are either inefficient or overly intricate (see sections 2 and 3). This paper proposes simple and efficient techniques for computing BO policies.

We develop a symbolic method for ordering rational functions near the point $x = 1$. This idea originates from the work of Hordijk et al. [18], who applied such orderings within the linear programming framework as part of a simplex-based method for solving MDPs over an entire range of discount factors. In contrast, we incorporate symbolic ordering directly into the dynamic programming framework. By treating the discount factor as a symbolic variable, we express value and action-value functions as rational functions and use their relative orderings to guide policy improvement. This enables us to reinterpret and extend classical algorithms—such as policy iteration—for computing BO policies. Crucially, our approach yields algorithms with provable efficiency and establishes the tightest known bounds to date for computing BO policies, independent of the input's bit representation.

1.1 Contributions

We use our ordering of rational functions to simulate the trajectory of various algorithms on an MDP with a sufficiently large discount factor. This methodology enables the following contributions:

- Post and Ye [39] showed that the Max-Gain simplex algorithm converges to the optimal policy in strongly polynomial time for deterministic MDPs, with a bound of $O(n^5k^2 \log^2 n)$ iterations.

  Madani et al. [27] extended the classical algorithm of Karp [24] to the discounted setting and achieved a bound of $O(n^2k)$ for solving deterministic MDPs.

  We generalise both of these algorithms to the Blackwell setting, preserving their respective bounds up to a polynomial factor. This yields the first strongly polynomial guarantees for computing Blackwell-optimal policies in deterministic MDPs.

- We obtain the first direct policy improvement procedure for computing BO policies that does not rely on Laurent series expansions. When combined with the Random-Facet algorithm [21, 34], this approach yields a subexponential expected bound of $\text{poly}(n, k) \cdot \exp(O(\sqrt{n \log n}))$ for general MDPs—the tightest known bound to date that is independent of the bit-size of the input.

- The switching rule used in policy iteration plays a crucial role in determining the algorithm's complexity. We analyse three switching rules that achieve the tightest known upper bounds for discounted MDPs, and generalise each to the Blackwell setting while preserving their bounds up to polynomial factors.

- For every MDP, there exists a threshold discount factor beyond which all discount-optimal policies are also Blackwell-optimal. A tight upper bound on this threshold facilitates the computation of BO policies, while a large lower bound highlights the inherent complexity of the problem and the limitations of certain algorithmic approaches discussed in section 6.

  We construct an MDP whose threshold discount factor is exponentially close to 1, thereby establishing the best-known lower bound on this threshold.

The next section introduces Blackwell optimality, providing background, motivation, and outlining the key challenges in computing BO policies. Section 3 reviews the relevant literature. Section 4 presents our rational function ordering framework, which forms the basis for the algorithms developed in Section 5. Finally, Section 6 concludes with a discussion and summary of our contributions.

2 Blackwell Optimality

The value function of a policy π, defined in (1), satisfies the recursive *Bellman equations*, which for $s \in S$ take the form:

$$V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s')V^\pi(s'). \tag{2}$$

Similarly, for $s \in S, a \in A$, the action-value function is defined as:

$$Q^\pi(s, a) = R(s, a) + \gamma \sum_{s'} T(s, a, s')V^\pi(s'). \tag{3}$$

Notation. We write V^π_γ and Q^π_γ to make the dependence on γ explicit, particularly in contexts where γ may not be fixed.

For any policy π, let P^π denote the $n \times n$ stochastic matrix with entries $P^\pi(s, s') = T(s, \pi(s), s')$. Similarly, let $\mathbf{r}^\pi$ denote the $n \times 1$ reward vector with components $R(s, \pi(s))$. The Bellman equation for the value vector can be expressed as:

$$\mathbf{v}^\pi = \mathbf{r}^\pi + \gamma P^\pi \mathbf{v}^\pi. \tag{4}$$

Solving this system using Cramer's rule, the value of a state s under policy π is given by:

$$\mathbf{v}^\pi_s = \frac{\mathbf{n}^\pi_s}{d_\pi}, \tag{5}$$

where: $d_\pi = |I - \gamma P^\pi|$ is the determinant of $I - \gamma P^\pi$, and $\mathbf{n}^\pi_s$ is the determinant of the matrix formed by replacing the s-th column of $I - \gamma P^\pi$ with $\mathbf{r}^\pi$. Similarly, the action-value function can be written in vectorised form using the reward vector $\mathbf{r}_a$ and transition matrix P_a for action a, with $\mathbf{v}^\pi$ substituted from Equation (5):

$$\mathbf{q}^\pi_a = \mathbf{r}_a + \gamma P_a \frac{\mathbf{n}^\pi}{d_\pi}. \tag{6}$$

2.1 BO Policies and Threshold Discount Factor

Definition 1. *A policy π is Blackwell-optimal if there exists $\gamma' \in [0, 1)$ such that $V^\pi_\gamma(s) \geq V^{\pi'}_\gamma(s),\ \forall s \in S, \forall \pi' \in \Pi$ and $\gamma \in [\gamma', 1)$.*

A BO policy is both discounted (as $\gamma \to 1$) and average-optimal, but the converse need not hold. See Figure 1a for an example.

Theorem 1 (Blackwell [5])**.** *Every finite MDP has at least one BO policy.*

The proof of this result becomes evident through the developments presented in this paper. See section 4.3.

Threshold discount factors play an important role in computation of BO policies. We define two existing threshold discount factors below. Let $\Pi^\star_{\text{bw}}$ denote the set of BO policies and $\Pi^\star_\gamma$ the set of discount-optimal policies with discount factor γ. The Blackwell discount factor γ_{bw}, introduced by Grand-Clément and Petrik [16], is defined as:

$$\gamma_{\text{bw}} \stackrel{\text{def}}{=} \inf \left\{ \gamma \in [0, 1) \;\middle|\; \forall \gamma' \in (\gamma, 1), \Pi^\star_{\gamma'} = \Pi^\star_{\text{bw}} \right\}.$$

That is, γ_{bw} is the smallest discount factor beyond which the set of discount-optimal policies coincides with the set of BO policies. It is guaranteed to exist in every finite MDP [16].

Mukherjee and Kalyanakrishnan [36] define a stronger threshold condition, requiring that the ordering of Q-values for every policy remains invariant beyond the threshold. Formally:

$$\gamma_{\text{Q}} \stackrel{\text{def}}{=} \sup_{\pi\in\Pi; s\in S; a,a'\in A} \left\{ \inf \left\{ \gamma \in [0,1) \;\middle|\; \forall \tau \in (\gamma, 1), \right. \right.$$
$$\left. \left. \left(Q^{\pi}_{\gamma}(s,a) > Q^{\pi}_{\gamma}(s,a') \implies Q^{\pi}_{\tau}(s,a) > Q^{\pi}_{\tau}(s,a')\right) \right\} \right\}.$$

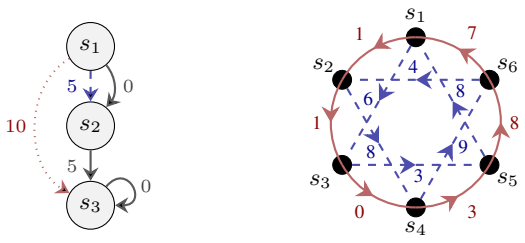
It can be shown that γ_{Q} exists for every MDP, and that $\gamma_{\text{bw}} \leq \gamma_{\text{Q}}$ [36].

2.2 The Case for BO Policies

Most Reinforcement Learning problems aim to maximise the cumulative sum of rewards for an agent. In infinite-horizon tasks without absorbing goal states, this sum may diverge unless rewards are discounted. Discounted optimality, therefore, is widely adopted and applied in domains such as obstacle avoidance for robots [30] and routing automated guided vehicles to serve multiple queues [46]. Despite its appeal, discounted optimality can yield suboptimal behaviour, favouring short-term mediocre rewards over more valuable long-term outcomes. For such continuing tasks, average reward is often a more suitable objective. However, algorithms for average optimality and their convergence guarantees usually rely on strong structural assumptions about the underlying Markov chain, such as unichain or ergodicity, which can be restrictive and hard to verify [47].

Consider the deterministic MDP in Fig. 1a, which admits three policies: π_1, π_2, and π_3. All three are gain-optimal, π_2 and π_3 are also bias-optimal, but only π_3 is Blackwell-optimal. This example shows that Blackwell optimality is strictly stronger than either gain or bias optimality—it selects policies that remain optimal for all sufficiently high discount factors, rather than only at a particular value or in the average-reward limit. Consequently, it provides a more robust and stable decision rule, especially when the discount factor is unknown, ill-defined, or subject to change. Moreover, in transient states, BO policies prioritise early reward collection, making them particularly effective when such states yield large immediate returns.

By unifying the strengths of the discounted and average-reward criteria, Blackwell optimality offers a versatile framework for decision-making. Computing a BO policy automatically yields an average-optimal policy, which has a wide range of real-world applications, including queuing networks [1], scheduling [3], inventory management [10], and transfer lines [31]. Accordingly, Dewanto et al. [15] identify advancing our understanding of Blackwell optimality as one of the pressing questions in reinforcement learning.



(a) (b)

Figure 1: **a)** Example of a DMDP with 3 states and 3 actions: action 1 (solid), action 2 (dashed), and action 3 (dotted). From states s_2 and s_3, all actions produce the same effect. Consider policies $\pi_1 = (1,1,1)$, $\pi_2 = (2,1,1)$, and $\pi_3 = (3,1,1)$. All three policies yield the same gain, $V^{\pi_i}_{\text{g}}(s_1) = 0$. Yet, their biases differ: $V^{\pi_2}_{\text{b}}(s_1) = V^{\pi_3}_{\text{b}}(s_1) = 10$, while $V^{\pi_1}_{\text{b}}(s_1) = 5$. Their discounted values are: $V^{\pi_1}_{\gamma}(s_1) = 5\gamma, V^{\pi_2}_{\gamma}(s_1) = 5+5\gamma, V^{\pi_3}_{\gamma}(s_1) = 10$. Since $0 < \gamma < 1$, it follows that $V^{\pi_1}_{\gamma}(s_1) < V^{\pi_2}_{\gamma}(s_1) < V^{\pi_3}_{\gamma}(s_1)$. Note that only π_3 is BO. **b)** Example of an DMDP with $S = \{s_i\}_{i=1}^{6}$ and $A = \{a_1 \text{ (solid)}, a_2 \text{ (dashed)}\}$. Transitions are deterministic; rewards are shown on edges. While the actual Blackwell discount factor is $\gamma_{\text{bw}} = 0.8541$, the upper bound of Grand-Clément and Petrik [16] is: $U = 1 - \frac{1}{2.19\times 10^{66}}$.

2.3 Challenges in Computing BO Policies

Algorithms for computing BO policies are either computationally expensive [49] or intricate and difficult to implement [38]. To address this, Grand-Clément and Petrik [16] derive an upper bound U on the threshold discount factor γ_{bw}, guaranteeing that for any $\gamma \geq U \geq \gamma_{\text{bw}}$, every discount-optimal policy is also BO. Their bound has the form $U = 1 - \frac{1}{O(n^n)\cdot r_\infty \cdot m^n}$, where m is the bit-size of the MDP instance and r_∞ the maximum absolute reward.

While this gives a direct recipe for computing BO policies for MDPs with rational data of finite precision, the bound is so conservative as to be impractical. For example, in Fig. 1b, the true threshold is $\gamma_{\text{bw}} \approx 0.85$, but the bound yields $U \approx 1 - O(10^{-66})$, a vast gap that limits its applicability.

In practice, the absence of tight bounds leads practitioners to use very large discount factors to approximate BO policies. To be safe, they may choose an extremely high γ and solve the problem via policy iteration, value iteration, or linear programming. However, as $\gamma \to 1$, convergence slows dramatically [29], the Bellman matrix becomes nearly singular, and solving the resulting system of equations becomes numerically unstable. For large MDPs, the combination of high γ and sparse transitions further exacerbates instability, increasing the risk of computational errors.

Our experiments show that value iteration slows markedly as the discount factor γ approaches 1, with runtime scaling roughly as $\frac{1}{1-\gamma}$. In contrast, both policy iteration and linear programming fail to converge beyond certain threshold values of γ. We implemented VI and PI using the Python package `MDPtoolbox` [9] with default parameters, and formulated the LP approach using the `cvxpy` library. Figure 2 presents results for a simple 2-state MDP with deterministic transitions, indicating the γ ranges where each method either slows dramatically or fails. The table summarises these failures, showing that none of the algorithms can compute the discount-optimal policy for $\gamma > \gamma_{\text{fail}}$. Consequently, when $\gamma_{\text{bw}} > \gamma_{\text{fail}}$, these methods cannot produce the BO policy. In our implementations, $\gamma_{\text{fail}} \approx 1 - 10^{-17}$ for all three algorithms. Our code and data are available at [37].

To illustrate the impact of this limitation, we examine two families of MDP instances with $\gamma_{\text{bw}} > \gamma_{\text{fail}}$: one with a provable exponential lower bound on γ_{bw} (Theorem 2), and another derived from practical applications (Example 1). These cases demonstrate that high-threshold instances occur both theoretically and in practice, where standard methods are guaranteed to fail. In contrast, our approach successfully computes the BO policy even in these challenging settings, underscoring its robustness and practical utility.

Theorem 2. *There exists an MDP M with n states such that the threshold discount factor satisfies $\gamma^{M}_{bw} \geq 1 - O(2^{-n/3})$.*

Proof. See Appendix A for a construction. □

Example 1 (Healthcare)**.** *We examine a simplified model of an MDP used in healthcare [4, 17], originally developed to simulate clinical decision-making using real patient data. The objective is to minimise patient mortality while also limiting the invasiveness of the prescribed drug dosage—low, medium, or high. The model consists of n*

states: the first $n-1$ represent progressively worsening health conditions, and the final state is an absorbing mortality state. The action set $\{low, medium, high\}$ corresponds to dosage levels. As shown in Figure 3, higher dosages increase the likelihood of recovery (i.e., transitions to earlier, healthier states). Rewards penalise more aggressive treatments: in states 1 through $n-1$, the rewards are 10, 8, and 6 for low, medium, and high actions, respectively.

Figure 3 illustrates the family of MDPs we consider, and the accompanying table lists six representative instances with increasing state counts and their corresponding threshold discount factors γ_{bw}. Notably, once the state count exceeds 25, the values of γ_{bw} surpass $1-10^{-17}$, highlighting the numerical extremity of the problem.

2.4 Key Idea

We take policy iteration as our motivating method. At each step, given a current policy π, we consider whether changing the action at some state s to an alternate action a would improve the policy. The standard approach evaluates this by comparing the value $V^\pi(s)$ under the current policy with the Q-value $Q^\pi(s,a)$, of taking action a and following π thereafter. Specifically, we check the sign of the difference: $Q^\pi(s,a) - V^\pi(s)$.

Our main idea is that we can determine the sign of this difference—i.e., whether switching to action a is beneficial—without computing the exact values of $V^\pi(s)$ or $Q^\pi(s,a)$. Instead, we treat both quantities as rational functions in the discount factor γ, and focus on the sign of their difference as a function of γ.

Let $P(\gamma) = Q^\pi_\gamma(s,a) - V^\pi_\gamma(s)$. This is a rational function, and our goal is to determine the sign of $P(\gamma)$ at a specific value of γ, typically very close to 1 (e.g., $\gamma = 0.999$).

The key observation is that if we identify the last root of $P(\gamma)$ before 1—say at $\gamma = 0.9$—then the sign of $P(\gamma)$ remains constant in $(0.9, 1)$, since there are no sign changes beyond that point. Thus, we can infer the sign of $P(0.999)$ by simply computing the sign of $P(1)$, bypassing exact evaluations of $V^\pi(s)$ or $Q^\pi(s,a)$.

This insight enables a symbolic approach to policy improvement, replacing numerical evaluation of value function with algebraic comparisons yielding efficient algorithms even as $\gamma \to 1$.

3 Literature Review

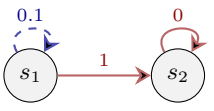
Blackwell [5] first demonstrated the existence of a Blackwell-optimal policy non-constructively. Building on this, Miller and Veinott [35] developed a policy iteration algorithm for finding the Blackwell-optimal policy, using the Laurent series expansion of V^π_γ around $\gamma = 1$. Such expansions connect the gain and bias to the expected total discounted reward and are a standard tool for analysing finite-state undiscounted models. However, the method is computationally expensive, with only an exponential upper bound on its runtime. In Appendix B, we show that this policy improvement procedure is equivalent to our rational function-based approach—both yield the same set of BO policies. Our algorithm, however, incorporates an additional step based on the Random-Facet method, which enables us to derive a subexponential bound.

Veinott [49] introduced a new family of optimality criteria, known as N-discount optimality. For $N = -1, 0, 1, \ldots$, a policy $\pi^\star$ is considered N-discount-optimal [49] if: $\lim_{\gamma\to 1}(1-\gamma)^{-N}[V^{\pi^\star}_\gamma(s) - V^\pi_\gamma(s)] \geq 0$, $\forall \pi, \forall s \in S$. This criterion captures the sensitivity of a policy's optimality with respect to the parameter N. Specifically, for $N = -1$, the condition corresponds to gain optimality, and for $N = 0$, it represents bias optimality. As N increases, the optimality condition becomes increasingly stringent, with Blackwell optimality being the most restrictive case. Veinott [49] showed that Blackwell optimality is equivalent to $|S|$-discount optimality.

As the sensitivity of the optimality criterion increases, designing efficient algorithms becomes more challenging. For $N = -1$ (gain optimality), the problem is efficiently solvable via policy improvement methods [19] or linear programming [32, 13, 11]. The case $N = 0$ (bias optimality) was tackled by Veinott [48] using policy improvement and by Denardo [12] through linear programming.

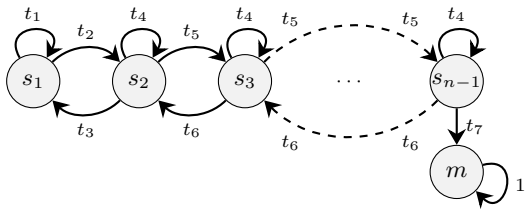
The fastest known algorithm for finding an N-optimal policy was developed by O'Sullivan and Veinott Jr [38]. Their method decomposes the problem into a linear sequence of subproblems. Each subproblem is either a Maximum Transient Value (MTV) problem, which optimises short-term rewards, or a Maximum Reward Rate (MRR) problem, which maximises long-term average rewards. The input to each subproblem is determined by the solution of the previous one. Although each subproblem admits a linear programming formulation of size $poly(n,k)$ and can be solved in weakly polynomial time, the overall method is highly intricate and has no known implementations. In contrast, our algorithms are simple and yield strong complexity bounds—independent of the bit-size of rewards.

Building on the work of Jeroslow [20], Hordijk et al. [18] developed a method for comparing rational functions near zero, which they applied to discounted MDPs. They constructed a simplex tableau in which the entries are expressed as rational functions of the parameter $\rho = \frac{1-\gamma}{\gamma}$. Using this symbolic representation, they applied Sturm's Theorem to identify a threshold ρ_0 such that the current tableau remains optimal for all $\rho \geq \rho_0$. This threshold determines the next set of basic variables, allowing the tableau to be updated accordingly. The process is then repeated, successively identifying intervals



u	3	4	5	6	7	8	...	17
VI Time (s)	0.1	1.1	13	153	1828	–	...	–
LP Status	✓	✓	✓	✓	✓	✗	...	✗
PI Status	✓	✓	✓	✓	✓	✓	...	✗

Figure 2: Figure shows a deterministic MDP with two states: two actions (solid and dashed) are available from state 1, and a single action from state 2. Edge labels indicate the corresponding rewards. The accompanying table reports the performance of VI, LP, and PI as the discount factor $\gamma = 1-10^{-u}$ increases. VI becomes impractically slow for $u \geq 8$, LP produces incorrect results, and PI fails due to matrix singularities for $u \geq 17$.



n	15	20	25	30	35	40
u	9.98	13.47	16.97	20.4	23.96	27.4

Figure 3: Family of healthcare-inspired MDPs [17] along with their corresponding Blackwell threshold discount factors. The model uses seven distinct transition probabilities $\{t_i\}_{i=1}^7$, across all states specified separately for each action: Low: $\{0.7, 0.3, 0.3, 0.4, 0.3, 0.3, 0.3\}$; Medium: $\{0.8, 0.2, 0.4, 0.4, 0.2, 0.4, 0.2\}$; High: $\{0.9, 0.1, 0.5, 0.4, 0.1, 0.5, 0.1\}$. For example, taking the medium action from state s_3 results in a transition to s_2, s_3, and s_4 with probabilities 0.4, 0.4, and 0.2, respectively. The table reports values of u such that the Blackwell threshold satisfies $\gamma_{\text{bw}} = 1-10^{-u}$, shown for increasing numbers of states n.

$[\rho_1, \rho_0]$, until $\rho_0 = 0$, at which point the algorithm terminates with the optimal policy for the entire range of discount factors. Moreover, by setting $\rho = 0$ directly and adjusting the pivot selection rule, the method can be adapted to compute the Blackwell-optimal policy. In contrast, our approach is simpler and integrates directly into the policy improvement framework, enabling generalisation to a broad class of efficient algorithms with provable upper bounds.

Smallwood [43] first introduced the concept of a threshold discount factor beyond which a discount-optimal policy becomes Blackwell-optimal. Grand-Clément and Petrik [16] demonstrated the existence of such a threshold for finite MDPs and provided an upper bound on $1/(1-\gamma_{\text{bw}})$, which is exponential in the number of states. However, no lower bound on this parameter was known. In this work, we provide the first exponential lower bound, showing that such thresholds can, in fact, be exponentially close to 1.

In a learning context, Mahadevan [28] introduced the first tabular Q-learning algorithm designed to achieve bias-optimal policies by optimising over the family of n-discount optimality criteria. Dewanto and Gallagher [14] presented a policy-gradient method for learning bias-optimal policies in unichain MDPs. Additionally, Schneckenreither [41] proposed a model-free tabular algorithm for computing bias-optimal policies in unichain MDPs.

Boone and Gaujal [6] studied the problem of identifying BO policies in deterministic MDPs within a fixed-confidence PAC-RL framework. They proved that this is impossible in general, unless the MDP satisfies uniqueness conditions on both the optimal cycle and the bias-optimal policy. For this maximal identifiable class, they proposed a sample-efficient algorithm based on generalised Bellman coefficients and structured confidence sets, achieving near-optimal bounds on the number of reward queries required.

4 Method of Rational Functions

In this section, we present the mathematical basis of our method for computing BO policies. We begin by defining an ordering of rational functions in the vicinity of 1. Using this ordering, we derive BO policies for both Deterministic MDPs (DMDPs) and general MDPs.

4.1 Ordering of rational functions

Consider two rational functions, $r_1(x) = \frac{p_1(x)}{q_1(x)}$, $r_2(x) = \frac{p_2(x)}{q_2(x)}$, where p_1, p_2, q_1, q_2 are polynomials with real coefficients and $q_1, q_2 \not\equiv 0$. Let $\tau(x) = r_1(x) - r_2(x)$, we define $\tau \doteq 0$ if and only if $p_1(x)q_2(x) \equiv p_2(x)q_1(x)$. Suppose:

$$\tau(x) = \frac{\eta(x)}{\delta(x)} = \frac{(1-x)^{c_1} \cdot \bar{\eta}(x)}{(1-x)^{c_2} \cdot \bar{\delta}(x)},$$

where $c_1, c_2 \in \mathbb{Z}_{\geq 0}$ denote the multiplicities of the root $x = 1$ in $\eta(x)$ and $\delta(x)$, respectively, and $\bar{\eta}(1) \neq 0, \bar{\delta}(1) \neq 0$. We define a total order $\succ$ on the set of rational functions, such that:

$$r_1 \succ r_2 \iff \bar{\eta}(1) \cdot \bar{\delta}(1) > 0.$$

We refer to this ordering as μ-ordering.

Lemma 3. *For finite rational functions r_1 and r_2, $r_1 \succ r_2 \iff \exists \sigma \in (0,1)$ such that $r_1(x) > r_2(x)\ \forall x \in (\sigma, 1)$.*

Proof. Since η and δ are finite polynomials they have a finite number of roots. Let σ_1 and σ_2 be the largest roots of η and δ respectively in $(0,1)$ and let $\max\{\sigma_1, \sigma_2\} = \sigma$. It is clear that τ cannot change sign in $(\sigma, 1)$. Since $\bar{\eta}(1) \neq 0$ and $\bar{\delta}(1) \neq 0$, the ratio $\bar{\tau} = \frac{\bar{\eta}}{\bar{\delta}}$ must be finite and non-zero. As $\bar{\tau}$ is continuous we have: $\bar{\tau}(\sigma_0) > 0 \iff \bar{\tau}(1) > 0,\ \forall \sigma_0 \in (\sigma, 1)$. Since $(1-\sigma_0) > 0$, τ and $\bar{\tau}$ have the same sign at σ_0 for all $\sigma_0 \in (\sigma, 1)$ and the result follows. □

Example 2. *Let $r_1 = \frac{(1-x)^2(5x-10)}{x-2}$ and $r_2 = \frac{(1-x)(x-5)}{x-4}$. Then $\tau = \frac{(1-x)(-5x^2+24x-15)}{x-4}$ and $\bar{\eta}(1)\bar{\delta}(1) = 4 \cdot (-3) < 0 \implies r_2 \succ r_1$. (see Figure 4).*

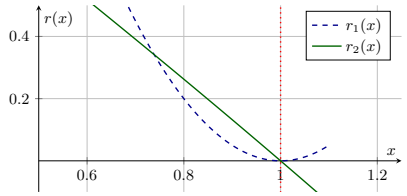


Figure 4: Plot of the functions r_1 and r_2. Note that $r_1(1) = r_2(1) = 0$, while $r_1(1-\epsilon) < r_2(1-\epsilon)$ for all $0 < \epsilon < \frac{1}{4}$.

4.2 Complexity

The complexity of comparing two rational functions depends on two primary operations:

- Computing the difference τ: This involves subtracting one rational function from the other. Assume that the polynomials involved have a degree of $O(d)$, the calculation of τ, which includes polynomial multiplication, can be performed in $O(d \log d)$ steps using efficient polynomial multiplication algorithms.
- Determining the multiplicity of 1 for τ: This requires evaluating the polynomial at 1 at most upto its d-th derivative. Using Horner's method, each evaluation at 1 takes $O(d)$ operations resulting in $O(d^2)$ complexity to determine the multiplicity.

Therefore, the overall complexity of comparing two rational functions is $O(d^2)$ operations.

Given a sequence having t elements, finding a maximal/minimal element requires at most t comparisons. Since each comparison between rational functions has a complexity of $O(d^2)$ under μ-ordering, the overall complexity of computing the maximal or minimal rational function from the sequence is $O(td^2)$. We use $\widehat{\max}$, $\widehat{\min}$ to denote such max and min operations respectively.

4.3 Application

In the following sections, we consider four existing algorithmic families $\{\mathcal{L}_i\}_{1 \leq i \leq 4}$, each generating a sequence of policies converging to an optimal policy $\pi^\star$ for a discounted MDP M_γ. The trajectory $\left(\pi^0_\gamma, \pi^1_\gamma, \ldots, \pi^\star_\gamma\right)$ depends on both the MDP and γ. For each $\mathcal{L}_i$, we define a corresponding γ-independent algorithm $\mathcal{L}'_i$ that produces the same sequence for all sufficiently large discount factors. The following theorem establishes the existence of such a sequence.

Theorem 4. *There exists a $\tau \in [0,1)$ such that $\mathcal{L}_i$ follows the trajectory: $\left(\pi^0_\tau, \pi^1_\tau, \ldots, \pi^\star_\tau\right)$ for all $\gamma \in [\tau, 1)$.*

Proof. Let the discount factor γ be treated as a symbolic variable. Then, equation (5) defines a value vector $\mathbf{v}^\pi_\gamma$, whose components (one per state) are rational functions of γ.

Define $f_s^{(i,j)}(\gamma) = V^{\pi_j}_\gamma(s) - V^{\pi_i}_\gamma(s)$, the value difference at state s between policies π_j and π_i. Since the MDP is finite, each $f_s^{(i,j)}(\gamma)$ is a ratio of finite-degree polynomials, and therefore has only finitely many roots. Let $\gamma_s^{(i,j)}$ denote the largest root of $f_s^{(i,j)}$ in the interval

$[0, 1)$. For any $\gamma > \gamma_s^{(i,j)}$, the sign of $f_s^{(i,j)}(\gamma)$ remains constant, meaning the relative ordering of $V_\gamma^{\pi_i}(s)$ and $V_\gamma^{\pi_j}(s)$ does not change beyond this point. Define the global threshold

$$\tau = \max_{s \in S,\, \pi_i, \pi_j \in \Pi} \gamma_s^{(i,j)}.$$

Then, for all $\gamma \in [\tau, 1)$, the value ordering—and hence policy preferences—remains invariant across policy pairs in Π_γ. □

Since μ-ordering provides an ordering or rational functions in the left-neighbourhood of 1, the theorem implies that $\mathcal{L}_i$ follows the same trajectory as when applied to $\mathcal{M}_{1-\epsilon}$ with $\epsilon \to 0$. Hence, both correctness and complexity guarantees of $\mathcal{L}_i$ in the discounted setting extend to the symbolic variant $\mathcal{L}_i'$, with only a polynomial overhead from symbolic evaluation.

5 Efficient Planning with Blackwell Optimality

In this section, we describe algorithms of type $\mathcal{L}_i'$, identify the rational functions and corresponding thresholds that govern their policy trajectories, and analyse their computational complexity.

5.1 Maximum Mean Weight Cycle

The algorithm of Madani et al. [27], inspired by Karp's method for finding the minimum mean-weight cycle in a graph [24], is the fastest known approach for solving DMDPs under the discounted criterion.

The algorithm proceeds in two stages, both based on Bellman-Ford-style updates. In the first stage, it computes $d_i(s)$, the maximum discounted cost of an i-edge path starting from state s, for all states. Using these d_i values, it evaluates

$$y_0(s) = \min_{0 \le i \le n} \frac{d_n(s) - \gamma^{n-i} d_i(s)}{1 - \gamma^{n-i}},$$

for each s. In the second stage, the $y_0(s)$ values serve as initial state values for updating $y_i(s)$, and the maximum $y_i(s)$ over all states yields the optimal values.

The discounted costs of paths and the ratios y_i are rational functions of γ. We apply μ-ordering replacing the $\max$ and $\min$ operators with $\widehat{\max}$ and $\widehat{\min}$ respectively. Since the original algorithm focuses solely on computing values, we introduce additional variables $\bar{\alpha}_i(s)$, $\bar{y}_0(s)$, and $\bar{a}_i(s)$ to track the actions taken during the updates of $d_i(s)$, $y_0(s)$, and $y_i(s)$, respectively. This modification enables recovery of the BO policy. The full procedure is given in Algorithm 1.

Let $D_\gamma^i(s, a, s') = r(s, a, s') + \gamma d_{i-1}(s')$ and $Y_\gamma^j(s, a, s') = r(s, a, s') + \gamma y_{j-1}(s')$ where d_i and y_j are as defined in Algorithm 1. Then the threshold discount factor γ_1 is defined as:

$$\gamma_1 \stackrel{\text{def}}{=} \sup_{\substack{1 \le i \le n,\, 1 \le j \le n-1 \\ s, s', s'' \in S,\, a, a' \in A}} \inf_{\gamma \in [0,1)} \Big\{ \forall \tau \in (\gamma, 1),\ \forall \Phi \in \{D^i, Y^j\} : \Phi_\gamma(s, a, s') > \Phi_\gamma(s, a', s'') \Rightarrow \Phi_\tau(s, a, s') > \Phi_\tau(s, a', s'') \Big\}.$$

Theorem 5. *The DetMDP2-Blackwell procedure computes a BO policy for a DMDP with a runtime complexity of $O(n^4 k)$.*

Proof. As demonstrated earlier, the complexity of comparing two rational functions is $O(n^2)$. Since the maximum is sought over the set of k actions, the complexity for finding the maximum at each step is $O(n^2 k)$. Given that the maximum is computed at most n^2 times throughout the algorithm, the runtime complexity is $O(n^4 k)$. □

Algorithm 1 DetMDP2-Blackwell

procedure $\Phi_D(M = (S, A, T, r))$
  for each $s \in S$ **do**
    $d_0(s) \leftarrow 0$
  for $i = 1$ to n **do**
    for each $s \in S$ **do**
      $d_i(s) \leftarrow \widehat{\max}_{s' \in S}\, r(s, a, s') + \gamma d_{i-1}(s')$
      $\bar{\alpha}_i(s) \leftarrow \arg\widehat{\max}_{a \in A}\, r(s, a, s') + \gamma d_{i-1}(s')$
  for each $s \in S$ **do**
    $y_0(s) \leftarrow \widehat{\min}_{0 \le i \le n} \frac{d_n(s) - \gamma^{n-i} d_i(s)}{1 - \gamma^{n-i}}$
    $\bar{y}_0(s) \leftarrow \bar{\alpha}_n(s)$
  for $i = 1$ to $n - 1$ **do**
    for each $s \in S$ **do**
      $y_i(s) \leftarrow \widehat{\max}_{s' \in S}\, r(s, a, s') + \gamma y_{i-1}(s')$
      $\bar{a}_i(s) \leftarrow \arg\widehat{\max}_{a \in A}\, r(s, a, s') + \gamma y_{i-1}(s')$
  for each $s \in S$ **do**
    $\bar{i}_s \leftarrow \arg\widehat{\max}_{0 \le i \le n}\, y_i(s)$
    $\pi^*_{\text{bw}}(s) \leftarrow \bar{a}_{\bar{i}_s}(s)$
  return π^*_{bw}

5.2 Max Gain

The Max-Gain Simplex (MGS) algorithm [39] computes the optimal policy for DMDPs in strongly polynomial time. For a DMDP with a discount factor γ, the algorithm iteratively selects the state-action pair with the highest gain to transition to a new policy. Specifically, starting from a policy π, it transitions to a new policy $\overline{\pi}$ defined as:

$$\overline{\pi}(s) = \begin{cases} \pi(s), & \text{if } s \neq \overline{s}, \\ \overline{a}, & \text{if } s = \overline{s}, \end{cases}$$

where $(\overline{s}, \overline{a})$ is chosen to satisfy:

$$(\overline{s}, \overline{a}) = \arg\max_{(s,a)} \left(Q^\pi(s, a) - V^\pi(s) \right).$$

5.2.1 Computing BO policies

Our algorithm for computing BO policies mirrors the procedure of MGS, but with an updated max operator. From (2) and (3), it is clear that the quantity $Q_\gamma^\pi(s, a) - V_\gamma^\pi(s)$ is a rational function of γ. Using μ-ordering, we redefine the max operator as:

$$(\overline{s}, \overline{a}) = \arg\widehat{\max}_{(s,a)} \left(Q_\gamma^\pi(s, a) - V_\gamma^\pi(s) \right).$$

Theorem 6. *The described procedure computes a BO policy for a DMDP in at most $O(n^7 k^3 \log^2 n)$ iterations.*

Proof. The MGS algorithm completes in at most $O(n^5 k^2 \log^2 n)$ iterations [39]. Our algorithm introduces an additional $O(n^2 k)$ steps per iteration to determine the max-gain. Hence, the total computational complexity for obtaining a BO policy is $O(n^7 k^3 \log^2 n)$. □

Since MGS and the subsequent algorithms compute Q-values to guide action selection, their threshold discount factor is γ_Q.

5.3 Random-Facet

Matoušek et al. [34] introduced the randomised pivot rule Random-Facet, which gives an upper bound of $2^{\sqrt{n \log m}}$ on the expected number of pivot steps to solve any linear program with n variables and

m constraints. Combining the algorithm with Clarkson's method [7] yields the tightest known bound of $O\left(n^2m + e^{O(\sqrt{n \log n})}\right)$.

Let $p = (s, a)$ be a state-action pair, and define $f^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. The pair p belongs to π if $\pi(s) = a$, and is *improving* for π iff $f^\pi(s, a) > 0$. The Random-Facet algorithm relies on determining whether a given pair p is improving for a policy π, which amounts to checking the sign of $f^\pi(s, a)$ at each step.

The algorithm can be adapted to MDPs as follows: given an MDP M and an initial policy π, select a state-action pair $p \notin \pi$, and recursively solve M without p, yielding a new policy π'. If p is not improving for π', then π' is guaranteed to be optimal. Otherwise, update π' by switching p, and repeat the process.

Let $\hat{f}^\pi_\gamma(s, a) = Q^\pi_\gamma(s, a) - V^\pi_\gamma(s)$ denote the rational function in γ. Our algorithm for identifying a BO policy adapts the Random-Facet procedure, applying μ-ordering on $\hat{f}$ to determine its sign in the vicinity of 1. The pseudocode is given in Algorithm 2. We denote the set of state-action pairs corresponding to M and π by M_p and π_p, respectively.

Algorithm 2 Random-Facet-Blackwell

```
1:  procedure Φ_R(M_p, π)
2:      if f̂^π(p) ⪯ 0, ∀p ∈ M_p then
3:          π*_bw ← π
4:          return π*_bw
5:      else
6:          Pick p ∈ M_p \ π_p uniformly at random
7:          π' ← Φ_R(M_p \ p, π)                    ▷ 1st call
8:          if f̂^π'(p) ≻ 0 then
9:              π'' ← switch(p, π')
10:             return Φ_R(M_p, π'')                ▷ 2nd call
11:         else
12:             return π'
```

Theorem 7. *The Random-Facet-Blackwell algorithm computes a BO policy for an MDP in at most poly$(n, k) \cdot e^{O(\sqrt{n \log n})}$ expected iterations.*

Proof. The Random-Facet algorithm, when combined with Clarkson's algorithm, has an expected runtime of $O(n^3k + e^{O(\sqrt{n \log n})})$ [34]. Our algorithm introduces an additional n^2 operations per recursive call to determine the sign of $\hat{f}^\pi$. Consequently, the overall runtime complexity increases by a polynomial factor in n, resulting in the desired bound. □

5.4 Generalisation: PI improvement

The approaches in Sections 5.2 and 5.3 extend naturally to any Policy Improvement (PI) procedure. A generic PI procedure operates as follows: given a policy π, define the set of improving state-action pairs: $J^\pi = \{(s, a) \mid Q^\pi(s, a) > V^\pi(s)\}$. At each iteration, select a subset $\Theta \subseteq J^\pi$ containing at most one action per state, and construct a new policy π' such that $\pi'(s) = a$ for all $(s, a) \in \Theta$. The rule for selecting Θ is algorithm-specific and crucial to the procedure's complexity. For example, in the Max-Gain method, Θ consists of the single pair (s, a) that maximises: $Q^\pi(s, a) - V^\pi(s)$.

As before, $Q^\pi_\gamma(s, a) - V^\pi_\gamma(s)$ is a rational function in γ, whose sign can be determined via μ-ordering. This leads to a PI procedure for computing BO policies in MDPs. We examine the three tightest known variants of *memoryless* PI algorithms—$\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$—with respective bounds $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$, each depending only on n and k under the discounted criterion. Applying μ-ordering, we generalise each variant to achieve bounds of $\text{poly}(n, k) \cdot \mathcal{B}_i$ under the Blackwell criterion. Below, we summarise $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ together with their corresponding bounds.

- $\mathcal{A}_1$: Batch-Switching policy iteration [22] partitions the state space into batches with a fixed ordering and, at each step, switches states from J^π in decreasing order of their batch index, yielding an iteration bound of $\mathcal{B}_1 = O(1.64^n)$.
- $\mathcal{A}_2$: Howard's policy iteration [19] is a greedy procedure that, at each iteration, switches every improvable state—i.e., it maximises the set Θ. While highly efficient in practice, its theoretical upper bound remains exponential in n, specifically $\mathcal{B}_2 = O(\frac{k^n}{n})$ [33].
- $\mathcal{A}_3$: Randomised Simple policy iteration [23] assumes an indexing of the states. At each iteration, it considers all improving pairs whose state has the highest index, and switches exactly one, chosen uniformly at random. Its iteration bound is $\mathcal{B}_3 = O((\log k)^n)$.

6 Summary and Discussion

In this paper, we used an ordering of rational functions near 1 to develop novel and efficient algorithms for computing BO policies in both MDPs and DMDPs. Our methods attain the tightest known complexity bounds, advancing the state of the art through a simple, theoretically robust framework for BO policy computation.

To illustrate the limitations of existing dynamic programming algorithms, we presented two examples, one of which gives an exponential lower bound on the threshold γ_{bw}. This bound not only underscores the complexity of computing BO policies but also limits the generalisability of existing proof techniques. In particular, Mukherjee and Kalyanakrishnan [36] prove a subexponential upper bound for Howard's policy iteration in DMDPs by analysing ratios of polynomials derived from Q-value comparisons and bounding the roots near 1 to control γ_Q. Our exponential lower bound on $\gamma_{\text{bw}} \leq \gamma_Q$ shows that such techniques cannot extend to the stochastic case.

We also implemented and tested a basic, unoptimised version of our symbolic policy iteration [37]. For DMDPs, the method is reasonably efficient, solving instances with up to 100 states in about three minutes on a standard desktop (AMD Ryzen 7 5700G, 16 GB RAM). For general MDPs, runtime increases more sharply with stochasticity due to the cost of symbolic matrix inversion: problems with up to 10-15 states solve within minutes, whereas 20-state instances may require over an hour. Benchmarking against prior BO algorithms is challenging—few exist, and those that do are rarely presented in a form amenable to straightforward implementation. This scarcity of practical baselines underscores the need for simple, implementable approaches such as ours. Although our current results serve primarily as a proof of concept, we plan to optimise and scale up the implementation in future work.

While our focus has been on planning with full model knowledge, the structural insights and algebraic techniques developed here may also inform learning-based approaches. In particular, model-based reinforcement learning could leverage efficient BO policy computation once an approximate model is inferred from data. However, as recent work has shown [6], identifying BO policies with limited samples is generally impossible without strong assumptions. A promising direction for future research is to investigate whether our algebraic characterisations can help delineate the class of MDPs where reliable identification is feasible, or guide the design of robust algorithms under uncertainty.

References

[1] F. Al-Ani, M. Wang, J. Charles, A. Ong, J. Forday, and V. Modi. Simulation-Driven Reinforcement Learning in Queuing Network Routing Optimization. *arXiv preprint arXiv:2507.18795*, 2025.

[2] C. Antonio, N. Muddasar, D. P. Giuseppe, and P. Giovanni. Reinforcement learning for intelligent healthcare applications: A survey. *Artificial Intelligence In Medicine*, 109:101964, 2020.

[3] M. E. Aydin and E. Öztemel. Dynamic job-shop scheduling using reinforcement learning agents. *Robotics Auton. Syst.*, 33(2-3):169–178, 2000.

[4] C. C. Bennett and K. K. Hauser. Artificial intelligence framework for simulating clinical decision-making: A markov decision process approach. *Artif. Intell. Medicine*, 57(1):9–19, 2013.

[5] D. Blackwell. Discrete Dynamic Programming. *The Annals of Mathematical Statistics*, pages 719–726, 1962.

[6] V. Boone and B. Gaujal. Identification of Blackwell Optimal Policies for Deterministic MDPs. In *International Conference on Artificial Intelligence and Statistics*, pages 7392–7424. PMLR, 2023.

[7] K. L. Clarkson. Las Vegas Algorithms for Linear and Integer Programming When the Dimension is Small. *J. ACM*, 42(2):488–499, 1995.

[8] B. Claudine, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixao, F. Mutz, et al. Self-Driving Cars: A Survey. *Expert Systems with Applications*, 165:113816, 2021.

[9] S. Cordwell, Y. Gonzalez, and T. Tulabandhula. Markov Decision Process (MDP) Toolbox for Python. *Website. URL: https://github.com/sawcordwell/pymdptoolbox (accessed on May 6, 2019)*, 2015.

[10] T. K. Das, A. Gosavi, S. Mahadevan, and N. Marchalleck. Solving Semi-Markov Decision Problems Using Average Reward Reinforcement Learning. *Management Science*, 45(4):560–574, 1999.

[11] E. V. Denardo. On Linear Programming in a Markov Decision Problem. *Management Science*, 16(5):281–288, 1970.

[12] E. V. Denardo. Computing a Bias-Optimal Policy in a Discrete-Time Markov Decision Problem. *Operations Research*, 18(2):279–289, 1970.

[13] C. Derman. On Sequential Decisions and Markov Chains. *Management Science*, 9(1):16–24, 1962.

[14] V. Dewanto and M. Gallagher. A nearly Blackwell-optimal policy gradient method. *arXiv preprint arXiv:2105.13609*, 2021.

[15] V. Dewanto, G. Dunn, A. Eshragh, M. Gallagher, and F. Roosta. Average-reward model-free reinforcement learning: a systematic review and literature mapping. *arXiv preprint arXiv:2010.08920*, 2020.

[16] J. Grand-Clément and M. Petrik. Reducing Blackwell and Average Optimality to Discounted MDPs via the Blackwell Discount Factor. In *Proc. NeurIPS 2023*, pages 52628–52647. Curran Associates, Inc., 2023.

[17] J. Grand-Clément, M. Petrik, and N. Vieille. Beyond discounted returns: Robust Markov decision processes with average and Blackwell optimality. *arXiv preprint arXiv:2312.03618*, 2023.

[18] A. Hordijk, R. Dekker, and L. C. M. Kallenberg. Sensitivity-Analysis in Discounted Markovian Decision Problems. *Operations-Research-Spektrum*, 7(3):143–151, 1985.

[19] R. A. Howard. *Dynamic Programming and Markov Processes.* MIT Press, 1960.

[20] R. Jeroslow. *An Algorithm for Discrete Dynamic Programming with Interest Rates Near Zero*. Management Sciences Research Group, Graduate School of Industrial Administration, Carnegie-Mellon University, 1972., 1972.

[21] G. Kalai. A Subexponential Randomized Simplex Algorithm (Extended Abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 475–482. ACM, 1992.

[22] S. Kalyanakrishnan, U. Mall, and R. Goyal. Batch-Switching Policy Iteration. In *Proc. IJCAI 2016*, pages 3147–3153. IJCAI/AAAI Press, 2016.

[23] S. Kalyanakrishnan, N. Misra, and A. Gopalan. Randomised Procedures for Initialising and Switching Actions in Policy Iteration. In *Proc. AAAI 2016*. AAAI Press, 2016.

[24] R. M. Karp. A Characterization of the Minimum Cycle Mean in a Digraph. *Discrete mathematics*, 23(3):309–311, 1978.

[25] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[26] J. Leahy, B. Kerimkulov, D. Siska, and L. Szpruch. Convergence of Policy Gradient for Entropy Regularized MDPs with Neural Network Approximation in the Mean-Field Regime. In *In Proc. ICML 2022*, volume 162, pages 12222–12252. PMLR, 2022.

[27] O. Madani, M. Thorup, and U. Zwick. Discounted Deterministic Markov Decision Processes and Discounted All-Pairs Shortest Paths. *ACM Trans. on Alg.*, 6(2):1–25, 2010.

[28] S. Mahadevan. Average Reward Reinforcement Learning: Foundations, Algorithms, and Empirical Results. *Mach. Learn.*, 22(1-3):159–195, 1996.

[29] S. Mahadevan. Sensitive Discount Optimality: Unifying Discounted and Average Reward Reinforcement Learning. In *ICML*, pages 328–336. Citeseer, 1996.

[30] S. Mahadevan and J. Connell. Automatic Programming of Behavior-Based Robots Using Reinforcement Learning. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 768–773. AAAI Press / The MIT Press, 1991.

[31] S. Mahadevan and G. Theocharous. Optimizing Production Manufacturing Using Reinforcement Learning. In D. J. Cook, editor, *Proceedings of the Eleventh International Florida Artificial Intelligence Research Society Conference*, pages 372–377. AAAI Press, 1998.

[32] A. S. Manne. Linear Programming and Sequential Decisions. *Management Science*, 6(3):259–267, 1960.

[33] Y. Mansour and S. Singh. On the Complexity of Policy Iteration. In *Proc. UAI 1999*, pages 401–408. Morgan Kaufmann, 1999.

[34] J. Matoušek, M. Sharir, and E. Welzl. A Subexponential Bound for Linear Programming. *Algorithmica*, 16(4/5):498–516, 1996.

[35] B. L. Miller and A. F. Veinott. Discrete Dynamic Programming with a Small Interest Rate. *The Annals of Mathematical Statistics*, 40(2): 366–370, 1969.

[36] D. Mukherjee and S. Kalyanakrishnan. Howard's Policy Iteration is Subexponential for Deterministic Markov Decision Problems with Rewards of Fixed Bit-size and Arbitrary Discount Factor. *arXiv preprint arXiv:2505.00795*, 2025.

[37] D. Mukherjee and S. Kalyanakrishnan. Code for "Efficient Computation of Blackwell Optimal Policies using Rational Functions". https://github.com/dib007/blackwell-pi, 2025. GitHub repository, last accessed 25 August 2025.

[38] M. O'Sullivan and A. F. Veinott Jr. Polynomial-Time Computation of Strong and n-Present-Value Optimal Policies in Markov Decision Chains. *Mathematics of Operations Research*, 42(3):577–598, 2017.

[39] I. Post and Y. Ye. The Simplex Method is Strongly Polynomial forDeterministic Markov Decision Processes. In *Proc. SODA 2013*, pages 1465–1473. SIAM, 2013.

[40] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[41] M. Schneckenreither. Average Reward Adjusted Discounted Reinforcement Learning Near-Blackwell-Optimal Policies for Real-World Applications. *arXiv preprint arXiv:2004.00857*, 2020.

[42] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[43] R. D. Smallwood. Optimum Policy Regions for Markov Processes with Discounting. *Operations Research*, 14(4):658–669, 1966.

[44] S. Sood, K. Papasotiriou, M. Vaiciulis, and T. Balch. Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization. *FinPlan*, 2023(2023):21, 2023.

[45] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[46] P. Tadepalli and D. Ok. H-learning: A Reinforcement Learning Method to Optimize Undiscounted Average Reward. Technical report, Computer Science Department, Oregon State University, 1994.

[47] J. N. Tsitsiklis. NP-Hardness of checking the unichain condition in average cost MDPs. *Oper. Res. Lett.*, 35(3):319–323, 2007.

[48] A. F. Veinott. On Finding Optimal Policies in Discrete Dynamic Programming with No Discounting. *The Annals of Mathematical Statistics*, 37(5):1284–1294, 1966.

[49] A. F. Veinott. Discrete Dynamic Programming with Sensitive Discount Optimality Criteria. *The Annals of Mathematical Statistics*, 40(5):1635–1660, 1969.

A γ_{bw} : A Lower Bound

In this section, we construct an MDP with a threshold discount factor γ_{bw} that is exponentially close to 1, as established in Theorem 2.

A.1 Construction

Consider the MDP shown in Figure 5, consisting of $n+7$ states. From the initial state $s_0 = u$, there are two available actions: a_0 and a_1. All other states allow only a single action a_0. There are two policies: $\pi_0 = a_0 a_0^{n-1}$ and $\pi_1 = a_1 a_0^{n-1}$. The value of π_0 from the start state is:

$$\begin{aligned} V^{\pi_0}(u) &= \gamma\left[(1-\epsilon)V^{\pi_0}(\alpha_0) + \epsilon V^{\pi_0}(\alpha_1)\right] \\ &= \epsilon\gamma^{\frac{n}{3}}\left[1 + V^{\pi_0}\left(\alpha_{\frac{n}{3}+1}\right)\right] \\ &= \epsilon\gamma^{\frac{n}{3}} \end{aligned}$$

And the Q-value of taking action a_1 from the same state under policy π_0 is given by:

$$\begin{aligned} Q^{\pi_0}(u, a_1) &= \gamma\left[\frac{1}{2}V^{\pi_0}(\beta_1) + \left(\frac{1}{2} - \epsilon\right)V^{\pi_0}(v) + \epsilon V^{\pi_0}(\delta_1)\right] \\ &= \gamma^{\frac{n}{3}}\left(\frac{1}{2}\right)^{\frac{n}{3}}\left[1 + V^{\pi_0}\left(\beta_{\frac{n}{3}+1}\right)\right] \\ &\qquad + \epsilon\gamma^{\frac{n}{3}+1}\left[1 + V^{\pi_0}\left(\delta_{\frac{n}{3}+2}\right)\right] \\ &= \gamma^{\frac{n}{3}}\left(\frac{1}{2}\right)^{\frac{n}{3}} + \epsilon\gamma^{\frac{n}{3}+1} \end{aligned}$$

Therefore:

$$Q^{\pi_0}(u, a_1) > V^{\pi_0}(u) \implies \epsilon\gamma^{\frac{n}{3}}\left(\frac{1}{\epsilon 2^{\frac{n}{3}}} + \gamma - 1\right) > 0$$

This implies that action a_1 is better than a_0 in state u whenever

$$\gamma > \gamma_0 = 1 - \tfrac{1}{\epsilon 2^{n/3}}.$$

Therefore π_1 remains Blackwell-optimal beyond the threshold γ_0. This gives a lower bound $\gamma_{\text{bw}} \geq 1 - O(2^{-n/3})$.

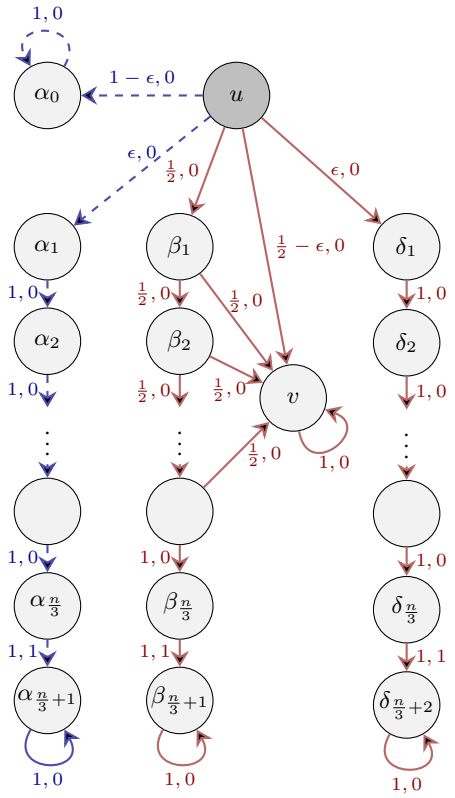


Figure 5: MDP with states $u, v, \alpha_i, \beta_i, \delta_i$. State u has 2 available actions a_0 (dashed) and a_1 (solid) and the other states have a single action. The edges are labelled: $T(s, a, s'), R(s, a, s')$ and $0 < \epsilon < \frac{1}{2}$.

B Equivalence of PI procedures

We present the policy improvement procedure of Miller and Veinott [35] and show that its sequence of visited policies matches that of our rational function-based approach. Our method further integrates the Random-Facet algorithm to obtain a subexponential bound. The steps for computing the difference vector $\mathbf{q}_a^\pi - \mathbf{v}^\pi$ for a given policy π and action a are outlined in Algorithm 3.

Algorithm 3 Policy Improvement

procedure COMPUTE_Q_MINUS_V($a, \pi, \mathcal{M}$)
  Compute P^π and $\mathbf{r}^\pi$
  $P^* \leftarrow \lim_{\rho \to 0} \rho \left(\rho I - (P^\pi - I)\right)^{-1}$
  $D^\pi \leftarrow (I - P^*)(P^* - (P^\pi - I))^{-1}$
  Define $\mathbf{y}_j^\pi$ as:

$$\mathbf{y}_j^\pi = \begin{cases} P^* \mathbf{r}^\pi, & \text{if } j = -1, \\ (-1)^j \left(D^\pi\right)^{j+1} \mathbf{r}^\pi, & \text{if } j \geq 0 \end{cases}$$

  Define $\psi_j^{a,\pi}$ as:

$$\psi_j^{a,\pi} = \begin{cases} (P_a - I)\, \mathbf{y}_j^\pi, & \text{if } j = -1, \\ \mathbf{r}_a + (P_a - I)\, \mathbf{y}_j^\pi - \mathbf{y}_{j-1}^\pi, & \text{if } j = 0, \\ (P_a - I)\, \mathbf{y}_j^\pi - \mathbf{y}_{j-1}^\pi, & \text{if } j \geq 0 \end{cases}$$

  Construct the matrix Ψ:

$$\Psi = \begin{bmatrix} \psi_{-1}^{a,\pi} & \psi_0^{a,\pi} & \cdots & \psi_{|S|}^{a,\pi} \end{bmatrix}$$

  Identify the first non-zero column of Ψ.
  Positive rows (states) in this column correspond to states s such that $Q^\pi(a, s) > V^\pi(s)$.

Theorem 8. *Let $\mathbf{\Delta}^\pi(\gamma) = \mathbf{q}_a^\pi(\gamma) - \mathbf{v}^\pi(\gamma)$. For a fixed state $s \in S$, write $\mathbf{\Delta}_s^\pi(\gamma) = \frac{A(\gamma)}{B(\gamma)}$, with polynomials A, B. Factor out the multiplicities of the root $\gamma = 1$: $A(\gamma) = (1-\gamma)^{c_1} A_1(\gamma),\ B(\gamma) = (1-\gamma)^{c_2} B_1(\gamma)$, where $c_1, c_2 \in \mathbb{Z}_{\geq 0}$ and $A_1(1) \neq 0,\ B_1(1) \neq 0$. Define $z_s(\gamma) = \frac{A_1(\gamma)}{B_1(\gamma)}$ Then, $z_s(1) = \psi_{j_0}^{a,\pi}(s),\ j_0 = \min\{j \mid \psi_j^{a,\pi}(s) \neq 0\}$ where $\psi^{a,\pi}$ is defined in Algorithm 3.*

Proof. The Laurent series expansion of the value function of a policy π and discount factor γ is given by [40]:

$$\mathbf{v}_\gamma^\pi = (1 + \rho) \sum_{j=-1}^{\infty} \rho^j \mathbf{y}_j \tag{7}$$

where $\rho = \frac{1-\gamma}{\gamma}$, and $\mathbf{y}_j = \mathbf{y}_j^\pi$ as defined in Algorithm 3.
Now consider the term:

$$\mathbf{q}_a^\pi - \mathbf{v}^\pi = \mathbf{r}_a + (\gamma P_a - I)\, \mathbf{v}^\pi$$

Replacing $\mathbf{v}^\pi$ from (7) we get:

$$\begin{aligned} \mathbf{q}_a^\pi - \mathbf{v}^\pi &= \mathbf{r}_a + [P_a - (1+\rho)I] \sum_{j=-1}^{\infty} \rho^j \mathbf{y}_j \\ &= \mathbf{r}_a + \sum_{j=-1}^{\infty} \rho^j \left[P_a - I - \rho I\right] \mathbf{y}_j \\ &= \mathbf{r}_a + \sum_{j=-1}^{\infty} \rho^j \left(P_a - I\right) \mathbf{y}_j - \rho^{j+1} \mathbf{y}_j \\ &= \rho^{-1}(P_a - I)\mathbf{y}_{-1} + [\mathbf{r}_a + (P_a - I)\mathbf{y}_0 - \mathbf{y}_{-1}] + \sum_{j=1}^{\infty} \rho^j \left[(P_a - I)\mathbf{y}_j - \mathbf{y}_{j-1}\right] \\ &= \sum_{j=-1}^{\infty} \rho^j \psi_j^{a,\pi} \end{aligned}$$

Again consider the term $\mathbf{q}_a^\pi - \mathbf{v}^\pi$ using equation (6). We have:

$$\mathbf{q}_a^\pi - \mathbf{v}^\pi = \mathbf{r}_a + (\gamma P_a - I)\, \frac{\mathbf{n}_\pi}{d_\pi}$$

Thus each entry of $\mathbf{q}_a^\pi - \mathbf{v}^\pi$ is a ratio of two polynomials say A and B. Write $A(\gamma) = (1-\gamma)^{m_1} A_1(\gamma)$ and $B(\gamma) = (1-\gamma)^{m_2} B_1(\gamma)$ where m_1, m_2 are integers greater than or equal to zero such that $A_1(1) \neq 0 \ \wedge \ B_1(1) \neq 0$. Now consider the function:

$$f(\gamma) = \frac{A(\gamma)}{B(\gamma)} = (1-\gamma)^{m_1 - m_2} \frac{A_1(\gamma)}{B_1(\gamma)}.$$

Let $t = m_1 - m_2$ and $z(\gamma) = \frac{A_1(\gamma)}{B_1(\gamma)}$, then we have two cases:
Case 1: $t \geq 0$
The Laurent series of f around $\gamma = 1$ is of the form:

$$c_t(1-\gamma)^t + c_{t+1}(1-\gamma)^{t+1} + ...$$

It is clear that $c_t = \frac{f^{(t)}(1)}{t!} = z(1)$.
Case 2: $t < 0$
The Laurent series of f around $\gamma = 1$ is of the form:

$$\frac{c_t}{(1-\gamma)^{-t}} + \frac{c_{t+1}}{(1-\gamma)^{-(t+1)}} + ...$$

Here we have: $c_t = \lim_{\gamma \to 1}(1-\gamma)^{-t} f(\gamma) = z(1)$.
Therefore the first non-zero term of the Laurent series expansion of $\mathbf{q}_a^\pi - \mathbf{v}^\pi$ at state s is given by: $\psi_{j_0}^{a,\pi}(s) = z(1)$, where j_0 is the index of the first non-zero term of $\psi^{a,\pi}$. $\square$