Reinforcement Learning: Theory and Methods

Sourangshu Ghosh[1]

[1] Department of Civil Engineering, Indian Institute of Science Bangalore,
sourangshug@iisc.ac.in

June 12, 2026

Abstract

Reinforcement learning (RL) provides a rigorous mathematical framework for sequential decision-making under uncertainty and has emerged as one of the foundational paradigms of modern artificial intelligence. This book presents a comprehensive and mathematically rigorous treatment of reinforcement learning, beginning with the measure-theoretic foundations of Markov Decision Processes (MDPs) and extending to modern deep reinforcement learning methods. The text develops the theory of measurable spaces, stochastic kernels, admissible policies, induced probability measures, and controlled stochastic processes in both finite and general state spaces. Building upon these foundations, the book systematically formulates value functions, return functionals, Bellman equations, and dynamic programming principles using tools from probability theory, stochastic processes, functional analysis, and operator theory. Particular emphasis is placed on contraction mappings, fixed-point theory, monotone operators, weighted norm formulations, spectral interpretations, and nonlinear operator geometry, thereby providing a rigorous analytical framework for understanding convergence, stability, and optimality in reinforcement learning algorithms. Classical methods such as value iteration, policy iteration, temporal-difference learning, Q-learning, and policy-gradient methods are derived and analyzed in a unified mathematical setting, highlighting the deep connections between reinforcement learning, stochastic control, and optimization theory.

The book further develops the mathematical principles underlying deep reinforcement learning, including stabilization mechanisms such as experience replay, target networks, Double DQN, dueling architectures, and prioritized replay, while providing geometric and operator-theoretic interpretations of their behavior. A major focus is devoted to the exploration–exploitation trade-off through regret minimization, Bayesian exploration, entropy-regularized control, optimism under uncertainty, and stochastic control perspectives. The text also addresses central theoretical and computational challenges in reinforcement learning, including sample inefficiency, instability under function approximation, reward shaping, and the curse of dimensionality. In addition, emerging research directions such as offline reinforcement learning, multi-agent systems, safe reinforcement learning, and theoretical generalization guarantees are examined within a unified mathematical framework. Supported by extensive theoretical derivations, rigorous proofs, and illustrative visualizations, the book is intended to serve both as an advanced graduate-level introduction and as a comprehensive reference for researchers and practitioners seeking a deep understanding of the mathematical foundations, algorithmic structures, and modern developments of reinforcement learning and sequential decision-making systems.

Keywords: Reinforcement Learning; Markov Decision Processes; Dynamic Programming; Bellman Equations; Deep Reinforcement Learning; Stochastic Control; Sequential Decision-Making; Operator Theory; Machine Learning; Artificial Intelligence

Contents

List of Figures

1 Introduction

Reinforcement Learning (RL) is a mathematical framework for modeling and solving problems of sequential decision-making under uncertainty. At its core, RL concerns an *agent* interacting with an *environment* over discrete time steps $t = 0, 1, 2, \ldots$, generating a stochastic process of states, actions, and rewards. The fundamental objective is to determine a decision-making strategy that maximizes cumulative reward over time.

The modern theory of reinforcement learning has been profoundly shaped by the foundational works of Sutton and Barto (1998) [1], Bertsekas and Tsitsiklis (1996) [2], and Puterman (2014) [3]. Sutton and Barto (1998) [1] developed the conceptual and algorithmic foundations of reinforcement learning by formalizing the interaction between an agent and an environment through rewards, value functions, temporal-difference learning, Monte Carlo methods, and policy optimization, presenting reinforcement learning as a computational framework for sequential decision-making under uncertainty. Bertsekas and Tsitsiklis (1996) [2] approached the subject from the perspective of stochastic control and dynamic programming, introducing the framework of neuro-dynamic programming, where function approximation and neural networks are combined with approximate dynamic programming techniques to overcome the curse of dimensionality in large-scale control and optimization problems. Puterman (2014) [3] provided the rigorous mathematical foundations of Markov decision processes, including measurable state spaces, Bellman optimality equations, contraction mappings, policy iteration, value iteration, and the existence of optimal stationary policies, thereby establishing the theoretical backbone upon which much of modern reinforcement learning is constructed. Together, these three works unified ideas from stochastic processes, optimal control, dynamic programming, and machine learning into a mathematically coherent framework that continues to underpin contemporary developments in reinforcement learning and deep reinforcement learning.

Formally, let $(\Omega, \mathcal{F}, \mathbb{P})$ be an underlying probability space. At each time t, the agent observes a state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$ according to a (possibly stochastic) decision rule, and subsequently receives a scalar reward $r_{t+1} \in \mathbb{R}$, while the environment transitions to a new state s_{t+1}. The evolution of the system induces a trajectory:

$$(s_0, a_0, r_1, s_1, a_1, r_2, s_2, \dots)$$

which is, in general, a stochastic process adapted to an appropriate filtration $\{\mathcal{F}_t\}_{t \geq 0}$.

A distinguishing feature of RL is that the agent does not have access to explicit supervision in the

form of labeled input-output pairs. Instead, learning is driven by scalar reward signals that may be *delayed*, *sparse*, and *noisy*. Consequently, the contribution of an individual action to long-term performance is not directly observable, leading to the classical *credit assignment problem*. This temporal coupling introduces significant mathematical and computational challenges that are absent in standard supervised learning.

To formalize the objective, define the (discounted) return:

$$G_t := \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \quad \gamma \in [0, 1) \tag{1}$$

whenever the series converges almost surely or in expectation. The goal is to identify a policy π—that is, a mapping from states to probability distributions over actions—such that the expected return is maximized:

$$\sup_{\pi} \mathbb{E}_{\pi} [G_0]$$

where the expectation is taken with respect to the probability measure induced by the interaction between the policy and the environment dynamics.

It is instructive to contrast RL with other dominant paradigms in statistical learning:

- **Supervised Learning:** Given a dataset $\{(x_i, y_i)\}_{i=1}^n$ drawn from an unknown distribution, the objective is to learn a function f minimizing a risk functional of the form

  $$\mathbb{E}_{(x,y)}[\ell(f(x), y)]$$

  Here, feedback is immediate and explicit via labels y_i.

- **Unsupervised Learning:** One seeks to uncover latent structure in data $\{x_i\}_{i=1}^n$, typically by optimizing an objective such as likelihood or reconstruction error, without access to target outputs.

- **Reinforcement Learning:** In contrast, the data-generating process depends on the learner's actions, and the objective functional involves expectations over entire trajectories:

  $$\mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right]$$

  Thus, RL inherently couples learning with control.

Another fundamental aspect of RL is the trade-off between *exploration* and *exploitation*. Since the agent typically lacks complete knowledge of the environment dynamics, it must balance:

- *Exploration:* selecting actions to acquire information about the environment,
- *Exploitation:* selecting actions believed to maximize expected reward based on current knowledge.

This trade-off can be formalized in various ways, often leading to stochastic optimization problems with partial information.

From a mathematical perspective, RL can be viewed as a synthesis of several disciplines, including stochastic processes, optimal control theory, dynamic programming, and statistical learning theory. In particular, the foundational structure of RL is typically formalized via Markov Decision Processes (MDPs), which we introduce in the subsequent section.

Finally, we emphasize that the central challenge in RL lies not merely in evaluating a fixed decision rule, but in solving a *nested optimization problem* in which the policy both influences and depends on the distribution of future data. This intrinsic feedback loop is what distinguishes RL from classical inference problems and underlies much of its theoretical richness and practical difficulty.

2 Markov Decision Processes

In order to formalize sequential decision-making problems, we adopt a measure-theoretic framework that accommodates both finite and general (possibly uncountable) state and action spaces.

2.1 Measurable Spaces and Stochastic Kernels

We begin by formalizing the underlying measurable structures required to define controlled stochastic processes in full generality.

The mathematical foundations of measurable spaces and stochastic kernels in reinforcement learning and stochastic control are rigorously developed in the works of Bertsekas and Shreve (1996) [4], Hernández-Lerma and Lasserre (2012) [5], and Cohn (2013) [6]. Bertsekas and Shreve (1996) [4] formulated discrete-time stochastic optimal control within a measure-theoretic framework, carefully defining measurable state and action spaces, stochastic transition kernels, admissible policies, and dynamic programming operators, thereby establishing a rigorous analytical basis for stochastic control and sequential decision-making problems. Hernández-Lerma and Lasserre (2012) [5] further expanded these ideas by developing the theory of discrete-time Markov control processes on Borel spaces, emphasizing measurable selection theorems, existence of optimal policies, discounted and average-cost criteria, and the probabilistic structure of controlled Markov processes under general state spaces. Complementing these works, Cohn (2013) [6] gave a treatment of the measure theory that provides the foundational mathematical machinery required for modern reinforcement learning theory, including sigma-algebras, measurable functions, product measures, integration theory, probability measures on measurable spaces, and convergence theorems that underpin stochastic kernels and transition probability models. Measure Theory Together, these works establish the rigorous measure-theoretic and probabilistic framework necessary for analyzing Markov decision processes, stochastic dynamic programming, and modern reinforcement learning algorithms in both finite and general state spaces.

2.1.1 Measurable and Standard Borel Spaces

Let $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$ be measurable spaces. In most applications, $\mathcal{S}$ and $\mathcal{A}$ are endowed with topological structures (e.g., metric spaces), and $\mathcal{B}(\cdot)$ denotes the associated Borel σ-algebra.

Definition 2.1 (Standard Borel Space)**.** A measurable space $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ is called a *standard Borel space* if there exists a Polish topology on $\mathcal{X}$ (i.e., a separable and completely metrizable topology) such that $\mathcal{B}(\mathcal{X})$ coincides with the Borel σ-algebra generated by this topology.

Standard Borel spaces constitute a sufficiently rich class for most applications in reinforcement learning and stochastic control, while ensuring the validity of key measurable selection theorems and disintegration results.

Definition 2.2 (Stochastic Kernel)**.** A stochastic kernel P on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ given $(\mathcal{S} \times \mathcal{A}, \mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A}))$ is a mapping

$$P : \mathcal{B}(\mathcal{S}) \times \mathcal{S} \times \mathcal{A} \to [0, 1]$$

satisfying:

1. For every $(s, a) \in \mathcal{S} \times \mathcal{A}$, the map
$$B \mapsto P(B|s, a)$$
defines a probability measure on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$.

2. For every $B \in \mathcal{B}(\mathcal{S})$, the function
$$(s, a) \mapsto P(B|s, a)$$
is $\mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A})$-measurable.

Thus, a stochastic kernel assigns to each state-action pair (s, a) a probability measure on the next-state space, in a manner that is measurable with respect to (s, a). Given a stochastic kernel P and any bounded measurable function $f : \mathcal{S} \to \mathbb{R}$, one defines the integral operator:

$$(Pf)(s, a) := \int_{\mathcal{S}} f(s')\, P(ds'|s, a) \tag{2}$$

This induces a mapping

$$P : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S} \times \mathcal{A})$$

where $\mathcal{B}_b(\mathcal{S})$ denotes the space of bounded measurable functions on $\mathcal{S}$. The measurability condition in the definition ensures that $(s, a) \mapsto (Pf)(s, a)$ is measurable.

2.1.2 Product Structure and Disintegration

Let μ be a probability measure on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and P a stochastic kernel. Then one can define a probability measure μP on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ via:

$$(\mu P)(B) := \int_{\mathcal{S}} P(B|s, a)\, \mu(ds) \tag{3}$$

when a is fixed, or more generally, for a joint distribution over (s, a). More generally, given a probability measure ν on $(\mathcal{S} \times \mathcal{A})$, one can construct a measure on $\mathcal{S} \times \mathcal{A} \times \mathcal{S}$ by:

$$\nu(ds, da)P(ds'|s, a)$$

which plays a central role in defining trajectory measures.

Conversely, under appropriate regularity conditions (e.g., standard Borel spaces), any joint probability measure on $(\mathcal{S} \times \mathcal{S})$ admits a disintegration into a marginal and a stochastic kernel, justifying the interpretation of P as a conditional distribution.

If $\mathcal{S}$ is finite, then $\mathcal{B}(\mathcal{S}) = 2^{\mathcal{S}}$, and a stochastic kernel reduces to a collection of transition probabilities:

$$P(s'|s, a), \quad s, s' \in \mathcal{S},\ a \in \mathcal{A}$$

satisfying:

$$\sum_{s' \in \mathcal{S}} P(s'|s, a) = 1 \tag{4}$$

The measurability condition is essential for ensuring that expectations and iterated integrals involving the kernel are well-defined, particularly when constructing stochastic processes via extension theorems. In the context of reinforcement learning, stochastic kernels serve as the mathematical abstraction of environment dynamics, encapsulating both deterministic and stochastic transitions within a unified framework.

2.2 Definition of Markov Decision Process

We now provide a mathematically rigorous formulation of a Markov Decision Process (MDP), suitable for both finite and general (uncountable) state and action spaces.

The theory of Markov decision processes (MDPs) has been systematically developed through the foundational works of Puterman (2014) [3], Feinberg and Shwartz (2012) [7], and Hernández-Lerma and Lasserre (2012) [8]. Puterman (2014) [3] provided one of the most comprehensive treatments of MDPs, rigorously formulating sequential decision-making problems through states, actions, transition probabilities, reward structures, and policies, while developing the mathematical foundations of Bellman equations, value iteration, policy iteration, discounted and average-cost criteria, and optimality theory for finite and countable state spaces. Feinberg and Shwartz (2012) [7] further expanded the theory through a broad collection of advanced topics in controlled stochastic systems, including continuous-time and discrete-time MDPs, partially observable decision processes, risk-sensitive control, constrained optimization, and computational methods, thereby connecting classical stochastic control theory with modern

reinforcement learning and operations research applications. Hernández-Lerma and Lasserre (2012) [8] extended the mathematical framework to more general Borel state and action spaces, emphasizing measurable selection theory, ergodic control, average-cost optimality, stability properties, and the existence and structure of optimal policies under highly general assumptions. Together, these works established the modern theoretical foundation of Markov decision processes and provided the rigorous probabilistic, analytical, and optimization framework that underlies contemporary reinforcement learning, stochastic control, and dynamic programming theory.

2.2.1 Basic Definition

Definition 2.3 (Markov Decision Process)**.** A Markov Decision Process (MDP) is a tuple

$$(\mathcal{S}, \mathcal{A}, P, R, \gamma),$$

where:

- $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ is the state space, assumed to be a standard Borel space,
- $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$ is the action space, also assumed to be a standard Borel space,
- $P(\cdot|s, a)$ is a stochastic kernel on $\mathcal{S}$ given $\mathcal{S} \times \mathcal{A}$,
- R is either:
  - a measurable function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, or
  - more generally, a stochastic kernel $\mathcal{R}(dr|s, a)$ on $\mathbb{R}$,
- $\gamma \in [0, 1)$ is a discount factor.

If $\mathcal{R}(dr|s, a)$ is a reward kernel, then the expected reward function is defined as

$$R(s, a) := \int_{\mathbb{R}} r \, \mathcal{R}(dr|s, a), \tag{5}$$

provided the integral exists. In many applications, it is assumed that rewards are bounded, i.e.,

$$\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} |R(s, a)| < \infty, \tag{6}$$

which ensures well-definedness of infinite-horizon discounted returns. To fully specify the stochastic system, one introduces an initial probability measure ρ on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$. The complete data of the decision process is then given by

$$(\mathcal{S}, \mathcal{A}, P, R, \gamma, \rho).$$

A policy π specifies how actions are selected based on available information. In the most general formulation, a policy is a sequence of stochastic kernels

$$\pi_t(da_t \mid h_t),$$

where $h_t = (s_0, a_0, \ldots, s_t)$ denotes the history up to time t. Such policies are called *history-dependent*.

Under standard assumptions (e.g., standard Borel spaces), it can be shown that restricting attention to *Markov policies* $\pi(da|s)$ entails no loss of optimality for discounted problems.

2.2.2 Construction of the Controlled Process

Given an initial distribution ρ and a policy π, the MDP induces a probability measure on the infinite trajectory space

$$\Omega := (\mathcal{S} \times \mathcal{A})^{\mathbb{N}},$$

equipped with the product σ-algebra. The existence of such a measure $\mathbb{P}_\rho^\pi$ is guaranteed by the *Ionescu–Tulcea extension theorem*, applied iteratively to the sequence of stochastic kernels:

$$s_0 \sim \rho, \tag{7}$$

$$a_t \mid h_t \sim \pi_t(\cdot \mid h_t), \tag{8}$$

$$s_{t+1} \mid (h_t, a_t) \sim P(\cdot \mid s_t, a_t). \tag{9}$$

This construction yields a well-defined stochastic process

$$\{(s_t, a_t)\}_{t \geq 0}$$

on $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$. Define the discounted return:

$$G_0 := \sum_{t=0}^{\infty} \gamma^t r_{t+1} \tag{10}$$

If the reward function is bounded, then the series converges absolutely almost surely, and:

$$|G_0| \leq \frac{\sup_{s,a} |R(s,a)|}{1-\gamma} \tag{11}$$

Thus, G_0 is integrable under $\mathbb{P}_\rho^\pi$, and the performance of a policy is well-defined as:

$$J(\pi) := \mathbb{E}_\rho^\pi[G_0] \tag{12}$$

The MDP can equivalently be described in terms of:

- A controlled Markov chain with transition kernel P and a stochastic control problem with cost functional $-G_0$,
- A dynamic programming system characterized by Bellman equations.

If $\mathcal{S}$ and $\mathcal{A}$ are finite, then:

- P reduces to a transition probability matrix and R is a finite table,
- All measurability issues are trivial and the trajectory measure reduces to a standard discrete-time Markov chain controlled by π.

The assumption that $\mathcal{S}$ and $\mathcal{A}$ are standard Borel spaces ensures the applicability of measurable selection theorems, which are essential for proving the existence of optimal policies. The restriction $\gamma < 1$ guarantees that the Bellman operator is a contraction in appropriate function spaces, forming the basis for convergence results in dynamic programming.

2.3 Policies

We now formalize the notion of policies in a general measure-theoretic framework. Policies specify how actions are selected based on available information and play a central role in defining the controlled stochastic process.

The mathematical theory of policies in stochastic control and reinforcement learning has been extensively developed in the works of Bertsekas (2012) [9], Puterman (2014) [3], and Hernández-Lerma and Lasserre (2012) [5]. Bertsekas (2012) [9] presents policies as fundamental decision rules governing sequential optimization in dynamic systems, rigorously analyzing deterministic and stochastic policies, stationary

and nonstationary strategies, and their roles in dynamic programming, Bellman optimality, and optimal control under discounted and finite-horizon formulations. Puterman (2014) [3] further formalized the theory of policies within the framework of Markov decision processes, carefully defining randomized and history-dependent policies, Markov policies, stationary optimal policies, and conditions under which optimal stationary strategies exist for discounted, finite-horizon, and average-cost criteria. Hernández-Lerma and Lasserre (2012) [5] extended these ideas to general Borel state and action spaces using a rigorous measure-theoretic framework, emphasizing measurable policies, stochastic kernels, measurable selection theorems, and the existence and structural properties of optimal control laws in discrete-time Markov control processes. Collectively, these works established the modern theoretical understanding of policies as measurable decision mappings that determine the evolution of controlled stochastic systems and form the central mathematical object underlying reinforcement learning, stochastic dynamic programming, and optimal control theory.

Let the history at time t be defined as

$$h_t := (s_0, a_0, s_1, a_1, \ldots, s_t) \in \mathcal{H}_t := (\mathcal{S} \times \mathcal{A})^t \times \mathcal{S}$$

We equip $\mathcal{H}_t$ with the product σ-algebra. Let us also define $\{\mathcal{F}_t\}_{t\geq 0}$ denote the natural filtration generated by the process, i.e.,

$$\mathcal{F}_t := \sigma(s_0, a_0, \ldots, s_t)$$

Definition 2.4 (History-Dependent Policy)**.** A (randomized) history-dependent policy is a sequence $\pi = \{\pi_t\}_{t\geq 0}$ of stochastic kernels

$$\pi_t(da_t \mid h_t),$$

where each π_t is a stochastic kernel on $\mathcal{A}$ given $\mathcal{H}_t$, i.e.,

- For each history $h_t \in \mathcal{H}_t$, $\pi_t(\cdot \mid h_t)$ is a probability measure on $\mathcal{A}$,
- For each measurable set $B \in \mathcal{B}(\mathcal{A})$, the map $h_t \mapsto \pi_t(B \mid h_t)$ is measurable.

Such policies are also called *non-anticipative*, since a_t depends only on past and present information.

2.3.1 Markov Policies

A central simplification in the theory of controlled stochastic processes is the restriction of admissible policies to those that depend only on the current state. This leads to the notion of Markov policies, which are defined in terms of stochastic kernels on the state space rather than the full history space.

Definition 2.5 (Markov Policy)**.** A policy $\pi = \{\pi_t\}_{t\geq 0}$ is called a *Markov policy* if there exists a sequence of stochastic kernels $\{\tilde{\pi}_t\}_{t\geq 0}$ on $\mathcal{A}$ given $\mathcal{S}$ such that

$$\pi_t(da_t \mid h_t) = \tilde{\pi}_t(da_t \mid s_t) \quad \text{for all } t \geq 0 \text{ and all histories } h_t. \tag{13}$$

Thus, under a Markov policy, the conditional distribution of the action at time t depends only on the current state s_t, and not on the entire past trajectory $h_t = (s_0, a_0, \ldots, s_t)$. In particular, for any measurable set $B \in \mathcal{B}(\mathcal{A})$, one has

$$\mathbb{P}^{\pi}_{\rho}(a_t \in B \mid \mathcal{F}_t) = \tilde{\pi}_t(B \mid s_t), \tag{14}$$

where $\{\mathcal{F}_t\}$ is the natural filtration generated by the process.

The measurability requirement for $\tilde{\pi}_t$ ensures that for each $B \in \mathcal{B}(\mathcal{A})$, the mapping

$$s \mapsto \tilde{\pi}_t(B \mid s)$$

is $\mathcal{B}(\mathcal{S})$-measurable. This guarantees that the composition of the policy with the state process yields a well-defined stochastic kernel on $\mathcal{A}$ adapted to the filtration. Under a Markov policy, the controlled

process $\{s_t\}$ evolves as a time-inhomogeneous Markov chain. Indeed, combining the Markov property of the transition kernel P with the structure of π, one obtains, for all measurable sets $B \in \mathcal{B}(\mathcal{S})$,

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid \mathcal{F}_t) = \int_{\mathcal{A}} P(B \mid s_t, a)\, \tilde{\pi}_t(da \mid s_t) \tag{15}$$

Defining the induced transition kernel

$$P_t^\pi(B \mid s) := \int_{\mathcal{A}} P(B \mid s, a)\, \tilde{\pi}_t(da \mid s) \tag{16}$$

it follows that

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid s_t = s) = P_t^\pi(B \mid s) \tag{17}$$

which shows that $\{s_t\}$ is a (generally time-inhomogeneous) Markov process under π. In the special case where $\tilde{\pi}_t \equiv \pi$ for all t, the policy is said to be stationary, and the induced kernel simplifies to

$$P^\pi(B \mid s) = \int_{\mathcal{A}} P(B \mid s, a)\, \pi(da \mid s) \tag{18}$$

so that $\{s_t\}$ becomes a time-homogeneous Markov chain.

A fundamental result in the theory of Markov Decision Processes is that, under standard assumptions (e.g., $\mathcal{S}$ and $\mathcal{A}$ are standard Borel spaces, rewards are bounded, and $\gamma < 1$), restricting attention to Markov policies entails no loss of optimality. More precisely, for any history-dependent policy π, there exists a Markov policy $\tilde{\pi}$ such that

$$\mathbb{E}_\rho^\pi\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] = \mathbb{E}_\rho^{\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \tag{19}$$

This result follows from conditional expectation arguments and the tower property, and reflects the fact that the state s_t is a sufficient statistic for optimal decision-making. From a functional perspective, a Markov policy induces a linear operator on bounded measurable functions $f : \mathcal{S} \to \mathbb{R}$ via

$$(P^\pi f)(s) := \int_{\mathcal{A}} \int_{\mathcal{S}} f(s')\, P(ds' \mid s, a)\, \pi(da \mid s) \tag{20}$$

which plays a central role in dynamic programming and the analysis of Bellman operators.

In summary, Markov policies reduce the complexity of admissible control strategies by restricting dependence to the current state, while preserving optimality and enabling a tractable operator-theoretic formulation of the decision process.

2.3.2 Stationary Policies

A particularly important class of policies arises when the decision rule does not vary with time. These are known as stationary (or time-homogeneous) policies and play a central role in infinite-horizon problems due to their structural simplicity and optimality properties.

Definition 2.6 (Stationary Policy)**.** A Markov policy $\pi = \{\pi_t\}_{t \geq 0}$ is said to be *stationary* if there exists a single stochastic kernel π on $\mathcal{A}$ given $\mathcal{S}$ such that

$$\pi_t(\cdot \mid s) = \pi(\cdot \mid s) \quad \text{for all } s \in \mathcal{S},\ t \geq 0. \tag{21}$$

Thus, under a stationary policy, the conditional distribution of the action depends only on the current state and is invariant across time. In particular, for any measurable set $B \in \mathcal{B}(\mathcal{A})$, one has

$$\mathbb{P}_\rho^\pi(a_t \in B \mid \mathcal{F}_t) = \pi(B \mid s_t), \tag{22}$$

for all $t \geq 0$, where $\{\mathcal{F}_t\}$ is the natural filtration generated by the process.

When a stationary policy is employed, the controlled state process $\{s_t\}_{t\geq 0}$ becomes a time-homogeneous Markov chain. Indeed, combining the stationarity of π with the transition kernel P, we obtain for all measurable sets $B \in \mathcal{B}(\mathcal{S})$:

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid \mathcal{F}_t) = \int_{\mathcal{A}} P(B \mid s_t, a)\, \pi(da \mid s_t). \tag{23}$$

Defining the induced transition kernel

$$P^\pi(B \mid s) := \int_{\mathcal{A}} P(B \mid s, a)\, \pi(da \mid s), \tag{24}$$

it follows that

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid s_t = s) = P^\pi(B \mid s), \tag{25}$$

which is independent of time t. Hence, $\{s_t\}$ is a time-homogeneous Markov process under π.

From the perspective of performance evaluation, stationary policies yield a time-invariant value function. Specifically, defining the value function associated with π as

$$V^\pi(s) := \mathbb{E}_s^\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right], \tag{26}$$

where $\mathbb{E}_s^\pi$ denotes expectation conditioned on $s_0 = s$, one obtains the fixed-point equation

$$V^\pi(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s, a) \right] \pi(da \mid s). \tag{27}$$

Equivalently, introducing the linear operator T^π acting on bounded measurable functions $V : \mathcal{S} \to \mathbb{R}$,

$$(T^\pi V)(s) := \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s, a) \right] \pi(da \mid s), \tag{28}$$

the value function satisfies

$$V^\pi = T^\pi V^\pi. \tag{29}$$

Under standard assumptions (e.g., bounded rewards and $\gamma \in [0,1)$), the operator T^π is a contraction on the Banach space $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$, satisfying

$$\|T^\pi V - T^\pi W\|_\infty \leq \gamma \|V - W\|_\infty, \tag{30}$$

for all bounded measurable functions V, W. Consequently, by the Banach fixed-point theorem, T^π admits a unique fixed point, which coincides with V^π.

A fundamental structural result in infinite-horizon discounted MDPs is that stationary policies are sufficient for optimality. More precisely, under standard regularity conditions, there exists a stationary policy π^* such that

$$V^{\pi^*}(s) = \sup_\pi V^\pi(s) \quad \text{for all } s \in \mathcal{S}. \tag{31}$$

This justifies restricting the search for optimal policies to the class of stationary Markov policies, significantly reducing the complexity of the optimization problem.

In summary, stationary policies provide a time-invariant and analytically tractable framework for decision-making, enabling the formulation of the control problem as a fixed-point equation and forming the foundation of dynamic programming methods in reinforcement learning.

2.3.3 Deterministic Policies

A particularly important subclass of policies is obtained when the randomness in action selection is removed entirely. In this case, the agent selects a single action in each state according to a measurable decision rule.

Definition 2.7 (Deterministic Policy)**.** A policy is called *deterministic* if there exists a measurable function $\mu : \mathcal{S} \to \mathcal{A}$ such that

$$\pi(\cdot|s) = \delta_{\mu(s)}(\cdot), \tag{32}$$

where $\delta_{\mu(s)}$ denotes the Dirac probability measure concentrated at the point $\mu(s)$.

Thus, under a deterministic policy, the conditional distribution of the action given the current state degenerates to a point mass. In particular, for any measurable set $B \in \mathcal{B}(\mathcal{A})$, one has

$$\pi(B \mid s) = \delta_{\mu(s)}(B) = \begin{cases} 1, & \text{if } \mu(s) \in B, \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Equivalently, the action process satisfies

$$a_t = \mu(s_t) \quad \text{almost surely for all } t \geq 0. \tag{34}$$

The measurability of μ with respect to $\mathcal{B}(\mathcal{S})$ and $\mathcal{B}(\mathcal{A})$ is essential to ensure that the induced mapping

$$s \mapsto \delta_{\mu(s)}(B)$$

is measurable for every $B \in \mathcal{B}(\mathcal{A})$, thereby guaranteeing that π is a valid stochastic kernel. This condition is nontrivial in general measurable spaces and is typically ensured by assuming that $\mathcal{S}$ and $\mathcal{A}$ are standard Borel spaces.

Under a deterministic policy μ, the controlled state process $\{s_t\}_{t\geq 0}$ evolves according to the induced transition kernel

$$P^{\mu}(B \mid s) := P(B \mid s, \mu(s)) \tag{35}$$

for all $B \in \mathcal{B}(\mathcal{S})$. Consequently, for any $t \geq 0$,

$$\mathbb{P}^{\mu}_{\rho}(s_{t+1} \in B \mid s_t = s) = P(B \mid s, \mu(s)) \tag{36}$$

which shows that $\{s_t\}$ is a (time-homogeneous) Markov process under μ. The performance of a deterministic policy is characterized by its value function

$$V^{\mu}(s) := \mathbb{E}^{\mu}_{s}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \tag{37}$$

which satisfies the Bellman equation

$$V^{\mu}(s) = R(s, \mu(s)) + \gamma \int_{\mathcal{S}} V^{\mu}(s')\, P(ds' \mid s, \mu(s)) \tag{38}$$

Equivalently, defining the operator T^{μ} acting on bounded measurable functions V as

$$(T^{\mu}V)(s) := R(s, \mu(s)) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s, \mu(s)) \tag{39}$$

one has

$$V^{\mu} = T^{\mu}V^{\mu}. \tag{40}$$

From an optimization perspective, deterministic policies are sufficient for optimality under broad conditions. In particular, when the action space is finite, or more generally when measurable selection

theorems apply (e.g., when $\mathcal{S}$ and $\mathcal{A}$ are standard Borel spaces and the reward and transition kernels satisfy appropriate regularity conditions), there exists an optimal policy μ^* such that

$$V^{\mu^*}(s) = \sup_{\pi} V^{\pi}(s) \quad \text{for all } s \in \mathcal{S} \tag{41}$$

and μ^* is deterministic and stationary. This result follows from the fact that the Bellman optimality operator involves a pointwise maximization:

$$V^*(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s,a) \right] \tag{42}$$

and measurable selection arguments ensure the existence of a measurable maximizer $\mu^*(s)$.

From a functional-analytic viewpoint, deterministic policies correspond to selecting extremal points of the convex set of stochastic kernels. Indeed, the set of all stochastic policies $\pi(\cdot|s)$ forms a convex set for each fixed s, and deterministic policies correspond to its extreme points. This observation is closely related to the fact that randomization is not required for optimality in standard MDPs.

In summary, deterministic policies represent the simplest class of admissible controls, eliminating randomness in action selection while preserving optimality properties. Their structure enables a direct connection between dynamic programming equations and pointwise optimization over the action space.

2.3.4 Classes of Policies

In the measure-theoretic formulation of controlled stochastic processes, it is useful to distinguish several classes of admissible policies according to the information they utilize and their structural restrictions. These classes form a hierarchy that reflects increasing generality at the cost of analytical and computational complexity.

Let Π denote the class of all admissible policies, where a policy $\pi \in \Pi$ is a sequence of stochastic kernels $\{\pi_t\}_{t \geq 0}$ with

$$\pi_t(da_t \mid h_t),$$

such that for each $t \geq 0$, π_t is a stochastic kernel on $\mathcal{A}$ given the history space $\mathcal{H}_t$, and for every $B \in \mathcal{B}(\mathcal{A})$, the mapping $h_t \mapsto \pi_t(B \mid h_t)$ is measurable. These policies are also referred to as history-dependent or non-anticipative, since the action at time t depends only on past and present observations.

A subclass of Π is given by the set Π_M of Markov policies, consisting of all policies $\pi = \{\pi_t\}$ for which there exist stochastic kernels $\tilde{\pi}_t$ on $\mathcal{A}$ given $\mathcal{S}$ satisfying

$$\pi_t(da_t \mid h_t) = \tilde{\pi}_t(da_t \mid s_t) \quad \text{for all } t \geq 0. \tag{43}$$

Thus, Markov policies restrict the dependence of the action to the current state s_t, which acts as a sufficient statistic for decision-making. A further restriction leads to the class Π_S of stationary Markov policies, defined as those policies for which there exists a single stochastic kernel π such that

$$\pi_t(\cdot \mid s) = \pi(\cdot \mid s) \quad \text{for all } t \geq 0. \tag{44}$$

In this case, the decision rule is invariant with respect to time, and the induced controlled process becomes time-homogeneous. Finally, the class Π_D of deterministic stationary policies consists of those policies for which there exists a measurable function $\mu : \mathcal{S} \to \mathcal{A}$ such that

$$\pi(da \mid s) = \delta_{\mu(s)}(da), \tag{45}$$

for all $s \in \mathcal{S}$. These policies correspond to selecting a single action in each state with probability one.

The above classes satisfy the inclusions

$$\Pi_D \subset \Pi_S \subset \Pi_M \subset \Pi. \tag{46}$$

Each inclusion follows directly from the definitions. Indeed, any deterministic stationary policy is a stationary policy by construction, since the Dirac measure $\delta_{\mu(s)}$ is a particular stochastic kernel. Similarly, any stationary policy is trivially a Markov policy with $\tilde{\pi}_t \equiv \pi$, and any Markov policy is a special case of a history-dependent policy where the dependence on history is reduced to the current state.

From an optimization perspective, this hierarchy is particularly significant because, under standard assumptions (e.g., $\mathcal{S}$ and $\mathcal{A}$ are standard Borel spaces, rewards are bounded, and $\gamma \in [0, 1)$), one can show that

$$\sup_{\pi \in \Pi} J(\pi) = \sup_{\pi \in \Pi_M} J(\pi) = \sup_{\pi \in \Pi_S} J(\pi) = \sup_{\pi \in \Pi_D} J(\pi), \tag{47}$$

where

$$J(\pi) := \mathbb{E}_\rho^\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right]. \tag{48}$$

This equality reflects the fact that increasingly restrictive policy classes do not reduce the achievable optimal value, despite significantly reducing the complexity of the admissible decision rules.

From a geometric viewpoint, for each fixed state $s \in \mathcal{S}$, the set of stochastic policies $\pi(\cdot \mid s)$ forms a convex set of probability measures on $\mathcal{A}$, while deterministic policies correspond to its extreme points. The fact that optimal policies can be chosen from Π_D is therefore closely related to the structure of the Bellman optimality operator, which involves pointwise maximization over $\mathcal{A}$.

In summary, the classification $\Pi_D \subset \Pi_S \subset \Pi_M \subset \Pi$ provides a rigorous framework for analyzing the complexity and structure of admissible policies, while the equivalence of their optimal values under standard assumptions justifies restricting attention to deterministic stationary policies in both theoretical analysis and algorithmic implementations.

2.3.5 Measurability and Admissibility

The notion of admissibility of a policy is fundamentally tied to the requirement that the controlled stochastic process be well-defined on an appropriate probability space. In particular, admissibility ensures that one can construct a probability measure on the infinite trajectory space and that all relevant random variables (states, actions, rewards) are measurable with respect to the induced filtration.

Let the trajectory space be defined as

$$\Omega := (\mathcal{S} \times \mathcal{A})^{\mathbb{N}}, \tag{49}$$

equipped with the product σ-algebra

$$\mathcal{F} := \bigotimes_{t=0}^{\infty} (\mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A})). \tag{50}$$

For each $t \geq 0$, define the canonical coordinate mappings

$$s_t(\omega) := \omega_{2t}, \quad a_t(\omega) := \omega_{2t+1}, \tag{51}$$

and the associated filtration

$$\mathcal{F}_t := \sigma(s_0, a_0, \ldots, s_t). \tag{52}$$

A policy $\pi = \{\pi_t\}_{t \geq 0}$ is said to be *admissible* if, for each $t \geq 0$, π_t is a stochastic kernel on $\mathcal{A}$ given $(\mathcal{H}_t, \mathcal{B}(\mathcal{H}_t))$, where $\mathcal{H}_t$ denotes the history space. That is, for every measurable set $B \in \mathcal{B}(\mathcal{A})$, the mapping

$$h_t \mapsto \pi_t(B \mid h_t)$$

is $\mathcal{B}(\mathcal{H}_t)$-measurable, and for each fixed h_t, $\pi_t(\cdot \mid h_t)$ is a probability measure on $\mathcal{A}$. This measurability condition is essential to ensure that the conditional distribution of a_t given the past is well-defined. More precisely, for each $t \geq 0$, the random variable a_t must satisfy

$$\mathbb{P}_\rho^\pi(a_t \in B \mid \mathcal{F}_t) = \pi_t(B \mid h_t), \tag{53}$$

for all $B \in \mathcal{B}(\mathcal{A})$, where $h_t = (s_0, a_0, \ldots, s_t)$ is the realized history. This requires that the mapping $h_t \mapsto \pi_t(B \mid h_t)$ be measurable so that it defines a valid version of a conditional probability.

The admissibility of π guarantees that the sequence of stochastic kernels

$$\rho, \quad \pi_0, \quad P, \quad \pi_1, \quad P, \quad \ldots$$

can be composed to define a unique probability measure $\mathbb{P}_\rho^\pi$ on $(\Omega, \mathcal{F})$. The existence and uniqueness of this measure follow from the Ionescu–Tulcea extension theorem, which applies to sequences of measurable stochastic kernels. Explicitly, for cylinder sets of the form

$$C = \{s_0 \in B_0, a_0 \in D_0, \ldots, s_t \in B_t\},$$

the measure $\mathbb{P}_\rho^\pi$ is defined recursively by

$$\mathbb{P}_\rho^\pi(C) = \int_{B_0} \rho(ds_0) \int_{D_0} \pi_0(da_0 \mid s_0) \int_{B_1} P(ds_1 \mid s_0, a_0) \cdots \int_{B_t} P(ds_t \mid s_{t-1}, a_{t-1}). \tag{54}$$

Moreover, admissibility ensures that the process $\{(s_t, a_t)\}_{t \geq 0}$ is adapted to the filtration $\{\mathcal{F}_t\}$ and that expectations of measurable functionals of the trajectory, such as the discounted return

$$G_0 = \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \tag{55}$$

are well-defined. In particular, if the reward function is measurable and bounded, then G_0 is a measurable random variable with respect to $\mathcal{F}$ and is integrable under $\mathbb{P}_\rho^\pi$.

It is important to emphasize that without the measurability condition on π_t, the composition of kernels may fail to produce a well-defined probability measure, and conditional expectations may not exist in a rigorous sense. Thus, admissibility is not merely a technical requirement but a fundamental condition ensuring the mathematical consistency of the reinforcement learning framework.

In summary, admissible policies are precisely those that are measurable with respect to the history and therefore compatible with the probabilistic structure of the controlled process. This guarantees the existence of a unique trajectory measure, the validity of conditional distributions, and the well-posedness of performance criteria.

2.3.6 Induced Probability Measure

Given an initial distribution ρ on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and an admissible policy $\pi = \{\pi_t\}_{t \geq 0}$, the controlled dynamics of the Markov Decision Process induce a probability measure on the infinite trajectory space. This construction is formalized using the Ionescu–Tulcea extension theorem.

Let the trajectory space be

$$\Omega := (\mathcal{S} \times \mathcal{A})^{\mathbb{N}}, \tag{56}$$

equipped with the product σ-algebra

$$\mathcal{F} := \bigotimes_{t=0}^{\infty} \left(\mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A})\right). \tag{57}$$

For each $t \geq 0$, define the coordinate projections

$$s_t(\omega) := \omega_{2t}, \quad a_t(\omega) := \omega_{2t+1}, \tag{58}$$

and the natural filtration

$$\mathcal{F}_t := \sigma(s_0, a_0, \ldots, s_t). \tag{59}$$

The evolution of the system is specified by the sequence of stochastic kernels

$$\rho, \quad \pi_0, \quad P, \quad \pi_1, \quad P, \quad \dots,$$

where ρ is the initial distribution on $\mathcal{S}$, each π_t is a stochastic kernel on $\mathcal{A}$ given $\mathcal{H}_t$, and P is the transition kernel on $\mathcal{S}$ given $\mathcal{S} \times \mathcal{A}$. These kernels define a consistent family of finite-dimensional distributions as follows.

For any $t \geq 0$ and measurable sets $B_0, \dots, B_t \in \mathcal{B}(\mathcal{S})$ and $D_0, \dots, D_{t-1} \in \mathcal{B}(\mathcal{A})$, define

$$\begin{aligned}&\mathbb{P}_\rho^\pi\big(s_0 \in B_0, a_0 \in D_0, \dots, s_t \in B_t\big) \\ &\quad := \int_{B_0} \rho(ds_0) \int_{D_0} \pi_0(da_0 \mid s_0) \int_{B_1} P(ds_1 \mid s_0, a_0) \int_{D_1} \pi_1(da_1 \mid h_1) \cdots \int_{B_t} P(ds_t \mid s_{t-1}, a_{t-1}),\end{aligned}$$

where $h_k = (s_0, a_0, \dots, s_k)$ denotes the history up to time k. The measurability assumptions on ρ, π_t, and P ensure that these iterated integrals are well-defined and that the resulting set function is a probability measure on cylinder sets. Moreover, these finite-dimensional distributions are consistent in the sense of Kolmogorov, i.e., they satisfy the compatibility condition under marginalization.

By the Ionescu–Tulcea extension theorem, there exists a unique probability measure $\mathbb{P}_\rho^\pi$ on $(\Omega, \mathcal{F})$ whose restriction to cylinder sets coincides with the above specification. This measure is called the *trajectory measure* induced by the policy π and the initial distribution ρ. Under the probability measure $\mathbb{P}_\rho^\pi$, the sequence of random variables $\{(s_t, a_t)\}_{t \geq 0}$ is well-defined and adapted to the filtration $\{\mathcal{F}_t\}$. Furthermore, the process satisfies the following conditional distributions almost surely:

$$\mathbb{P}_\rho^\pi(a_t \in D \mid \mathcal{F}_t) = \pi_t(D \mid h_t), \tag{60}$$

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid \mathcal{F}_t, a_t) = P(B \mid s_t, a_t), \tag{61}$$

for all $B \in \mathcal{B}(\mathcal{S})$ and $D \in \mathcal{B}(\mathcal{A})$.

This construction ensures that expectations of measurable functionals of the trajectory, such as

$$\mathbb{E}_\rho^\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right],$$

are well-defined, provided appropriate integrability conditions hold (e.g., bounded rewards). The measure $\mathbb{P}_\rho^\pi$ thus provides the rigorous probabilistic foundation for evaluating and comparing policies in reinforcement learning.

In summary, the sequence of kernels (ρ, π, P) uniquely determines a probability measure on the infinite trajectory space, and the Ionescu–Tulcea theorem guarantees both existence and uniqueness of this measure, thereby ensuring the well-posedness of the controlled stochastic process.

2.3.7 Structural Results

Under standard assumptions—namely that $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$ are standard Borel spaces, the reward function R is bounded and measurable, and the discount factor satisfies $\gamma \in [0, 1)$—the Markov Decision Process admits strong structural simplifications. In particular, although the class Π of admissible policies includes highly general history-dependent randomized strategies, it suffices to restrict attention to significantly smaller subclasses without loss of optimality.

Recall that the performance of a policy $\pi \in \Pi$ is given by

$$J(\pi) := \mathbb{E}_\rho^\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right] \tag{62}$$

which is well-defined under the boundedness of rewards. The first fundamental structural result asserts that dependence on the full history is unnecessary.

Theorem 2.8 (Sufficiency of Markov Policies)**.** *For any history-dependent policy $\pi \in \Pi$, there exists a Markov policy $\tilde{\pi} \in \Pi_M$ such that*

$$J(\tilde{\pi}) = J(\pi). \tag{63}$$

The key idea underlying this result is that the current state s_t is a sufficient statistic for optimal decision-making. More precisely, one constructs $\tilde{\pi}$ by taking conditional expectations of the original policy with respect to the σ-algebra generated by s_t. Formally, for each $t \geq 0$ and $B \in \mathcal{B}(\mathcal{A})$, define

$$\tilde{\pi}_t(B \mid s) := \mathbb{E}_\rho^\pi\big[\mathbf{1}_{\{a_t \in B\}} \mid s_t = s\big]. \tag{64}$$

By standard properties of conditional expectation on standard Borel spaces, this defines a stochastic kernel on $\mathcal{A}$ given $\mathcal{S}$. The tower property of conditional expectation then ensures that the induced trajectory measure under $\tilde{\pi}$ yields the same joint distribution of states and rewards as under π, and hence

$$\mathbb{E}_\rho^\pi\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] = \mathbb{E}_\rho^{\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right]. \tag{65}$$

The second structural result concerns the existence of optimal policies within the class of stationary Markov policies.

Theorem 2.9 (Existence of Optimal Stationary Policies)**.** *There exists a stationary Markov policy $\pi^* \in \Pi_S$ such that*

$$J(\pi^*) = \sup_{\pi \in \Pi} J(\pi). \tag{66}$$

The proof of this result relies on the contraction properties of the Bellman optimality operator. Define the operator T on bounded measurable functions $V : \mathcal{S} \to \mathbb{R}$ by

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s, a)\right]. \tag{67}$$

Under the boundedness of R and $\gamma < 1$, the operator T is a contraction on $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$, satisfying

$$\|TV - TW\|_\infty \leq \gamma \|V - W\|_\infty, \tag{68}$$

for all bounded measurable functions V, W. By the Banach fixed-point theorem, there exists a unique fixed point V^* such that

$$V^* = TV^*. \tag{69}$$

The optimal value function V^* satisfies the Bellman optimality equation

$$V^*(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s, a)\right]. \tag{70}$$

Under standard measurability conditions, measurable selection theorems (e.g., Kuratowski–Ryll-Nardzewski) guarantee the existence of a measurable function $\mu^* : \mathcal{S} \to \mathcal{A}$ such that

$$\mu^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s, a)\right]. \tag{71}$$

Defining the stationary deterministic policy

$$\pi^*(\cdot \mid s) := \delta_{\mu^*(s)}(\cdot), \tag{72}$$

one verifies that $V^{\pi^*} = V^*$ and hence

$$J(\pi^*) = \sup_{\pi \in \Pi} J(\pi). \tag{73}$$

Combining the two theorems, we obtain the equivalence

$$\sup_{\pi \in \Pi} J(\pi) = \sup_{\pi \in \Pi_M} J(\pi) = \sup_{\pi \in \Pi_S} J(\pi), \tag{74}$$

and, under additional regularity conditions, the supremum is attained within the class of deterministic stationary policies.

These structural results justify restricting attention to stationary Markov policies in infinite-horizon discounted MDPs, as they achieve optimal performance while significantly simplifying both theoretical analysis and algorithmic design.

2.3.8 Remarks

The restriction to Markov or stationary policies is not merely a modeling convenience but is a consequence of intrinsic structural properties of the controlled stochastic system. In particular, the state variable s_t serves as a sufficient statistic for the history $h_t = (s_0, a_0, \ldots, s_t)$ in the sense that the conditional distribution of future trajectories depends on the past only through the present state. Formally, for any admissible policy $\pi \in \Pi$, one has

$$\mathbb{P}_\rho^\pi\big(s_{t+1} \in B \mid \mathcal{F}_t\big) = \mathbb{P}_\rho^\pi\big(s_{t+1} \in B \mid s_t\big), \tag{75}$$

for all $B \in \mathcal{B}(\mathcal{S})$, provided the policy is Markov. The sufficiency of Markov policies follows from the fact that, for any history-dependent policy π, one can construct a Markov policy $\tilde{\pi}$ via conditional expectation,

$$\tilde{\pi}_t(B \mid s) := \mathbb{E}_\rho^\pi\big[\mathbf{1}_{\{a_t \in B\}} \mid s_t = s\big], \tag{76}$$

which preserves the joint distribution of (s_t, a_t) and hence the value functional $J(\pi)$. This reduction relies critically on the tower property of conditional expectation,

$$\mathbb{E}\big[\mathbb{E}[X \mid s_t]\big] = \mathbb{E}[X], \tag{77}$$

and on the Markov property of the transition kernel P. Consequently, the optimal control problem admits an equivalent formulation over the reduced class Π_M, and further over Π_S due to time-homogeneity in infinite-horizon discounted settings.

From a dynamic programming perspective, the optimal value function V^* is characterized as the unique fixed point of the Bellman operator

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s, a) \right], \tag{78}$$

and optimal policies arise from measurable maximizers of the pointwise optimization problem

$$\mu^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s, a) \right]. \tag{79}$$

This characterization is inherently Markovian and stationary, reflecting the fact that optimal decisions depend only on the current state and not on time or past history.

In practical reinforcement learning algorithms, policies are typically represented through parametric families. That is, one considers a collection $\{\pi_\theta\}_{\theta \in \Theta}$ of stochastic kernels indexed by parameters θ belonging to a measurable space $(\Theta, \mathcal{B}(\Theta))$. For each $\theta \in \Theta$, the mapping

$$(s, B) \mapsto \pi_\theta(B \mid s) \tag{80}$$

defines a stochastic kernel on $\mathcal{A}$ given $\mathcal{S}$. In many modern applications, the parameterization takes the form

$$\pi_\theta(da \mid s) = p_\theta(a \mid s)\, da, \tag{81}$$

where $p_\theta(\cdot \mid s)$ is a density function with respect to a reference measure on $\mathcal{A}$, often implemented via a neural network. The measurability requirement is then expressed as

$$(s, \theta) \mapsto p_\theta(a \mid s) \quad \text{is measurable for each fixed } a \in \mathcal{A}, \tag{82}$$

ensuring that $(s, \theta) \mapsto \pi_\theta(B \mid s)$ is measurable for all $B \in \mathcal{B}(\mathcal{A})$.

Despite the use of finite-dimensional parameterizations, the underlying mathematical object remains a stochastic kernel, and all probabilistic constructions—such as the induced trajectory measure $\mathbb{P}_\rho^{\pi_\theta}$ and the value functional

$$J(\theta) := \mathbb{E}_\rho^{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \tag{83}$$

—are defined in terms of this kernel. Optimization over policies is thus recast as optimization over parameters:

$$\sup_{\pi\in\Pi} J(\pi) \;\approx\; \sup_{\theta\in\Theta} J(\theta), \tag{84}$$

which introduces approximation error due to the restriction to the parametric class.

In summary, the prominence of Markov and stationary policies is rooted in deep probabilistic and functional-analytic properties of the MDP framework, while practical implementations approximate the infinite-dimensional space of stochastic kernels using parameterized families without altering the underlying mathematical structure.

2.4 Induced Stochastic Process

Fix an initial distribution ρ on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and an admissible policy $\pi = \{\pi_t\}_{t\geq 0}$. The Markov Decision Process then induces a stochastic process on an appropriate probability space, which we now construct rigorously.

The probabilistic foundations underlying induced stochastic processes, the Markov property, and reward-based sequential decision systems are deeply rooted in the works of Norris (1998) [10], Revuz (2008) [11], and Dynkin and Yushkevich (1979) [12]. Norris (1998) [10] developed a rigorous introduction to the theory of Markov chains, emphasizing transition probabilities, Chapman–Kolmogorov equations, recurrence and transience, invariant distributions, convergence behavior, and the fundamental probabilistic structure that characterizes memoryless stochastic evolution. Revuz (2008) [11] provided a mathematically sophisticated treatment of discrete-time and continuous-time Markov chains, focusing on stopping times, martingales, hitting probabilities, potential theory, and long-term asymptotic behavior, thereby establishing many of the analytical tools used in stochastic process theory and reinforcement learning. Dynkin and Yushkevich (1979) [12]'s pioneering work on controlled Markov processes extended classical Markov chain theory to decision-dependent stochastic systems, introducing controlled transition mechanisms, reward criteria, admissible policies, and dynamic optimization principles that later became foundational to Markov decision processes and stochastic control theory. Together, these works established the rigorous probabilistic and analytical framework for understanding how policies induce stochastic trajectories satisfying the Markov property, how rewards accumulate along such trajectories, and how controlled stochastic evolution can be optimized through sequential decision-making principles that now form the basis of reinforcement learning and dynamic programming.

Let the trajectory space be

$$\Omega := (\mathcal{S}\times\mathcal{A})^{\mathbb{N}}, \tag{85}$$

equipped with the product σ-algebra

$$\mathcal{F} := \bigotimes_{t=0}^{\infty} \left(\mathcal{B}(\mathcal{S})\otimes\mathcal{B}(\mathcal{A})\right). \tag{86}$$

Define the canonical coordinate mappings

$$s_t(\omega) := \omega_{2t}, \quad a_t(\omega) := \omega_{2t+1}, \tag{87}$$

and the associated filtration

$$\mathcal{F}_t := \sigma(s_0, a_0, \ldots, s_t), \quad t \geq 0. \tag{88}$$

The stochastic evolution of the system is specified by the initial distribution ρ, the policy π, and the transition kernel P. These objects determine a sequence of conditional distributions given recursively by

$$s_0 \sim \rho, \tag{89}$$

$$a_t \mid \mathcal{F}_t \sim \pi_t(\cdot \mid h_t), \tag{90}$$

$$s_{t+1} \mid (\mathcal{F}_t, a_t) \sim P(\cdot \mid s_t, a_t), \tag{91}$$

where $h_t = (s_0, a_0, \ldots, s_t)$ denotes the history up to time t. In the case of a Markov policy, the second relation simplifies to

$$a_t \mid s_t \sim \pi(\cdot \mid s_t). \tag{92}$$

By the Ionescu–Tulcea extension theorem, the sequence of stochastic kernels

$$\rho, \quad \pi_0, \quad P, \quad \pi_1, \quad P, \quad \ldots$$

defines a unique probability measure $\mathbb{P}_\rho^\pi$ on $(\Omega, \mathcal{F})$. Under this measure, the coordinate process

$$\{(s_t, a_t)\}_{t \geq 0} \tag{93}$$

is well-defined and adapted to the filtration $\{\mathcal{F}_t\}$. The process satisfies the following conditional distribution properties almost surely: for all $B \in \mathcal{B}(\mathcal{A})$ and $C \in \mathcal{B}(\mathcal{S})$,

$$\mathbb{P}_\rho^\pi(a_t \in B \mid \mathcal{F}_t) = \pi_t(B \mid h_t), \tag{94}$$

$$\mathbb{P}_\rho^\pi(s_{t+1} \in C \mid \mathcal{F}_t, a_t) = P(C \mid s_t, a_t). \tag{95}$$

These relations characterize the process as a controlled stochastic system.

Furthermore, the joint process $\{(s_t, a_t)\}$ is generally not Markov in its entirety unless the policy is Markov. However, under a Markov policy π, the state process $\{s_t\}$ becomes a Markov process with transition kernel

$$P_t^\pi(C \mid s) := \int_{\mathcal{A}} P(C \mid s, a)\, \pi_t(da \mid s), \tag{96}$$

and in the stationary case,

$$P^\pi(C \mid s) := \int_{\mathcal{A}} P(C \mid s, a)\, \pi(da \mid s). \tag{97}$$

The induced trajectory

$$(s_0, a_0, s_1, a_1, s_2, \ldots)$$

is thus a realization of a controlled stochastic process governed by the interaction between the policy and the environment dynamics. This process forms the probabilistic foundation for defining expectations of path-dependent functionals, such as the discounted return

$$G_0 = \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \tag{98}$$

which is measurable with respect to $\mathcal{F}$ and integrable under $\mathbb{P}_\rho^\pi$ under standard boundedness assumptions.

In summary, the combination of the initial distribution, the policy, and the transition kernel uniquely determines a probability measure on the infinite trajectory space, and hence a well-defined stochastic process that captures the evolution of states and actions over time in the Markov Decision Process.

2.5 Markov Property

The defining structural feature of a Markov Decision Process is that the evolution of the state depends on the past only through the current state-action pair. This property is encoded in the transition kernel P and manifests as a conditional independence relation under the induced trajectory measure.

Definition 2.10 (Markov Property)**.** The controlled process $\{(s_t, a_t)\}_{t \geq 0}$ satisfies the Markov property if, for all $t \geq 0$ and all measurable sets $B \in \mathcal{B}(\mathcal{S})$,

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid s_0, a_0, \ldots, s_t, a_t) = \mathbb{P}_\rho^\pi(s_{t+1} \in B \mid s_t, a_t) = P(B \mid s_t, a_t), \tag{99}$$

almost surely.

Equivalently, letting $\mathcal{F}_t := \sigma(s_0, a_0, \ldots, s_t)$ denote the natural filtration, the Markov property can be expressed as

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid \mathcal{F}_t, a_t) = P(B \mid s_t, a_t), \quad \forall B \in \mathcal{B}(\mathcal{S}), \tag{100}$$

which asserts that s_{t+1} is conditionally independent of the past $\mathcal{F}_t$ given the present pair (s_t, a_t). In terms of conditional expectations, for any bounded measurable function $f : \mathcal{S} \to \mathbb{R}$,

$$\mathbb{E}_\rho^\pi[f(s_{t+1}) \mid \mathcal{F}_t, a_t] = \int_{\mathcal{S}} f(s')\, P(ds' \mid s_t, a_t) \quad \text{a.s.} \tag{101}$$

This property follows directly from the construction of the induced probability measure $\mathbb{P}_\rho^\pi$ via the sequence of stochastic kernels (ρ, π, P) and the Ionescu–Tulcea theorem. In particular, by definition of the kernel P, the conditional law of s_{t+1} given (s_t, a_t) is precisely $P(\cdot \mid s_t, a_t)$, and admissibility of the policy ensures that this conditional distribution is well-defined with respect to $\mathcal{F}_t$.

An equivalent and often useful formulation is the following factorization of finite-dimensional distributions: for measurable sets $B_0, \ldots, B_{t+1} \in \mathcal{B}(\mathcal{S})$ and $D_0, \ldots, D_t \in \mathcal{B}(\mathcal{A})$,

$$\begin{aligned}
&\mathbb{P}_\rho^\pi(s_0 \in B_0, a_0 \in D_0, \ldots, s_{t+1} \in B_{t+1}) \\
&\quad = \int_{B_0} \rho(ds_0) \int_{D_0} \pi_0(da_0 \mid s_0) \int_{B_1} P(ds_1 \mid s_0, a_0) \cdots \int_{D_t} \pi_t(da_t \mid h_t) \int_{B_{t+1}} P(ds_{t+1} \mid s_t, a_t),
\end{aligned}$$

which shows that the dependence on the past enters only through the current state-action pair in the final transition.

While the joint process $\{(s_t, a_t)\}$ is not, in general, Markov (since the distribution of a_{t+1} may depend on the entire history through the policy), the state process $\{s_t\}$ becomes Markov under a Markov policy. Indeed, if π is Markov, then for all $B \in \mathcal{B}(\mathcal{S})$,

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid \mathcal{F}_t) = \int_{\mathcal{A}} P(B \mid s_t, a)\, \pi_t(da \mid s_t) =: P_t^\pi(B \mid s_t), \tag{102}$$

and in the stationary case,

$$\mathbb{P}_\rho^\pi(s_{t+1} \in B \mid s_t = s) = \int_{\mathcal{A}} P(B \mid s, a)\, \pi(da \mid s) =: P^\pi(B \mid s), \tag{103}$$

so that $\{s_t\}$ is a time-inhomogeneous (respectively, time-homogeneous) Markov chain.

In summary, the Markov property asserts that the transition dynamics of the environment are memoryless given the present state-action pair. This conditional independence structure is the cornerstone of dynamic programming, as it allows the decomposition of multi-stage decision problems into recursive one-step optimization problems.

2.6 Reward Structure

In a general measure-theoretic formulation of Markov Decision Processes, the reward need not be deterministic but may instead be modeled as a random variable whose distribution depends on the current state-action pair. This leads naturally to the notion of a reward kernel, which generalizes the deterministic reward function.

Let $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$ denote the measurable space of real-valued rewards. A reward kernel is a stochastic kernel

$$\mathcal{R}(dr \mid s, a)$$

on $\mathbb{R}$ given $(\mathcal{S} \times \mathcal{A}, \mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A}))$, satisfying:

- For each $(s, a) \in \mathcal{S} \times \mathcal{A}$, the mapping $B \mapsto \mathcal{R}(B \mid s, a)$ is a probability measure on $\mathcal{B}(\mathbb{R})$,
- For each $B \in \mathcal{B}(\mathbb{R})$, the mapping $(s, a) \mapsto \mathcal{R}(B \mid s, a)$ is measurable.

Under this formulation, the reward at time t is a random variable r_{t+1} satisfying the conditional distribution

$$r_{t+1} \mid (s_t, a_t) \sim \mathcal{R}(\cdot \mid s_t, a_t). \tag{104}$$

Thus, for any measurable set $B \subseteq \mathbb{R}$,

$$\mathbb{P}_\rho^\pi(r_{t+1} \in B \mid s_t, a_t) = \mathcal{R}(B \mid s_t, a_t), \tag{105}$$

almost surely. The deterministic reward function arises as the expectation of the reward kernel. Specifically, if the first moment exists, one defines

$$R(s, a) := \int_{\mathbb{R}} r \, \mathcal{R}(dr \mid s, a), \tag{106}$$

which yields a measurable function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. In this case, the reward random variable satisfies

$$\mathbb{E}_\rho^\pi[r_{t+1} \mid s_t, a_t] = R(s_t, a_t). \tag{107}$$

More generally, for any bounded measurable function $g : \mathbb{R} \to \mathbb{R}$, one has the integral representation

$$\mathbb{E}_\rho^\pi[g(r_{t+1}) \mid s_t, a_t] = \int_{\mathbb{R}} g(r) \, \mathcal{R}(dr \mid s_t, a_t), \tag{108}$$

which characterizes the conditional distribution of rewards in full generality.

To ensure well-posedness of the infinite-horizon return

$$G_0 = \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \tag{109}$$

it is standard to impose integrability or boundedness conditions on the reward kernel. A common assumption is the existence of a constant $M < \infty$ such that

$$\int_{\mathbb{R}} |r| \, \mathcal{R}(dr \mid s, a) \leq M \quad \text{for all } (s, a) \in \mathcal{S} \times \mathcal{A}, \tag{110}$$

which implies

$$\mathbb{E}_\rho^\pi[|r_{t+1}|] \leq M \quad \text{for all } t \geq 0. \tag{111}$$

Under this condition and $\gamma \in [0, 1)$, the discounted return G_0 is absolutely convergent almost surely and satisfies

$$\mathbb{E}_\rho^\pi[|G_0|] \leq \frac{M}{1 - \gamma}. \tag{112}$$

An alternative but equivalent formulation incorporates the reward into the transition dynamics by defining an augmented kernel on $\mathcal{S} \times \mathbb{R}$:

$$\tilde{P}(ds', dr \mid s, a) := P(ds' \mid s, a) \, \mathcal{R}(dr \mid s, a), \tag{113}$$

which describes the joint evolution of the next state and reward. This perspective is often useful when analyzing the joint process $\{(s_t, r_{t+1})\}$.

In the special case where the reward is deterministic, the kernel reduces to a Dirac measure:

$$\mathcal{R}(dr \mid s, a) = \delta_{R(s,a)}(dr), \tag{114}$$

so that

$$r_{t+1} = R(s_t, a_t) \quad \text{almost surely.} \tag{115}$$

In summary, the reward kernel formulation provides a flexible and mathematically complete description of reward generation in MDPs, encompassing both deterministic and stochastic rewards, and ensuring compatibility with the general theory of stochastic kernels and controlled processes.

2.7 Finite MDPs as a Special Case

A particularly important and widely studied subclass of Markov Decision Processes arises when both the state space $\mathcal{S}$ and the action space $\mathcal{A}$ are finite sets. In this setting, the abstract measure-theoretic framework simplifies considerably, and all objects admit explicit finite-dimensional representations.

Let $\mathcal{S} = \{s_1, \dots, s_n\}$ and $\mathcal{A} = \{a_1, \dots, a_m\}$. Then the σ-algebras $\mathcal{B}(\mathcal{S})$ and $\mathcal{B}(\mathcal{A})$ coincide with the power sets $2^{\mathcal{S}}$ and $2^{\mathcal{A}}$, respectively, and every function defined on these spaces is automatically measurable. Consequently, all measurability conditions imposed in the general theory become trivial.

The transition kernel P reduces to a collection of transition probabilities

$$P(s' \mid s, a), \quad s, s' \in \mathcal{S},\ a \in \mathcal{A},$$

satisfying

$$P(s' \mid s, a) \geq 0, \quad \sum_{s' \in \mathcal{S}} P(s' \mid s, a) = 1. \tag{116}$$

For each fixed action $a \in \mathcal{A}$, one can represent $P(\cdot \mid \cdot, a)$ as an $n \times n$ stochastic matrix P^a with entries

$$(P^a)_{ij} := P(s_j \mid s_i, a). \tag{117}$$

Similarly, a stochastic policy π is represented by a collection of probability vectors

$$\pi(\cdot \mid s) = \big(\pi(a_1 \mid s), \dots, \pi(a_m \mid s)\big), \quad s \in \mathcal{S}, \tag{118}$$

satisfying

$$\pi(a \mid s) \geq 0, \quad \sum_{a \in \mathcal{A}} \pi(a \mid s) = 1. \tag{119}$$

A deterministic policy corresponds to a function $\mu : \mathcal{S} \to \mathcal{A}$, which can equivalently be viewed as selecting a standard basis vector in $\mathbb{R}^m$ for each state.

The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is represented as a finite table of values $R(s, a)$, or equivalently as a collection of vectors $\{R^a\}_{a \in \mathcal{A}}$ with

$$R^a_i := R(s_i, a). \tag{120}$$

Under a stationary policy π, the induced transition matrix on $\mathcal{S}$ is given by

$$P^\pi(s' \mid s) := \sum_{a \in \mathcal{A}} P(s' \mid s, a)\, \pi(a \mid s), \tag{121}$$

which defines an $n \times n$ stochastic matrix P^π. The expected immediate reward under π is

$$R^\pi(s) := \sum_{a \in \mathcal{A}} R(s, a)\, \pi(a \mid s). \tag{122}$$

The value function $V^\pi : \mathcal{S} \to \mathbb{R}$ can be identified with a vector $V^\pi \in \mathbb{R}^n$ satisfying the Bellman equation

$$V^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in \mathcal{S}} P^\pi(s' \mid s)\, V^\pi(s'), \tag{123}$$

or, in vector form,

$$V^\pi = R^\pi + \gamma P^\pi V^\pi. \tag{124}$$

Rearranging, one obtains

$$(I - \gamma P^\pi) V^\pi = R^\pi, \tag{125}$$

and since $\gamma \in [0, 1)$ and P^π is stochastic, the matrix $(I - \gamma P^\pi)$ is invertible, yielding the explicit solution

$$V^\pi = (I - \gamma P^\pi)^{-1} R^\pi. \tag{126}$$

The Bellman optimality equation similarly reduces to

$$V^*(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s' \mid s,a)\, V^*(s') \right], \tag{127}$$

which is a system of n nonlinear equations. In vector notation, defining the Bellman optimality operator T by

$$(TV)(s) := \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s' \mid s,a)\, V(s') \right], \tag{128}$$

one has

$$V^* = TV^*. \tag{129}$$

In this finite setting, all expectations reduce to finite sums, all integrals to summations, and all stochastic kernels to matrices or vectors. As a result, the theory becomes computationally tractable and forms the foundation for classical dynamic programming algorithms such as value iteration and policy iteration.

In summary, finite MDPs provide a concrete and fully explicit realization of the general framework, where all abstract measure-theoretic constructs reduce to linear algebraic objects, while preserving the essential structure of the decision problem.

2.8 Remarks

The above formulation is sufficiently general to encompass both classical discrete-time stochastic control problems and modern reinforcement learning models. In particular, the evolution of the system is described by a sequence of stochastic kernels $(\rho, \pi, P, \mathcal{R})$, which define a probability measure on the trajectory space via the Ionescu–Tulcea theorem. The performance of a policy π is evaluated through the discounted return

$$G_0 := \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \tag{130}$$

and the corresponding value functional

$$J(\pi) := \mathbb{E}_\rho^\pi[G_0]. \tag{131}$$

The restriction $\gamma \in [0,1)$ plays a crucial role in ensuring that this infinite series is well-defined. Indeed, if the reward function is uniformly bounded, i.e., there exists $M < \infty$ such that $|r_{t+1}| \le M$ almost surely, then

$$|G_0| \le \sum_{t=0}^{\infty} \gamma^t M = \frac{M}{1-\gamma}, \tag{132}$$

which implies that G_0 converges absolutely almost surely and belongs to $L^1(\mathbb{P}_\rho^\pi)$. Consequently, the value functional $J(\pi)$ is finite for all admissible policies. Moreover, the discount factor induces a contraction structure in the associated dynamic programming operators. For instance, the Bellman operator T defined by

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s,a) \right] \tag{133}$$

satisfies

$$\|TV - TW\|_\infty \le \gamma \|V - W\|_\infty, \tag{134}$$

for all bounded measurable functions V, W. This contraction property guarantees the existence and uniqueness of a fixed point V^* and underpins the convergence of value iteration algorithms.

The assumption that $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$ are standard Borel (in particular, Polish) spaces further ensures that the framework is well-posed from a measure-theoretic standpoint. In this setting, all relevant stochastic kernels (transition kernels, policy kernels, and reward kernels) are assumed to be Borel measurable, which guarantees that compositions of kernels remain measurable and that conditional expectations are well-defined.

These topological and measurability assumptions enable the application of powerful analytical tools. For example, measurable selection theorems ensure the existence of measurable functions $\mu^* : \mathcal{S} \to \mathcal{A}$ satisfying

$$\mu^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s,a) \right], \tag{135}$$

thereby yielding deterministic stationary optimal policies. Similarly, the contraction mapping principle applies in the Banach space $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$, ensuring that the Bellman operator admits a unique fixed point.

Furthermore, the standard Borel structure guarantees the existence of regular conditional probabilities and disintegrations, which are essential for defining policies via conditional distributions and for constructing Markov reductions of history-dependent policies. It also ensures that probability measures on trajectory spaces are tight and that limits of sequences of measures behave well under weak convergence.

In summary, the combination of the discount condition $\gamma < 1$, boundedness of rewards, and the assumption of Polish (standard Borel) spaces provides a mathematically robust framework in which all components of the MDP are well-defined, and powerful tools from functional analysis, probability theory, and measurable selection theory can be rigorously applied.

3 Policies and Return

In this section, we provide a rigorous formulation of policies and return in Markov Decision Processes, emphasizing their probabilistic structure, measurability, and analytical properties.

The concepts of policies and return form the core of reinforcement learning and stochastic decision theory, and their mathematical and algorithmic foundations have been extensively developed by Sutton and Barto (1998) [1], Bertsekas (2012) [9], and Szepesvári (2022) [13]. Sutton and Barto (1998) [1] formalized the interaction between an agent and its environment through policies that map states to actions and returns that quantify cumulative future rewards, introducing foundational ideas such as discounted return, episodic and continuing tasks, value functions, and policy optimization within the reinforcement learning paradigm. Bertsekas (2012) [9] approached these concepts from the perspective of dynamic programming and optimal control, rigorously analyzing the role of policies in sequential optimization problems and establishing the mathematical structure of return functions through Bellman equations, cost-to-go functions, and recursive optimality principles for deterministic and stochastic systems. Szepesvári (2022) [13] further unified the theoretical and algorithmic aspects of reinforcement learning by systematically studying policy evaluation, exploration–exploitation trade-offs, temporal-difference learning, Monte Carlo methods, and convergence properties of reinforcement learning algorithms under finite and infinite-horizon formulations. Together, these works established the modern understanding of policies as decision-making mechanisms governing agent behavior and returns as quantitative measures of long-term performance, thereby forming the theoretical foundation for reinforcement learning, stochastic control, and sequential optimization.

3.1 Policies

A policy encodes the decision-making mechanism of the agent and determines how actions are selected in response to states (or, in greater generality, histories). In the most fundamental formulation, a policy is modeled as a stochastic kernel, thereby allowing randomized decision rules and ensuring compatibility with the measure-theoretic structure of the Markov Decision Process.

Definition 3.1 (Policy)**.** A (stochastic) policy is a stochastic kernel π on $\mathcal{A}$ given $\mathcal{S}$, i.e.,

$$\pi : \mathcal{B}(\mathcal{A}) \times \mathcal{S} \to [0,1], \tag{136}$$

such that:

- For each $s \in \mathcal{S}$, the mapping $B \mapsto \pi(B|s)$ is a probability measure on $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$,
- For each $B \in \mathcal{B}(\mathcal{A})$, the mapping $s \mapsto \pi(B|s)$ is $\mathcal{B}(\mathcal{S})$-measurable.

The first condition ensures that, for each fixed state s, the policy defines a valid probability distribution over actions, while the second condition guarantees that the policy is measurable as a function of the state, which is essential for the well-definedness of conditional expectations and for the construction of the induced probability measure on trajectories.

Equivalently, the policy can be viewed as an operator acting on measurable functions $f : \mathcal{A} \to \mathbb{R}$. For each $s \in \mathcal{S}$, define

$$(\pi f)(s) := \int_{\mathcal{A}} f(a)\, \pi(da|s), \tag{137}$$

whenever the integral is well-defined. This representation emphasizes that π maps functions on $\mathcal{A}$ to functions on $\mathcal{S}$, and in particular,

$$(\pi f)(s) = \mathbb{E}[f(a) \mid s], \tag{138}$$

where the expectation is taken with respect to the distribution $\pi(\cdot|s)$. Thus, a policy induces a conditional expectation operator, which plays a central role in the analysis of value functions and Bellman equations.

From a probabilistic perspective, if $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ is the probability space induced by the policy π, then for each $t \geq 0$, the action a_t satisfies

$$\mathbb{P}_\rho^\pi(a_t \in B \mid s_t) = \pi(B \mid s_t), \quad \forall B \in \mathcal{B}(\mathcal{A}), \tag{139}$$

almost surely. This relation characterizes π as the conditional distribution of the action given the current state.

A deterministic policy arises as a special case of a stochastic policy in which all randomness is eliminated. Specifically, a policy is deterministic if there exists a measurable function $\mu : \mathcal{S} \to \mathcal{A}$ such that

$$\pi(da|s) = \delta_{\mu(s)}(da), \tag{140}$$

where $\delta_{\mu(s)}$ denotes the Dirac measure concentrated at $\mu(s)$. In this case, for any measurable function $f : \mathcal{A} \to \mathbb{R}$, one has

$$\int_{\mathcal{A}} f(a)\, \pi(da|s) = f(\mu(s)), \tag{141}$$

so that the conditional expectation reduces to evaluation at the selected action.

More generally, one may consider policies that depend on the entire history $h_t = (s_0, a_0, \ldots, s_t)$, in which case π_t is a stochastic kernel on $\mathcal{A}$ given the history space. However, under standard assumptions, such generality is not required for optimality, as one can restrict attention to Markov or even stationary policies without loss of performance.

In summary, a policy is a measurable mapping from states to probability measures over actions, and it serves as the fundamental object governing the stochastic evolution of the controlled process. Its formulation as a stochastic kernel ensures both mathematical rigor and compatibility with the probabilistic structure of Markov Decision Processes.

3.2 Controlled Process Under a Policy

Given an initial distribution ρ on $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$, a policy $\pi = \{\pi_t\}_{t \geq 0}$, and a transition kernel P on $\mathcal{S}$ given $\mathcal{S} \times \mathcal{A}$, one constructs a controlled stochastic process on a canonical path space. Let

$$\Omega := (\mathcal{S} \times \mathcal{A})^{\mathbb{N}}, \qquad \mathcal{F} := \bigotimes_{t=0}^{\infty} \left(\mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A})\right), \tag{142}$$

and define the coordinate projections

$$s_t(\omega) := \omega_{2t}, \qquad a_t(\omega) := \omega_{2t+1}. \tag{143}$$

The natural filtration is

$$\mathcal{F}_t := \sigma(s_0, a_0, \dots, s_t), \quad t \geq 0. \tag{144}$$

The dynamics are specified recursively by the sequence of stochastic kernels

$$\rho, \quad \pi_0, \quad P, \quad \pi_1, \quad P, \quad \dots,$$

which determine the conditional distributions

$$s_0 \sim \rho, \tag{145}$$

$$a_t \mid \mathcal{F}_t \sim \pi_t(\cdot \mid h_t), \tag{146}$$

$$s_{t+1} \mid (\mathcal{F}_t, a_t) \sim P(\cdot \mid s_t, a_t), \tag{147}$$

where $h_t = (s_0, a_0, \dots, s_t)$ denotes the history. In the case of a Markov policy, the second relation simplifies to

$$a_t \mid s_t \sim \pi(\cdot \mid s_t). \tag{148}$$

Under standard measurability assumptions, the Ionescu–Tulcea theorem guarantees the existence of a unique probability measure $\mathbb{P}_\rho^\pi$ on $(\Omega, \mathcal{F})$ such that, for any measurable sets $B_0, \dots, B_t \in \mathcal{B}(\mathcal{S})$ and $D_0, \dots, D_{t-1} \in \mathcal{B}(\mathcal{A})$,

$$\mathbb{P}_\rho^\pi(s_0 \in B_0, a_0 \in D_0, \dots, s_t \in B_t) \tag{149}$$

$$= \int_{B_0} \rho(ds_0) \int_{D_0} \pi_0(da_0 \mid s_0) \int_{B_1} P(ds_1 \mid s_0, a_0) \cdots \int_{B_t} P(ds_t \mid s_{t-1}, a_{t-1}). \tag{150}$$

This specifies all finite-dimensional distributions and hence the law of the entire process. The resulting trajectory

$$(s_0, a_0, s_1, a_1, \dots)$$

is a realization of the controlled process. Its distribution is completely determined by the triple (ρ, π, P), and no additional randomness is required beyond that encoded in these kernels.

Moreover, the process satisfies the following conditional independence relations almost surely:

$$\mathbb{P}_\rho^\pi(a_t \in B \mid \mathcal{F}_t) = \pi_t(B \mid h_t), \tag{151}$$

$$\mathbb{P}_\rho^\pi(s_{t+1} \in C \mid \mathcal{F}_t, a_t) = P(C \mid s_t, a_t), \tag{152}$$

for all measurable sets $B \subseteq \mathcal{A}$ and $C \subseteq \mathcal{S}$. These relations characterize the process as a controlled Markov system in which the policy governs the selection of actions and the kernel P governs the evolution of states.

In summary, the combination of the initial distribution, the policy, and the transition kernel induces a unique probability measure on the infinite trajectory space, thereby defining a well-posed stochastic process that captures the interaction between the agent and the environment.

3.3 Return

The return is a path-dependent functional that aggregates the sequence of rewards obtained along a trajectory generated by a policy. Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space induced by the initial distribution ρ, policy π, transition kernel P, and (possibly stochastic) reward mechanism. For $\omega \in \Omega$, define the reward sequence $\{R_t(\omega)\}_{t \geq 1}$, where R_{t+1} is the reward realized after executing action a_t in state s_t.

Definition 3.2 (Return)**.** The (discounted) return starting at time t is defined by

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \qquad \gamma \in [0, 1). \tag{153}$$

The random variable G_t is $\mathcal{F}$-measurable provided each R_t is measurable, and it depends on the entire tail of the trajectory $(s_t, a_t, s_{t+1}, a_{t+1}, \dots)$. The discount factor γ induces a geometric weighting that attenuates the contribution of distant rewards and ensures summability under mild conditions.

Assume that the rewards are uniformly bounded, i.e., there exists $M < \infty$ such that $|R_t| \leq M$ almost surely for all t. Then, for each $\omega \in \Omega$,

$$|G_t(\omega)| \leq \sum_{k=0}^{\infty} \gamma^k |R_{t+k+1}(\omega)| \leq \sum_{k=0}^{\infty} \gamma^k M = \frac{M}{1-\gamma}, \tag{154}$$

which implies that G_t is finite almost surely and $G_t \in L^\infty(\mathbb{P}_\rho^\pi) \subset L^1(\mathbb{P}_\rho^\pi)$. Consequently, expectations of G_t are well-defined. More generally, if rewards are not uniformly bounded but satisfy the integrability condition

$$\sup_{t \geq 0} \mathbb{E}_\rho^\pi[|R_{t+1}|] \leq M < \infty, \tag{155}$$

then by Tonelli's theorem,

$$\mathbb{E}_\rho^\pi[|G_t|] \leq \sum_{k=0}^{\infty} \gamma^k \mathbb{E}_\rho^\pi[|R_{t+k+1}|] \leq \sum_{k=0}^{\infty} \gamma^k M = \frac{M}{1-\gamma}, \tag{156}$$

so that $G_t \in L^1(\mathbb{P}_\rho^\pi)$.

The return admits a useful recursive decomposition. For all $t \geq 0$,

$$G_t = R_{t+1} + \gamma G_{t+1}, \tag{157}$$

almost surely. This identity follows by separating the first term of the series:

$$G_t = R_{t+1} + \sum_{k=1}^{\infty} \gamma^k R_{t+k+1} = R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+1+k+1} = R_{t+1} + \gamma G_{t+1}. \tag{158}$$

Taking conditional expectations with respect to $\mathcal{F}_t$ yields

$$\mathbb{E}_\rho^\pi[G_t \mid \mathcal{F}_t] = \mathbb{E}_\rho^\pi[R_{t+1} \mid \mathcal{F}_t] + \gamma\, \mathbb{E}_\rho^\pi[G_{t+1} \mid \mathcal{F}_t], \tag{159}$$

which forms the basis for dynamic programming and Bellman-type recursions. If the reward is generated via a reward kernel $\mathcal{R}(\cdot \mid s, a)$ with finite first moment, then

$$\mathbb{E}_\rho^\pi[R_{t+1} \mid s_t, a_t] = \int_{\mathbb{R}} r\, \mathcal{R}(dr \mid s_t, a_t) =: R(s_t, a_t), \tag{160}$$

and hence

$$\mathbb{E}_\rho^\pi[G_t \mid s_t] = \mathbb{E}_\rho^\pi\left[\sum_{k=0}^{\infty} \gamma^k R(s_{t+k}, a_{t+k}) \,\middle|\, s_t \right]. \tag{161}$$

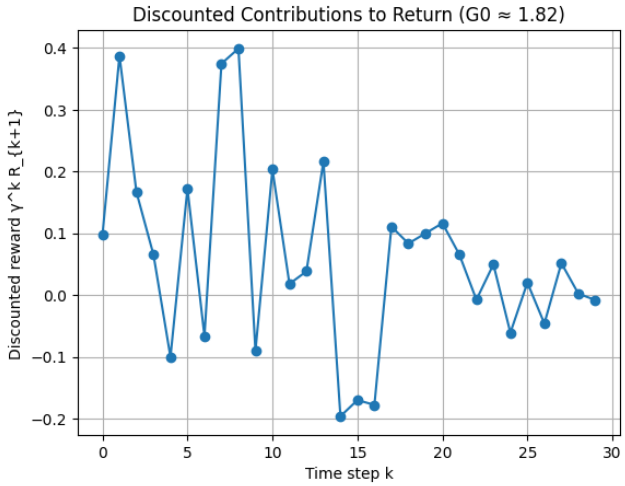
The behavior of the return G_0 can be understood by decomposing it into its individual discounted contributions. By definition,

$$G_0 = \sum_{k=0}^{\infty} \gamma^k R_{k+1}, \tag{162}$$

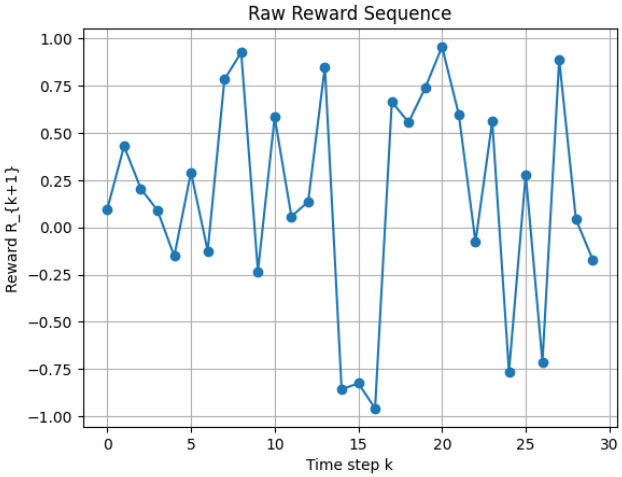
so that each term $\gamma^k R_{k+1}$ represents the contribution of the reward received at time $k+1$, scaled by the discount factor. Referring to the *first figure* (Figure 1a), the plotted sequence corresponds precisely to the terms

$$\{\gamma^k R_{k+1}\}_{k \geq 0}, \tag{163}$$

which constitute the summands of G_0. The decay in magnitude of these terms as k increases reflects the geometric weighting γ^k, which suppresses the influence of rewards farther in the future. Even when the raw rewards fluctuate in magnitude, their discounted counterparts diminish over time due to the factor γ^k.



(a) **Discounted contributions** $\gamma^k R_{k+1}$



(b) **Raw rewards** R_{k+1}

Figure 1: **Comparison between raw rewards and their discounted contributions to the return** $G_0 = \sum_{k=0}^{\infty} \gamma^k R_{k+1}$.

In contrast, the *second figure* (Figure 1b) displays the raw reward sequence

$$\{R_{k+1}\}_{k\geq 0}, \tag{164}$$

without any attenuation. This sequence does not exhibit systematic decay and may remain of comparable magnitude across time. The comparison between the two figures makes explicit the role of the discount factor: while R_{k+1} may be large for large k, its contribution to the return is effectively reduced to

$$\gamma^k R_{k+1}, \tag{165}$$

which tends to zero as $k \to \infty$ whenever $\gamma \in [0, 1)$.

Thus, the figures visually demonstrate that the return G_0 is dominated by near-term rewards, with distant rewards contributing progressively less according to the geometric decay:

$$|\gamma^k R_{k+1}| \leq \gamma^k M \to 0 \quad \text{as } k \to \infty, \tag{166}$$

under boundedness $|R_{k+1}| \leq M$. This illustrates the fundamental role of discounting in ensuring both convergence of the series and temporal prioritization of rewards.

The two figures (Figure 1a and 1b) provide a direct visualization of the structure of the return as defined in your subsection. Recall that the return is given by

$$G_0 = \sum_{k=0}^{\infty} \gamma^k R_{k+1}, \qquad \gamma \in [0, 1). \tag{167}$$

The *first plot* (Figure 1a) displays the individual terms $\gamma^k R_{k+1}$, i.e., the discounted contributions to G_0. Each point corresponds to a term in the series, making explicit how the geometric factor γ^k progressively reduces the magnitude of contributions as k increases. This visually illustrates the bound

$$\left|\gamma^k R_{k+1}\right| \leq \gamma^k M, \tag{168}$$

whenever $|R_{k+1}| \leq M$, and hence explains the summability of the series.

The *second plot* (Figure 1b) shows the raw reward sequence $R_{k+1 k\geq 0}$, without any attenuation. In contrast to the first plot, the magnitudes here do not decay, and fluctuations persist across time steps.

Comparing the two figures makes clear that the discounting mechanism transforms the raw sequence into a geometrically weighted one, effectively emphasizing near-term rewards while suppressing distant contributions.

Together, the plots illustrate that the return can be interpreted as a weighted series where the weights decay geometrically:

$$G_0 = R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots, \tag{169}$$

and hence

$$|G_0| \leq \sum_{k=0}^{\infty} \gamma^k |R_{k+1}|. \tag{170}$$

In summary, the second figure (Figure 1b) represents the underlying reward process, while the first figure (Figure 1a) shows how this process is transformed into a convergent series through discounting, thereby making explicit the analytical role of γ in ensuring finiteness and emphasizing short-term outcomes.

In summary, the return G_t is a well-defined, integrable random variable under standard boundedness or integrability assumptions, admits a recursive structure essential for analysis, and serves as the fundamental quantity whose expectation is optimized in reinforcement learning and stochastic control.

3.4 Value Functions

The performance of a policy is quantified through value functions, which are expectations of the return under the probability measure induced by the policy and the system dynamics. These functions encode the long-term consequences of decisions and form the central objects in both theoretical analysis and algorithmic design.

Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space induced by a policy π. For a fixed initial state $s \in \mathcal{S}$, define the conditional probability measure $\mathbb{P}_s^\pi := \mathbb{P}_\rho^\pi(\cdot \mid s_0 = s)$, and the corresponding expectation $\mathbb{E}_s^\pi[\cdot]$. The *state-value function* associated with a policy π is defined as

$$V^\pi(s) := \mathbb{E}_s^\pi[G_0], \tag{171}$$

where

$$G_0 = \sum_{t=0}^{\infty} \gamma^t R_{t+1}. \tag{172}$$

Under standard boundedness or integrability assumptions on the reward process, $V^\pi(s)$ is well-defined and finite for all $s \in \mathcal{S}$. Similarly, the *action-value function* is defined by conditioning additionally on the initial action:

$$Q^\pi(s, a) := \mathbb{E}^\pi \left[G_0 \mid s_0 = s, a_0 = a \right]. \tag{173}$$

This function represents the expected return obtained by first taking action a in state s, and thereafter following the policy π.

The value functions admit fundamental recursive decompositions. Using the identity

$$G_0 = R_1 + \gamma G_1, \tag{174}$$

and conditioning appropriately, one obtains

$$V^\pi(s) = \mathbb{E}_s^\pi[R_1] + \gamma \, \mathbb{E}_s^\pi[G_1]. \tag{175}$$

Applying the law of total expectation and the definition of the policy,

$$\mathbb{E}_s^\pi[R_1] = \int_{\mathcal{A}} \mathbb{E}[R_1 \mid s_0 = s, a_0 = a] \, \pi(da \mid s), \tag{176}$$

and similarly,

$$\mathbb{E}_s^\pi[G_1] = \int_{\mathcal{A}} \int_{\mathcal{S}} V^\pi(s') \, P(ds' \mid s, a) \, \pi(da \mid s). \tag{177}$$

Combining these expressions yields the Bellman expectation equation

$$V^\pi(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a) \right] \pi(da \mid s). \tag{178}$$

The action-value function satisfies a similar recursion. Conditioning on (s_0, a_0) and using the same decomposition,

$$Q^\pi(s,a) = \mathbb{E}[R_1 \mid s_0 = s, a_0 = a] + \gamma\, \mathbb{E}[G_1 \mid s_0 = s, a_0 = a]. \tag{179}$$

The first term is simply $R(s,a)$, while the second term can be expressed as

$$\mathbb{E}[G_1 \mid s_0 = s, a_0 = a] = \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a). \tag{180}$$

Thus,

$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a). \tag{181}$$

The relationship between V^π and Q^π is given by

$$V^\pi(s) = \int_{\mathcal{A}} Q^\pi(s,a)\, \pi(da \mid s), \tag{182}$$

which expresses the state-value function as the expectation of the action-value function under the policy.

These recursive equations characterize V^π and Q^π as fixed points of linear operators on appropriate function spaces. In particular, defining the Bellman operator T^π by

$$(T^\pi V)(s) := \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s,a) \right] \pi(da \mid s), \tag{183}$$

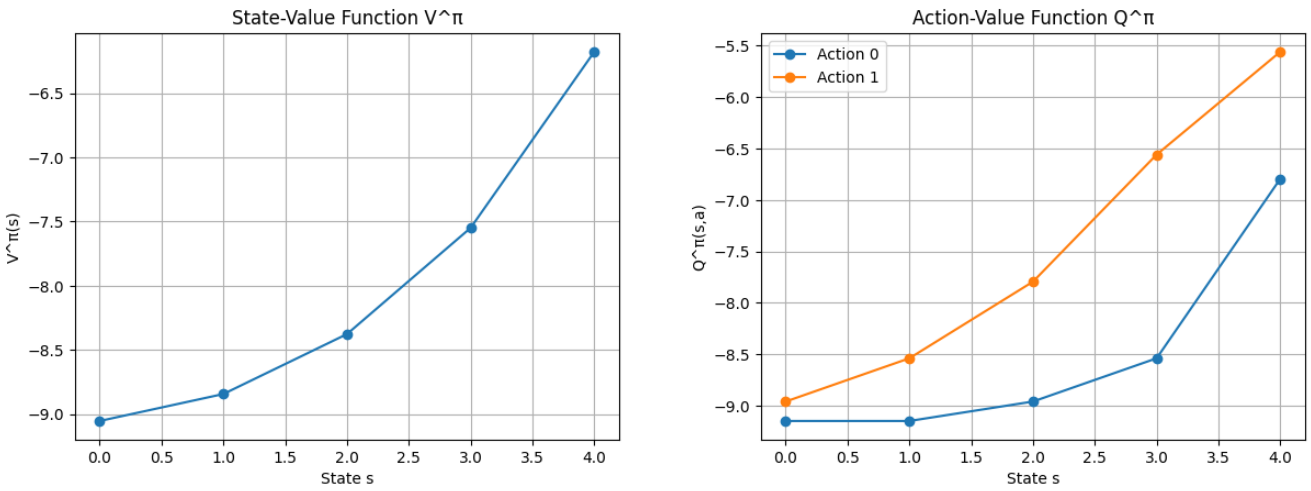
one has

$$V^\pi = T^\pi V^\pi. \tag{184}$$

Under the sup-norm $\|\cdot\|_\infty$, the operator T^π is a contraction:

$$\|T^\pi V - T^\pi W\|_\infty \le \gamma \|V - W\|_\infty, \tag{185}$$

which guarantees the existence and uniqueness of V^π as its fixed point.



(a) **State-value function** $V^\pi(s)$ (b) **Action-value function** $Q^\pi(s,a)$

Figure 2: **Visualization of value functions: the state-value function** $V^\pi(s)$ **and action-value function** $Q^\pi(s,a)$**, illustrating the relationship** $V^\pi(s) = \sum_a \pi(a|s)\, Q^\pi(s,a)$**.**

The two figures (Figure 2a and 2b) provide a direct numerical instantiation of the value-function framework for a finite MDP. The *first plot* (Figure 2a) depicts the state-value function

$$V^{\pi}(s) = \mathbb{E}_s^{\pi}! \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right], \tag{186}$$

which is computed via iterative application of the Bellman operator

$$V_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a), V_k(s') \right], \tag{187}$$

until convergence to the fixed point V^{π}. The resulting curve over states represents the expected long-term return when following the policy π, thereby encoding the global performance of the policy as a function of the starting state.

The *second plot* (Figure 2b) shows the action-value function

$$Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a), V^{\pi}(s'), \tag{188}$$

evaluated for each action a. Unlike V^{π}, this produces multiple curves (one per action), explicitly revealing how different choices at the current state influence the long-term outcome. The separation between these curves reflects the advantage or disadvantage of specific actions, thereby enabling direct comparison at the decision level.

The fundamental relationship between these two quantities,

$$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s), Q^{\pi}(s,a), \tag{189}$$

is clearly manifested in the plots: the state-value curve lies between the action-value curves, since it represents a convex combination of them under the policy π. In the case of a uniform policy, this corresponds to an equal-weight average, so $V^{\pi}(s)$ appears visually as an intermediate curve between the action-specific values. The figures (Figure 2a and 2b) illustrate that

1. V^{π} provides a state-level summary of long-term performance,
2. Q^{π} refines this by resolving action-level consequences, and
3. Their relationship through expectation under π is directly visible in the geometry of the plotted curves.

In summary, the value functions provide a complete characterization of the performance of a policy, admit recursive representations fundamental to dynamic programming, and form the analytical backbone of reinforcement learning theory.

3.5 Objective Functional

The central problem in reinforcement learning and stochastic control is to determine a policy that maximizes the expected cumulative reward over an infinite horizon. This objective is formalized through a functional defined on the space of admissible policies.

Let Π denote the class of admissible policies and let $(\Omega, \mathcal{F}, \mathbb{P}_{\rho}^{\pi})$ be the probability space induced by an initial distribution ρ and a policy $\pi \in \Pi$. The performance of π is quantified by the objective functional

$$J(\pi) := \mathbb{E}_{\rho}^{\pi}[G_0], \tag{190}$$

where

$$G_0 = \sum_{t=0}^{\infty} \gamma^t R_{t+1}. \tag{191}$$

Under standard assumptions (e.g., bounded rewards or suitable integrability conditions and $\gamma \in [0, 1)$), the random variable G_0 belongs to $L^1(\mathbb{P}_\rho^\pi)$, ensuring that $J(\pi)$ is well-defined and finite. The optimal control problem is then formulated as

$$\sup_{\pi \in \Pi} J(\pi), \tag{192}$$

which seeks a policy achieving the maximal expected return. In many cases, this supremum is attained within a restricted class of policies (e.g., stationary Markov policies), as established by structural results.

When the initial state is fixed at $s \in \mathcal{S}$, one considers the conditional expectation

$$V^\pi(s) := \mathbb{E}_s^\pi[G_0], \tag{193}$$

and the objective reduces pointwise to

$$\sup_{\pi \in \Pi} V^\pi(s). \tag{194}$$

This leads to the definition of the *optimal value function*

$$V^*(s) := \sup_{\pi \in \Pi} V^\pi(s), \quad s \in \mathcal{S}. \tag{195}$$

The function V^* can be characterized as the maximal fixed point of the Bellman optimality operator. Define the operator T on bounded measurable functions $V : \mathcal{S} \to \mathbb{R}$ by

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s, a) \right]. \tag{196}$$

Then, under standard assumptions, T is a contraction mapping on $(\mathcal{B}_b(\mathcal{S}), \| \cdot \|_\infty)$ with modulus γ, i.e.,

$$\|TV - TW\|_\infty \leq \gamma \|V - W\|_\infty. \tag{197}$$

By the Banach fixed-point theorem, there exists a unique function V^* such that

$$V^* = TV^*, \tag{198}$$

which explicitly yields the Bellman optimality equation

$$V^*(s) = \sup_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s, a) \right]. \tag{199}$$

Moreover, for any policy $\pi \in \Pi$, one has the pointwise inequality

$$V^\pi(s) \leq V^*(s), \quad \forall s \in \mathcal{S}, \tag{200}$$

which establishes V^* as the upper envelope of all achievable value functions. Under additional regularity conditions (e.g., standard Borel spaces and measurable reward and transition kernels), measurable selection theorems ensure the existence of a policy π^* such that

$$V^{\pi^*}(s) = V^*(s), \quad \forall s \in \mathcal{S}, \tag{201}$$

and such a policy can be chosen to be stationary and deterministic:

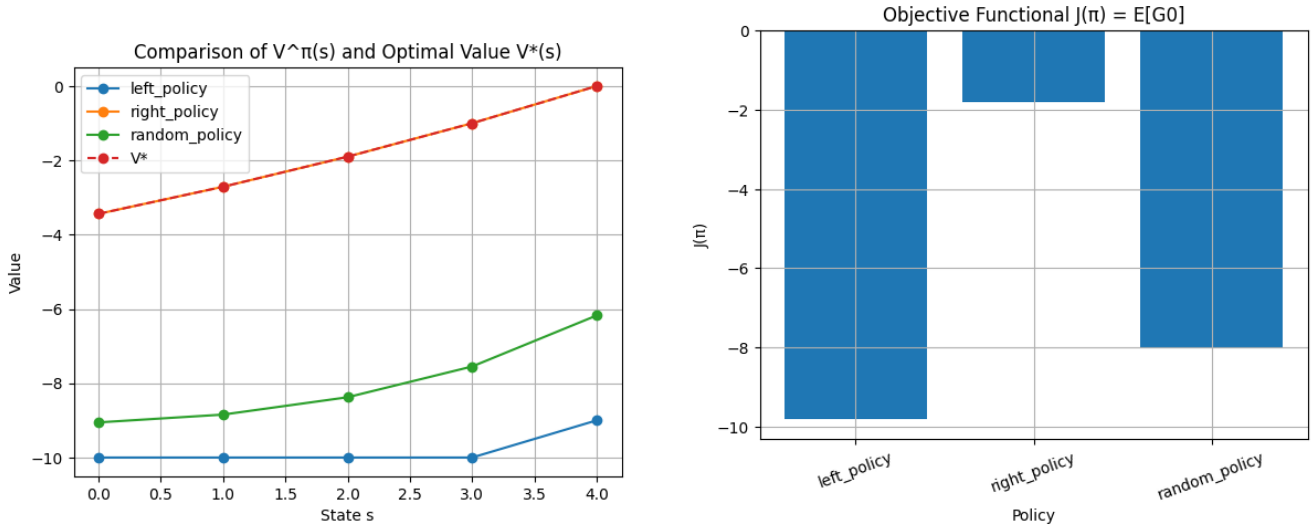
$$\pi^*(\cdot \mid s) = \delta_{\mu^*(s)}(\cdot), \tag{202}$$

where

$$\mu^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds' \mid s, a) \right]. \tag{203}$$

The two figures (Figure 3a and 3b) provide a concrete numerical realization of the objective functional and the optimal value function within a finite Markov Decision Process. For any policy π, the performance is quantified by the objective functional

$$J(\pi) = \mathbb{E}_\rho^\pi[G_0] = \sum_{s \in \mathcal{S}} \rho(s), V^\pi(s), \tag{204}$$



(a) **Comparison of $V^\pi(s)$ and $V^*(s)$**

(b) **Objective functional $J(\pi)$ across policies**

Figure 3: **Visualization of the objective functional: state-value functions under different policies and their corresponding performance $J(\pi) = \mathbb{E}_\rho^\pi[G_0]$.**

where the state-value function satisfies the Bellman expectation equation

$$V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a), V^\pi(s') \right]. \tag{205}$$

The *first figure* (Figure 3a) plots $V^\pi(s)$ for several policies together with the optimal value function V^*. The optimal function is obtained as the fixed point of the Bellman optimality operator,

$$V^*(s) = \max_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a), V^*(s') \right], \tag{206}$$

and satisfies the pointwise dominance property

$$V^\pi(s) \le V^*(s), \quad \forall s \in \mathcal{S}, ; \forall \pi. \tag{207}$$

This ordering is clearly reflected in the plot, where the curve corresponding to V^* forms an upper envelope over all V^π, illustrating the principle of optimality.

The *second figure* (Figure 3b) visualizes the objective functional $J(\pi)$ for each policy, computed under a uniform initial distribution ρ. Numerically, this corresponds to

$$J(\pi) = \sum_{s\in\mathcal{S}} \rho(s), V^\pi(s) = \rho^\top V^\pi, \tag{208}$$

which aggregates the state-wise values into a single scalar performance metric. The bar plot shows how different policies induce different expected returns, thereby illustrating the optimization problem

$$\sup_{\pi\in\Pi} J(\pi). \tag{209}$$

The policy achieving the highest bar corresponds (up to numerical approximation) to the optimal policy π^*, for which

$$J(\pi^*) = \max_{\pi\in\Pi} J(\pi), \quad \text{and} \quad V^{\pi^*} = V^*. \tag{210}$$

In summary, the first figure (Figure 3a) illustrates the structure and ordering of value functions across policies, while the second figure (Figure 3b) demonstrates how these functions are aggregated into the scalar objective $J(\pi)$. Together, they provide a complete visualization of the mapping

$$\pi \longmapsto V^\pi \longmapsto J(\pi), \tag{211}$$

and its maximization leading to the optimal value function V^* and optimal policy π^*.

In summary, the objective functional provides a rigorous optimization criterion over the space of policies, the optimal value function characterizes the maximal achievable performance, and the Bellman optimality equation furnishes a recursive and computationally tractable characterization of optimal behavior.

3.6 Interpretation

The return G_t is a path-dependent functional that aggregates the future rewards along a trajectory and therefore encodes the long-term consequences of decisions. For a given trajectory $\omega = (s_0, a_0, s_1, a_1, \dots) \in \Omega$, the quantity

$$G_t(\omega) = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}(\omega) \tag{212}$$

depends on the entire future evolution of the system beyond time t. Thus, unlike immediate rewards, G_t captures delayed effects and trade-offs between short-term and long-term gains.

The policy π governs the stochastic selection of actions and thereby determines, together with the transition kernel P and initial distribution ρ, a probability measure $\mathbb{P}_\rho^\pi$ on the trajectory space $(\Omega, \mathcal{F})$. Under this measure, the sequence $\{(s_t, a_t)\}_{t\geq 0}$ becomes a well-defined stochastic process. Consequently, the expectation

$$\mathbb{E}_\rho^\pi[G_t] = \int_\Omega G_t(\omega)\, \mathbb{P}_\rho^\pi(d\omega) \tag{213}$$

represents an average over all possible trajectories, weighted according to their likelihood under the policy and the environment dynamics.

More explicitly, this expectation can be written in terms of iterated integrals induced by the sequence of kernels (ρ, π, P):

$$\begin{aligned}\mathbb{E}_\rho^\pi[G_0] = \int_{\mathcal{S}} \rho(ds_0) \int_{\mathcal{A}} \pi(da_0 \mid s_0) \int_{\mathcal{S}} P(ds_1 \mid s_0, a_0) \int_{\mathcal{A}} \pi(da_1 \mid s_1) \cdots \\ \times \left(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right),\end{aligned}$$

which makes explicit how both the stochasticity of the policy and the randomness of the environment contribute to the overall expectation. From this perspective, a policy π induces a distribution over trajectories:

$$\pi \;\longmapsto\; \mathbb{P}_\rho^\pi \in \mathcal{P}(\Omega), \tag{214}$$

where $\mathcal{P}(\Omega)$ denotes the space of probability measures on $(\Omega, \mathcal{F})$. The return G_t is a measurable functional $G_t : \Omega \to \mathbb{R}$, and the objective functional can be viewed as the composition

$$J(\pi) = \int_\Omega G_0(\omega)\, \mathbb{P}_\rho^\pi(d\omega). \tag{215}$$

Thus, the reinforcement learning problem can be interpreted as an optimization problem over probability measures induced by admissible policies.

This viewpoint highlights that two distinct sources of randomness are present:

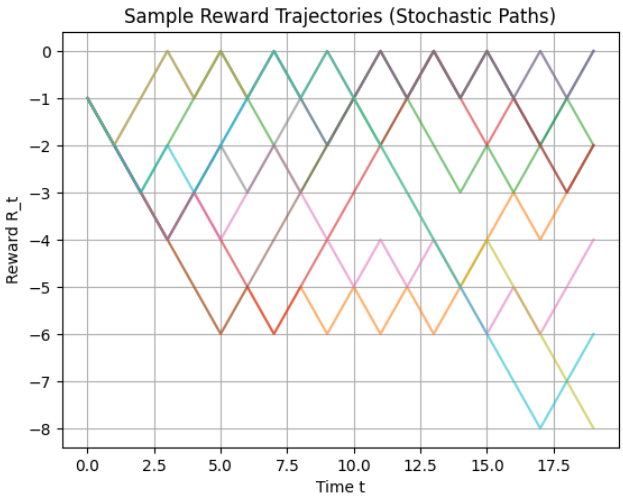
- *Environmental randomness*, arising from the transition kernel P (and possibly the reward kernel),
- *Policy-induced randomness*, arising from the stochastic kernel π.

The expectation $\mathbb{E}_\rho^\pi[G_t]$ integrates over both sources simultaneously, reflecting the combined uncertainty in the system.

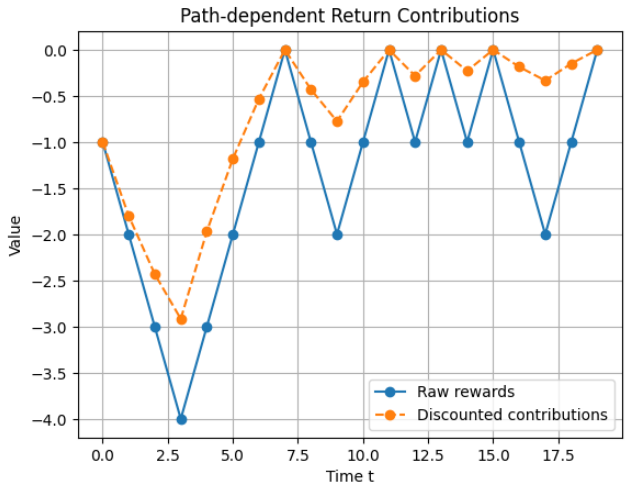
Furthermore, the recursive structure of the return,

$$G_t = R_{t+1} + \gamma G_{t+1}, \tag{216}$$

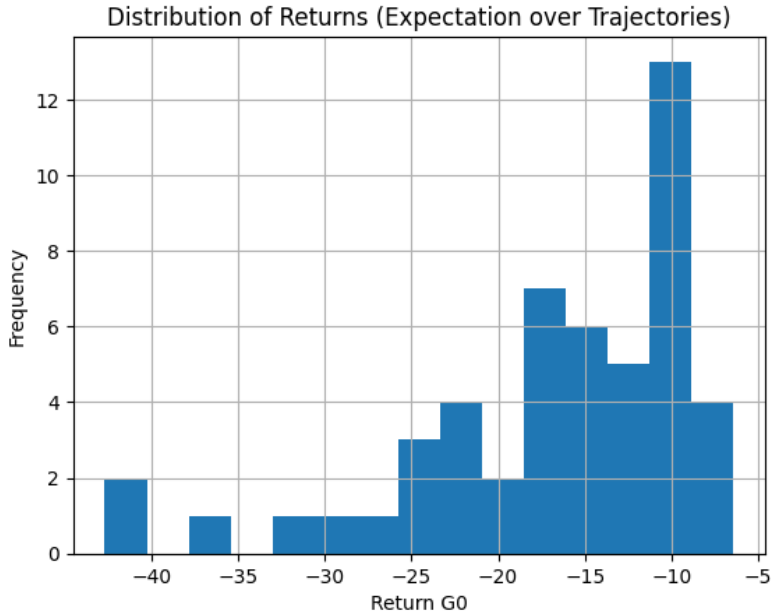
together with the Markov property, enables a decomposition of the global objective into local decisions. This is the foundation of dynamic programming, where one replaces the trajectory-level optimization problem by a sequence of one-step optimization problems characterized by the Bellman equations.



(a) **Sample reward trajectories**



(b) **Discounted contributions** $\gamma^k R_{k+1}$



(c) **Distribution of returns** G_0

Figure 4: **Visualization of the interpretation of return: stochastic trajectories, discounted contributions, and the induced distribution of returns approximating** $\mathbb{E}_\rho^\pi[G_0]$**.**

The three figures (Figures 4a, 4b, and 4c) provide a probabilistic interpretation of the return as a random variable defined on trajectory space. For each trajectory $\omega = (s_0, a_0, s_1, a_1, \dots) \in \Omega$, the return is given by

$$G_0(\omega) = \sum_{k=0}^{\infty} \gamma^k R_{k+1}(\omega), \tag{217}$$

which is a path-dependent functional. The *first figure* (Figure 4a) illustrates the inherent randomness in trajectories generated under the probability measure $\mathbb{P}_\rho^\pi$, i.e.,

$$\omega \sim \mathbb{P}_\rho^\pi, \tag{218}$$

showing that different realizations ω produce different sequences of states, actions, and rewards, and hence different values of $G_0(\omega)$.

The *second figure* (Figure 4b) focuses on the structure of the return itself by plotting the discounted contributions $\gamma^k R_{k+1}$, thereby making explicit the decomposition

$$G_0 = \sum_{k=0}^{\infty} \gamma^k R_{k+1}. \tag{219}$$

This visualization highlights the role of the discount factor $\gamma \in [0, 1)$, which geometrically attenuates the influence of rewards as (k) increases, ensuring convergence and emphasizing near-term rewards over distant ones.

Finally, the *third figure* (Figure 4c) illustrates the expectation of the return under the trajectory distribution. The objective functional is given by

$$\mathbb{E}_\rho^\pi[G_0] = \int_\Omega G_0(\omega)\, \mathbb{P}_\rho^\pi(d\omega), \tag{220}$$

and the plot approximates this expectation via empirical averaging over sampled trajectories:

$$\frac{1}{N} \sum_{i=1}^{N} G_0(\omega^{(i)}) \xrightarrow[N\to\infty]{} \mathbb{E}_\rho^\pi[G_0]. \tag{221}$$

Thus, the figure demonstrates how averaging over many realizations reduces variability and converges to the expected return. The three figures (Figures 4a, 4b, and 4c) collectively show that

1. Randomness in trajectories induces variability in returns,
2. The return aggregates discounted rewards along each trajectory, and
3. The expected return arises as an average over the probability measure on trajectory space, thereby connecting sample paths, functional evaluation, and expectation in a unified probabilistic framework.

In summary, policies define probability measures on trajectory space, the return assigns a cumulative reward to each trajectory, and the objective functional evaluates policies by integrating this reward over all possible trajectories. This unified probabilistic framework underlies both the theoretical analysis of optimal control problems and the development of practical reinforcement learning algorithms.

4 Value Functions

Value functions provide a precise quantitative characterization of the long-term performance of a policy by assigning to each state (or state-action pair) the expected cumulative reward that can be obtained starting from that configuration. Their definition relies on the return random variable and the probability measure induced by the policy and system dynamics.

Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space associated with a policy π, and let $\{(s_t, a_t)\}_{t\geq 0}$ be the induced stochastic process. For each $t \geq 0$, define the return

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{222}$$

which is assumed to be integrable under standard boundedness or moment conditions on the reward.

Definition 4.1 (State Value Function)**.** The state-value function associated with a policy π is defined as

$$V^\pi(s) = \mathbb{E}^\pi[G_t \mid s_t = s]. \tag{223}$$

This definition is independent of t in the infinite-horizon stationary setting due to time-homogeneity. More precisely, if π is stationary and the transition kernel does not depend on time, then

$$\mathbb{E}^\pi[G_t \mid s_t = s] = \mathbb{E}^\pi[G_0 \mid s_0 = s], \tag{224}$$

so that V^π is well-defined as a function on $\mathcal{S}$.

Definition 4.2 (Action Value Function)**.** The action-value function associated with a policy π is defined as

$$Q^\pi(s,a) = \mathbb{E}^\pi[G_t \mid s_t = s, a_t = a]. \tag{225}$$

The function $Q^\pi(s,a)$ represents the expected return when the agent starts in state s, takes action a at time t, and subsequently follows the policy π. It refines the information provided by V^π by resolving the contribution of individual actions. These value functions are related through the law of total expectation. Conditioning on the action selected at time t, one obtains

$$V^\pi(s) = \mathbb{E}^\pi\big[\,\mathbb{E}^\pi[G_t \mid s_t = s, a_t]\,\big|\, s_t = s\big] = \int_{\mathcal{A}} Q^\pi(s,a)\,\pi(da \mid s). \tag{226}$$

Furthermore, both V^π and Q^π satisfy recursive (Bellman) equations derived from the decomposition

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{227}$$

Taking conditional expectations yields

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} \mid s_t = s] + \gamma\,\mathbb{E}^\pi[G_{t+1} \mid s_t = s]. \tag{228}$$

Using the law of total expectation and the Markov property,

$$\mathbb{E}^\pi[R_{t+1} \mid s_t = s] = \int_{\mathcal{A}} R(s,a)\,\pi(da \mid s), \tag{229}$$

and

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s] = \int_{\mathcal{A}}\int_{\mathcal{S}} V^\pi(s')\,P(ds' \mid s,a)\,\pi(da \mid s). \tag{230}$$

Thus,

$$V^\pi(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\,P(ds' \mid s,a)\right] \pi(da \mid s). \tag{231}$$

Similarly, for the action-value function,

$$Q^\pi(s,a) = \mathbb{E}^\pi[R_{t+1} \mid s_t = s, a_t = a] + \gamma\,\mathbb{E}^\pi[G_{t+1} \mid s_t = s, a_t = a], \tag{232}$$

which simplifies to

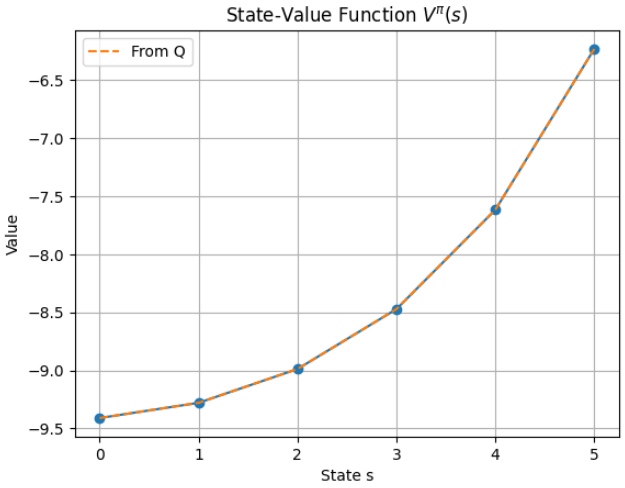
$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\,P(ds' \mid s,a). \tag{233}$$

The three figures (Figures 5a, 5b, and 5c) provide a concrete numerical realization of the abstract definitions and recursive structure of value functions. Figure 5a visualizes the state-value function, defined by

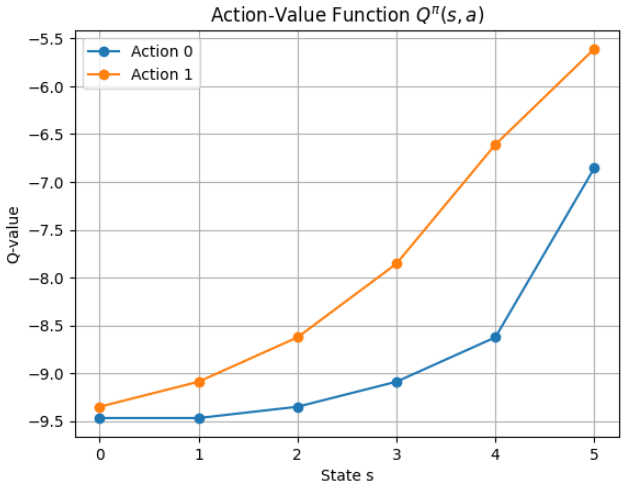
$$V^\pi(s) = \mathbb{E}^\pi\left[G_t \mid s_t = s\right], \tag{234}$$

thereby representing the long-term expected return starting from state s. The plot empirically confirms the consistency relation

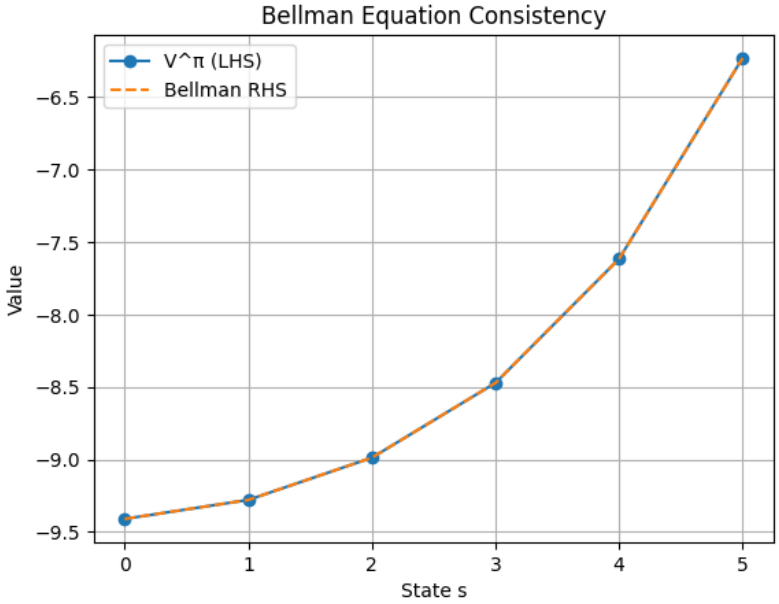
$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s), Q^\pi(s,a), \tag{235}$$



(a) **State-value function $V^\pi(s)$**



(b) **Action-value function $Q^\pi(s,a)$**



(c) **Bellman equation consistency**

Figure 5: **Visualization of value functions and their Bellman consistency. The first row shows V^π and Q^π, while the second row verifies the Bellman expectation equation.**

showing that the value of a state is obtained as an average of action-values weighted by the policy. Figure 5b depicts the action-value function, governed by the Bellman relation

$$Q^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a), V^\pi(s'), \tag{236}$$

and makes explicit how different actions induce different long-term returns; in particular, it reveals the dependence of future value on the immediate decision a. Finally, Figure 5c verifies the Bellman expectation equation for the state-value function,

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a), V^\pi(s') \right], \tag{237}$$

demonstrating numerically that the global expectation defining V^π coincides with its local recursive decomposition. Conceptually, these plots (Figures 5a, 5b, and 5c) illustrate a hierarchy:

1. V^π encodes the long-term performance of states,

2. Q^π refines this by resolving action-level consequences, and

3. The Bellman equation shows that this global notion of return is entirely characterized by one-step reward plus discounted continuation value.

In summary, the value functions encode the expected long-term return of a policy, admit recursive characterizations through Bellman equations, and serve as the fundamental quantities for both policy evaluation and optimization in reinforcement learning.

4.1 State Value Function

The state value function is a fundamental object in reinforcement learning and stochastic control, as it quantifies the long-term expected performance of a policy when the system starts from a given state. It provides a mapping from the state space to the real numbers, encoding the expected cumulative reward obtained by following a fixed policy thereafter.

The theory of the state value function occupies a central role in reinforcement learning, dynamic programming, and stochastic control, and has been systematically developed in the works of Sutton and Barto (1998) [1], Puterman (2014) [3], and Bertsekas and Tsitsiklis (1996) [2]. Sutton and Barto (1998) [1] introduced the state value function as the expected cumulative return obtained when an agent follows a given policy from a particular state, emphasizing its importance in policy evaluation, temporal-difference learning, Monte Carlo estimation, and Bellman recursion within reinforcement learning systems. Puterman (2014) [3] provided a rigorous mathematical formulation of value functions within the framework of Markov decision processes, analyzing discounted, finite-horizon, and average-cost criteria, and establishing the theoretical foundations of optimal value functions through Bellman optimality equations, contraction mappings, and existence theorems for optimal policies. Bertsekas and Tsitsiklis (1996) [2] further extended these ideas to large-scale and approximate dynamic programming settings, studying value function approximation, neuro-dynamic programming, convergence properties of iterative methods, and computational approaches for estimating value functions in high-dimensional stochastic control problems. Together, these works established the state value function as a fundamental mathematical object that quantifies long-term expected performance under a policy and serves as the basis for policy evaluation, optimal control, and modern reinforcement learning algorithms.

Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space induced by a policy π, and let $\{(s_t, a_t)\}_{t\geq 0}$ denote the controlled stochastic process. For each $t \geq 0$, define the return

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{238}$$

where $\gamma \in [0, 1)$ is the discount factor and $\{R_t\}$ is the reward sequence. Under standard assumptions (e.g., bounded rewards), G_t is integrable.

The state value function associated with a policy π is defined as

$$V^\pi(s) := \mathbb{E}^\pi[G_t \mid s_t = s], \tag{239}$$

for all $s \in \mathcal{S}$. In the infinite-horizon, time-homogeneous setting, this definition is independent of t, and one may equivalently write

$$V^\pi(s) = \mathbb{E}_s^\pi[G_0], \tag{240}$$

where $\mathbb{E}_s^\pi[\cdot]$ denotes expectation conditioned on $s_0 = s$. The function $V^\pi : \mathcal{S} \to \mathbb{R}$ is measurable and, under boundedness of rewards, satisfies

$$\|V^\pi\|_\infty \leq \frac{M}{1-\gamma}, \tag{241}$$

whenever $|R_t| \leq M$ almost surely. Thus, V^π belongs to the Banach space of bounded measurable functions. A key property of the state value function is its recursive characterization. Using the decomposition

$$G_t = R_{t+1} + \gamma G_{t+1}, \tag{242}$$

and applying conditional expectation, one obtains

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} \mid s_t = s] + \gamma\, \mathbb{E}^\pi[G_{t+1} \mid s_t = s]. \tag{243}$$

By the law of total expectation and the definition of the policy,

$$\mathbb{E}^\pi[R_{t+1} \mid s_t = s] = \int_{\mathcal{A}} R(s,a)\, \pi(da \mid s), \tag{244}$$

and

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s] = \int_{\mathcal{A}} \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a)\, \pi(da \mid s). \tag{245}$$

Combining these expressions yields the Bellman expectation equation

$$V^\pi(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a) \right] \pi(da \mid s). \tag{246}$$

Equivalently, defining the Bellman operator T^π acting on bounded measurable functions V by

$$(T^\pi V)(s) := \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds' \mid s,a) \right] \pi(da \mid s), \tag{247}$$

the state value function is characterized as the unique fixed point:

$$V^\pi = T^\pi V^\pi. \tag{248}$$

Moreover, T^π is a contraction mapping under the sup-norm:

$$\|T^\pi V - T^\pi W\|_\infty \le \gamma \|V - W\|_\infty, \tag{249}$$

which guarantees existence and uniqueness of V^π.

The three plots (Figures 6a, 6b, and 6c) provide a coherent numerical validation of the operator-theoretic and probabilistic structure underlying the state-value function. The *first plot* (Figure 6a) illustrates the convergence of the iterative scheme

$$V_{k+1} = T^\pi V_k, \tag{250}$$

where T^π is the Bellman expectation operator. Since T^π is a contraction on $(\mathcal{B} * b(\mathcal{S}), |\cdot| * \infty)$ with modulus $\gamma \in [0,1)$, one has the geometric error bound

$$|V_k - V^\pi| * \infty \le \gamma^k |V_0 - V^\pi| * \infty. \tag{251}$$

The trajectories in the first plot visibly stabilize as k increases, demonstrating convergence to the unique fixed point V^π satisfying

$$V^\pi = T^\pi V^\pi. \tag{252}$$

This empirical behavior directly reflects the contraction property and the Banach fixed-point theorem.

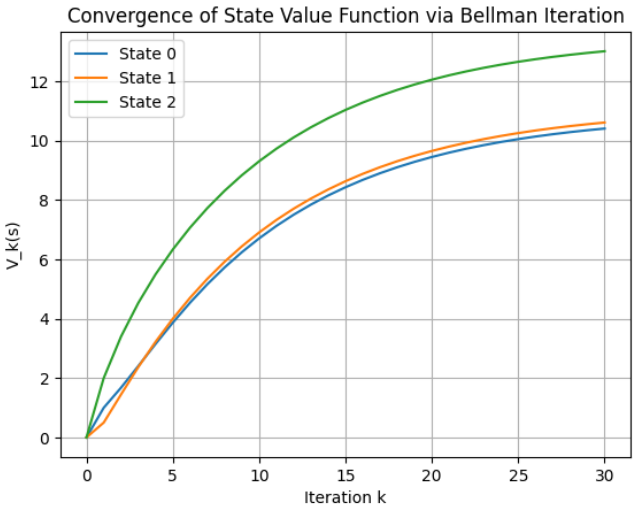
The *second plot* (Figure 6b) verifies the Bellman expectation equation itself. Writing the transition kernel under policy π as

$$P^\pi(s,s') := \sum_{a \in \mathcal{A}} \pi(a|s), P(s'|s,a), \tag{253}$$

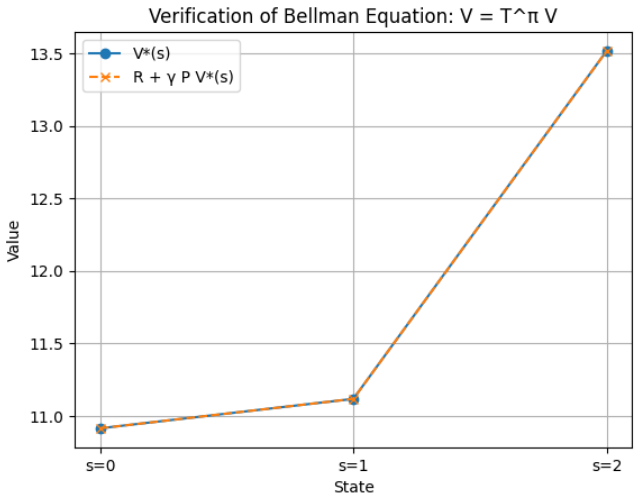
and the expected reward as $R^\pi(s) := \sum_a \pi(a|s) R(s,a)$, the Bellman equation can be expressed as

$$V^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in \mathcal{S}} P^\pi(s,s') V^\pi(s'). \tag{254}$$

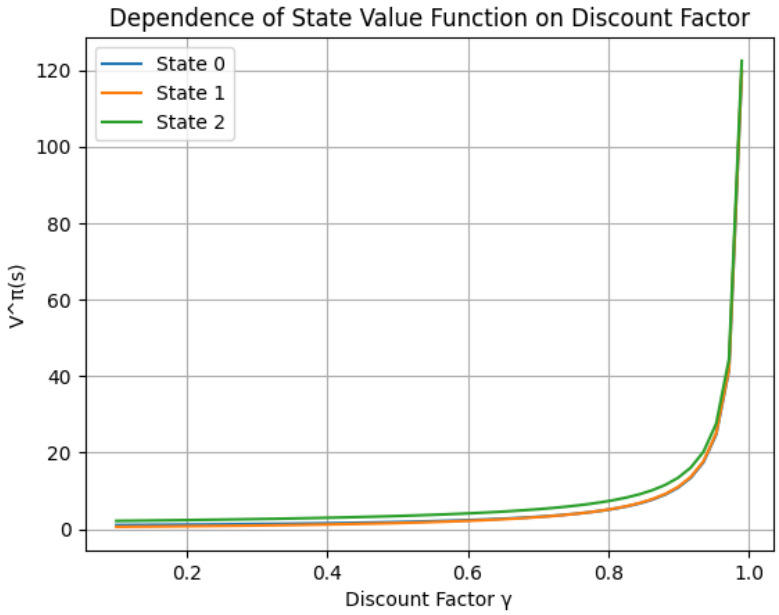
The plot compares the left-hand side $V^\pi(s)$ with the right-hand side computed from the model, showing numerical agreement across states. This confirms that the computed value function indeed satisfies the fixed-point relation pointwise.



(a) **Convergence of V_k**



(b) **Bellman consistency**



(c) **Effect of discount factor γ**

Figure 6: **Visualization of the state-value function: convergence, Bellman equation, and dependence on γ.**

The *third plot* (Figure 6c) highlights the role of the discount factor γ in shaping the value function. In vector form, the Bellman equation can be written as

$$V^\pi = R^\pi + \gamma P^\pi V^\pi, \tag{255}$$

which yields the explicit solution

$$V^\pi = (I - \gamma P^\pi)^{-1} R^\pi, \tag{256}$$

provided $(I - \gamma P^\pi)$ is invertible. The plot demonstrates how V^π varies with γ: as γ increases, the inverse $(I - \gamma P^\pi)^{-1}$ places greater weight on future transitions, amplifying the contribution of long-term rewards. This is reflected in the growth and reshaping of the value function curves, emphasizing the increasing importance of delayed outcomes. Taken together, the figures (Figures 6a, 6b, and 6c) show that

1. Iterative application of T^π converges geometrically to V^π,
2. The resulting function satisfies the Bellman equation exactly, and

3. The discount factor γ governs the balance between immediate and future rewards through the resolvent operator $(I - \gamma P^\pi)^{-1}$.

In summary, the state value function provides a complete characterization of the expected long-term return under a given policy, admits a recursive representation via the Bellman equation, and plays a central role in both theoretical analysis and computational methods in reinforcement learning.

4.2 Action Value Function

The action value function refines the notion of the state value function by incorporating the effect of an explicitly chosen action at the current state. It quantifies the expected cumulative reward obtained when the agent starts in a given state, executes a specified action, and subsequently follows a fixed policy. This function plays a central role in optimal control and reinforcement learning, as it allows direct comparison of alternative actions.

The concept of the action value function, or Q-function, forms one of the foundational components of modern reinforcement learning and was pioneered in the work of Watkins (1989) [14], and later systematically developed by Sutton and Barto (1998) [1] and by Szepesvári (2022) [13]. Watkins (1989) [14] introduced the action value function in the context of Q-learning, where the expected cumulative return associated with taking a particular action in a given state and subsequently following an optimal policy is recursively estimated through temporal-difference updates, thereby enabling model-free learning directly from delayed rewards without requiring explicit knowledge of transition probabilities. Sutton and Barto (1998) [1] further formalized the action value function within the broader reinforcement learning framework, emphasizing its role in policy evaluation, control, temporal-difference learning, Monte Carlo methods, SARSA, Q-learning, and Bellman optimality equations, while demonstrating how action-value methods enable agents to learn optimal behavior through interaction with stochastic environments. Szepesvári (2022) [13] provided a rigorous algorithmic and theoretical treatment of action-value learning methods, analyzing convergence properties, exploration–exploitation trade-offs, stochastic approximation techniques, and finite-sample behavior of reinforcement learning algorithms based on Q-functions and approximate dynamic programming principles. Together, these works established the action value function as a central mathematical and computational object in reinforcement learning, enabling optimal sequential decision-making through recursive estimation of long-term expected rewards associated with state–action pairs.

Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space induced by a policy π, and let $\{(s_t, a_t)\}_{t\geq 0}$ denote the associated controlled process. For each $t \geq 0$, define the return

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{257}$$

where $\gamma \in [0, 1)$ and $\{R_t\}$ is the reward sequence. Under standard boundedness or integrability assumptions, $G_t \in L^1(\mathbb{P}_\rho^\pi)$. The action value function associated with a policy π is defined by

$$Q^\pi(s, a) := \mathbb{E}^\pi[G_t \mid s_t = s,\, a_t = a]\,, \quad (s, a) \in \mathcal{S} \times \mathcal{A}. \tag{258}$$

In the infinite-horizon, time-homogeneous setting, this definition is independent of t, and one may equivalently write

$$Q^\pi(s, a) = \mathbb{E}^\pi[G_0 \mid s_0 = s,\, a_0 = a]\,. \tag{259}$$

The mapping $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is measurable and, if $|R_t| \leq M$ almost surely, satisfies the bound

$$|Q^\pi(s, a)| \leq \frac{M}{1 - \gamma}, \tag{260}$$

so that Q^π is bounded.

The action value function admits a recursive characterization derived from the decomposition

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{261}$$

Taking conditional expectation with respect to (s_t, a_t) yields

$$Q^\pi(s,a) = \mathbb{E}[R_{t+1} \mid s_t = s, a_t = a] + \gamma\, \mathbb{E}^\pi[G_{t+1} \mid s_t = s, a_t = a]. \tag{262}$$

If the reward admits a conditional expectation $R(s,a)$, then

$$\mathbb{E}[R_{t+1} \mid s_t = s, a_t = a] = R(s,a), \tag{263}$$

and by the Markov property,

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s, a_t = a] = \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a), \tag{264}$$

where V^π is the state value function. Consequently,

$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds' \mid s,a). \tag{265}$$

The relationship between Q^π and V^π is given by the law of total expectation:

$$V^\pi(s) = \int_{\mathcal{A}} Q^\pi(s,a)\, \pi(da \mid s), \tag{266}$$

which shows that the state value function is the expectation of the action value function under the policy.

Alternatively, Q^π can be expressed purely in terms of itself by substituting the expression for V^π:

$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} \left(\int_{\mathcal{A}} Q^\pi(s',a')\, \pi(da' \mid s') \right) P(ds' \mid s,a). \tag{267}$$

In operator form, defining T_Q^π acting on bounded measurable functions $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ by

$$(T_Q^\pi Q)(s,a) := R(s,a) + \gamma \int_{\mathcal{S}} \left(\int_{\mathcal{A}} Q(s',a')\, \pi(da' \mid s') \right) P(ds' \mid s,a), \tag{268}$$

one has the fixed-point relation

$$Q^\pi = T_Q^\pi Q^\pi. \tag{269}$$

Moreover, T_Q^π is a contraction under the sup-norm:

$$\|T_Q^\pi Q_1 - T_Q^\pi Q_2\|_\infty \le \gamma \|Q_1 - Q_2\|_\infty, \tag{270}$$

which ensures the existence and uniqueness of Q^π.

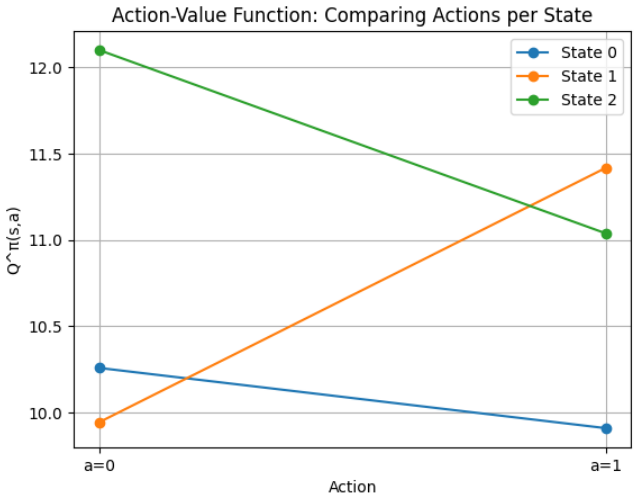
The three figures (Figures 7a, 7b, and 7c) provide a concrete numerical interpretation of the action-value formulation and its relation to the state-value function and Bellman recursion. The *first plot* (Figure 7a) illustrates the definition of the action-value function

$$Q^\pi(s,a) = \mathbb{E}^\pi \left[G_0 \mid s_0 = s, , a_0 = a \right], \tag{271}$$

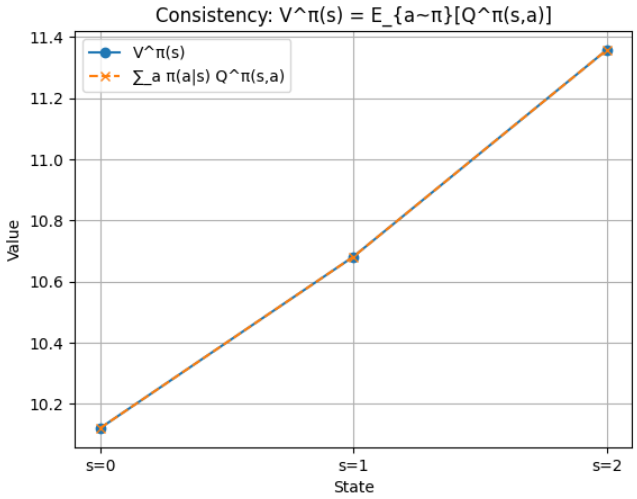
by explicitly comparing different actions at a fixed state. The separation between curves corresponding to distinct actions reflects how the choice of a influences the entire future trajectory and hence the long-term return. In particular, even if immediate rewards are similar, the plot shows that downstream transitions induce different cumulative outcomes, thereby justifying the need for Q^π as a finer evaluation criterion than V^π.

The *second plot* (Figure 7b) demonstrates the relationship between the state-value and action-value functions. According to the law of total expectation,

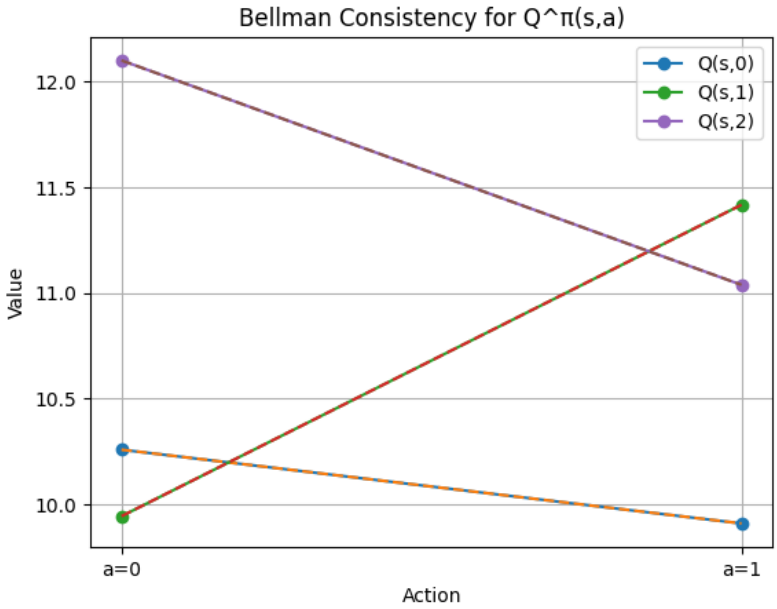
$$V^\pi(s) = \mathbb{E}^\pi[G_0 \mid s_0 = s] = \sum_{a \in \mathcal{A}} \pi(a|s), Q^\pi(s,a), \tag{272}$$



(a) **Action-value comparison:** $Q^\pi(s,a)$ **across actions**



(b) **Consistency:** $V^\pi(s) = \sum_a \pi(a|s) Q^\pi(s,a)$



(c) **Bellman fixed-point consistency for** Q^π

Figure 7: **Visualization of the action-value function** Q^π**: comparison across actions, relationship with** V^π**, and Bellman recursion.**

which expresses $V^\pi(s)$ as a policy-weighted average of action-values. Numerically, the plot verifies this identity by showing that the aggregated value coincides with the weighted combination of the $Q^\pi(s,a)$ curves. This confirms that V^π does not introduce new information, but rather summarizes Q^π under the stochastic policy π.

The *third plot* (Figure 7c) verifies the recursive (Bellman) structure of the action-value function. From the decomposition of the return,

$$G_0 = R_1 + \gamma G_1, \tag{273}$$

one obtains the fixed-point equation

$$Q^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s',a'), \tag{274}$$

which can be written compactly as

$$Q^\pi = T_Q^\pi Q^\pi. \tag{275}$$

The plot shows that successive iterates $Q_{k+1} = T_Q^\pi Q_k$ converge and stabilize, indicating that the fixed-point condition is satisfied numerically:

$$|Q_k - T_Q^\pi Q_k|_\infty \to 0. \tag{276}$$

This provides empirical evidence of the contraction property and the recursive consistency of Q^π. The figures (Figures 7a, 7b, and 7c) jointly demonstrate that

1. Action-values distinguish long-term consequences of actions,
2. State-values arise as expectations over action-values under the policy, and
3. The action-value function satisfies a Bellman fixed-point equation, thereby aligning the numerical behavior with the theoretical structure of the subsection.

In summary, the action value function provides a detailed evaluation of state-action pairs, admits recursive representations fundamental to dynamic programming, and enables direct comparison of alternative actions, making it central to both theoretical analysis and algorithmic implementations in reinforcement learning.

5 Bellman Equations

The Bellman equations provide a recursive characterization of value functions and form the analytical backbone of dynamic programming and reinforcement learning. They arise from the decomposition of the return into an immediate reward and a discounted future return, combined with the Markov property of the controlled process.

Let $(\Omega, \mathcal{F}, \mathbb{P}_\rho^\pi)$ be the probability space induced by a policy π, and recall the definition of the return

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{277}$$

Taking conditional expectations yields a one-step recursive relation for value functions.

Theorem 5.1 (Bellman Expectation Equation)**.** *For any policy π, the state-value function V^π satisfies*

$$V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a) V^\pi(s') \right]. \tag{278}$$

Derivation. Starting from

$$V^\pi(s) = \mathbb{E}^\pi[G_t \mid s_t = s], \tag{279}$$

and using the decomposition of G_t, we obtain

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} \mid s_t = s] + \gamma\, \mathbb{E}^\pi[G_{t+1} \mid s_t = s]. \tag{280}$$

Conditioning on the action a_t and applying the law of total expectation,

$$\mathbb{E}^\pi[R_{t+1} \mid s_t = s] = \sum_{a\in\mathcal{A}} \pi(a|s) R(s,a), \tag{281}$$

and

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s] = \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} P(s'|s,a) V^\pi(s'). \tag{282}$$

Combining these expressions yields the result. Equivalently, defining the Bellman operator T^π by

$$(T^\pi V)(s) := \sum_{a\in\mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s') \right], \tag{283}$$

the value function V^π is the unique fixed point:

$$V^\pi = T^\pi V^\pi. \tag{284}$$

Moreover, T^π is a contraction under the sup-norm:

$$\|T^\pi V - T^\pi W\|_\infty \leq \gamma \|V - W\|_\infty, \tag{285}$$

which guarantees existence and uniqueness.

Theorem 5.2 (Bellman Optimality Equation)**.** *The optimal value function V^* satisfies*

$$V^*(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^*(s') \right]. \tag{286}$$

Derivation and Interpretation. By definition,

$$V^*(s) = \sup_\pi V^\pi(s). \tag{287}$$

Using the Bellman expectation equation and the fact that the supremum over policies can be taken pointwise over actions, one obtains

$$V^*(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right]. \tag{288}$$

This expresses the principle of optimality: an optimal policy must choose, at each state, an action that maximizes the sum of immediate reward and discounted optimal future value. Defining the Bellman optimality operator T by

$$(TV)(s) := \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s') \right], \tag{289}$$

one has the fixed-point relation

$$V^* = TV^*. \tag{290}$$

The operator T is also a contraction:

$$\|TV - TW\|_\infty \leq \gamma \|V - W\|_\infty, \tag{291}$$

which ensures the existence and uniqueness of V^*. Furthermore, if the maximum is attained for each s, then there exists an optimal (deterministic stationary) policy π^* satisfying

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right], \tag{292}$$

and such a policy achieves

$$V^{\pi^*}(s) = V^*(s), \quad \forall s \in \mathcal{S}. \tag{293}$$

The three figures (Figures 8a, 8b, and 8c) provide a direct numerical realization of the Bellman equations and their fixed-point structure. The *first figure* (Figure 8a) corresponds to policy evaluation and illustrates the iterative scheme

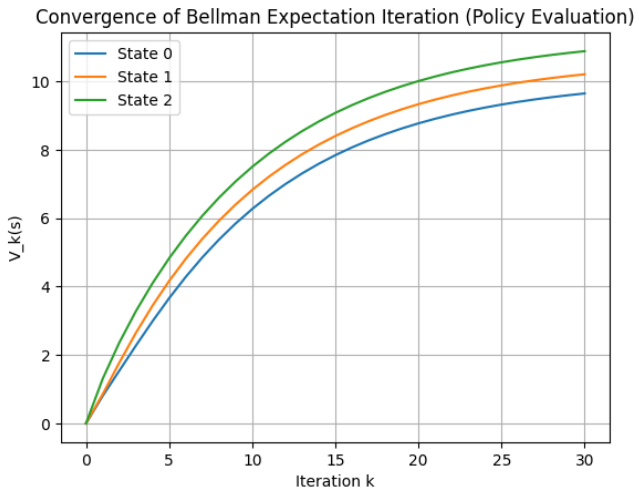
$$V_{k+1} = T^\pi V_k, \tag{294}$$

where T^π is the Bellman expectation operator. Since T^π is a γ-contraction on $(\mathcal{B} * b(\mathcal{S}), |\cdot| * \infty)$, one has

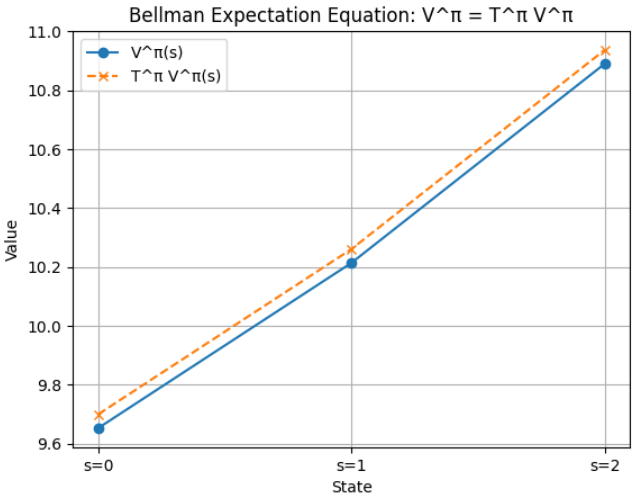
$$|V_{k+1} - V^\pi| * \infty = |T^\pi V_k - T^\pi V^\pi| * \infty \leq \gamma |V_k - V^\pi|_\infty, \tag{295}$$

which implies geometric convergence

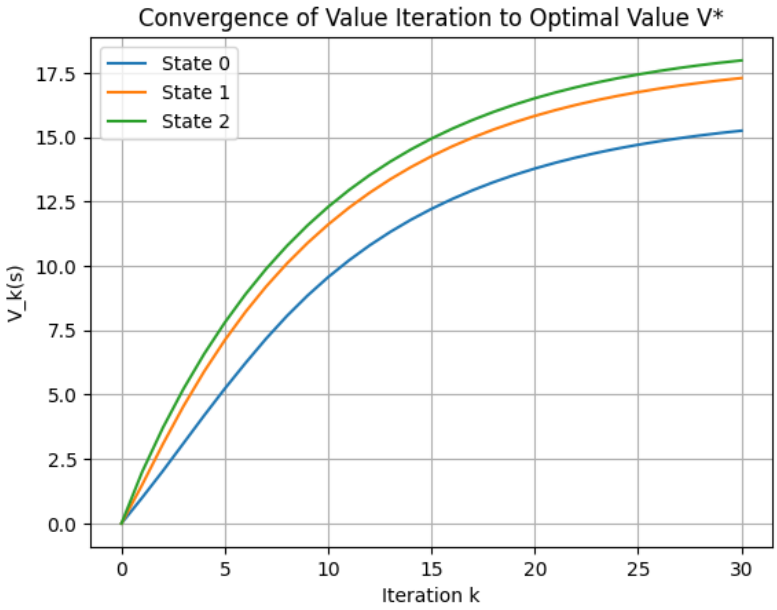
$$V_k \xrightarrow[k \to \infty]{} V^\pi. \tag{296}$$



(a) **Convergence of $V_k \to V^\pi$ (Policy Evaluation)**



(b) **Verification of $V^\pi = T^\pi V^\pi$**



(c) **Convergence of $V_k \to V^*$ (Value Iteration)**

Figure 8: **Visualization of the Bellman equations: policy evaluation, Bellman expectation consistency, and optimal value convergence.**

This behavior is reflected in the first plot, where the iterates progressively stabilize, demonstrating convergence to the policy-specific value function.

The *second figure* (Figure 8b) provides a numerical verification of the Bellman expectation equation itself. At convergence, the value function satisfies the fixed-point relation

$$V^\pi = T^\pi V^\pi, \tag{297}$$

or explicitly,

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^\pi(s') \right]. \tag{298}$$

The plot confirms that the discrepancy between V^π and $T^\pi V^\pi$ vanishes numerically, i.e.,

$$|V^\pi - T^\pi V^\pi|_\infty \approx 0, \tag{299}$$

thus validating the fixed-point characterization.

The *third figure* (Figure 8c) corresponds to value iteration and illustrates the convergence of the sequence

$$V_{k+1} = TV_k, \tag{300}$$

where T is the Bellman optimality operator,

$$(TV)(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s') \right]. \tag{301}$$

By the contraction property,

$$|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*|_\infty, \tag{302}$$

and hence

$$V_k \xrightarrow[k \to \infty]{} V^*, \tag{303}$$

where V^* satisfies the Bellman optimality equation

$$V^* = TV^*. \tag{304}$$

The third plot (Figure 8c) shows this convergence, illustrating the principle of optimality: the limiting function represents the maximal achievable value obtained by selecting actions that optimize the one-step reward plus discounted future value. The three plots (Figures 8a, 8b, and 8c) collectively demonstrate:

1. Contraction-driven convergence in policy evaluation,
2. Numerical verification of the Bellman expectation fixed point, and
3. Convergence of value iteration to the optimal value function, thereby tightly aligning empirical behavior with the underlying theoretical framework.

In summary, the Bellman expectation equation characterizes value functions for a fixed policy, while the Bellman optimality equation characterizes the maximal achievable value. Together, they reduce a global optimization problem over trajectories to local recursive relations, forming the theoretical foundation of dynamic programming and reinforcement learning algorithms.

5.1 Bellman Operators and Contraction Mapping

The Bellman equations admit a natural interpretation as fixed-point equations of operators acting on suitable function spaces. This perspective is fundamental, as it allows one to invoke powerful tools from functional analysis—most notably the Banach fixed-point theorem—to establish existence, uniqueness, and convergence properties of value functions.

The theory of Bellman operators and contraction mappings forms the mathematical backbone of dynamic programming and reinforcement learning, originating in the pioneering work of Bellman (1957) [15] and later rigorously developed by Bertsekas (2012) [9] and Puterman (2014) [3]. Bellman (1957) [15] introduced the principle of optimality and formulated dynamic programming through recursive functional equations, now known as Bellman equations, in which the optimal value function is characterized as the fixed point of an operator acting on a suitable function space. Bertsekas (2012) [9] further developed this framework using tools from nonlinear analysis and fixed-point theory, rigorously analyzing Bellman operators under discounted and stochastic settings and establishing their contraction properties with respect to appropriate norms, thereby guaranteeing the existence and uniqueness of optimal value functions as well as convergence of iterative schemes such as value iteration and policy iteration. Puterman (2014) [3] systematically extended these ideas within the theory of Markov decision processes, providing a comprehensive treatment of Bellman optimality operators, monotonicity properties, contraction mappings in supremum norms, and convergence theorems for discounted, finite-horizon, and average-cost stochastic control problems. Together, these works established the operator-theoretic foundation of sequential

decision-making by showing that optimal control and reinforcement learning problems can be formulated as fixed-point problems whose solutions are obtained through recursive contractions on spaces of value functions.

Let $\mathcal{B}_b(\mathcal{S})$ denote the space of all bounded, $\mathcal{B}(\mathcal{S})$-measurable functions $V : \mathcal{S} \to \mathbb{R}$ equipped with the supremum norm

$$\|V\|_\infty := \sup_{s\in\mathcal{S}} |V(s)|. \tag{305}$$

Then $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$ is a Banach space. For a fixed policy π, define the Bellman operator $T^\pi : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$ by

$$(T^\pi V)(s) := \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right] \pi(da|s). \tag{306}$$

The operator T^π is well-defined: if $V \in \mathcal{B}_b(\mathcal{S})$, then $T^\pi V$ is measurable and bounded, since

$$|(T^\pi V)(s)| \le \int_{\mathcal{A}} \left[|R(s,a)| + \gamma \|V\|_\infty \right] \pi(da|s) \le M + \gamma \|V\|_\infty, \tag{307}$$

whenever $|R(s,a)| \le M$.

To establish the contraction property, let $V, W \in \mathcal{B}_b(\mathcal{S})$. Then for each $s \in \mathcal{S}$,

$$\begin{aligned} |(T^\pi V)(s) - (T^\pi W)(s)| &= \left| \int_{\mathcal{A}} \gamma \int_{\mathcal{S}} \big(V(s') - W(s')\big) P(ds'|s,a)\, \pi(da|s) \right| \\ &\le \gamma \int_{\mathcal{A}} \int_{\mathcal{S}} |V(s') - W(s')| P(ds'|s,a)\, \pi(da|s) \\ &\le \gamma \|V - W\|_\infty. \end{aligned}$$

Taking supremum over $s \in \mathcal{S}$ yields

$$\|T^\pi V - T^\pi W\|_\infty \le \gamma \|V - W\|_\infty. \tag{308}$$

Since $\gamma \in [0,1)$, T^π is a strict contraction.Consequently, by the Banach fixed-point theorem, there exists a unique function $V^\pi \in \mathcal{B}_b(\mathcal{S})$ such that

$$V^\pi = T^\pi V^\pi, \tag{309}$$

and for any initial $V_0 \in \mathcal{B}_b(\mathcal{S})$, the sequence defined by $V_{k+1} = T^\pi V_k$ converges uniformly to V^π:

$$\|V_k - V^\pi\|_\infty \le \gamma^k \|V_0 - V^\pi\|_\infty. \tag{310}$$

Similarly, define the optimal Bellman operator $T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$ by

$$(TV)(s) := \sup_{a\in\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right]. \tag{311}$$

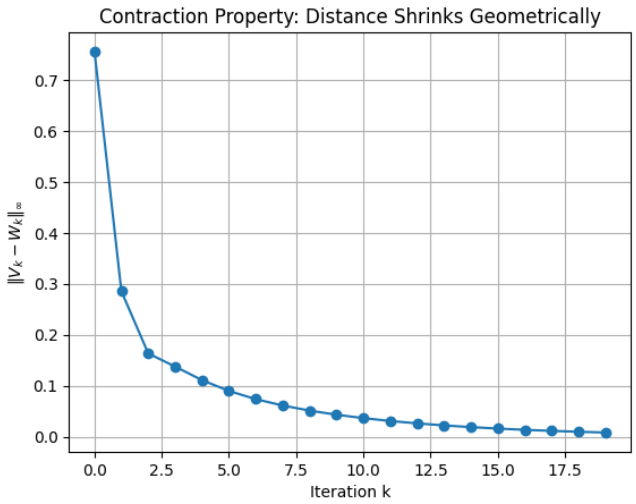
Under standard assumptions (e.g., bounded rewards and measurable maximization), T maps $\mathcal{B}_b(\mathcal{S})$ into itself.

To verify the contraction property, observe that for any $V, W \in \mathcal{B}_b(\mathcal{S})$ and any $s \in \mathcal{S}$,

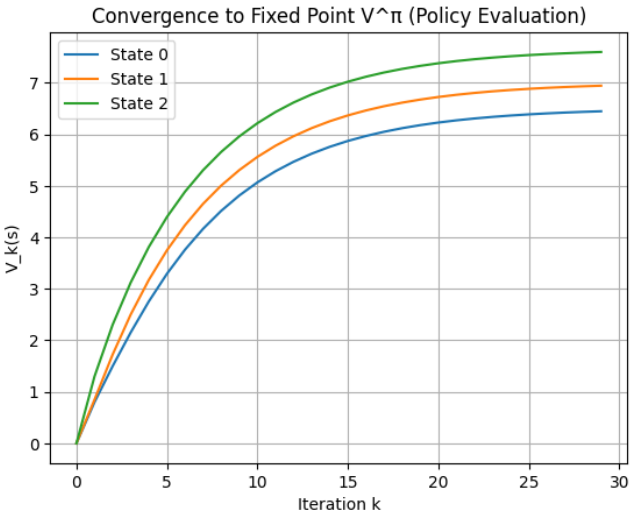
$$\begin{aligned} |(TV)(s) - (TW)(s)| &= \left| \sup_a \left[R(s,a) + \gamma \int V(s') P(ds'|s,a) \right] - \sup_a \left[R(s,a) + \gamma \int W(s') P(ds'|s,a) \right] \right| \\ &\le \sup_{a\in\mathcal{A}} \left| \gamma \int_{\mathcal{S}} (V(s') - W(s')) P(ds'|s,a) \right| \\ &\le \gamma \|V - W\|_\infty. \end{aligned}$$

Hence,

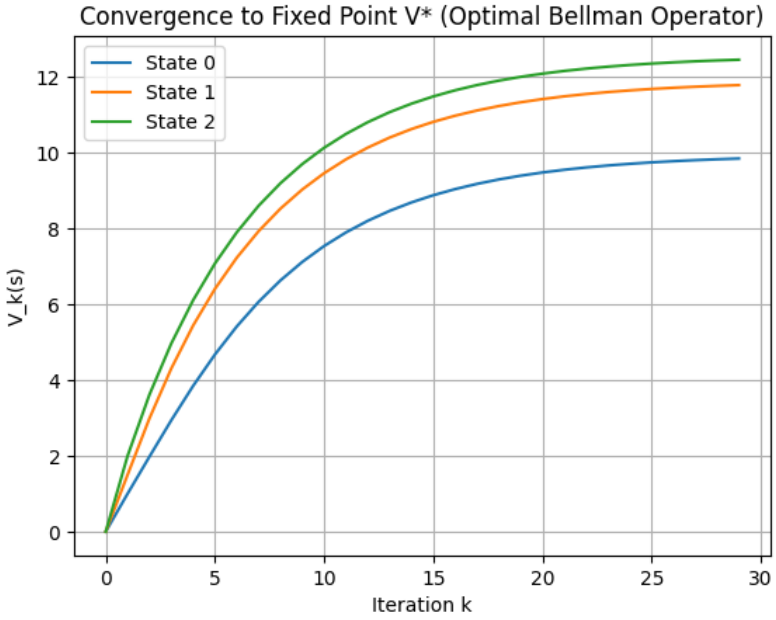
$$\|TV - TW\|_\infty \le \gamma \|V - W\|_\infty. \tag{312}$$



(a) **Contraction: $\|V_k - W_k\|_\infty$ decays geometrically**



(b) **Convergence to V^π under T^π**



(c) **Convergence to V^* under optimal Bellman operator T**

Figure 9: **Illustration of Bellman operators as contraction mappings: geometric decay of distances, convergence to V^π, and convergence to V^*.**

Therefore, T is also a contraction mapping, and by the Banach fixed-point theorem, there exists a unique $V^* \in \mathcal{B}_b(\mathcal{S})$ such that

$$V^* = TV^*. \tag{313}$$

This fixed point is precisely the optimal value function. Moreover, the contraction property implies that iterative schemes such as value iteration converge globally and at a geometric rate.
The three figures (Figure 9a, 9b, and 9c) provide a concrete numerical interpretation of the operator-theoretic properties established in the subsection on Bellman operators and contraction mappings. First, the *first figure* (Figure 9a) empirically verifies the contraction property of the Bellman operator T^π. For any two functions $V, W \in \mathcal{B} * b(\mathcal{S})$, one has

$$|T^\pi V - T^\pi W| * \infty \leq \gamma |V - W| * \infty, \quad \gamma \in [0, 1), \tag{314}$$

which implies that successive applications of T^π shrink distances between functions at a geometric rate. This is precisely what is observed in the plot: the distance between iterates decreases exponentially

across iterations, illustrating the inequality

$$|(T^\pi)^k V - (T^\pi)^k W| * \infty \leq \gamma^k |V - W|_\infty. \tag{315}$$

Second, the *second figure* (Figure 9b) demonstrates the fixed-point convergence associated with policy evaluation. Starting from an arbitrary initial function V_0, the iteration

$$V_{k+1} = T^\pi V_k \tag{316}$$

generates a sequence that converges uniformly to the unique fixed point V^π satisfying

$$V^\pi = T^\pi V^\pi. \tag{317}$$

The trajectories shown in the figure stabilize as $k \to \infty$, confirming that

$$\lim_{k\to\infty} |V_k - V^\pi|_\infty = 0, \tag{318}$$

which is a direct consequence of the contraction mapping theorem.

Finally, the *third figure* (Figure 9c) illustrates the analogous fixed-point convergence for the Bellman optimality operator T. The value iteration scheme

$$V_{k+1} = TV_k \tag{319}$$

produces a sequence converging to the optimal value function V^*, characterized by

$$V^* = TV^*. \tag{320}$$

The plotted trajectories again exhibit stabilization, indicating convergence:

$$\lim_{k\to\infty} |V_k - V^*| * \infty = 0. \tag{321}$$

This convergence is guaranteed by the contraction property

$$|TV - TW| * \infty \leq \gamma |V - W|_\infty, \tag{322}$$

which ensures both existence and uniqueness of the fixed point. The figures (Figure 9a, 9b, and 9c) collectively illustrate

1. The geometric contraction of Bellman operators,
2. The convergence of policy evaluation to V^π, and
3. The convergence of value iteration to V^*, thereby providing a unified empirical validation of the fixed-point framework underlying reinforcement learning.

In summary, the Bellman operators T^π and T provide a functional-analytic formulation of policy evaluation and optimal control. Their contraction properties guarantee well-posedness, uniqueness of solutions, and the convergence of iterative algorithms, thereby forming a rigorous foundation for reinforcement learning theory.

5.2 Value Iteration

Value iteration is a fundamental dynamic programming method for computing the optimal value function by successive approximation. It is based on repeated application of the Bellman optimality operator, exploiting its contraction property on the Banach space $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$.

Value iteration is one of the most fundamental algorithms in dynamic programming and reinforcement learning, originating from the pioneering work of Bellman (1957) [15] and later rigorously analyzed by Puterman (2014) [3] and by Bertsekas and Tsitsiklis (1996) [2]. Bellman (1957) [15] introduced value

iteration through the principle of optimality, formulating sequential decision problems as recursive optimization equations in which repeated application of the Bellman operator progressively approximates the optimal value function. Puterman (2014) [3] developed a comprehensive mathematical treatment of value iteration within the framework of Markov decision processes, establishing convergence theorems for discounted and finite-horizon problems, analyzing contraction properties of Bellman operators in suitable normed spaces, and studying computational aspects of iterative dynamic programming methods for stochastic control. Bertsekas and Tsitsiklis (1996) [2] further extended the theory to large-scale and approximate dynamic programming settings, investigating asynchronous value iteration, approximate value function representations, stochastic approximation methods, and the role of value iteration in neuro-dynamic programming and reinforcement learning algorithms for high-dimensional systems. Together, these works established value iteration as a recursive fixed-point computational method for obtaining optimal value functions and policies, thereby forming one of the central algorithmic foundations of stochastic control, dynamic programming, and modern reinforcement learning.

Let $T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$ be the Bellman optimality operator defined by

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right]. \tag{323}$$

Starting from an arbitrary initial function $V_0 \in \mathcal{B}_b(\mathcal{S})$, one defines the iterative sequence

$$V_{k+1} := TV_k, \quad k \geq 0. \tag{324}$$

Explicitly, this recursion takes the form

$$V_{k+1}(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V_k(s')\, P(ds'|s,a) \right], \quad s \in \mathcal{S}. \tag{325}$$

The well-definedness of the sequence $\{V_k\}$ follows from the fact that T maps $\mathcal{B}_b(\mathcal{S})$ into itself. Indeed, if V_k is bounded, then

$$|V_{k+1}(s)| \leq \sup_{a \in \mathcal{A}} [|R(s,a)| + \gamma \|V_k\|_\infty] \leq M + \gamma \|V_k\|_\infty, \tag{326}$$

whenever $|R(s,a)| \leq M$, so that V_{k+1} remains bounded.

A crucial property underlying value iteration is that T is a contraction mapping:

$$\|TV - TW\|_\infty \leq \gamma \|V - W\|_\infty, \quad \forall V, W \in \mathcal{B}_b(\mathcal{S}), \tag{327}$$

with modulus $\gamma \in [0,1)$. Consequently, by the Banach fixed-point theorem, T admits a unique fixed point V^* satisfying

$$V^* = TV^*, \tag{328}$$

which coincides with the optimal value function. Applying the contraction property iteratively yields

$$\|V_{k+1} - V^*\|_\infty = \|TV_k - TV^*\|_\infty \leq \gamma \|V_k - V^*\|_\infty. \tag{329}$$

By induction, one obtains the geometric error bound

$$\|V_k - V^*\|_\infty \leq \gamma^k \|V_0 - V^*\|_\infty. \tag{330}$$

Since $\gamma < 1$, it follows that

$$\lim_{k \to \infty} \|V_k - V^*\|_\infty = 0, \tag{331}$$

i.e., the convergence is uniform over $\mathcal{S}$. In addition to computing the value function, value iteration induces a sequence of greedy policies. For each $k \geq 0$, define

$$\pi_k(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V_k(s')\, P(ds'|s,a) \right]. \tag{332}$$

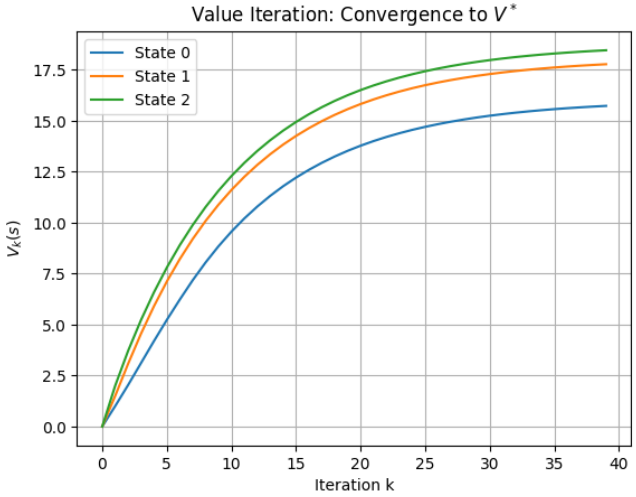
Under suitable conditions (e.g., existence of maximizers), any limit point of $\{\pi_k\}$ is an optimal policy π^* satisfying

$$V^{\pi^*}(s) = V^*(s), \quad \forall s \in \mathcal{S}. \tag{333}$$

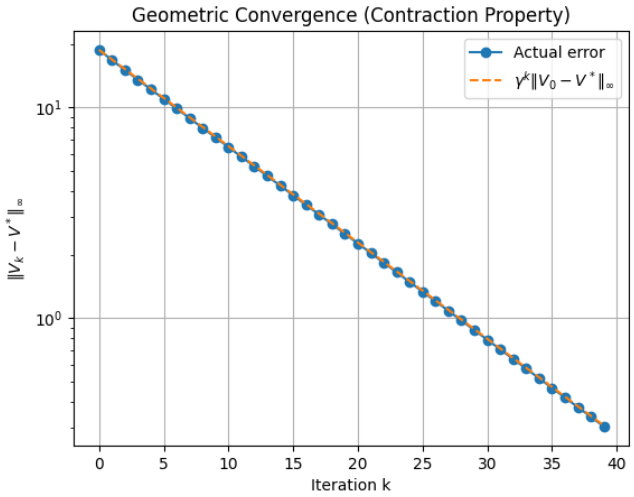
Furthermore, one can quantify the suboptimality of the greedy policy π_k in terms of the approximation error:

$$\|V^{\pi_k} - V^*\|_\infty \leq \frac{2\gamma}{1-\gamma}\|V_k - V^*\|_\infty, \tag{334}$$

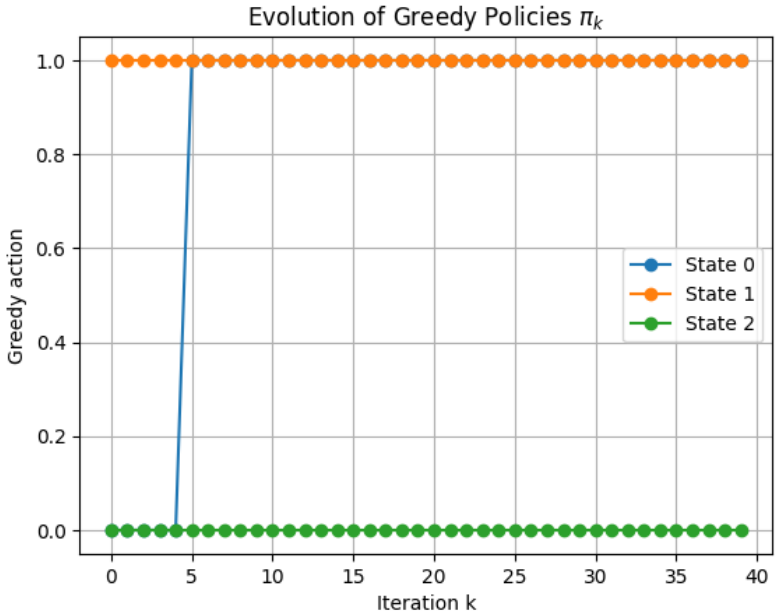
which shows that accurate value estimates lead to near-optimal policies.



(a) **Convergence of $V_k \to V^*$**



(b) **Geometric decay of $\|V_k - V^*\|_\infty$**



(c) **Evolution of greedy policies π_k**

Figure 10: **Illustration of value iteration: convergence to the optimal value function, geometric error decay due to contraction, and stabilization of greedy policies.**

The three figures (Figure 10a, 10b, and 10c) provide a direct empirical counterpart to the theoretical properties of value iteration governed by the Bellman optimality operator. First, the iterative scheme

$$V_{k+1} = TV_k \tag{335}$$

generates a sequence that converges to the unique fixed point V^* satisfying

$$V^* = TV^*. \tag{336}$$

By the Banach fixed-point theorem, this implies

$$V_k \xrightarrow[k\to\infty]{} V^* \quad \text{in } |\cdot|_\infty. \tag{337}$$

This fixed-point convergence is illustrated in the *first figure* (Figure 10a), where the trajectories $V_k(s)_k$ for representative states stabilize toward limiting values, visually confirming convergence to V^*.

Second, the Bellman operator T is a γ-contraction under the supremum norm, yielding the quantitative error bound

$$|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*| * \infty. \tag{338}$$

Taking logarithms gives

$$\log |V_k - V^*| * \infty \leq k \log \gamma + \log |V_0 - V^*|_\infty, \tag{339}$$

which predicts linear decay on a log-scale. This behavior is confirmed in the *second figure* (Figure 10b), where the error plotted against iteration index exhibits an approximately straight line, verifying geometric (exponential) convergence.

Finally, value iteration induces a sequence of greedy policies defined by

$$\pi_k(s) \in \text{argmax}_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s' \mid s,a), V_k(s') \right]. \tag{340}$$

As $V_k \to V^*$, these policies converge (up to tie-breaking) to an optimal policy π^* satisfying

$$\pi^*(s) \in \text{argmax}_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s' \mid s,a), V^*(s') \right]. \tag{341}$$

This convergence is illustrated in the *third figure* (Figure 10c), where the selected actions $\pi_k(s)$ stabilize after a finite number of iterations, indicating that the policy has reached optimality. The figures (Figure 10a, 10b, and 10c) collectively demonstrate

1. Fixed-point convergence of value iteration,
2. Validate the contraction-induced geometric rate of error decay, and
3. Show the emergence and stabilization of the optimal greedy policy, thereby providing a complete numerical illustration of the theoretical framework.

In summary, value iteration provides a globally convergent and computationally tractable method for solving the Bellman optimality equation. Its convergence is guaranteed by the contraction property of the Bellman operator, and its geometric rate ensures efficiency in practice, especially in finite or discretized settings.

5.3 Policy Iteration

Policy iteration is a fundamental method for solving Markov Decision Processes that alternates between evaluating a policy and improving it. Unlike value iteration, which directly iterates on value functions, policy iteration operates in the space of policies and exploits the structure of the Bellman equations to achieve rapid convergence.

Policy iteration is a foundational algorithm in dynamic programming and Markov decision processes that was first systematically developed by Howard (1960) [16] and later rigorously analyzed by Puterman (2014) [3] and Bertsekas (2012) [9]. Howard (1960) [16] introduced policy iteration as an iterative optimization procedure consisting of alternating policy evaluation and policy improvement steps,

demonstrating that sequential refinement of policies leads toward optimal decision rules in controlled Markov processes. Puterman (2014) [3] expanded the theoretical framework of policy iteration within discrete-time Markov decision processes by establishing convergence properties, optimality conditions, computational complexity considerations, and extensions to discounted, finite-horizon, and average-cost stochastic control problems. Bertsekas (2012) [9] further developed the mathematical and computational analysis of policy iteration through the lens of dynamic programming and optimal control, studying exact and approximate policy iteration schemes, asynchronous implementations, and their relationships to Bellman operators, contraction mappings, and reinforcement learning methods in large-scale optimization problems. Together, these works established policy iteration as one of the central algorithms for sequential decision-making, providing a rigorous framework for computing optimal policies through recursive evaluation and improvement procedures that remain fundamental in stochastic control and reinforcement learning theory.

Let π_0 be an arbitrary initial policy. The policy iteration algorithm generates a sequence $\{\pi_k\}_{k\geq 0}$ through two steps: policy evaluation and policy improvement. In the policy evaluation step, given π_k, one computes its associated value function V^{π_k} as the unique solution to the Bellman expectation equation:

$$V^{\pi_k} = T^{\pi_k} V^{\pi_k}, \tag{342}$$

where the Bellman operator T^{π_k} is defined by

$$(T^{\pi_k} V)(s) := \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right] \pi_k(da|s). \tag{343}$$

Equivalently, V^{π_k} satisfies

$$V^{\pi_k}(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^{\pi_k}(s')\, P(ds'|s,a) \right] \pi_k(da|s), \quad \forall s \in \mathcal{S}. \tag{344}$$

In the policy improvement step, a new policy π_{k+1} is defined by acting greedily with respect to V^{π_k}:

$$\pi_{k+1}(s) \in \operatorname{argmax}_{a\in\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^{\pi_k}(s')\, P(ds'|s,a) \right]. \tag{345}$$

This construction ensures that π_{k+1} is at least as good as π_k. The key property underlying policy iteration is the policy improvement theorem. Define, for any policy π, the action-value function

$$Q^{\pi}(s,a) := R(s,a) + \gamma \int_{\mathcal{S}} V^{\pi}(s')\, P(ds'|s,a). \tag{346}$$

Then the improvement step guarantees that

$$Q^{\pi_k}(s, \pi_{k+1}(s)) \geq V^{\pi_k}(s), \quad \forall s \in \mathcal{S}. \tag{347}$$

From this inequality, one can show that

$$V^{\pi_{k+1}}(s) \geq V^{\pi_k}(s), \quad \forall s \in \mathcal{S}, \tag{348}$$

so that the sequence $\{V^{\pi_k}\}$ is monotonically non-decreasing:

$$V^{\pi_0} \leq V^{\pi_1} \leq V^{\pi_2} \leq \cdots \leq V^{*}. \tag{349}$$

Moreover, strict improvement occurs unless π_k is already optimal. Indeed, if

$$\pi_{k+1}(s) = \pi_k(s), \quad \forall s \in \mathcal{S}, \tag{350}$$

then π_k satisfies the Bellman optimality condition and hence

$$V^{\pi_k} = V^{*}. \tag{351}$$

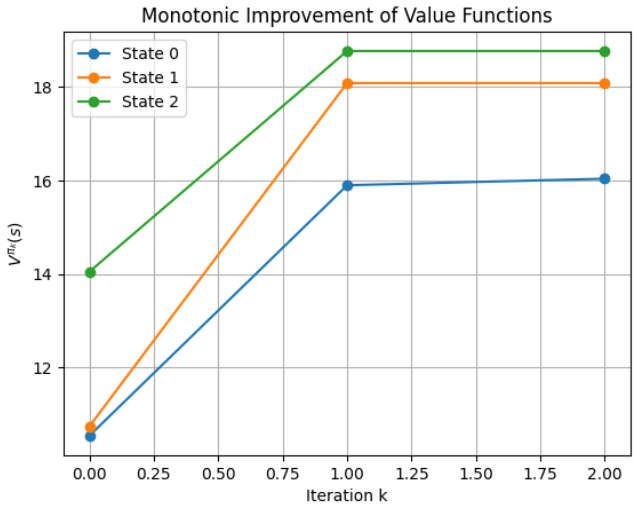
In the case where $\mathcal{S}$ and $\mathcal{A}$ are finite, the number of distinct deterministic policies is finite, say $|\mathcal{A}|^{|\mathcal{S}|}$. Since each iteration produces a strictly improved policy unless optimality is reached, the algorithm must terminate in a finite number of steps. At termination, one obtains a policy π^* satisfying

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^*(s') \right], \tag{352}$$

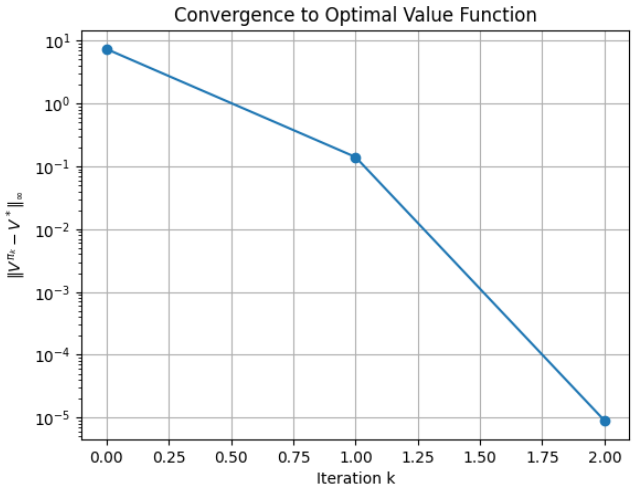
and

$$V^{\pi^*}(s) = V^*(s), \quad \forall s \in \mathcal{S}. \tag{353}$$

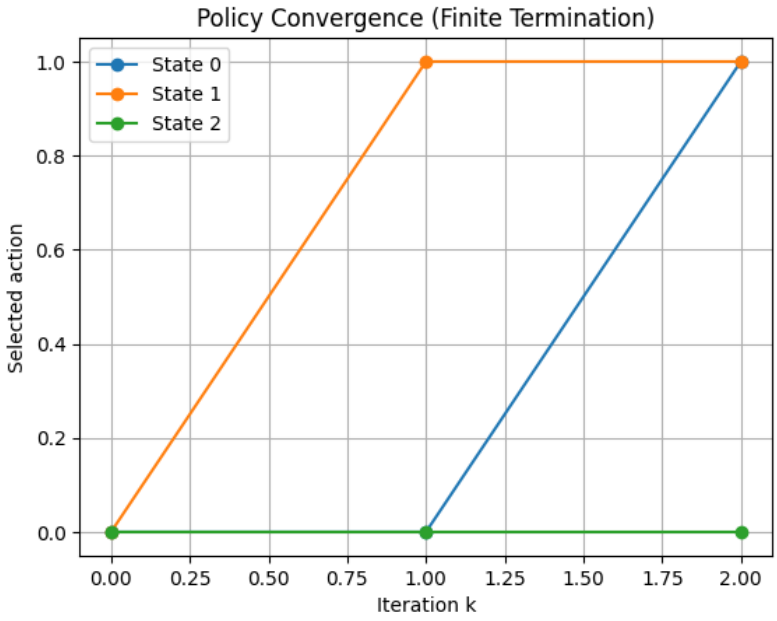
From an operator-theoretic viewpoint, policy iteration can be interpreted as alternating between computing the fixed point of T^{π_k} and updating the policy to be greedy with respect to this fixed point. This often leads to faster convergence than value iteration, as each policy evaluation step solves the Bellman equation exactly (or approximately in practical implementations).



(a) **Monotonic improvement of V^{π_k}**



(b) **Convergence to V^***



(c) **Finite-time convergence of policies π_k**

Figure 11: **Illustration of policy iteration: monotonic improvement of value functions, convergence to the optimal value, and stabilization of policies in a finite number of iterations.**

The three figures (Figure 11a, 11b, and 11c) provide a direct visual validation of the fundamental properties of policy iteration as described in the subsection. First, the sequence of policies π_k generated by alternating policy evaluation and policy improvement yields a corresponding sequence of value functions V^{π_k} that is monotonically non-decreasing. That is,

$$V^{\pi_0}(s) \leq V^{\pi_1}(s) \leq \cdots \leq V^*(s), \quad \forall s \in \mathcal{S}, \tag{354}$$

with strict inequality unless optimality is reached. This monotonic improvement follows from the policy improvement theorem, since

$$Q^{\pi_k}(s, \pi_{k+1}(s)) \geq V^{\pi_k}(s), \tag{355}$$

which implies $V^{\pi_{k+1}} \geq V^{\pi_k}$. This behavior is clearly illustrated in the *first figure* (Figure 11a), where the value functions increase across iterations and progressively approach their limiting values.

Second, the sequence V^{π_k} converges to the optimal value function V^*:

$$\lim_{k \to \infty} V^{\pi_k}(s) = V^*(s), \quad \forall s \in \mathcal{S}. \tag{356}$$

Equivalently, the approximation error vanishes:

$$|V^{\pi_k} - V^*|_\infty \to 0. \tag{357}$$

This convergence is depicted in the *second figure* (Figure 11b), where the error $|V^{\pi_k} - V^*|_\infty$ decreases toward zero, reflecting the fact that successive policies become increasingly optimal.

Finally, in the case of finite state and action spaces, policy iteration terminates in a finite number of steps. Since the number of deterministic policies is finite, strict improvement cannot continue indefinitely. If at some iteration k,

$$\pi_{k+1}(s) = \pi_k(s), \quad \forall s \in \mathcal{S}, \tag{358}$$

then π_k satisfies the Bellman optimality condition and is therefore optimal:

$$\pi_k = \pi^*, \qquad V^{\pi_k} = V^*. \tag{359}$$

This finite termination property is illustrated in the *third figure* (Figure 11c), where the sequence of policies stabilizes after a finite number of iterations, indicating that no further improvement is possible. The figures (Figure 11a, 11b, and 11c) collectively demonstrate

1. The monotonic improvement of value functions,
2. Convergence to the optimal value function, and
3. Finite-step stabilization of policies—core theoretical guarantees that make policy iteration a powerful and reliable method for solving Markov Decision Processes.

In summary, policy iteration provides a conceptually clear and computationally powerful procedure for solving Markov Decision Processes. Its monotonic improvement property, finite termination in the finite setting, and strong optimality guarantees make it one of the central algorithms in reinforcement learning and stochastic control.

5.4 Action-Value Bellman Equations

In addition to the state-value formulation, the Bellman equations admit a representation in terms of the action-value function, which provides a more refined characterization of the decision-making process by explicitly evaluating state–action pairs. This formulation is particularly useful in algorithmic settings where direct comparison between actions is required.

The theory of action-value Bellman equations forms a central component of modern reinforcement learning and was fundamentally shaped by the works of Watkins (1989) [14], Sutton and Barto (1998) [1],

and Bertsekas and Tsitsiklis. Watkins (1989) [14] introduced the action-value formulation through Q-learning, where the optimal action-value function satisfies a recursive Bellman optimality relation that enables agents to learn optimal behavior directly from sampled transitions and delayed rewards without requiring an explicit model of the environment. Sutton and Barto (1998) [1] systematically developed the theory of action-value Bellman equations within the broader reinforcement learning framework, showing how recursive relationships between state–action values and future expected returns underpin temporal-difference learning, SARSA, Q-learning, expected SARSA, and policy optimization methods. Bertsekas and Tsitsiklis extended these ideas to approximate dynamic programming and neuro-dynamic programming, rigorously analyzing convergence properties, stochastic approximation schemes, asynchronous updates, and function approximation methods associated with Bellman operators acting on action-value functions in high-dimensional stochastic control problems. Together, these works established the action-value Bellman equation as a recursive fixed-point characterization of optimal sequential decision-making, providing the mathematical and algorithmic foundation for model-free reinforcement learning and many contemporary deep reinforcement learning algorithms.

Recall that the action-value function associated with a policy π is defined as

$$Q^\pi(s,a) := \mathbb{E}^\pi[G_t \mid s_t = s,\, a_t = a]\,, \tag{360}$$

which represents the expected return when the agent starts in state s, takes action a, and thereafter follows policy π. Using the decomposition of the return,

$$G_t = R_{t+1} + \gamma G_{t+1}, \tag{361}$$

and taking conditional expectation with respect to (s_t, a_t), one obtains

$$Q^\pi(s,a) = \mathbb{E}[R_{t+1} \mid s_t = s, a_t = a] + \gamma\, \mathbb{E}^\pi[G_{t+1} \mid s_t = s, a_t = a]. \tag{362}$$

By the definition of the reward function and the Markov property,

$$\mathbb{E}[R_{t+1} \mid s_t = s, a_t = a] = R(s,a), \tag{363}$$

and

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s, a_t = a] = \int_{\mathcal{S}} V^\pi(s')\, P(ds'|s,a), \tag{364}$$

which yields the relation

$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds'|s,a). \tag{365}$$

Substituting the representation of the state-value function

$$V^\pi(s') = \int_{\mathcal{A}} Q^\pi(s',a')\, \pi(da'|s'), \tag{366}$$

one obtains the Bellman expectation equation in action-value form:

$$Q^\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} \left(\int_{\mathcal{A}} Q^\pi(s',a')\, \pi(da'|s') \right) P(ds'|s,a). \tag{367}$$

This equation characterizes Q^π as the unique fixed point of the operator T_Q^π defined by

$$(T_Q^\pi Q)(s,a) := R(s,a) + \gamma \int_{\mathcal{S}} \left(\int_{\mathcal{A}} Q(s',a')\, \pi(da'|s') \right) P(ds'|s,a). \tag{368}$$

Under the sup-norm on bounded measurable functions on $\mathcal{S} \times \mathcal{A}$, this operator is a contraction:

$$\|T_Q^\pi Q_1 - T_Q^\pi Q_2\|_\infty \le \gamma \|Q_1 - Q_2\|_\infty, \tag{369}$$

ensuring existence and uniqueness of Q^π.

The optimal action-value function Q^* is defined by

$$Q^*(s,a) := \sup_\pi Q^\pi(s,a), \tag{370}$$

and satisfies the Bellman optimality equation

$$Q^*(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} \max_{a' \in \mathcal{A}} Q^*(s',a')\, P(ds'|s,a). \tag{371}$$

This equation reflects the principle of optimality: after taking action a in state s, the agent transitions to a new state s' and thereafter behaves optimally, selecting the action that maximizes the continuation value. Defining the optimal Bellman operator T_Q by

$$(T_Q Q)(s,a) := R(s,a) + \gamma \int_{\mathcal{S}} \max_{a' \in \mathcal{A}} Q(s',a')\, P(ds'|s,a), \tag{372}$$

one has the fixed-point relation

$$Q^* = T_Q Q^*, \tag{373}$$

and T_Q is also a contraction:

$$\|T_Q Q_1 - T_Q Q_2\|_\infty \leq \gamma \|Q_1 - Q_2\|_\infty. \tag{374}$$

An important advantage of the action-value formulation is that the optimal policy can be recovered directly from Q^* without requiring knowledge of the state-value function. Specifically,

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s,a), \tag{375}$$

which defines an optimal (deterministic stationary) policy.

The three figures (Figures 12a, 12b, and 12c) collectively provide a concrete illustration of the theoretical properties underlying value iteration and its connection to the Bellman optimality operator. First, the iteration

$$V_{k+1} = TV_k \tag{376}$$

generates a sequence in the Banach space $(\mathcal{B} * b(\mathcal{S}), |\cdot| * \infty)$ that converges to the unique fixed point V^* satisfying

$$V^* = TV^*. \tag{377}$$

This fixed-point convergence is a direct consequence of the contraction property of T, and is visually demonstrated in the *first figure* (Figure 12a), where the trajectories $V_k(s)_k$ for different states stabilize to limiting values, illustrating

$$V_k \xrightarrow[k \to \infty]{} V^*. \tag{378}$$

Second, the contraction mapping property yields a quantitative bound on the rate of convergence. For all $k \geq 0$,

$$|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*| * \infty, \tag{379}$$

which shows that the error decays at a geometric rate governed by $\gamma \in [0,1)$. This behavior is explicitly confirmed in the *second figure* (Figure 12b), where the error $|V_k - V^*| * \infty$ is plotted on a logarithmic scale and exhibits an approximately linear decay, consistent with exponential convergence:

$$\log |V_k - V^*|_\infty \sim k \log \gamma. \tag{380}$$

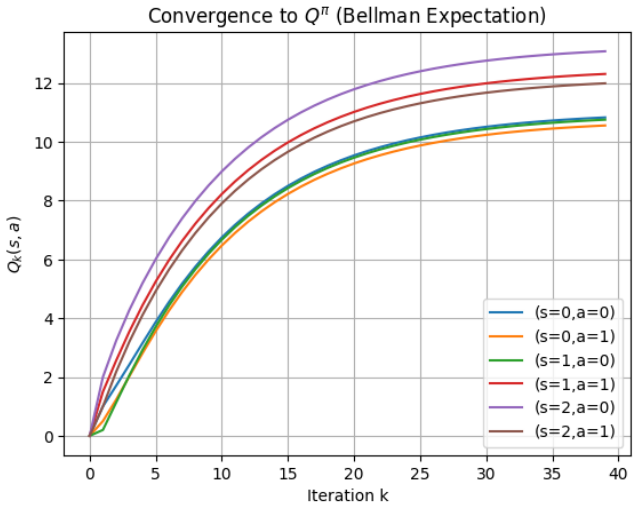
Finally, value iteration induces a sequence of greedy policies defined by

$$\pi_k(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s' \mid s,a) V_k(s') \right]. \tag{381}$$

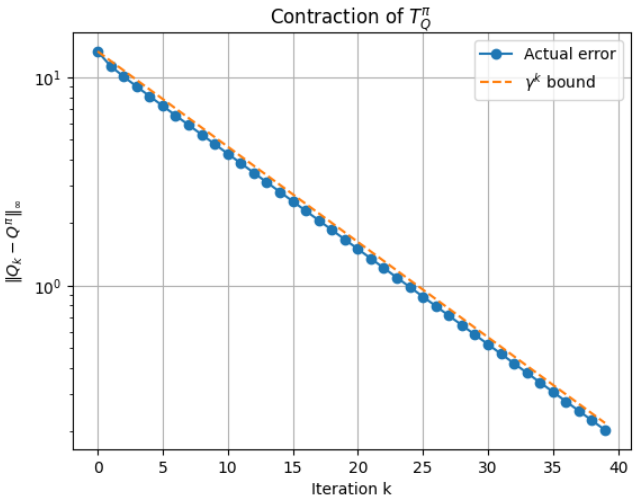
As $V_k \to V^*$, these policies converge (up to possible ties) to an optimal policy π^* satisfying

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^*(s') \right]. \tag{382}$$

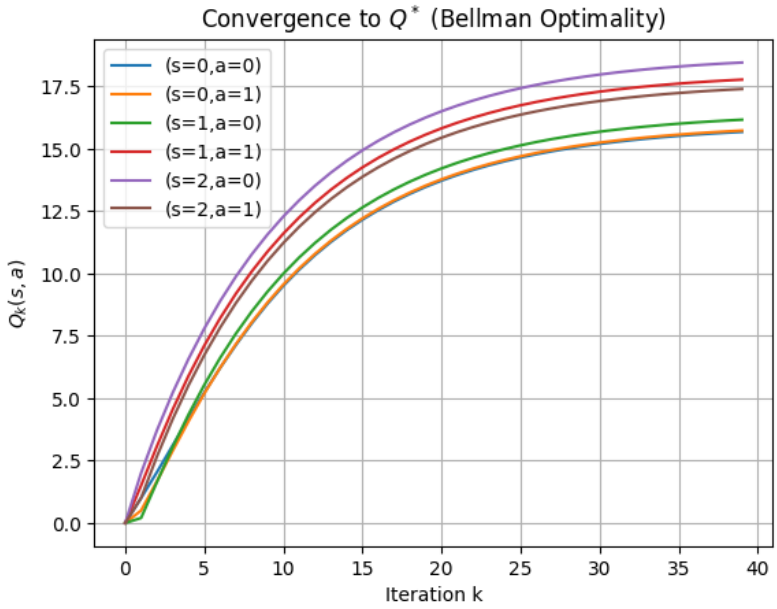
This phenomenon is illustrated in the *third figure* (Figure 12c), where the selected actions $\pi_k(s)$ stabilize after a finite number of iterations, reflecting convergence to optimal decision rules. The figures (Figures 12a, 12b, and 12c) jointly validate the theoretical framework:



(a) **Convergence to Q^π**



(b) **Contraction of T_Q^π**



(c) **Convergence to Q^* (Bellman optimality)**

Figure 12: **Illustration of action-value Bellman equations: convergence of Q_k to Q^π, geometric contraction behavior, and convergence to the optimal action-value function Q^*.**

1. The first demonstrates fixed-point convergence,
2. The second confirms the contraction-induced geometric rate, and
3. The third shows how optimal policies emerge naturally from the limiting value function.

In summary, the action-value Bellman equations provide a complete and self-contained characterization of optimal behavior at the level of state–action pairs. They are central to many reinforcement learning algorithms, particularly those that operate without explicit models of the transition dynamics, such as Q-learning.

5.5 Existence of Optimal Policies

A central question in stochastic control and reinforcement learning is whether there exists a policy that attains the optimal value function. Under standard structural assumptions, one can establish not only existence but also a strong structural characterization of optimal policies.

The existence of optimal policies in stochastic control and Markov decision processes has been rigorously established through the foundational works of Hernández-Lerma and Lasserre (2012) [5], Puterman (2014) [3], and Bertsekas and Shreve (1996) [4]. Hernández-Lerma and Lasserre (2012) [5] developed a comprehensive measure-theoretic framework for discrete-time Markov control processes on general Borel spaces, employing measurable selection theorems, compactness arguments, and stochastic kernel analysis to derive conditions guaranteeing the existence of optimal stationary and nonstationary policies under discounted and average-cost criteria. Puterman (2014) [3] systematically analyzed the existence and structure of optimal policies within finite and countable-state Markov decision processes, establishing rigorous results for discounted, finite-horizon, and average-reward formulations through Bellman optimality theory, contraction mappings, and dynamic programming principles. Bertsekas and Shreve (1996) [4] extended these ideas to stochastic optimal control in general measurable spaces, emphasizing the interplay between measurable policies, stochastic transition kernels, and recursive optimization equations while providing rigorous proofs for the existence of optimal controls under broad probabilistic and analytical assumptions. Together, these works established the mathematical foundations ensuring that optimal decision rules exist for a wide class of controlled stochastic systems, thereby providing the theoretical basis for dynamic programming, stochastic control, and reinforcement learning.

Assume that $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$ are standard Borel spaces, the reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is bounded and measurable, the transition kernel $P(\cdot|s, a)$ is Borel measurable, and the discount factor satisfies $\gamma \in [0, 1)$. Under these assumptions, the Bellman optimality operator

$$(TV)(s) := \sup_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s, a) \right] \tag{383}$$

is a contraction on $\mathcal{B}_b(\mathcal{S})$ and admits a unique fixed point V^* satisfying

$$V^* = TV^*. \tag{384}$$

The function V^* is the optimal value function:

$$V^*(s) = \sup_{\pi \in \Pi} V^\pi(s), \quad \forall s \in \mathcal{S}. \tag{385}$$

The existence of an optimal policy π^* such that

$$V^{\pi^*}(s) = V^*(s), \quad \forall s \in \mathcal{S}, \tag{386}$$

follows from the ability to select, for each state s, an action that attains the supremum in the Bellman optimality equation.

Define the set-valued mapping

$$\Gamma(s) := \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds'|s, a) \right]. \tag{387}$$

Under the stated assumptions (in particular, measurability and compactness-type conditions or upper semicontinuity when required), the mapping Γ admits a measurable selector $\mu^* : \mathcal{S} \to \mathcal{A}$ such that

$$\mu^*(s) \in \Gamma(s), \quad \forall s \in \mathcal{S}. \tag{388}$$

Using this selector, define a policy π^* by

$$\pi^*(\cdot|s) = \delta_{\mu^*(s)}(\cdot), \tag{389}$$

i.e., π^* is deterministic and stationary. By construction, for every $s \in \mathcal{S}$,

$$V^*(s) = R(s, \mu^*(s)) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds'|s, \mu^*(s)). \tag{390}$$

To verify optimality, define the Bellman operator associated with π^*:

$$(T^{\pi^*}V)(s) = R(s, \mu^*(s)) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s, \mu^*(s)). \tag{391}$$

Then the above relation implies

$$V^* = T^{\pi^*}V^*. \tag{392}$$

Since T^{π^*} is a contraction with a unique fixed point, it follows that

$$V^{\pi^*} = V^*, \tag{393}$$

establishing that π^* is optimal.

Thus, under standard assumptions, there exists an optimal policy that is both stationary and deterministic. This is a remarkable structural result, as it shows that the search over the large class of history-dependent stochastic policies Π can be restricted, without loss of optimality, to the much smaller class of deterministic stationary policies.

The key mathematical ingredient enabling this result is the measurable selection theorem, which guarantees the existence of a measurable function μ^* selecting maximizers from the set-valued mapping $\Gamma(s)$. This ensures that the resulting policy is admissible and compatible with the underlying measurable structure of the problem.

The three figures (Figure 13a, 13b, and 13c) provide a concrete numerical verification of the existence and structural characterization of an optimal policy. The optimal value function V^* is obtained as the unique fixed point of the Bellman optimality operator,

$$V^*(s) = (TV^*)(s) := \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a), V^*(s') \right]. \tag{394}$$

The *third figure* (Figure 13c) illustrates this fixed-point property by plotting the Bellman residual,

$$|V^* - TV^*| * \infty = \sup * s \in \mathcal{S} \, |V^*(s) - (TV^*)(s)| \,, \tag{395}$$

which is numerically close to zero, thereby confirming that V^* satisfies the Bellman equation.

Given V^*, one defines the set of maximizers

$$\Gamma(s) := \text{argmax}_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s'} P(s'|s, a), V^*(s') \right], \tag{396}$$

and selects a measurable function $\mu^*(s) \in \Gamma(s)$. The *second figure* (Figure 13b) visualizes this selector as a deterministic mapping $s \mapsto \mu^*(s)$, thereby illustrating the existence of a stationary deterministic policy. This induces the policy

$$\pi^*(\cdot|s) = \delta_{\mu^*(s)}(\cdot), \tag{397}$$

which assigns probability one to the maximizing action.

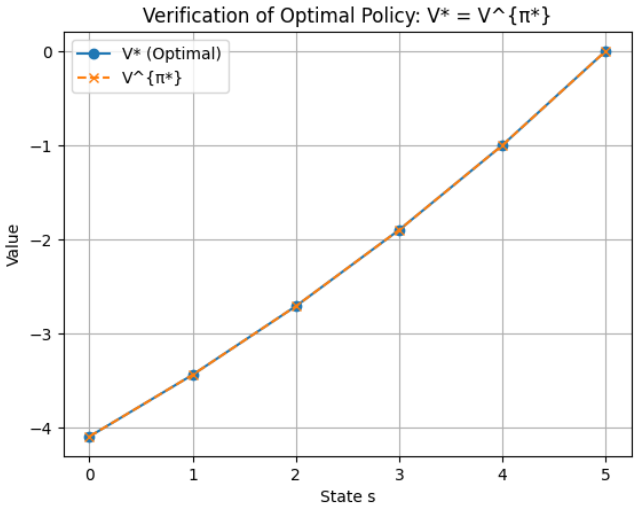
To verify optimality, consider the value function of this policy, which satisfies

$$V^{\pi^*}(s) = R(s, \mu^*(s)) + \gamma \sum_{s'} P(s'|s, \mu^*(s)), V^{\pi^*}(s'). \tag{398}$$

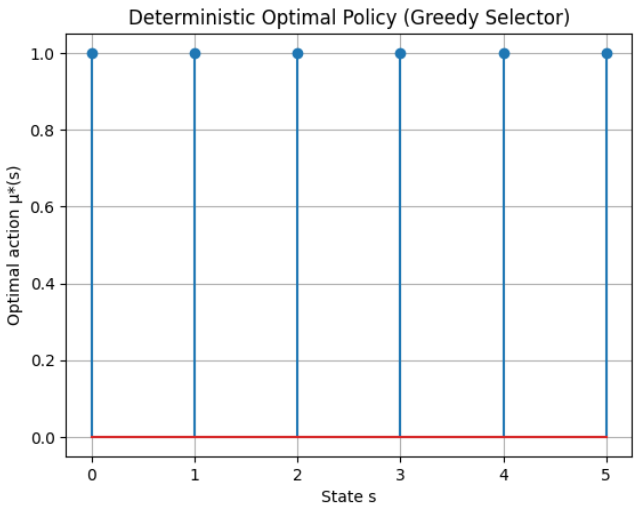
Since V^* satisfies the same relation and the associated Bellman operator is a contraction, uniqueness of the fixed point implies

$$V^{\pi^*} = V^*. \tag{399}$$

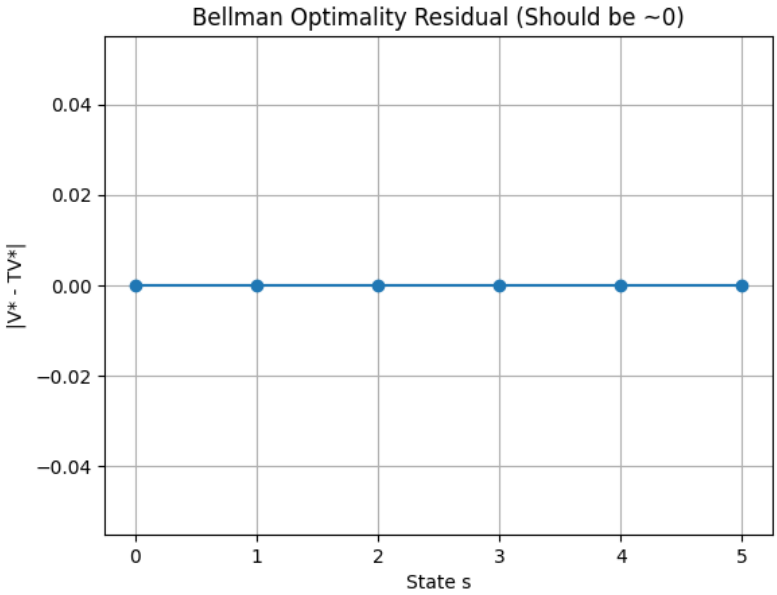
This equality is confirmed in the *first figure* (Figure 13a), where the curves corresponding to V^{π^*} and V^* coincide (up to numerical precision). The figures (Figure 13a, 13b, and 13c) collectively demonstrate that



(a) **Optimal value vs policy value**



(b) **Greedy policy** $\mu^*(s)$



(c) **Bellman optimality residual**

Figure 13: **Visualization of existence of an optimal deterministic stationary policy.**

1. The Bellman operator admits a fixed point V^*,
2. A measurable selector μ^* yields a deterministic stationary policy, and
3. The induced policy π^* attains the optimal value, thereby providing a complete numerical illustration of the existence of optimal policies.

In summary, the existence of optimal stationary deterministic policies provides a rigorous foundation for focusing on such policies in both theoretical analysis and algorithmic design, significantly simplifying the structure of optimal control problems in reinforcement learning.

5.6 Interpretation of Bellman Equations

The Bellman equations encode the *principle of optimality*, which asserts that an optimal policy must be composed of optimal decisions at every stage. Formally, the global optimization over trajectories can be decomposed into a sequence of local one-step optimization problems.

The interpretation of Bellman equations as recursive characterizations of optimal sequential decision-making lies at the heart of dynamic programming and reinforcement learning, and has been profoundly developed in the works of Bellman (1957) [15], Sutton and Barto (1998) [1], and Bertsekas (2012) [9]. Bellman (1957) [15] introduced the principle of optimality and showed that complex multistage optimization problems can be decomposed into simpler subproblems through recursive functional equations relating the value of a state to the immediate reward and the value of future states. Sutton and Barto (1998) [1] interpreted Bellman equations within the reinforcement learning framework as consistency relations for value functions and action-value functions, emphasizing their role in temporal-difference learning, policy evaluation, control, and the propagation of reward information through stochastic environments. Bertsekas (2012) [9] further developed the mathematical and computational interpretation of Bellman equations by viewing them as nonlinear fixed-point equations associated with dynamic programming operators, analyzing their contraction properties, convergence behavior, and applications to deterministic and stochastic optimal control problems. Together, these works established Bellman equations as the fundamental recursive structure underlying optimal decision-making, demonstrating how long-term optimal behavior can be characterized through local one-step optimization principles that form the theoretical foundation of dynamic programming and reinforcement learning.

For the optimal value function, this principle is expressed as

$$V^*(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds'|s,a) \right], \tag{400}$$

which states that the optimal value at state s is obtained by choosing an action that maximizes the sum of the immediate reward and the discounted optimal continuation value. This decomposition transforms a high-dimensional stochastic optimization problem—originally defined over entire trajectories

$$(s_0, a_0, s_1, a_1, \dots) \in \Omega$$

—into a recursive functional equation defined pointwise on the state space. In particular, instead of optimizing

$$\sup_{\pi \in \Pi} \mathbb{E}_\rho^\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right],$$

one solves the fixed-point equation

$$V^* = TV^*, \tag{401}$$

where T is the Bellman optimality operator. This reduction is the key to tractability in dynamic programming.

From a probabilistic viewpoint, the Bellman equations arise from conditioning arguments. For any policy π, using the decomposition

$$G_t = R_{t+1} + \gamma G_{t+1}, \tag{402}$$

and taking conditional expectation with respect to $s_t = s$, one obtains

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} + \gamma G_{t+1} \mid s_t = s]\,. \tag{403}$$

Applying the tower property of conditional expectation,

$$\mathbb{E}^\pi[G_{t+1} \mid s_t = s] = \mathbb{E}^\pi[\mathbb{E}^\pi[G_{t+1} \mid s_{t+1}] \mid s_t = s] = \mathbb{E}^\pi[V^\pi(s_{t+1}) \mid s_t = s]\,, \tag{404}$$

which yields the identity

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s]\,. \tag{405}$$

Using the Markov property, this can be expressed explicitly as

$$V^\pi(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\, P(ds'|s,a) \right] \pi(da|s). \tag{406}$$

Thus, the Bellman equation admits a dual interpretation:

- As a *fixed-point equation* $V^\pi = T^\pi V^\pi$ (or $V^* = TV^*$), emphasizing its functional-analytic structure,
- As a *conditional expectation identity*, emphasizing its probabilistic meaning.

From a dynamic perspective, the Bellman equation expresses a temporal consistency condition: the value assigned to a state must agree with the expected value obtained by taking one step forward and then evaluating the continuation. This consistency is captured by the recursion

$$V^\pi(s_t) = \mathbb{E}^\pi[R_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t], \tag{407}$$

which holds almost surely. Furthermore, the optimal Bellman equation embodies a local optimality condition:

$$V^*(s) = \sup_{a\in\mathcal{A}} Q^*(s,a), \quad \text{where} \quad Q^*(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} V^*(s')\, P(ds'|s,a). \tag{408}$$

This shows that optimal control can be understood as a sequence of greedy decisions with respect to the optimal continuation value.

The three figures (Figure 14a, 14b, and 14c) provide a concrete numerical realization of the analytical interpretation of the Bellman equations. The *first plot* (Figure 14a) illustrates the fixed-point nature of value iteration. Starting from an arbitrary initial function V_0, the sequence defined by

$$V_{k+1} = TV_k, \quad \text{where} \quad (TV)(s) = \max_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a), V(s') \right], \tag{409}$$

converges to the unique fixed point V^*. The trajectories in the plot show $V_k \to V^*$, thereby illustrating the contraction property and the convergence

$$V_k \xrightarrow[k\to\infty]{} V^*. \tag{410}$$

The *second plot* (Figure 14b) verifies the Bellman optimality equation as a consistency condition. It compares the left-hand side and right-hand side of

$$V^*(s) = \max_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a), V^*(s') \right], \tag{411}$$

demonstrating numerically that

$$V^* = TV^*. \tag{412}$$

The agreement of both sides across all states confirms that the computed function indeed satisfies the fixed-point equation.

The *third plot* (Figure 14c) illustrates the principle of optimality through the action-value function

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a), V^*(s'), \tag{413}$$

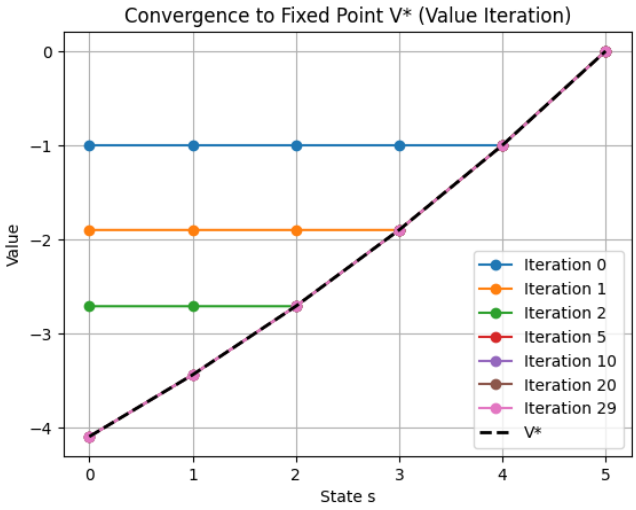
together with the relation

$$V^*(s) = \max_{a\in\mathcal{A}} Q^*(s,a). \tag{414}$$

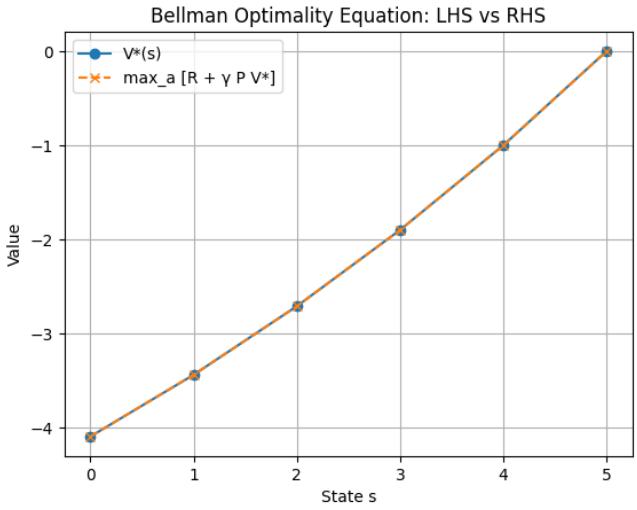
Each curve in the plot corresponds to a specific action a and represents the long-term consequence of that local decision, while the upper envelope corresponds to the optimal value function V^*. This makes explicit that global optimality is achieved by selecting, at each state, the action that maximizes the local one-step objective.

Taken together, the figures (Figure 14a, 14b, and 14c) illustrate the central conceptual reduction underlying dynamic programming:

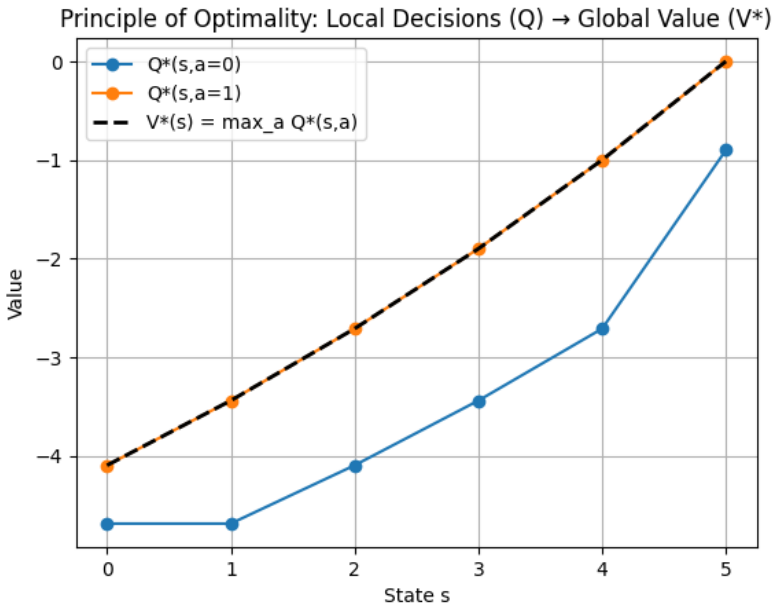
$$\sup_{\pi\in\Pi} \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t \right] \quad \Longleftrightarrow \quad V^* = TV^*, \tag{415}$$



(a) **Fixed-point convergence** $V_k \to V^*$



(b) **Bellman consistency** $V^* = TV^*$



(c) **Principle of optimality:** $V^*(s) = \max_a Q^*(s,a)$

Figure 14: **Visualization of the Bellman equations: convergence to the fixed point, verification of the Bellman optimality condition, and the principle of optimality via action-value functions.**

transforming a trajectory-level optimization problem into a pointwise functional equation. From a probabilistic perspective, this is rooted in the identity

$$V^\pi(s) = \mathbb{E}^\pi \left[R_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s \right], \tag{416}$$

which expresses the Bellman equation as a conditional expectation. Consequently, the value function satisfies a temporal consistency relation,

$$\text{value at } s = \text{immediate reward} + \text{discounted continuation value}, \tag{417}$$

providing the fundamental link between stochastic processes, recursion, and optimal control that is visualized across the three plots.

In summary, the Bellman equations unify probabilistic conditioning, recursive structure, and optimization. They reduce a complex stochastic control problem to a system of functional equations whose

solutions characterize optimal behavior, thereby forming the conceptual and mathematical foundation of reinforcement learning.

6 Operator-Theoretic Perspective

The Bellman equations admit a natural formulation in terms of nonlinear operators acting on function spaces. This operator-theoretic viewpoint provides a rigorous and unifying framework for analyzing existence, uniqueness, and convergence properties of value functions, and forms the mathematical foundation of dynamic programming methods.

Let $\mathcal{B}_b(\mathcal{S})$ denote the Banach space of bounded measurable functions $V : \mathcal{S} \to \mathbb{R}$ equipped with the supremum norm

$$\|V\|_\infty := \sup_{s\in\mathcal{S}} |V(s)|. \tag{418}$$

Define the Bellman optimality operator $T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$ by

$$(TV)(s) = \sup_{a\in\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right]. \tag{419}$$

In the finite state-action setting, this reduces to

$$(TV)(s) = \max_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V(s') \right]. \tag{420}$$

The operator T maps $\mathcal{B}_b(\mathcal{S})$ into itself. Indeed, if $V \in \mathcal{B}_b(\mathcal{S})$ and $|R(s,a)| \le M$, then

$$|(TV)(s)| \le \sup_{a\in\mathcal{A}} [|R(s,a)| + \gamma\|V\|_\infty] \le M + \gamma\|V\|_\infty, \tag{421}$$

so that TV is bounded. Measurability follows from the measurability of R, P, and standard arguments involving supremum over measurable functions.

Theorem 6.1 (Contraction Mapping)**.** *The Bellman operator T is a contraction on $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$, i.e.,*

$$\|TV - TW\|_\infty \le \gamma\|V - W\|_\infty, \quad \forall V, W \in \mathcal{B}_b(\mathcal{S}). \tag{422}$$

Proof. Let $V, W \in \mathcal{B}_b(\mathcal{S})$ and fix $s \in \mathcal{S}$. Then

$$|(TV)(s) - (TW)(s)| = \left| \sup_{a\in\mathcal{A}} \left[R(s,a) + \gamma \int V(s')P(ds'|s,a) \right] - \sup_{a\in\mathcal{A}} \left[R(s,a) + \gamma \int W(s')P(ds'|s,a) \right] \right|.$$

Using the general inequality

$$|\sup_x f(x) - \sup_x g(x)| \le \sup_x |f(x) - g(x)|, \tag{423}$$

we obtain

$$\begin{aligned} |(TV)(s) - (TW)(s)| &\le \sup_{a\in\mathcal{A}} \left| \gamma \int_{\mathcal{S}} \big(V(s') - W(s')\big) P(ds'|s,a) \right| \\ &\le \gamma \sup_{a\in\mathcal{A}} \int_{\mathcal{S}} |V(s') - W(s')| P(ds'|s,a). \end{aligned}$$

Since $P(\cdot|s,a)$ is a probability measure,

$$\int_{\mathcal{S}} |V(s') - W(s')| P(ds'|s,a) \le \|V - W\|_\infty, \tag{424}$$

and therefore

$$|(TV)(s) - (TW)(s)| \le \gamma\|V - W\|_\infty. \tag{425}$$

Taking the supremum over $s \in \mathcal{S}$ yields

$$\|TV - TW\|_\infty \le \gamma\|V - W\|_\infty. \tag{426}$$

□

Since $\gamma \in [0, 1)$, the operator T is a strict contraction. Therefore, by the Banach fixed-point theorem, there exists a unique function $V^* \in \mathcal{B}_b(\mathcal{S})$ such that

$$V^* = TV^*. \tag{427}$$

This fixed point coincides with the optimal value function.

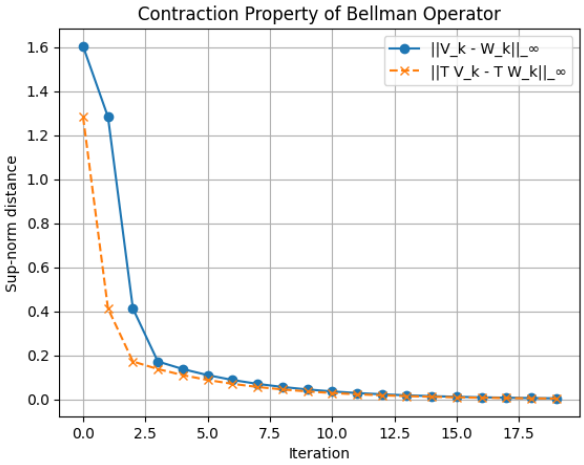
Moreover, for any initial function $V_0 \in \mathcal{B}_b(\mathcal{S})$, the sequence defined by successive application of T,

$$V_{k+1} = TV_k, \tag{428}$$

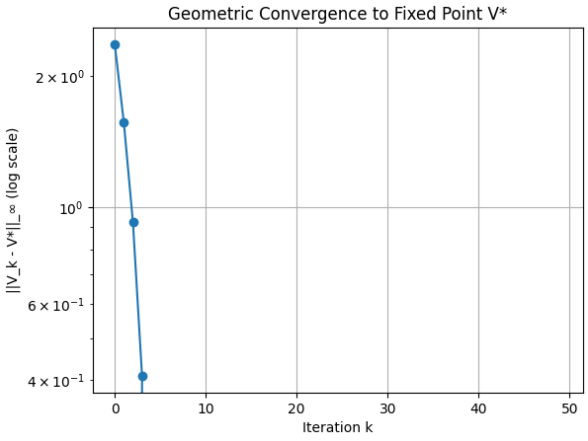
converges uniformly to V^*:

$$\|V_k - V^*\|_\infty \leq \gamma^k \|V_0 - V^*\|_\infty. \tag{429}$$

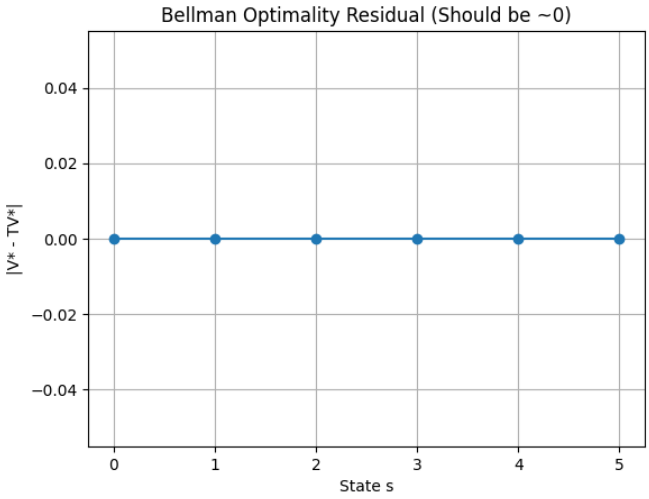
From an operator-theoretic standpoint, the reinforcement learning problem reduces to solving a nonlinear fixed-point equation in a Banach space. The contraction property ensures well-posedness and stability, while also providing the theoretical justification for iterative numerical schemes such as value iteration.



(a) **Contraction property of the Bellman operator**



(b) **Geometric convergence to the fixed point V^***



(c) **Bellman optimality residual**

Figure 15: **Operator-theoretic interpretation of the Bellman operator: contraction, convergence, and fixed-point verification.**

The three figures (Figure 15a and 15b, and 15c) provide a precise numerical illustration of the operator-theoretic structure of the Bellman optimality operator $T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$. The *first plot* (Figure 15a) demonstrates the contraction property

$$|TV - TW|_\infty \leq \gamma |V - W|_\infty, \tag{430}$$

by tracking the evolution of two sequences $\{V_k\}$ and $\{W_k\}$ under repeated application of T. Empirically, one observes

$$|TV_k - TW_k|_\infty < |V_k - W_k|_\infty, \tag{431}$$

indicating that the distance between iterates shrinks geometrically, in accordance with the contraction modulus $\gamma \in (0, 1)$.

The *second plot* (Figure 15b) illustrates convergence to the unique fixed point. Starting from an arbitrary initial function V_0, the iteration

$$V_{k+1} = TV_k \tag{432}$$

produces a sequence converging to V^*, i.e.,

$$V_k \xrightarrow[k\to\infty]{} V^*, \tag{433}$$

with the quantitative error bound

$$|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*|_\infty. \tag{434}$$

The approximately linear decay observed on the logarithmic scale confirms this geometric convergence rate predicted by the Banach fixed-point theorem.

The *third plot* (Figure 15c) verifies the fixed-point identity itself. By evaluating the residual

$$|V^*(s) - (TV^*)(s)|, \tag{435}$$

one observes that it is numerically negligible for all states $s \in \mathcal{S}$, thereby confirming that

$$V^* = TV^*. \tag{436}$$

Taken together, these figures (Figure 15a and 15b, and 15c) illustrate the core operator-theoretic principles underlying dynamic programming. The Bellman operator acts on the Banach space $\mathcal{B}_b(\mathcal{S})$, and its contraction property ensures, via the Banach fixed-point theorem, the existence and uniqueness of a solution V^* satisfying

$$\exists\, V^* \in \mathcal{B}_b(\mathcal{S}) \quad \text{such that} \quad V^* = TV^*. \tag{437}$$

Moreover, the iterative scheme

$$V_{k+1} = TV_k \tag{438}$$

is guaranteed to converge globally to V^* from any initial condition. Thus, the plots collectively provide a concrete visualization of how the abstract functional-analytic framework translates into stable and convergent computational procedures.

In summary, the Bellman operator encapsulates the dynamics, reward structure, and optimization criterion into a single functional object. Its contraction property is the key structural feature that guarantees the existence, uniqueness, and computability of optimal value functions.

6.1 Policy Evaluation Operator and Fixed Points

The mathematical analysis of policy evaluation operators and fixed-point formulations in reinforcement learning and dynamic programming is deeply connected to the foundations of functional analysis developed by Kreyszig (1991) [17] and the stochastic control frameworks established by Puterman (2014) [3] and Bertsekas and Tsitsiklis (1996) [2]. Kreyszig (1991) [17] provided the fundamental tools of functional analysis, including normed linear spaces, Banach spaces, bounded linear operators, contraction mappings, and fixed-point theorems, which form the mathematical basis for analyzing the convergence and stability properties of Bellman and policy evaluation operators. Puterman (2014) [3] applied these ideas within the theory of Markov decision processes by rigorously formulating policy evaluation as a fixed-point problem in spaces of value functions, establishing existence and uniqueness results through contraction properties of Bellman operators under discounted reward criteria. Bertsekas and Tsitsiklis

(1996) [2] further extended the operator-theoretic framework to approximate dynamic programming and neuro-dynamic programming, analyzing iterative policy evaluation schemes, asynchronous algorithms, function approximation methods, and convergence behavior of stochastic iterative procedures associated with value-function fixed points. Together, these works established the modern understanding of policy evaluation operators as contraction mappings on suitable function spaces whose fixed points correspond to value functions, thereby providing the rigorous analytical foundation for dynamic programming, reinforcement learning, and approximate stochastic control methods.

Fix a policy π and consider the Banach space $(\mathcal{B}_b(\mathcal{S}), \|\cdot\|_\infty)$ of bounded measurable functions. The Bellman operator associated with π is defined by

$$(T^\pi V)(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right] \pi(da|s), \quad s \in \mathcal{S}. \tag{439}$$

This operator encodes the one-step lookahead under the policy π, combining the immediate reward with the discounted expected continuation value.

The mapping T^π is well-defined on $\mathcal{B}_b(\mathcal{S})$. Indeed, if $V \in \mathcal{B}_b(\mathcal{S})$ and $|R(s,a)| \le M$, then

$$|(T^\pi V)(s)| \le \int_{\mathcal{A}} [|R(s,a)| + \gamma \|V\|_\infty]\, \pi(da|s) \le M + \gamma \|V\|_\infty, \tag{440}$$

so that $T^\pi V$ is bounded. Measurability follows from the measurability of R, P, and π, together with standard properties of integration with respect to stochastic kernels. The operator T^π is affine in V, in the sense that for any $V, W \in \mathcal{B}_b(\mathcal{S})$ and $\lambda \in [0,1]$,

$$T^\pi(\lambda V + (1-\lambda)W) = \lambda T^\pi V + (1-\lambda) T^\pi W, \tag{441}$$

since the dependence on V is linear through the integral term. Furthermore, T^π is monotone: if $V(s) \le W(s)$ for all $s \in \mathcal{S}$, then

$$(T^\pi V)(s) \le (T^\pi W)(s), \quad \forall s \in \mathcal{S}, \tag{442}$$

which follows from the positivity of the transition kernel.

A fundamental property is that T^π is a contraction mapping under the sup-norm. For any $V, W \in \mathcal{B}_b(\mathcal{S})$ and any $s \in \mathcal{S}$,

$$\begin{aligned} |(T^\pi V)(s) - (T^\pi W)(s)| &= \left| \int_{\mathcal{A}} \gamma \int_{\mathcal{S}} (V(s') - W(s')) P(ds'|s,a)\, \pi(da|s) \right| \\ &\le \gamma \int_{\mathcal{A}} \int_{\mathcal{S}} |V(s') - W(s')| P(ds'|s,a)\, \pi(da|s) \\ &\le \gamma \|V - W\|_\infty. \end{aligned}$$

Taking supremum over s yields

$$\|T^\pi V - T^\pi W\|_\infty \le \gamma \|V - W\|_\infty. \tag{443}$$

Since $\gamma \in [0,1)$, T^π is a strict contraction. Therefore, by the Banach fixed-point theorem, there exists a unique function $V^\pi \in \mathcal{B}_b(\mathcal{S})$ such that

$$V^\pi = T^\pi V^\pi. \tag{444}$$

This fixed point coincides with the value function associated with the policy π, i.e.,

$$V^\pi(s) = \mathbb{E}_s^\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{445}$$

Moreover, the contraction property implies that the fixed point can be computed via successive approximation. For any initial function $V_0 \in \mathcal{B}_b(\mathcal{S})$, define

$$V_{k+1} := T^\pi V_k. \tag{446}$$

Then, by induction,

$$\|V_k - V^\pi\|_\infty \leq \gamma^k \|V_0 - V^\pi\|_\infty, \tag{447}$$

which shows that V_k converges uniformly and geometrically fast to V^π. An alternative characterization of V^π can be obtained by iterating the operator:

$$V^\pi = \lim_{k\to\infty} (T^\pi)^k V_0, \tag{448}$$

for any $V_0 \in \mathcal{B}_b(\mathcal{S})$. Expanding $(T^\pi)^k$ reveals that V^π aggregates rewards along trajectories:

$$(T^\pi)^k V_0(s) = \mathbb{E}_s^\pi \left[\sum_{t=0}^{k-1} \gamma^t R(s_t, a_t) + \gamma^k V_0(s_k) \right], \tag{449}$$

and letting $k \to \infty$ yields the infinite-horizon value function.

Figures 16a, 16b, 16c, 16d, 16e and 16f illustrate the fundamental mathematical structure underlying policy evaluation and Bellman fixed-point theory in reinforcement learning. Figure 16a depicts the action of the Bellman operator

$$(T^\pi V)(s) = \int_{\mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a) \right] \pi(da|s), \tag{450}$$

which transforms a value function V into a new function obtained through one-step reward accumulation and discounted continuation. The figure illustrates how repeated application of T^π progressively reshapes an arbitrary initial value function toward the fixed-point solution corresponding to the policy π.

Figure 16b illustrates the contraction property of the Bellman operator. For any bounded measurable functions $V, W \in \mathcal{B}_b(\mathcal{S})$,

$$\|T^\pi V - T^\pi W\|_\infty \leq \gamma \|V - W\|_\infty, \tag{451}$$

where $\gamma \in [0, 1)$ is the discount factor. Since $\gamma < 1$, the distance between successive iterates decreases geometrically, as shown by the exponentially decaying trajectories in the figure. This contraction property is the key mechanism guaranteeing convergence and stability of policy evaluation.

Figure 16c demonstrates successive approximation toward the unique fixed point. Starting from an arbitrary initial function V_0, one defines the iterative sequence

$$V_{k+1} = T^\pi V_k. \tag{452}$$

By the Banach fixed-point theorem, there exists a unique value function V^π satisfying

$$V^\pi = T^\pi V^\pi. \tag{453}$$

Moreover, the convergence is geometric:

$$\|V_k - V^\pi\|_\infty \leq \gamma^k \|V_0 - V^\pi\|_\infty. \tag{454}$$

The figure illustrates how the iterates approach the fixed point exponentially fast as the number of iterations increases.

Figure 16d provides a geometric interpretation of the fixed-point equation. The fixed point is characterized by the intersection between the identity mapping

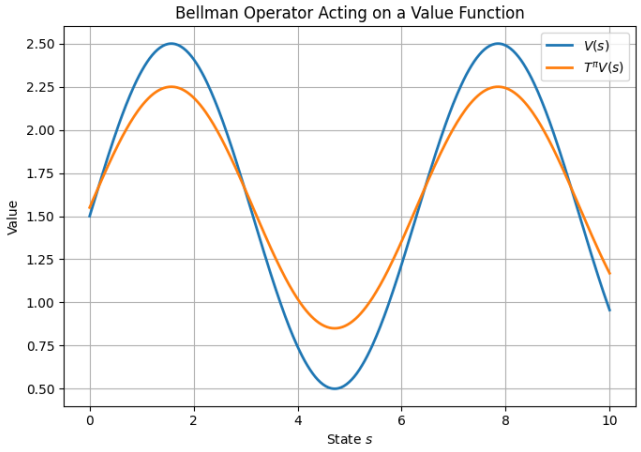
$$V \tag{455}$$

and the Bellman operator

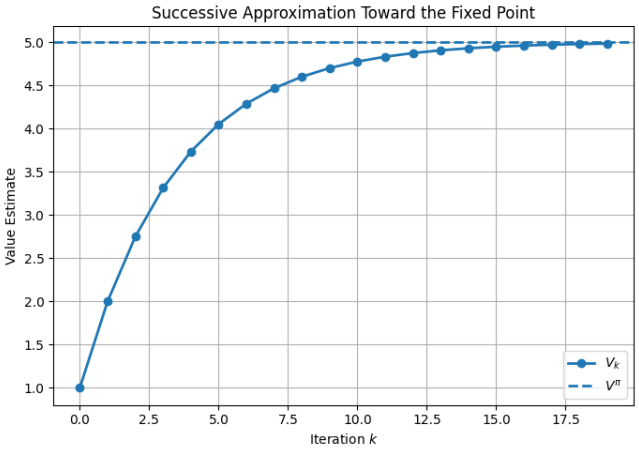
$$T^\pi V. \tag{456}$$

At the intersection,

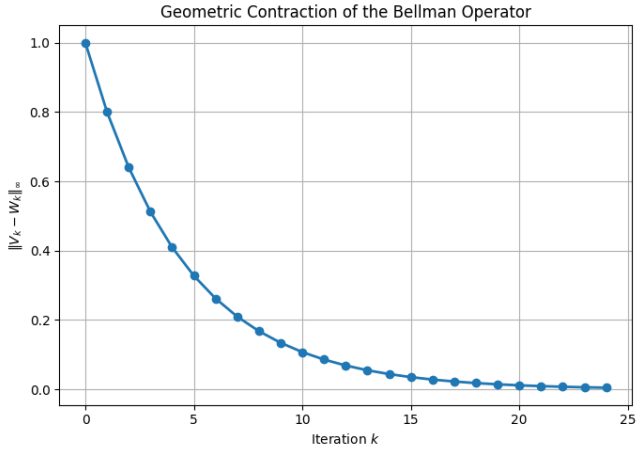
$$V^\pi = T^\pi V^\pi, \tag{457}$$



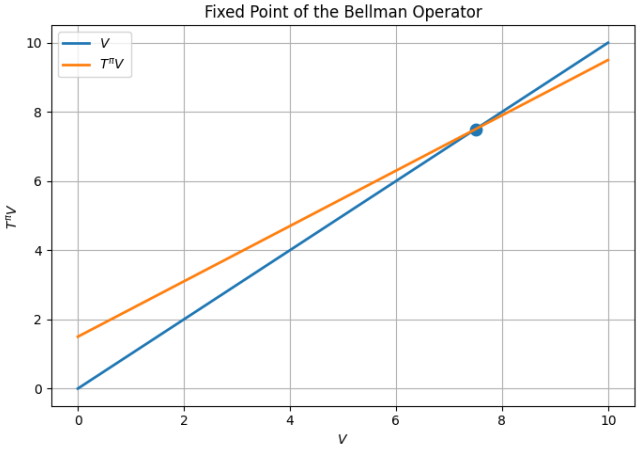
(a) **Bellman operator contraction mapping**



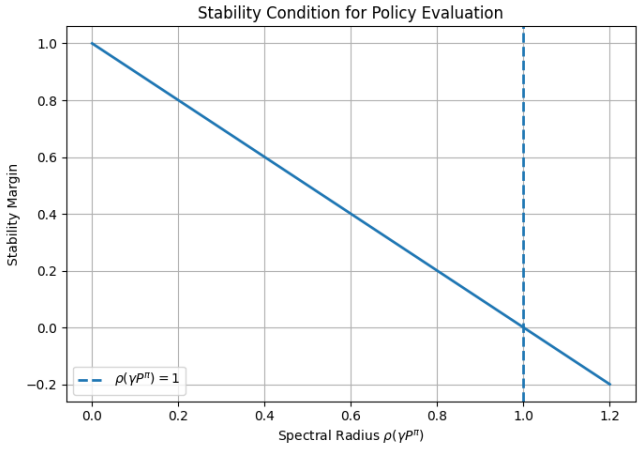
(b) **Successive approximation convergence**



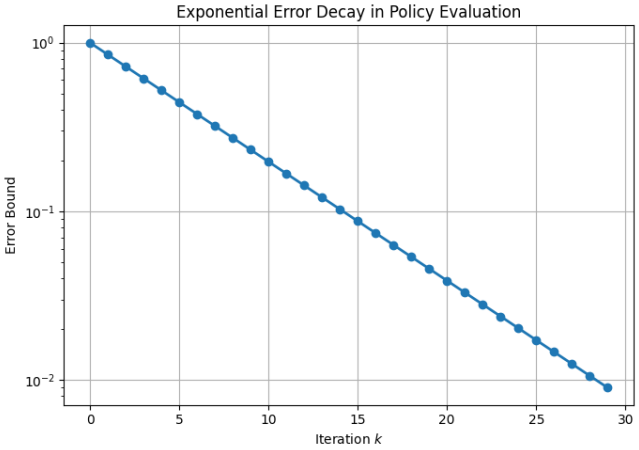
(c) **Geometric decay of evaluation error**



(d) **Fixed-point iteration trajectory**



(e) **Spectral radius and stability**



(f) **Matrix inverse solution structure**

Figure 16: **Illustrations of Bellman operators, contraction mappings, fixed-point convergence, geometric error decay, spectral stability, and linear algebraic formulations in policy evaluation.**

meaning that the value function remains invariant under Bellman updates. This graphical viewpoint illustrates how policy evaluation can be interpreted as solving a nonlinear operator equation in function space.

Figure 16e illustrates the exponential decay of the approximation error. Because the Bellman operator is a contraction, the evaluation error satisfies

$$\|V_k - V^\pi\|_\infty = \mathcal{O}(\gamma^k). \tag{458}$$

The semilogarithmic decay shown in the figure reflects the geometric convergence rate implied by the contraction coefficient γ. Smaller values of γ lead to faster convergence, whereas values close to 1 slow down the iterative process due to weaker contraction.

Figure 16f illustrates the linear algebraic formulation of policy evaluation in finite state-action spaces. Let P^π denote the transition matrix induced by the policy π, and let r^π denote the expected reward vector. The Bellman equation becomes

$$V^\pi = r^\pi + \gamma P^\pi V^\pi, \tag{459}$$

which may be rewritten as

$$(I - \gamma P^\pi)V^\pi = r^\pi. \tag{460}$$

Provided that the spectral radius satisfies

$$\rho(\gamma P^\pi) < 1, \tag{461}$$

the inverse exists and the unique solution is

$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi. \tag{462}$$

The figure illustrates how stability depends on the spectral properties of the discounted transition operator and highlights the connection between reinforcement learning and classical linear operator theory.

Collectively, Figures 16a, 16b, 16c, 16d, 16e and 16f visualize the core theoretical principles of policy evaluation, including Bellman operators, contraction mappings, geometric convergence, fixed-point structure, error decay, and matrix-based representations of value functions.

In the finite state-action setting, T^π admits a linear algebraic representation. Let P^π and r^π denote the transition matrix and reward vector induced by π. Then

$$V^\pi = r^\pi + \gamma P^\pi V^\pi, \tag{463}$$

which can be rewritten as

$$(I - \gamma P^\pi)V^\pi = r^\pi. \tag{464}$$

Since $\rho(\gamma P^\pi) < 1$, the inverse exists and

$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi. \tag{465}$$

In summary, the policy evaluation problem reduces to solving a linear fixed-point equation in a Banach space. The contraction property ensures existence, uniqueness, and stability of the solution, while also providing a constructive method for computation via successive approximations.

6.2 Monotonicity and Order Structure

The concepts of monotonicity and order structure play a fundamental role in the mathematical analysis of dynamic programming and stochastic control, and were systematically developed in the works of Hernández-Lerma and Lasserre (2012) [5], Puterman (2014) [3], and Bertsekas (2012) [9]. Hernández-Lerma and Lasserre (2012) [5] established a rigorous measure-theoretic framework for discrete-time

Markov control processes in which Bellman operators preserve partial orderings on spaces of measurable value functions, enabling the analysis of monotone convergence, optimality equations, and stability properties in general Borel state and action spaces. Puterman (2014) [3] further analyzed monotonicity properties of value iteration and policy iteration operators within Markov decision processes, demonstrating how order-preserving mappings and contraction structures guarantee convergence toward optimal value functions and stationary policies under discounted and average-cost criteria. Bertsekas (2012) [9] developed the operator-theoretic interpretation of dynamic programming by studying Bellman operators as monotone nonlinear mappings acting on ordered function spaces, emphasizing their roles in fixed-point theory, convergence analysis, and computational optimization methods for deterministic and stochastic control problems. Together, these works established the importance of monotonicity and order structures in ensuring stability, convergence, and optimality in dynamic programming and reinforcement learning, thereby providing a rigorous analytical framework for recursive stochastic optimization.

The space $\mathcal{B}_b(\mathcal{S})$ of bounded measurable functions admits a natural partial order defined pointwise: for $V, W \in \mathcal{B}_b(\mathcal{S})$, one writes

$$V \leq W \quad \Longleftrightarrow \quad V(s) \leq W(s),\ \forall s \in \mathcal{S}. \tag{466}$$

This endows $\mathcal{B}_b(\mathcal{S})$ with the structure of an ordered Banach space (indeed, a Banach lattice). The Bellman operators T and T^π are *monotone* (order-preserving) with respect to this partial order.

To verify this property, let $V, W \in \mathcal{B}_b(\mathcal{S})$ satisfy $V \leq W$. Then for every $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$\int_{\mathcal{S}} V(s')\, P(ds'|s, a) \ \leq \ \int_{\mathcal{S}} W(s')\, P(ds'|s, a), \tag{467}$$

since $P(\cdot|s, a)$ is a probability measure and $V(s') \leq W(s')$ pointwise. It follows that

$$R(s, a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s, a) \ \leq \ R(s, a) + \gamma \int_{\mathcal{S}} W(s')\, P(ds'|s, a). \tag{468}$$

Taking the supremum over $a \in \mathcal{A}$ yields

$$(TV)(s) \leq (TW)(s), \quad \forall s \in \mathcal{S}. \tag{469}$$

Similarly, integrating with respect to $\pi(da|s)$ gives

$$(T^\pi V)(s) \leq (T^\pi W)(s), \quad \forall s \in \mathcal{S}. \tag{470}$$

Thus, both T and T^π are monotone operators on $\mathcal{B}_b(\mathcal{S})$. This property has several important consequences.

First, the iterates generated by value iteration preserve order. If $V_0 \leq V_1$, then by induction,

$$TV_0 \leq TV_1, \quad T^2V_0 \leq T^2V_1, \quad \ldots, \tag{471}$$

and hence

$$T^kV_0 \leq T^kV_1, \quad \forall k \geq 0. \tag{472}$$

This shows that the mapping $V \mapsto T^kV$ is also monotone for every k.

Second, monotonicity enables the construction of monotone sequences that converge to the fixed point. For instance, if $V_0 \leq TV_0$, then the sequence $\{V_k\}$ defined by $V_{k+1} = TV_k$ satisfies

$$V_0 \leq V_1 \leq V_2 \leq \cdots, \tag{473}$$

and is non-decreasing. Since $\{V_k\}$ is also bounded above by V^* (by standard comparison arguments), it converges pointwise and, by contraction, also in the sup-norm to V^*. Similarly, if $V_0 \geq TV_0$, one obtains a non-increasing sequence converging to V^*.

Third, monotonicity yields a *comparison principle*. If $V \in \mathcal{B}_b(\mathcal{S})$ satisfies

$$V \leq TV, \tag{474}$$

then by iterating,

$$V \leq TV \leq T^2V \leq \cdots \leq V^*, \tag{475}$$

which implies

$$V \leq V^*. \tag{476}$$

Conversely, if $V \geq TV$, then

$$V \geq V^*. \tag{477}$$

Thus, subsolutions and supersolutions of the Bellman equation provide lower and upper bounds on the optimal value function. Fourth, monotonicity is closely tied to the positivity of the underlying transition operator. For a fixed policy π, define the linear operator

$$(P^\pi V)(s) := \int_{\mathcal{S}} V(s')\, P^\pi(ds'|s), \tag{478}$$

where

$$P^\pi(ds'|s) := \int_{\mathcal{A}} P(ds'|s,a)\, \pi(da|s). \tag{479}$$

Then P^π is a positive linear operator, meaning $V \geq 0$ implies $P^\pi V \geq 0$. The Bellman operator can be written as

$$T^\pi V = r^\pi + \gamma P^\pi V, \tag{480}$$

and its monotonicity follows directly from the positivity of P^π.

Finally, the order-preserving structure plays a crucial role in both theoretical analysis and algorithm design. It underlies convergence proofs, stability properties, and error bounds, and is particularly important in approximate dynamic programming, where monotonicity helps ensure that approximations remain well-behaved.

The three figures (Figure 17a, 17b, and 17c) provide a rigorous visualization of the order-theoretic structure underlying Bellman operators and complement the contraction-based analysis by revealing monotonicity and comparison principles.

In Plot 1 (Figure 17a), the inequality

$$V \leq W; \Longrightarrow; TV \leq TW \tag{481}$$

is verified pointwise across the state space. The curves show that if $V(s) \leq W(s)$ for all s, then after applying the Bellman operator the ordering is preserved. This property follows directly from the positivity of the transition operator:

$$(P^\pi V)(s) = \int_{\mathcal{S}} V(s'), P^\pi(ds'|s), \tag{482}$$

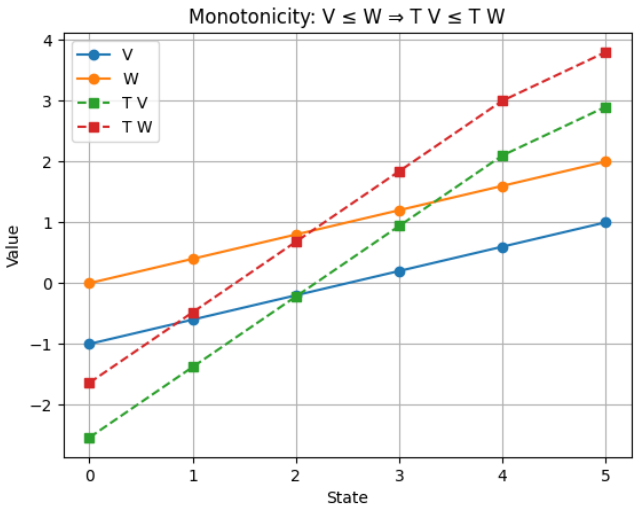
which implies that integration against a probability measure preserves inequalities. Hence, the Bellman operator

$$(TV)(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int V(s'), P(ds'|s,a) \right] \tag{483}$$

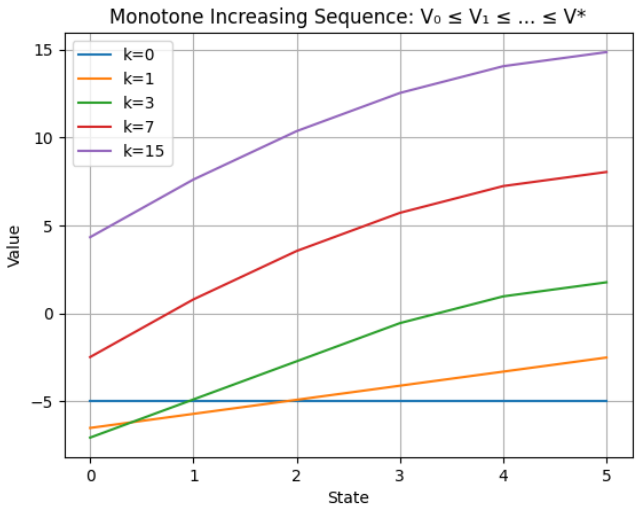
is order-preserving, reflecting its monotonicity on the partially ordered space.

In Plot 2 (Figure 17b), the sequence generated by value iteration,

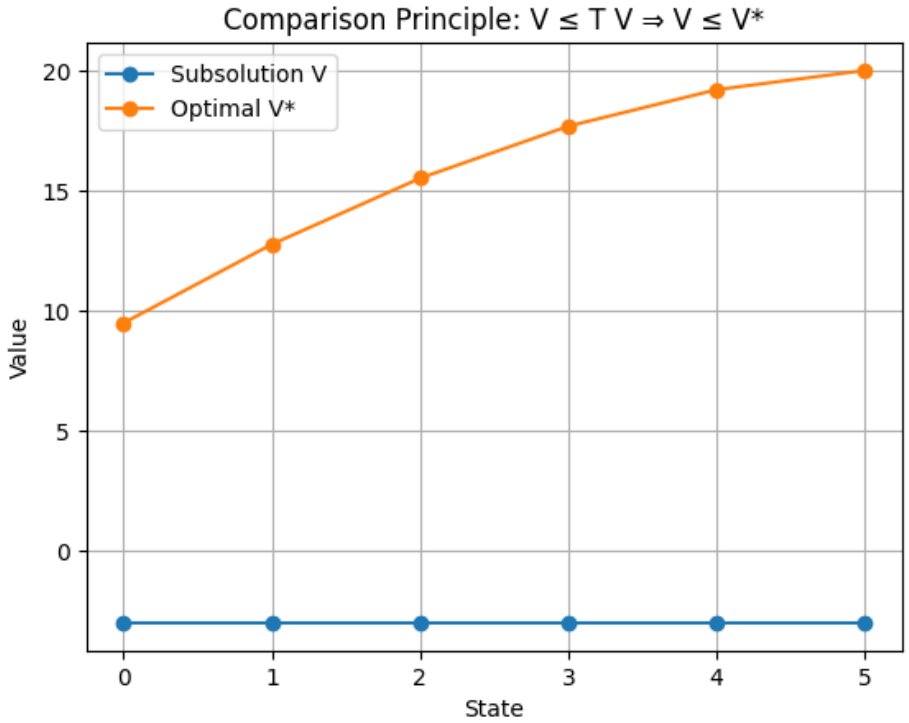
$$V_{k+1} = TV_k, \tag{484}$$



(a) **Order preservation of the Bellman operator: if $V \leq W$, then $TV \leq TW$, illustrating monotonicity.**



(b) **Monotone value iteration: starting from $V_0 \leq TV_0$, the sequence V_k satisfies $V_0 \leq V_1 \leq \cdots \leq V^*$ and converges increasingly to the fixed point.**



(c) **Comparison principle: any subsolution V satisfying $V \leq TV$ provides a lower bound $V \leq V^*$ on the optimal value function.**

Figure 17: **Monotonicity and Order Structure of Bellman Operators.**

is initialized from a subsolution satisfying

$$V_0 \leq TV_0. \tag{485}$$

The figure shows that the iterates form a monotone increasing sequence:

$$V_0 \leq V_1 \leq V_2 \leq \cdots \leq V^*, \tag{486}$$

which converges to the unique fixed point V^*. The strict upward movement and eventual stabilization of the curves provide a geometric realization of this lattice-theoretic construction. Unlike purely metric arguments, this demonstrates convergence through the order structure:

$$V_k \uparrow V^*. \tag{487}$$

In Plot 3 (Figure 17c), the comparison principle is illustrated. For any function V satisfying

$$V \leq TV, \tag{488}$$

iteration yields

$$V \leq TV \leq T^2V \leq \cdots \leq V^*, \tag{489}$$

and hence

$$V \leq V^*. \tag{490}$$

The plot shows such a subsolution lying entirely below the optimal value function. Similarly, if

$$V \geq TV, \tag{491}$$

then

$$V \geq V^*. \tag{492}$$

Thus, subsolutions and supersolutions provide lower and upper bounds on the fixed point.

Taken together, these figures (Figure 17a, 17b, and 17c) encode the deeper structure of the Bellman framework. The space $\mathcal{B}_b(\mathcal{S})$ is an ordered Banach space with partial order

$$V \leq W \iff V(s) \leq W(s), \quad \forall s, \tag{493}$$

and the Bellman operator

$$T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S}) \tag{494}$$

is monotone. The iteration

$$V_{k+1} = TV_k \tag{495}$$

therefore defines a monotone dynamical system whose trajectories converge to the fixed point from above or below depending on initialization. The comparison principle

$$V \leq TV \;\Rightarrow\; V \leq V^*, \qquad V \geq TV \;\Rightarrow\; V \geq V^* \tag{496}$$

provides a powerful tool for bounding solutions.

The key insight revealed by the plots is that convergence is not merely a consequence of contraction in the metric sense, but also of order-theoretic structure: contraction ensures uniqueness and convergence rate, while monotonicity provides directional structure, bounds, and stability. This dual perspective connects reinforcement learning with Banach lattice theory, positive operator theory, and monotone dynamical systems, and underlies many modern approaches to stability and approximation in stochastic control.

In summary, the monotonicity of Bellman operators reflects the fundamental positivity of the underlying stochastic dynamics and provides a powerful tool for analyzing convergence, bounding errors, and constructing iterative solution methods.

6.3 Supremum Norm vs Weighted Norms

The analysis of supremum norms and weighted norms has played a crucial role in understanding the convergence and approximation properties of dynamic programming and reinforcement learning algorithms. Puterman (2014) [3] established the classical framework for analyzing Bellman operators under the supremum norm, showing that discounted dynamic programming operators are contractions in the $|\cdot|_\infty$ norm, thereby guaranteeing the existence, uniqueness, and convergence of optimal value functions in Markov decision processes. Bertsekas and Tsitsiklis (1996) [2] extended these ideas to approximate dynamic programming and neuro-dynamic programming, demonstrating that in large-scale or unbounded stochastic systems the standard supremum norm may be insufficient, motivating the introduction of weighted norms that better capture state-dependent growth conditions and stability properties of approximate value iteration and policy evaluation schemes. Munos (2003) [18] further investigated these issues in the context of approximate policy iteration by deriving rigorous error bounds that depend on weighted norm structures and concentrability coefficients, thereby clarifying how approximation errors propagate through iterative reinforcement learning algorithms and how the choice of norm influences convergence guarantees and performance loss estimates. Together, these works established the theoretical importance of both supremum and weighted norms in analyzing Bellman operators, approximation error propagation, and the stability of reinforcement learning algorithms in high-dimensional and approximate settings.

The contraction property of the Bellman operator T under the supremum norm $\|\cdot\|_\infty$ relies crucially on the boundedness of the reward function and the compactness (or effective boundedness) of the state space. In more general settings—such as when $\mathcal{S}$ is non-compact or the reward function is unbounded—$\mathcal{B}_b(\mathcal{S})$ may no longer be an appropriate function space. In such cases, it is natural to introduce *weighted sup-norms* that control growth at infinity.

Let $w : \mathcal{S} \to [1, \infty)$ be a measurable weight function, and define the weighted norm

$$\|V\|_w := \sup_{s \in \mathcal{S}} \frac{|V(s)|}{w(s)}. \tag{497}$$

The associated function space is

$$\mathcal{B}_w(\mathcal{S}) := \left\{ V : \mathcal{S} \to \mathbb{R} \;\middle|\; \|V\|_w < \infty \right\}, \tag{498}$$

which is a Banach space under $\|\cdot\|_w$. Functions in $\mathcal{B}_w(\mathcal{S})$ are allowed to grow, but only at a rate controlled by w.

To ensure that the Bellman operator T is well-defined on $\mathcal{B}_w(\mathcal{S})$, one typically imposes a *drift condition* (or Lyapunov condition) on the transition kernel. Specifically, assume that there exist constants $\alpha \in [0, 1)$ and $b \geq 0$ such that

$$\int_{\mathcal{S}} w(s')\, P(ds'|s, a) \leq \alpha w(s) + b, \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A}. \tag{499}$$

This condition ensures that, on average, the process does not grow too rapidly relative to the weight function w. Assume further that the reward function satisfies a growth condition of the form

$$|R(s, a)| \leq c\, w(s), \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A}, \tag{500}$$

for some constant $c \geq 0$. Then for any $V \in \mathcal{B}_w(\mathcal{S})$, one has

$$\begin{aligned}
|(TV)(s)| &= \sup_{a \in \mathcal{A}} \left| R(s, a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s, a) \right| \\
&\leq \sup_{a \in \mathcal{A}} \left[|R(s, a)| + \gamma \int_{\mathcal{S}} |V(s')|\, P(ds'|s, a) \right] \\
&\leq c\, w(s) + \gamma \|V\|_w \int_{\mathcal{S}} w(s')\, P(ds'|s, a) \\
&\leq c\, w(s) + \gamma \|V\|_w (\alpha w(s) + b).
\end{aligned}$$

Dividing both sides by $w(s)$ yields

$$\frac{|(TV)(s)|}{w(s)} \leq c + \gamma\alpha\|V\|_w + \gamma b\|V\|_w \frac{1}{w(s)}. \tag{501}$$

Since $w(s) \geq 1$, it follows that

$$\|TV\|_w \leq c + \gamma(\alpha + b)\|V\|_w, \tag{502}$$

so that T maps $\mathcal{B}_w(\mathcal{S})$ into itself. To establish a contraction property, consider $V, W \in \mathcal{B}_w(\mathcal{S})$. Then

$$\begin{aligned}
\frac{|(TV)(s) - (TW)(s)|}{w(s)} &\leq \gamma \sup_{a\in\mathcal{A}} \frac{1}{w(s)} \int_{\mathcal{S}} |V(s') - W(s')|\, P(ds'|s,a) \\
&\leq \gamma\|V - W\|_w \sup_{a\in\mathcal{A}} \frac{1}{w(s)} \int_{\mathcal{S}} w(s')\, P(ds'|s,a) \\
&\leq \gamma\|V - W\|_w \left(\alpha + \frac{b}{w(s)}\right).
\end{aligned}$$

Taking the supremum over $s \in \mathcal{S}$, we obtain

$$\|TV - TW\|_w \leq \gamma(\alpha + b)\|V - W\|_w. \tag{503}$$

If $\beta := \gamma(\alpha + b) < 1$, then T is a contraction on $(\mathcal{B}_w(\mathcal{S}), \|\cdot\|_w)$:

$$\|TV - TW\|_w \leq \beta\|V - W\|_w. \tag{504}$$

Thus, under suitable drift and growth conditions, the contraction mapping framework extends naturally beyond bounded settings. The weighted norm compensates for possible growth in the value function, allowing one to recover existence, uniqueness, and convergence of solutions to the Bellman equation even when $\mathcal{S}$ is unbounded or rewards are not uniformly bounded.

The three figures (Figure 18a, 18b, and 18c) provide a precise numerical realization of the transition from the classical supremum norm framework to the weighted norm setting required for unbounded state spaces.

In Figure 1 (Figure 18a), the iterates generated by value iteration,

$$V_{k+1} = TV_k, \tag{505}$$

are plotted as functions of the state s. One observes that $V_k(s)$ grows with $|s|$, reflecting the fact that the reward and dynamics allow unbounded accumulation. Consequently, the supremum norm

$$|V_k|_\infty = \sup_{s\in\mathcal{S}} |V_k(s)| \tag{506}$$

fails to remain finite, illustrating the breakdown of the classical contraction framework. In fact, at the operator level, this corresponds to the instability

$$|TV|_\infty = \infty, \tag{507}$$

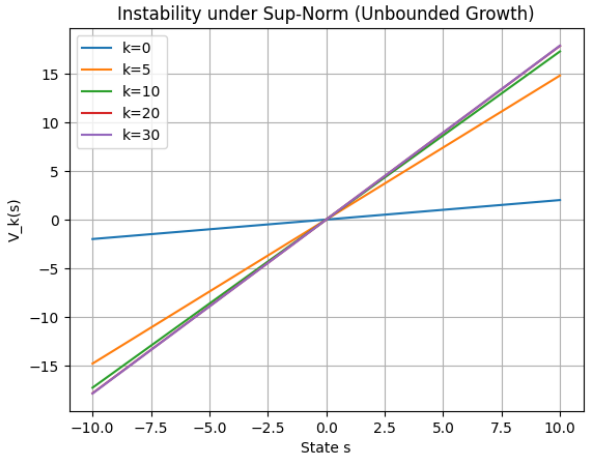
showing that T does not map bounded functions into bounded functions in such settings.

In Figure 2 (Figure 18b), the same iterates are normalized by a weight function $w(s)$, yielding the rescaled quantity

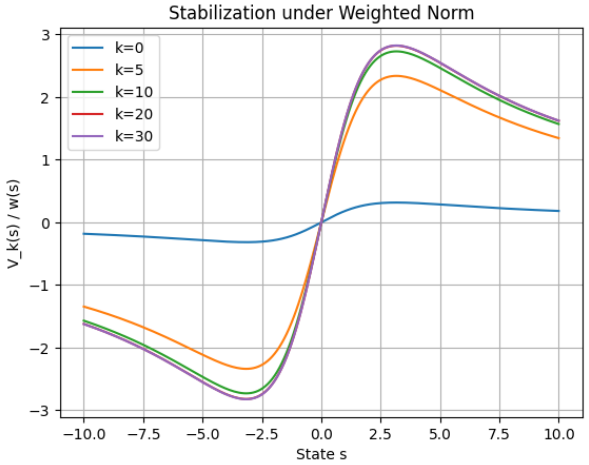
$$\frac{V_k(s)}{w(s)}. \tag{508}$$

The plots demonstrate that this ratio remains bounded, confirming that

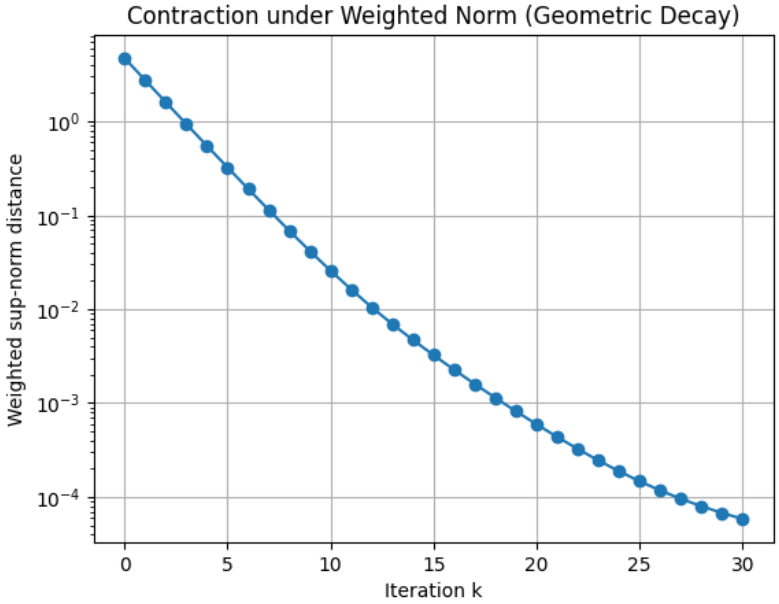
$$|V_k|_w = \sup_{s\in\mathcal{S}} \frac{|V_k(s)|}{w(s)} < \infty. \tag{509}$$



(a) **Breakdown of the supremum norm: the iterates $V_k(s)$ exhibit unbounded growth as $|s| \to \infty$, illustrating that $\|V_k\|_\infty$ may diverge in unbounded state spaces.**



(b) **Stabilization under the weighted norm $\|V\|_w = \sup_s \frac{|V(s)|}{w(s)}$: normalization by $w(s)$ controls growth and keeps the iterates bounded.**



(c) **Contraction in the weighted norm: the distance $\|V_k - W_k\|_w$ decays geometrically, verifying $\|TV - TW\|_w \leq \beta\|V - W\|_w$ with $\beta < 1$.**

Figure 18: **Supremum Norm vs Weighted Norms.**

Thus, although V_k itself may grow, its growth is controlled relative to w. This reflects the key role of the weight function as a Lyapunov function, compensating for the unboundedness of the state space and restoring well-posedness:

$$|TV|_w < \infty. \tag{510}$$

The mechanism underlying this stabilization is made explicit by the drift condition

$$\mathbb{E}[w(s_{t+1}) \mid s_t = s] \leq \alpha w(s) + b, \tag{511}$$

which ensures that, on average, the process does not grow faster than the weight function. This condition guarantees that the Bellman operator respects the weighted space.

In Figure 3 (Figure 18c), the contraction property is verified empirically in the weighted norm. Considering two sequences V_k and W_k, the plotted quantity corresponds to

$$|V_k - W_k|_w = \sup_{s \in \mathcal{S}} \frac{|V_k(s) - W_k(s)|}{w(s)}. \tag{512}$$

The log-scale linear decay demonstrates geometric convergence, confirming the inequality

$$|TV - TW|_w \leq \beta |V - W|_w, \tag{513}$$

where

$$\beta = \gamma(\alpha + b) < 1. \tag{514}$$

Thus, the Bellman operator is a contraction on $(\mathcal{B}_w(\mathcal{S}), |\cdot|_w)$, ensuring convergence of value iteration even in unbounded domains.

Taken together, the figures (Figure 18a, 18b, and 18c) illustrate the fundamental dichotomy:

$$|TV|_\infty = \infty \quad \text{(failure)}, \qquad |TV|_w < \infty \quad \text{(restored stability)}. \tag{515}$$

This demonstrates that the supremum norm is too rigid to accommodate growth, whereas the weighted norm adapts to the geometry of the state space.

From a deeper perspective, these visualizations encode several foundational ideas simultaneously. The weight function acts as a Lyapunov function, enforcing stability of the underlying Markov process. The drift condition ensures controlled growth, linking probabilistic dynamics with functional bounds. The contraction property in the weighted norm reflects a generalized Banach fixed-point theorem in non-compact spaces. Altogether, this framework provides the mathematical foundation for analyzing reinforcement learning and stochastic control problems in continuous or high-dimensional settings, where classical boundedness assumptions no longer hold.

This framework is fundamental in the study of controlled Markov processes on general state spaces, where Lyapunov functions (weights) play a central role in ensuring stability and ergodicity. It also provides the mathematical foundation for analyzing reinforcement learning algorithms in continuous or high-dimensional environments where classical boundedness assumptions fail.

6.4 Spectral Interpretation in Finite Spaces

The spectral interpretation of dynamic programming and reinforcement learning operators in finite-dimensional spaces is deeply connected to the theory of matrices and linear operators developed by Meyer (2023) [19], Horn and Johnson (2012) [20], and Bertsekas and Tsitsiklis (1996) [2]. Meyerr (2023) [19] provided a comprehensive treatment of matrix analysis and applied linear algebra, including eigenvalues, eigenvectors, spectral radius, matrix norms, stochastic matrices, and iterative methods, which are fundamental for analyzing the convergence behavior of Bellman operators and Markov transition matrices in finite-state systems. Horn and Johnson (2012) [20] further developed the mathematical foundations of spectral theory by rigorously studying matrix decompositions, Perron–Frobenius theory, nonnegative matrices, operator norms, and spectral properties of linear transformations, thereby providing the analytical

tools required for understanding contraction behavior, stability, and asymptotic dynamics in stochastic control and reinforcement learning. Bertsekas and Tsitsiklis (1996) [2] applied these operator-theoretic and spectral concepts to dynamic programming and neuro-dynamic programming, analyzing the spectral structure of transition operators, convergence of iterative value-function algorithms, asynchronous computations, and approximate dynamic programming methods in finite-state Markov decision processes. Together, these works established the spectral viewpoint of reinforcement learning and dynamic programming, showing how eigenstructure, spectral radii, and operator norms govern the convergence, stability, and approximation behavior of recursive stochastic optimization algorithms in finite-dimensional settings.

In the finite state-action setting, the Bellman equations admit a precise linear algebraic formulation, allowing tools from matrix analysis and spectral theory to be applied. Let $|\mathcal{S}| = n$ and $|\mathcal{A}| < \infty$. For a fixed policy π, define the transition matrix $P^\pi \in \mathbb{R}^{n\times n}$ and reward vector $r^\pi \in \mathbb{R}^n$ by

$$P^\pi(s,s') := \sum_{a\in\mathcal{A}} \pi(a|s)\, P(s'|s,a), \quad r^\pi(s) := \sum_{a\in\mathcal{A}} \pi(a|s)\, R(s,a). \tag{516}$$

Then the Bellman expectation equation can be written in vector form as

$$V^\pi = r^\pi + \gamma P^\pi V^\pi, \tag{517}$$

where $V^\pi \in \mathbb{R}^n$ is the value vector. Rearranging terms yields a linear system:

$$(I - \gamma P^\pi)V^\pi = r^\pi. \tag{518}$$

Thus, policy evaluation reduces to solving a system of linear equations. The matrix P^π is row-stochastic, i.e.,

$$P^\pi(s,s') \geq 0, \quad \sum_{s'} P^\pi(s,s') = 1, \tag{519}$$

which implies that its spectral radius satisfies

$$\rho(P^\pi) = 1. \tag{520}$$

Consequently,

$$\rho(\gamma P^\pi) = \gamma < 1, \tag{521}$$

since $\gamma \in [0,1)$.

This spectral property ensures that the matrix $(I - \gamma P^\pi)$ is invertible. Indeed, using the Neumann series for matrices,

$$(I - \gamma P^\pi)^{-1} = \sum_{k=0}^{\infty} (\gamma P^\pi)^k, \tag{522}$$

which converges because $\rho(\gamma P^\pi) < 1$. Therefore, the value function admits the explicit representation

$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi = \sum_{k=0}^{\infty} (\gamma P^\pi)^k r^\pi. \tag{523}$$

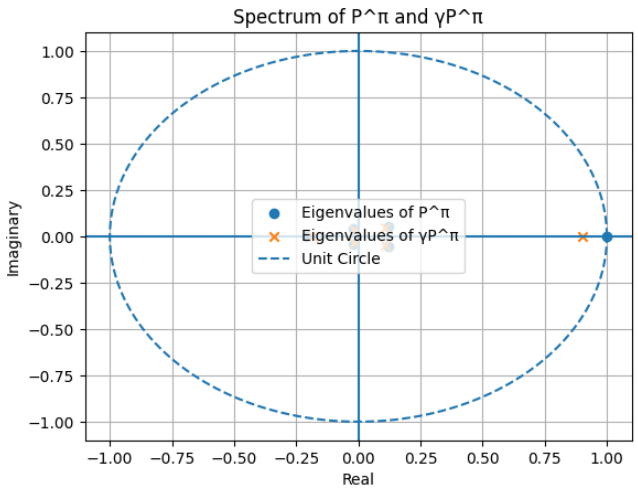
This series expansion has a natural probabilistic interpretation. The k-th term $(P^\pi)^k r^\pi$ represents the expected reward received at time k, and thus

$$V^\pi = \sum_{k=0}^{\infty} \gamma^k (P^\pi)^k r^\pi \tag{524}$$

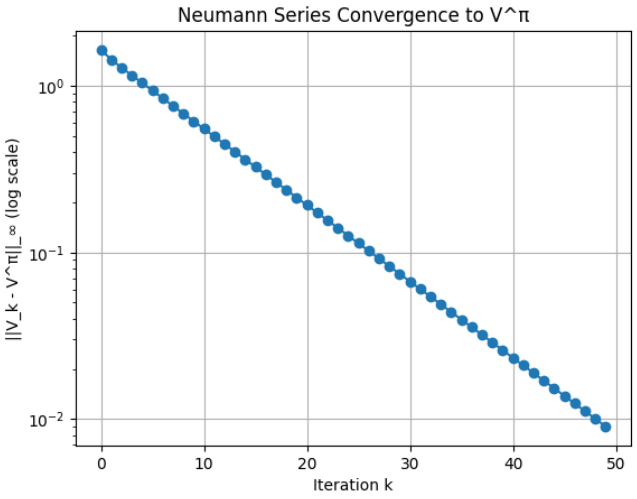
corresponds to the total discounted expected reward accumulated along trajectories governed by π.

From an operator-theoretic perspective, P^π acts as a linear operator on $\mathbb{R}^n$, and the Bellman equation corresponds to a resolvent equation of the form

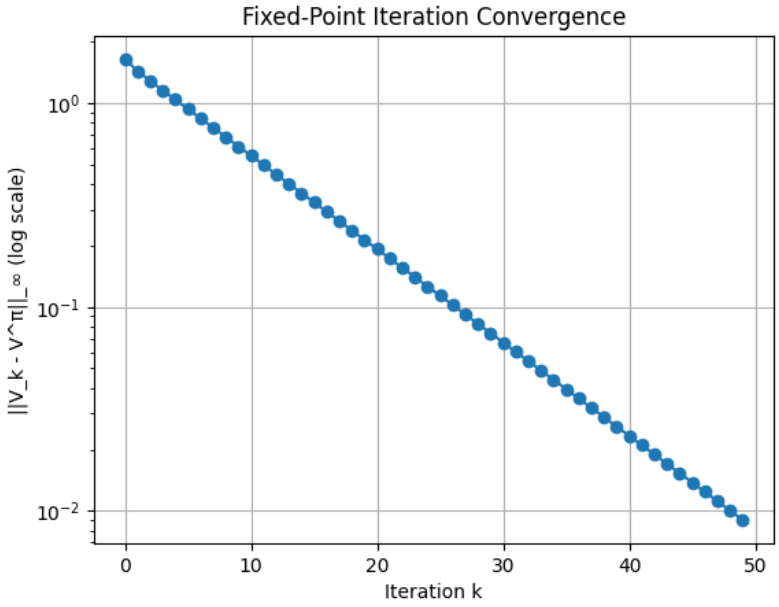
$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi. \tag{525}$$



(a) **Spectrum of P^π and γP^π, illustrating $\rho(P^\pi) = 1$ and $\rho(\gamma P^\pi) = \gamma < 1$, which ensures the invertibility of $(I - \gamma P^\pi)$.**



(b) **Geometric convergence of the truncated Neumann series $V_K = \sum_{k=0}^{K} (\gamma P^\pi)^k r^\pi$ toward V^π, consistent with $\rho(\gamma P^\pi) < 1$.**



(c) **Convergence of the fixed-point iteration $V_{k+1} = r^\pi + \gamma P^\pi V_k$ to the unique solution $V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$, with geometric error decay**

Figure 19: **Spectral and Operator-Theoretic Structure of Policy Evaluation.**

The operator $(I - \gamma P^\pi)^{-1}$ is known as the *resolvent* of P^π at γ^{-1}, and plays a central role in the analysis of Markov chains and dynamic systems. Furthermore, the contraction property of the Bellman operator is reflected in the spectral properties of γP^π. Specifically, for any vector norm compatible with the matrix norm, one has

$$\|\gamma P^\pi\| < 1, \tag{526}$$

which guarantees stability of the fixed-point iteration

$$V_{k+1} = r^\pi + \gamma P^\pi V_k. \tag{527}$$

The first plot (Figure 19a) illustrates the spectral contraction induced by γP^π, ensuring stability via $\rho(\gamma P^\pi) < 1$, which ensures the invertibility of $(I - \gamma P^\pi)$. The second plot (Figure 19b) verifies Geometric

convergence of the Neumann series representation

$$V^\pi = \sum_{k=0}^{\infty} (\gamma P^\pi)^k r^\pi \tag{528}$$

toward V^π, consistent with $\rho(\gamma P^\pi) < 1$. The third plot (Figure 19c) demonstrates convergence of the fixed-point iteration

$$V_{k+1} = r^\pi + \gamma P^\pi V_k \tag{529}$$

to the unique solution

$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi \tag{530}$$

with geometric error decay. Together, these visualizations (Figure 19a, 19b, and 19c) unify the linear algebraic, spectral, operator theoretic, and probabilistic interpretations of the Bellman equation.

In summary, the finite-dimensional setting reveals that policy evaluation is fundamentally a linear algebra problem, where the value function is obtained by solving a linear system or equivalently by summing a convergent operator series. This connection provides deep insight into the structure of reinforcement learning and enables the use of efficient numerical linear algebra techniques for computation.

6.5 Nonlinear Operator and Fixed-Point Geometry

The study of nonlinear operators and fixed-point geometry forms a central mathematical foundation for dynamic programming and reinforcement learning, and has been significantly developed in the works of Ortega and Rheinboldt (2000) [21], Bertsekas (2012) [9], Puterman (2014) [3], and Ghosh (2025) [22]. Ortega and Rheinboldt (2000) [21] provided a rigorous treatment of iterative methods for nonlinear equations, developing the theory of fixed-point iterations, contraction mappings, convergence analysis, stability, and geometric properties of nonlinear operator equations in normed spaces. Puterman (2014) [3] applied these ideas within the framework of Markov decision processes by formulating Bellman optimality equations as nonlinear fixed-point problems on spaces of value functions, establishing existence, uniqueness, and convergence results for iterative methods such as value iteration and policy iteration under discounted stochastic control settings. Bertsekas (2012) [9] further developed the operator-theoretic interpretation of dynamic programming by analyzing Bellman operators as monotone nonlinear mappings whose fixed points characterize optimal cost-to-go functions, emphasizing the geometric structure of recursive optimization problems, contraction behavior in suitable norms, and convergence properties of exact and approximate dynamic programming algorithms. The operator-theoretic interpretation of deep neural networks and nonlinear optimization landscapes has also been explored in Ghosh (2025) [22], where mathematical structures underlying nonlinear mappings, iterative learning dynamics, and convergence behavior in deep learning systems are studied from a rigorous analytical perspective. Together, these works established a rigorous nonlinear fixed-point framework for understanding dynamic programming and reinforcement learning, demonstrating how optimal sequential decision-making problems can be interpreted geometrically through iterative convergence toward fixed points of nonlinear Bellman operators.

The optimal Bellman operator $T : \mathcal{B}_b(\mathcal{S}) \to \mathcal{B}_b(\mathcal{S})$ is inherently nonlinear due to the maximization over actions. It can be written as

$$(TV)(s) = \sup_{a \in \mathcal{A}} F_a(V)(s), \quad \text{where} \quad F_a(V)(s) := R(s,a) + \gamma \int_{\mathcal{S}} V(s')\, P(ds'|s,a). \tag{531}$$

For each fixed $a \in \mathcal{A}$, the mapping F_a is an affine (indeed, linear up to a constant shift) operator on $\mathcal{B}_b(\mathcal{S})$. Thus, T is the pointwise supremum of a family of affine operators $\{F_a\}_{a \in \mathcal{A}}$.

This representation immediately implies several structural properties. First, T is monotone, since each F_a is monotone and the supremum preserves order. Second, T is *convex* (sublinear up to translation): for any $V, W \in \mathcal{B}_b(\mathcal{S})$ and $\lambda \in [0,1]$,

$$T(\lambda V + (1-\lambda)W)(s) = \sup_{a \in \mathcal{A}} F_a(\lambda V + (1-\lambda)W)(s) = \sup_{a \in \mathcal{A}} \big[\lambda F_a(V)(s) + (1-\lambda) F_a(W)(s)\big] \le \lambda (TV)(s) + (1-\lambda)(TW)(s), \tag{532}$$

which yields

$$T(\lambda V + (1-\lambda)W) \leq \lambda TV + (1-\lambda)TW. \tag{533}$$

Thus, T is convex but not linear, reflecting the underlying optimization structure.

From a geometric viewpoint, for each fixed $s \in \mathcal{S}$, the map $V \mapsto (TV)(s)$ is the upper envelope of the family $\{F_a(V)(s)\}_{a\in\mathcal{A}}$. Consequently, the graph of T can be interpreted as the pointwise supremum of affine functionals, analogous to support functions in convex analysis. This reveals a close connection between Bellman operators and convex duality. The fixed-point equation

$$V^* = TV^* \tag{534}$$

can therefore be interpreted geometrically as a self-consistency condition: V^* lies on the upper envelope generated by the family $\{F_a\}$, and at each state s, there exists at least one action $a^*(s)$ such that

$$V^*(s) = F_{a^*(s)}(V^*)(s). \tag{535}$$

Equivalently,

$$a^*(s) \in \operatorname{argmax}_{a\in\mathcal{A}} F_a(V^*)(s). \tag{536}$$

Thus, V^* is supported pointwise by one of the affine components, and the optimal policy selects the supporting action at each state. This perspective leads naturally to a variational inequality formulation. Define the residual

$$\mathcal{R}(V)(s) := (TV)(s) - V(s). \tag{537}$$

Then V^* is characterized by the condition

$$\mathcal{R}(V^*)(s) = 0, \quad \forall s \in \mathcal{S}, \tag{538}$$

while for any V,

$$V \leq TV \quad \Longleftrightarrow \quad \mathcal{R}(V) \geq 0, \tag{539}$$

and

$$V \geq TV \quad \Longleftrightarrow \quad \mathcal{R}(V) \leq 0. \tag{540}$$

Hence, V^* can be viewed as the unique function that simultaneously satisfies upper and lower envelope conditions, analogous to a saddle point or equilibrium in variational analysis.

Furthermore, the nonlinearity of T is closely related to the fact that optimal control problems involve selection among competing linear dynamics. Each operator F_a corresponds to fixing an action, while T performs a pointwise maximization over these choices. Thus, the Bellman optimality equation can be interpreted as a nonlinear eigenvalue problem in which the eigenfunction V^* is selected as the maximal fixed point of a family of affine maps.

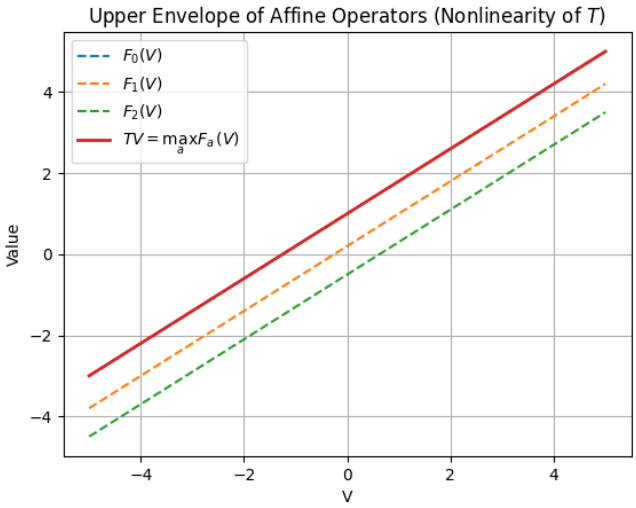
The three figures (Figure 20a, 20b, and 20c) provide a concrete geometric realization of the nonlinear Bellman operator and its fixed-point structure. As shown in Figure 1 (Figure 20a), the operator

$$(TV)(s) = \sup_{a\in\mathcal{A}} F_a(V)(s), \quad \text{with} \quad F_a(V)(s) = r_a + \gamma V, \tag{541}$$

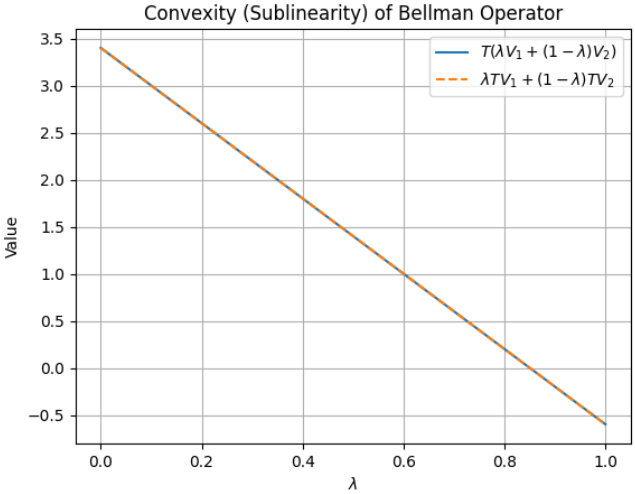
is obtained as the pointwise supremum of a family of affine mappings. Each F_a appears as a straight line (affine functional in V), while T emerges as their upper envelope. This visualization makes the nonlinearity explicit: although each component is affine, the maximization induces kinks where the maximizing action changes. These kinks correspond precisely to switching regions in which the optimal action changes, and the envelope interpretation shows that T behaves like a support function in convex analysis, with each F_a acting as a supporting hyperplane.

In Figure 2 (Figure 20b), the convex (sublinear) structure of the Bellman operator is verified numerically through the inequality

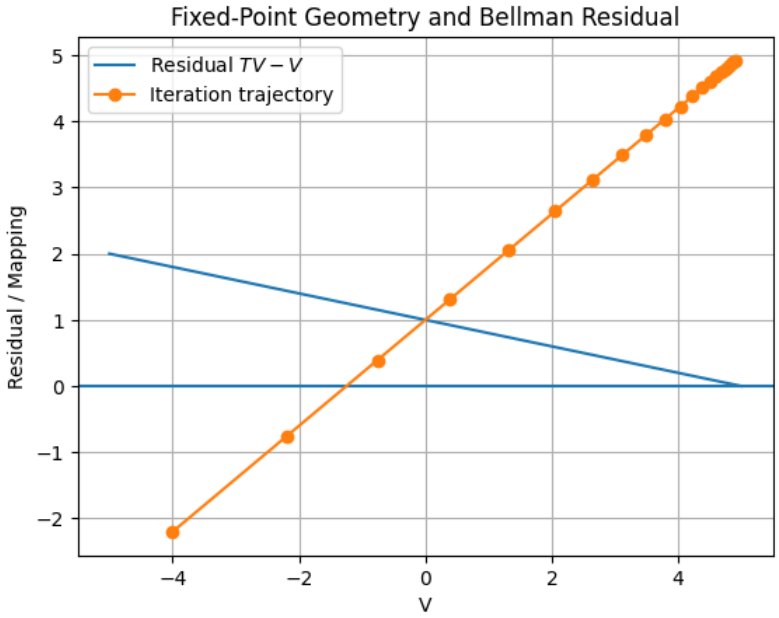
$$T(\lambda V + (1-\lambda)W) \leq \lambda, TV + (1-\lambda), TW, \quad \lambda \in [0,1]. \tag{542}$$



(a) **Upper envelope representation** $TV = \sup_a F_a(V)$**: illustrating the nonlinearity of the Bellman operator as the pointwise maximum of affine maps.**



(b) **Convexity (sublinearity) of** T **verifying** $T(\lambda V + (1-\lambda)W) \leq \lambda TV + (1-\lambda)TW$**.**



(c) **Fixed-point geometry of the Bellman operator: convergence of** $V_{k+1} = TV_k$ **to** V^***, characterized by the zero residual condition** $TV^* - V^* = 0$**.**

Figure 20: **Nonlinear Operator Geometry of the Bellman Mapping:** $T = \sup_a F_a$ **defines a convex, monotone contraction. The plots illustrate envelope structure, sublinearity, and fixed-point characterization** $V^* = TV^*$**.**

The solid curve represents the left-hand side, while the dashed curve corresponds to the right-hand side. The visible gap between them reflects the strict convexity induced by the supremum operation. This confirms that T is not linear but convex and monotone, a direct consequence of being the supremum of affine operators.

Figure 3 (Figure 20c) illustrates the fixed-point geometry of the Bellman operator. The optimal value function V^* is characterized by

$$V^* = TV^*, \qquad \mathcal{R}(V) := TV - V, \tag{543}$$

so that V^* is the unique solution of the variational condition

$$\mathcal{R}(V^*) = 0. \tag{544}$$

Graphically, this corresponds to the intersection between the curve TV and the identity line V. The iterative scheme

$$V_{k+1} = TV_k \tag{545}$$

generates a sequence that converges toward this intersection point, as seen in the trajectory plotted in the figure. The zero-crossing of the residual $TV - V$ marks the fixed point, confirming the Banach fixed-point structure in this nonlinear setting.

Taken together, the figures (Figure 20a, 20b, and 20c) provide a unified geometric interpretation of the Bellman operator: it is constructed from an affine family

$$F_a(V) = r_a + \gamma V, \tag{546}$$

combined through a nonlinear supremum

$$T = \sup_{a \in \mathcal{A}} F_a. \tag{547}$$

The resulting operator defines an upper-envelope geometry, where the optimal value V^* is the point at which this envelope becomes self-consistent. At this point, there exists an action $a^*(s)$ such that

$$V^*(s) = F_{a^*(s)}(V^*)(s), \quad \text{with} \quad a^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} F_a(V^*)(s), \tag{548}$$

revealing how the optimal policy emerges as the supporting action at the fixed point.

This visualization is particularly powerful because it simultaneously encodes multiple deep structures: the envelope construction from convex analysis Figure 20a), the nonlinear contraction from operator theory (Figure 20b), the residual condition from variational inequalities (Figure 20c), and the action selection mechanism underlying Bellman optimality.

In summary, the Bellman operator combines linear expectation (via F_a) with nonlinear optimization (via the supremum), yielding a convex, monotone, and contractive operator. Its fixed point V^* can be understood geometrically as the intersection of upper envelopes of affine operators, linking reinforcement learning with convex analysis, variational inequalities, and nonlinear operator theory.

6.6 Residuals and Error Bounds

The analysis of residuals and error bounds is fundamental to understanding the stability and performance of approximate dynamic programming and reinforcement learning algorithms, and has been extensively developed by Munos (2003) [18], Bertsekas and Tsitsiklis (1996) [2], Scherrer (2014) [23] and Ghosh (2025) [22]. Bertsekas and Tsitsiklis (1996) [2] established the foundational framework for studying approximation errors in neuro-dynamic programming, analyzing how Bellman residuals, stochastic approximation errors, and function approximation inaccuracies affect the convergence and stability of iterative value-function algorithms in large-scale Markov decision processes. Munos (2003) [18] further advanced this theory by deriving rigorous error bounds for approximate policy iteration, showing how

approximation and projection errors propagate through repeated Bellman updates and how concentrability properties and weighted norm structures influence the performance loss of approximate reinforcement learning methods. Scherrer (2014) [23] extended these ideas by establishing refined performance bounds for approximate policy iteration algorithms, developing tighter analytical estimates relating Bellman residuals, approximation quality, and policy suboptimality, thereby clarifying the theoretical relationship between local approximation errors and global decision-making performance in reinforcement learning systems. Issues related to approximation residuals, convergence accuracy, and error propagation in deep neural architectures are further discussed in Ghosh (2025) [22], emphasizing the mathematical foundations of learning errors and stability in high-dimensional optimization problems. Together, these works established the modern theoretical framework for residual analysis and error propagation in approximate dynamic programming, providing rigorous guarantees for the convergence, robustness, and performance of reinforcement learning algorithms under imperfect value-function approximation.

In practical settings, one often works with an approximate value function $\tilde{V}$ rather than the exact fixed point V^*. A natural way to quantify the quality of $\tilde{V}$ is through the *Bellman residual*, defined by

$$\mathcal{R}(\tilde{V}) := \|T\tilde{V} - \tilde{V}\|_\infty. \tag{549}$$

This quantity measures the extent to which $\tilde{V}$ fails to satisfy the Bellman optimality equation. In particular, $\mathcal{R}(\tilde{V}) = 0$ if and only if $\tilde{V} = V^*$, since V^* is the unique fixed point of T.

The residual provides a direct bound on the approximation error. Specifically, one has

$$\|\tilde{V} - V^*\|_\infty \leq \frac{1}{1-\gamma}\mathcal{R}(\tilde{V}). \tag{550}$$

To derive this inequality, observe that

$$\begin{aligned}\|\tilde{V} - V^*\|_\infty &= \|\tilde{V} - T\tilde{V} + T\tilde{V} - TV^*\|_\infty \\ &\leq \|\tilde{V} - T\tilde{V}\|_\infty + \|T\tilde{V} - TV^*\|_\infty \\ &\leq \mathcal{R}(\tilde{V}) + \gamma\|\tilde{V} - V^*\|_\infty,\end{aligned}$$

where the last inequality uses the contraction property of T. Rearranging terms yields

$$(1-\gamma)\|\tilde{V} - V^*\|_\infty \leq \mathcal{R}(\tilde{V}), \tag{551}$$

and hence the desired bound. This inequality shows that a small Bellman residual guarantees proximity to the optimal value function. In particular, the factor $(1-\gamma)^{-1}$ quantifies the amplification of local (one-step) errors into global (long-term) performance loss.

A related notion is the *policy residual*. Given a policy π and an approximate value function $\tilde{V}$, define

$$\mathcal{R}^\pi(\tilde{V}) := \|T^\pi\tilde{V} - \tilde{V}\|_\infty. \tag{552}$$

Then one similarly obtains

$$\|\tilde{V} - V^\pi\|_\infty \leq \frac{1}{1-\gamma}\mathcal{R}^\pi(\tilde{V}), \tag{553}$$

which provides an error bound for policy evaluation. Furthermore, residuals can be used to bound the performance loss of greedy policies. Let $\tilde{\pi}$ be a policy greedy with respect to $\tilde{V}$:

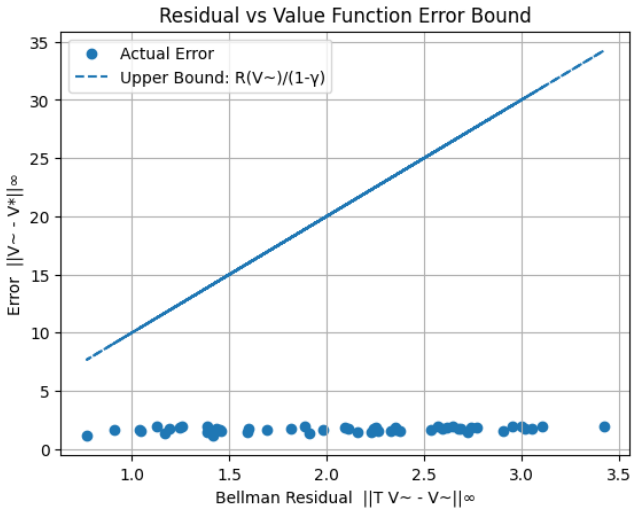
$$\tilde{\pi}(s) \in \operatorname{argmax}_{a\in\mathcal{A}}\left[R(s,a) + \gamma\int_{\mathcal{S}}\tilde{V}(s')\,P(ds'|s,a)\right]. \tag{554}$$

Then one can show that

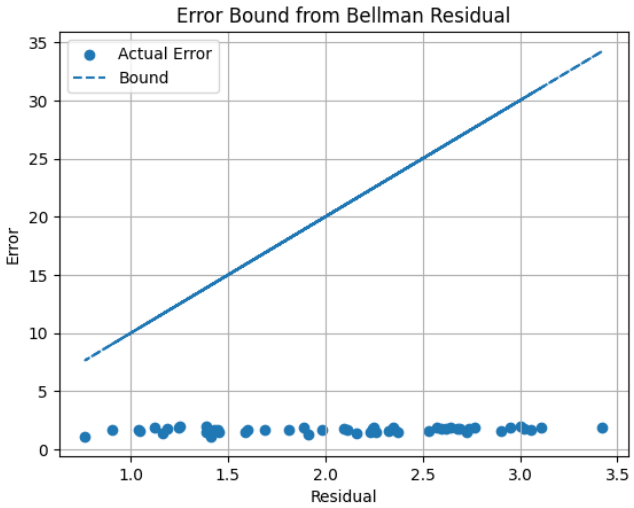
$$\|V^{\tilde{\pi}} - V^*\|_\infty \leq \frac{2\gamma}{1-\gamma}\|\tilde{V} - V^*\|_\infty, \tag{555}$$

which, combined with the residual bound, yields

$$\|V^{\tilde{\pi}} - V^*\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\mathcal{R}(\tilde{V}). \tag{556}$$



(a) **Verification of the residual-based error bound $\|\tilde{V} - V^*\|_\infty \leq \frac{1}{1-\gamma}\mathcal{R}(\tilde{V})$. Each point corresponds to an approximate value function, showing that the Bellman residual controls the approximation error.**



(b) **Stability of the fixed point under perturbations. The inequality $(1-\gamma)\|\tilde{V} - V^*\|_\infty \leq \mathcal{R}(\tilde{V})$ illustrates Lipschitz continuity of the solution with respect to Bellman residuals.**

(c) **Performance loss of the greedy policy induced by $\tilde{V}$. The bound $\|V^{\tilde{\pi}} - V^*\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\mathcal{R}(\tilde{V})$ shows how approximation errors propagate to control performance.**

Figure 21: **Bellman Residuals and Error Bounds: The Bellman residual $\mathcal{R}(\tilde{V}) = \|T\tilde{V} - \tilde{V}\|_\infty$ provides a computable measure of deviation from optimality.**

From an operator-theoretic perspective, the residual $\mathcal{R}(\tilde{V})$ can be interpreted as the norm of the defect in the fixed-point equation. The inequality above is a stability result: it shows that the fixed point of a contraction mapping depends continuously on perturbations, with Lipschitz constant $(1-\gamma)^{-1}$.

The three figures (Figure 21a, 21b, and 21c) provide a rigorous quantitative link between approximation, operator-theoretic stability, and control performance through the notion of the *Bellman residual*. Let the residual of an approximate value function $\tilde{V}$ be defined by

$$\mathcal{R}(\tilde{V}) := \|T\tilde{V} - \tilde{V}\|_\infty, \tag{557}$$

which measures the defect in the fixed-point equation

$$V = TV. \tag{558}$$

In Figure 1 (Figure 21a), this residual is shown to directly control the value-function error via the inequality

$$\|\tilde{V} - V^*\|_\infty \leq \frac{1}{1-\gamma}\,\mathcal{R}(\tilde{V}). \tag{559}$$

Each plotted point corresponds to a different approximation $\tilde{V}$, while the dashed line represents the theoretical upper bound. The visualization confirms that the residual acts as a *certificate of accuracy*: as $\mathcal{R}(\tilde{V}) \to 0$, one necessarily has $\tilde{V} \to V^*$. Thus, the Bellman residual provides a computable proxy for the otherwise unknown error $\|\tilde{V} - V^*\|_\infty$.

Figure 2 (Figure 21b) presents the same inequality in its equivalent rearranged form

$$(1-\gamma)\|\tilde{V} - V^*\|_\infty \leq \mathcal{R}(\tilde{V}), \tag{560}$$

highlighting the *Lipschitz stability* of the fixed point with respect to perturbations. From a geometric perspective, the residual represents the distance from satisfying the fixed-point condition

$$V = TV, \tag{561}$$

and the contraction property

$$\|TV - TW\|_\infty \leq \gamma\|V - W\|_\infty \tag{562}$$

ensures that deviations from optimality cannot grow arbitrarily. Instead, the fixed point depends continuously on perturbations, with stability modulus $(1-\gamma)^{-1}$.

In Figure 3 (Figure 21c), the impact of approximation error on control performance is quantified through the bound

$$\|V^{\tilde{\pi}} - V^*\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\,\mathcal{R}(\tilde{V}), \tag{563}$$

where $\tilde{\pi}$ is the greedy policy induced by $\tilde{V}$. This figure demonstrates that approximation error propagates into *policy degradation*, and crucially, that this degradation is amplified by the factor $\frac{2\gamma}{(1-\gamma)^2}$. Thus, even small residual errors can lead to larger control errors when γ is close to 1, reflecting the accumulation of long-term effects.

Taken together, the figures (Figure 21a, 21b, and 21c) illustrate the fundamental chain

$$\mathcal{R}(\tilde{V}) \;\Rightarrow\; \|\tilde{V} - V^*\|_\infty \;\Rightarrow\; \|V^{\tilde{\pi}} - V^*\|_\infty, \tag{564}$$

which connects *local consistency error* (residual) to *global value error*, and ultimately to *control suboptimality*. This provides a precise operator-theoretic interpretation: the residual is the norm of the defect in the nonlinear fixed-point equation, the error bound follows from contraction stability, and the policy loss bound quantifies how approximation inaccuracies propagate through the Bellman optimality structure.

Conceptually, these visualizations encode several deep principles simultaneously: the *stability of contraction mappings*, the *propagation of errors across time horizons*, and the *link between approximation*

quality and decision-making performance. From a practical standpoint, they justify using the Bellman residual as a stopping criterion in algorithms, since controlling $\mathcal{R}(\tilde{V})$ guarantees both value accuracy and near-optimal control.

In summary, the Bellman residual provides a computable and theoretically meaningful measure of approximation quality. It bridges the gap between numerical approximation and control performance, and plays a central role in approximate dynamic programming, reinforcement learning, and error analysis of value function approximation schemes.

7 Dynamic Programming

Dynamic programming provides a collection of recursive methods for solving Markov Decision Processes by exploiting the Bellman equations. The central idea is to compute value functions and optimal policies via successive approximations based on local consistency conditions. These methods rely fundamentally on the contraction and monotonicity properties of Bellman operators, ensuring convergence to the optimal solution.

Dynamic programming constitutes one of the foundational mathematical frameworks for sequential decision-making and optimal control, originating in the pioneering work of Bellman (1957) [15] and subsequently expanded by Howard (1960) [16] and Bertsekas (2012) [9]. Bellman (1957) [15] introduced the principle of optimality and formulated dynamic programming as a recursive decomposition methodology in which complex multistage optimization problems are reduced to sequences of simpler subproblems characterized through Bellman equations and value functions. Howard (1960) [16] extended these ideas to controlled Markov processes by systematically developing policy iteration methods, probabilistic decision models, and recursive optimization procedures for stochastic systems, thereby establishing important computational and theoretical links between dynamic programming and Markov decision processes. Bertsekas (2012) [9] further developed the modern theory of dynamic programming and optimal control by rigorously analyzing deterministic and stochastic systems, contraction mappings, Bellman operators, approximate dynamic programming, and large-scale optimization methods, while also connecting dynamic programming with reinforcement learning and neuro-dynamic programming techniques. Together, these works established dynamic programming as a universal recursive optimization framework for sequential decision-making under uncertainty, providing the theoretical and computational foundations for stochastic control, Markov decision processes, and modern reinforcement learning.

We begin with *policy evaluation*, which computes the value function associated with a fixed policy π. Starting from an initial function $V_0 \in \mathcal{B}_b(\mathcal{S})$, define the iterative scheme

$$V_{k+1}(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a) V_k(s') \right]. \tag{565}$$

Equivalently, in operator form,

$$V_{k+1} = T^\pi V_k. \tag{566}$$

By the contraction property of T^π, the sequence $\{V_k\}$ converges geometrically to the unique fixed point V^π, i.e.,

$$\lim_{k\to\infty} \|V_k - V^\pi\|_\infty = 0. \tag{567}$$

Thus, policy evaluation reduces to solving a linear fixed-point equation via successive approximation.

Next, *policy improvement* constructs a new policy that is greedy with respect to the current value function. Given a policy π and its value function V^π, define the action-value function

$$Q^\pi(s,a) = R(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a) V^\pi(s'). \tag{568}$$

A greedy policy π' is then defined by

$$\pi'(s) \in \operatorname{argmax}_{a\in\mathcal{A}} Q^\pi(s,a). \tag{569}$$

This step ensures that

$$V^{\pi'}(s) \geq V^{\pi}(s), \quad \forall s \in \mathcal{S}, \tag{570}$$

which is known as the policy improvement property. Combining policy evaluation and policy improvement yields the *policy iteration* algorithm. Starting from an initial policy π_0, one generates a sequence $\{\pi_k\}$ via:

$$\text{Policy Evaluation: } V^{\pi_k} = T^{\pi_k} V^{\pi_k}, \tag{571}$$

$$\text{Policy Improvement: } \pi_{k+1}(s) \in \text{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^{\pi_k}(s') \right]. \tag{572}$$

This produces a monotonically improving sequence of value functions:

$$V^{\pi_0} \leq V^{\pi_1} \leq V^{\pi_2} \leq \cdots \leq V^*. \tag{573}$$

In the finite state-action setting, policy iteration converges to an optimal policy π^* in a finite number of steps.

An alternative approach is *value iteration*, which directly applies the optimal Bellman operator. Starting from an arbitrary initial function V_0, define

$$V_{k+1}(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_k(s') \right]. \tag{574}$$

In operator form,

$$V_{k+1} = T V_k. \tag{575}$$

By the contraction property of T, the sequence $\{V_k\}$ converges uniformly to the optimal value function V^*:

$$\lim_{k \to \infty} \|V_k - V^*\|_\infty = 0, \quad \|V_k - V^*\|_\infty \leq \gamma^k \|V_0 - V^*\|_\infty. \tag{576}$$

An approximate optimal policy can then be extracted via

$$\pi_k(s) \in \text{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_k(s') \right]. \tag{577}$$

From a conceptual standpoint, policy iteration alternates between exact evaluation and greedy improvement, while value iteration performs a single-step lookahead at each iteration. Both methods are rooted in the same fixed-point structure but differ in computational trade-offs: policy iteration typically converges in fewer iterations but requires solving a full evaluation step, whereas value iteration uses simpler updates but may require more iterations.

In summary, dynamic programming transforms the global optimization problem of reinforcement learning into recursive local computations. Through policy evaluation, policy improvement, and value iteration, one systematically approaches the optimal value function and policy by exploiting the structure of the Bellman equations.

7.1 Policy Evaluation

Policy evaluation is the problem of computing the value function V^π associated with a fixed policy π. In the dynamic programming framework, this is achieved by solving the Bellman expectation equation

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^\pi(s') \right], \quad \forall s \in \mathcal{S}. \tag{578}$$

This equation characterizes V^π as the unique fixed point of the linear Bellman operator T^π.

A natural computational approach is to construct a sequence $\{V_k\}$ via successive approximation:

$$V_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_k(s') \right], \tag{579}$$

or equivalently,

$$V_{k+1} = T^\pi V_k. \tag{580}$$

This iteration is known as *iterative policy evaluation*. Starting from an arbitrary initial function V_0, the sequence $\{V_k\}$ converges to V^π due to the contraction property of T^π:

$$\|V_{k+1} - V^\pi\|_\infty = \|T^\pi V_k - T^\pi V^\pi\|_\infty \leq \gamma \|V_k - V^\pi\|_\infty. \tag{581}$$

By induction, one obtains the geometric convergence bound

$$\|V_k - V^\pi\|_\infty \leq \gamma^k \|V_0 - V^\pi\|_\infty, \tag{582}$$

which shows that the error decays exponentially fast.

The iteration admits a probabilistic interpretation. Expanding recursively,

$$V_k(s) = \mathbb{E}_s^\pi \left[\sum_{t=0}^{k-1} \gamma^t R(s_t, a_t) + \gamma^k V_0(s_k) \right].$$

As $k \to \infty$, the terminal term $\gamma^k V_0(s_k)$ vanishes (under boundedness assumptions), yielding

$$V^\pi(s) = \mathbb{E}_s^\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{583}$$

Thus, iterative policy evaluation accumulates expected rewards over increasingly long horizons. In the finite state-action setting, policy evaluation can also be expressed as a linear system. Let P^π and r^π denote the transition matrix and reward vector induced by π. Then

$$V^\pi = r^\pi + \gamma P^\pi V^\pi, \tag{584}$$

or equivalently,

$$(I - \gamma P^\pi) V^\pi = r^\pi. \tag{585}$$

This system has a unique solution since $\rho(\gamma P^\pi) < 1$. Iterative policy evaluation corresponds to applying the fixed-point iteration

$$V_{k+1} = r^\pi + \gamma P^\pi V_k, \tag{586}$$

which converges to the exact solution.

From an algorithmic perspective, policy evaluation may be implemented either synchronously (updating all states simultaneously using V_k) or asynchronously (updating states one at a time using the most recent values). Both schemes converge under standard conditions, with asynchronous updates often offering computational advantages in large-scale problems.

In summary, policy evaluation is the fundamental step in dynamic programming that computes the expected return under a fixed policy. It reduces to solving a linear fixed-point equation and serves as a building block for more advanced methods such as policy iteration and actor-critic algorithms.

7.2 Policy Iteration

Policy iteration is a fundamental dynamic programming algorithm that computes an optimal policy by iteratively alternating between *policy evaluation* and *policy improvement*. Starting from an initial policy π_0, the method constructs a sequence of policies $\{\pi_k\}$ that monotonically improve with respect to their value functions.

Given a policy π_k, the first step is to evaluate its value function V^{π_k} by solving the Bellman expectation equation

$$V^{\pi_k}(s) = \sum_{a \in \mathcal{A}} \pi_k(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^{\pi_k}(s') \right], \quad \forall s \in \mathcal{S}. \tag{587}$$

This step may be carried out exactly (by solving a linear system) or approximately via iterative methods.

The second step is *policy improvement*. Using the evaluated value function V^{π_k}, define the action-value function

$$Q^{\pi_k}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^{\pi_k}(s'). \tag{588}$$

A new policy π_{k+1} is then constructed by acting greedily with respect to Q^{π_k}:

$$\pi_{k+1}(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q^{\pi_k}(s,a), \quad \forall s \in \mathcal{S}. \tag{589}$$

Equivalently,

$$\pi_{k+1}(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^{\pi_k}(s') \right]. \tag{590}$$

A key property of this procedure is the *policy improvement theorem*, which guarantees that

$$V^{\pi_{k+1}}(s) \geq V^{\pi_k}(s), \quad \forall s \in \mathcal{S}. \tag{591}$$

To see this, note that for all s,

$$V^{\pi_k}(s) = \sum_a \pi_k(a|s) Q^{\pi_k}(s,a) \leq \max_a Q^{\pi_k}(s,a) = Q^{\pi_k}(s, \pi_{k+1}(s)). \tag{592}$$

Using the Bellman equation for $V^{\pi_{k+1}}$, one can propagate this inequality forward in time to obtain the monotonic improvement.

Thus, policy iteration generates a sequence of value functions satisfying

$$V^{\pi_0} \leq V^{\pi_1} \leq V^{\pi_2} \leq \cdots \leq V^*, \tag{593}$$

where V^* is the optimal value function. Since the number of deterministic policies is finite in the finite state-action setting, this sequence must terminate after a finite number of steps at an optimal policy π^* satisfying

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right]. \tag{594}$$

From an operator-theoretic perspective, policy iteration can be viewed as alternating between solving the fixed-point equation

$$V^{\pi_k} = T^{\pi_k} V^{\pi_k}, \tag{595}$$

and updating the policy according to

$$\pi_{k+1} \in \mathcal{G}(V^{\pi_k}), \tag{596}$$

where $\mathcal{G}$ denotes the greedy operator. This process can be interpreted as a form of Newton-like method for solving the nonlinear Bellman optimality equation.

In practice, exact policy evaluation may be computationally expensive, especially in large state spaces. This leads to the *modified policy iteration* variant, where policy evaluation is performed approximately using a finite number of iterations:

$$V_{k,0} := V^{\pi_k}, \quad V_{k,m+1} := T^{\pi_k} V_{k,m}, \quad m = 0, 1, \ldots, M, \tag{597}$$

followed by policy improvement using $V_{k,M}$.

In summary, policy iteration is a powerful and conceptually simple algorithm that leverages the structure of the Bellman equations. By iteratively evaluating and improving policies, it converges to an optimal policy in a finite number of steps in the finite setting, and forms the basis for many reinforcement learning algorithms.

7.3 Value Iteration

Value iteration is a fundamental dynamic programming method for computing the optimal value function V^* by directly iterating the Bellman optimality operator. Starting from an arbitrary initial function $V_0 \in \mathcal{B}_b(\mathcal{S})$, one constructs a sequence $\{V_k\}$ via

$$V_{k+1}(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_k(s') \right], \quad \forall s \in \mathcal{S}. \tag{598}$$

Equivalently, in operator form,

$$V_{k+1} = TV_k, \tag{599}$$

where T is the Bellman optimality operator.

The iteration arises from the fixed-point equation

$$V^* = TV^*, \tag{600}$$

which characterizes the optimal value function. Since T is a contraction mapping under the supremum norm, one has

$$\|TV - TW\|_\infty \leq \gamma \|V - W\|_\infty, \quad \forall V, W \in \mathcal{B}_b(\mathcal{S}), \tag{601}$$

with $\gamma \in [0,1)$. Therefore, by the Banach fixed-point theorem, T admits a unique fixed point V^*, and the sequence $\{V_k\}$ converges to V^* for any initial V_0.

More precisely, the convergence is geometric:

$$\|V_k - V^*\|_\infty \leq \gamma^k \|V_0 - V^*\|_\infty. \tag{602}$$

This bound shows that the error decreases exponentially with the number of iterations, with rate determined by the discount factor γ. The iteration admits a natural interpretation as successive refinement of finite-horizon value functions. Define

$$V_k(s) = \sup_\pi \mathbb{E}_s^\pi \left[\sum_{t=0}^{k-1} \gamma^t R(s_t, a_t) \right]. \tag{603}$$

Then one can show by induction that the value iteration recursion computes exactly these k-stage optimal returns, and as $k \to \infty$, $V_k(s) \to V^*(s)$. An important feature of value iteration is that it simultaneously performs policy evaluation and policy improvement in a single step. At each iteration, one may extract a greedy policy π_k defined by

$$\pi_k(s) \in \operatorname{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_k(s') \right]. \tag{604}$$

Under mild conditions, the sequence $\{\pi_k\}$ converges to an optimal policy π^*. In the finite state-action setting, value iteration can be implemented efficiently using vector and matrix operations:

$$V_{k+1} = \max_{a \in \mathcal{A}} \left(r_a + \gamma P_a V_k \right), \tag{605}$$

where P_a and r_a denote the transition matrix and reward vector corresponding to action a, and the maximum is taken componentwise.

From a computational perspective, value iteration has lower per-iteration cost compared to policy iteration, since it avoids solving a full linear system. However, it may require more iterations to converge, particularly when γ is close to 1. In practice, one often terminates the iteration when the Bellman residual becomes small:

$$\|TV_k - V_k\|_\infty \leq \varepsilon, \tag{606}$$

which ensures that V_k is within $\mathcal{O}(\varepsilon/(1-\gamma))$ of V^*.

Finally, value iteration can be extended to asynchronous and stochastic variants, where updates are performed on subsets of states or using sampled transitions. These extensions form the basis of many reinforcement learning algorithms, including Q-learning and approximate dynamic programming methods.

In summary, value iteration is a direct and theoretically grounded method for solving the Bellman optimality equation. By iteratively applying the optimal Bellman operator, it converges to the unique optimal value function and yields an optimal policy through greedy action selection.

8 Model-Free Methods

Model-free methods in reinforcement learning aim to learn value functions and optimal policies directly from sampled trajectories, without requiring explicit knowledge of the transition kernel P or the reward function R. These methods rely on stochastic approximation and bootstrapping, using observed data to iteratively refine estimates of value functions.

A central class of model-free methods is *Temporal Difference (TD) learning*, which combines ideas from Monte Carlo methods and dynamic programming. Given a trajectory $\{(s_t, a_t, R_{t+1}, s_{t+1})\}$ generated under a policy π, the TD(0) update for the state-value function is

$$V(s_t) \leftarrow V(s_t) + \alpha_t \delta_t, \tag{607}$$

where $\alpha_t > 0$ is a step-size parameter and

$$\delta_t := R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{608}$$

is the *temporal-difference error*. This error represents the discrepancy between the current estimate $V(s_t)$ and a one-step bootstrap target $R_{t+1} + \gamma V(s_{t+1})$. The update can be interpreted as a stochastic approximation to the Bellman expectation equation

$$V^\pi(s) = \mathbb{E}^\pi \left[R_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s \right]. \tag{609}$$

From a stochastic approximation perspective, the TD update seeks to solve the fixed-point equation $V = T^\pi V$ by replacing expectations with single-sample realizations. Under standard conditions (e.g., Robbins–Monro step sizes satisfying $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$, and sufficient exploration), one can show that

$$V(s) \to V^\pi(s) \quad \text{almost surely.} \tag{610}$$

A closely related method is *Q-learning*, which directly estimates the optimal action-value function Q^* without requiring a model or a fixed policy. The update rule is

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \delta_t^Q, \tag{611}$$

where the temporal-difference error is defined as

$$\delta_t^Q := R_{t+1} + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t). \tag{612}$$

This update corresponds to a stochastic approximation of the Bellman optimality equation

$$Q^*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right]. \tag{613}$$

An important feature of Q-learning is that it is *off-policy*: the updates use the greedy target $\max_a Q(s_{t+1}, a)$ regardless of the behavior policy used to generate the data. This allows the algorithm to learn the optimal policy while exploring the environment using a possibly different policy (e.g., ε-greedy exploration).

Under suitable assumptions—such as finite state-action spaces, sufficient exploration (each state-action pair is visited infinitely often), and diminishing step sizes—Q-learning converges almost surely to the optimal action-value function:

$$Q(s,a) \to Q^*(s,a), \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}. \tag{614}$$

An optimal policy can then be recovered via

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s,a). \tag{615}$$

Both TD learning and Q-learning rely on the principle of *bootstrapping*, where current estimates are updated using other learned estimates rather than waiting for complete returns. This leads to low-variance updates and efficient online learning, but introduces bias that must be controlled through appropriate step-size schedules and exploration strategies.

From an operator-theoretic viewpoint, these methods approximate the fixed points of the Bellman operators T^π and T using stochastic iterative schemes of the form

$$V_{k+1} = V_k + \alpha_k \left(\widehat{T^\pi V_k} - V_k \right), \quad Q_{k+1} = Q_k + \alpha_k \left(\widehat{TQ_k} - Q_k \right), \tag{616}$$

where $\widehat{T^\pi V_k}$ and $\widehat{TQ_k}$ denote noisy, sample-based approximations of the Bellman operators.

In summary, model-free methods provide a powerful framework for learning in unknown environments. By replacing expectations with sampled transitions, they enable the approximation of value functions and optimal policies directly from data, forming the foundation of modern reinforcement learning algorithms.

8.1 Temporal Difference Learning

Temporal Difference (TD) learning is a fundamental model-free method for estimating the value function V^π of a given policy π directly from sampled trajectories, without requiring knowledge of the transition kernel or reward distribution. The key idea is to combine bootstrapping (as in dynamic programming) with sampling (as in Monte Carlo methods), thereby forming an efficient online learning scheme.

Temporal-difference (TD) learning is one of the central algorithmic and theoretical foundations of modern reinforcement learning, originating in the seminal work of Sutton (1988) [24] and later rigorously analyzed by Sutton and Barto (1998) [1] and by Tsitsiklis and Van Roy (1996) [25]. Sutton (1988) [24] introduced temporal-difference learning as a class of prediction methods that combine ideas from Monte Carlo estimation and dynamic programming by updating value estimates through bootstrapping from subsequent predictions, thereby enabling efficient online learning directly from incomplete sequences of experience. Sutton and Barto (1998) [1] subsequently developed TD learning within the broader reinforcement learning framework, presenting algorithms such as TD(0), TD(λ), SARSA, and Q-learning, while emphasizing the role of temporal-difference errors in value-function approximation, policy evaluation, and sequential decision-making under uncertainty. Tsitsiklis and Van Roy (1996) [25] provided a rigorous mathematical analysis of temporal-difference methods using tools from stochastic approximation and linear systems theory, establishing convergence properties, stability conditions, and approximation behavior for TD algorithms under linear function approximation settings. Analysis of Temporal-Difference Learning Together, these works established temporal-difference learning as a fundamental computational mechanism for recursively estimating value functions from experience, thereby forming a cornerstone of modern reinforcement learning and approximate dynamic programming.

Given a trajectory $\{(s_t, a_t, R_{t+1}, s_{t+1})\}_{t \geq 0}$ generated under policy π, the TD(0) update rule is

$$V(s_t) \leftarrow V(s_t) + \alpha_t \left[R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \right], \tag{617}$$

where $\alpha_t \in (0,1]$ is a step-size (learning rate), and the quantity

$$\delta_t := R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{618}$$

is called the *temporal-difference (TD) error*. This error measures the discrepancy between the current estimate $V(s_t)$ and a one-step lookahead estimate based on the observed reward and the estimated value of the next state.

From a theoretical standpoint, TD learning can be interpreted as a stochastic approximation scheme for solving the Bellman expectation equation

$$V^\pi(s) = \mathbb{E}^\pi \left[R_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s \right]. \tag{619}$$

Defining the Bellman operator T^π, the update can be written as

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t \left(\widehat{T^\pi V_t}(s_t) - V_t(s_t) \right), \tag{620}$$

where

$$\widehat{T^\pi V_t}(s_t) := R_{t+1} + \gamma V_t(s_{t+1}) \tag{621}$$

is a single-sample (noisy) estimate of $(T^\pi V_t)(s_t)$. Thus, TD learning replaces the expectation in the Bellman equation with an unbiased sample, yielding an incremental update.

Under standard assumptions—finite state space, bounded rewards, and step sizes satisfying the Robbins–Monro conditions

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty, \tag{622}$$

together with sufficient exploration ensuring that each state is visited infinitely often—one can show that

$$V(s) \to V^\pi(s) \quad \text{almost surely.} \tag{623}$$

This convergence result is typically established using stochastic approximation theory and the contraction property of T^π. A useful interpretation of TD learning arises from unrolling the recursion. Iterating the update yields

$$V(s_t) \approx \mathbb{E}^\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \right], \tag{624}$$

but unlike Monte Carlo methods, TD learning updates $V(s_t)$ after observing only one step, using the estimate $V(s_{t+1})$ as a proxy for future returns. This *bootstrapping* mechanism reduces variance but introduces bias, leading to a bias-variance trade-off.

More generally, TD methods can be extended to multi-step versions. For example, the n-step TD target is

$$G_t^{(n)} := \sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n V(s_{t+n}), \tag{625}$$

and the corresponding update is

$$V(s_t) \leftarrow V(s_t) + \alpha_t \left[G_t^{(n)} - V(s_t) \right]. \tag{626}$$

Taking $n \to \infty$ recovers Monte Carlo methods, while $n = 1$ yields TD(0). Intermediate values of n interpolate between these extremes. An important extension is TD(λ), which forms a weighted average of n-step returns:

$$G_t^{(\lambda)} := (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}, \quad \lambda \in [0, 1]. \tag{627}$$

The corresponding update incorporates eligibility traces and provides a continuum of algorithms between TD(0) and Monte Carlo. From an operator-theoretic perspective, TD learning can be viewed as a stochastic fixed-point iteration:

$$V_{k+1} = V_k + \alpha_k (T^\pi V_k - V_k + \text{noise}), \tag{628}$$

where the noise arises from sampling. The contraction of T^π ensures stability, while stochastic approximation guarantees convergence under suitable conditions.

In summary, temporal difference learning provides an efficient and scalable method for policy evaluation in unknown environments. By combining sampling with bootstrapping, it enables online, incremental updates that converge to the true value function while maintaining low computational complexity.

8.2 Q-Learning

Q-learning is a canonical model-free algorithm for directly estimating the optimal action-value function Q^* without requiring knowledge of the transition kernel or reward distribution. Unlike policy evaluation methods, Q-learning is *off-policy*, meaning that it learns the optimal policy independently of the behavior policy used to generate data.

Q-learning is one of the most influential model-free reinforcement learning algorithms and was pioneered in the work of Watkins (1989) [14], later formalized with Dayan (1992) [26], and rigorously analyzed by Jaakkola, Jordan, and Singh (1993) [27]. Watkins (1989) [14] introduced Q-learning as a temporal-difference control method in which optimal action-value functions are recursively estimated from sampled transitions and delayed rewards without requiring prior knowledge of the environment's transition dynamics. Watkins and Dayan (1992) [26] subsequently provided the formal algorithmic and theoretical development of Q-learning, establishing the Bellman optimality-based update rule and proving convergence to the optimal action-value function under suitable assumptions on exploration and learning rates. Jaakkola, Jordan, and Singh (1993) [27] further strengthened the mathematical foundations of stochastic iterative dynamic programming methods by developing rigorous convergence analyses for reinforcement learning algorithms, including Q-learning, using stochastic approximation techniques and probabilistic convergence theory in asynchronous and noisy learning environments. Together, these works established Q-learning as a foundational algorithm for model-free optimal control, demonstrating how agents can asymptotically learn optimal policies directly from interaction with stochastic environments through recursive action-value estimation.

Given a trajectory $\{(s_t, a_t, R_{t+1}, s_{t+1})\}_{t\geq 0}$, the Q-learning update is

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \left[R_{t+1} + \gamma \max_{a\in\mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right], \tag{629}$$

where $\alpha_t \in (0, 1]$ is a step-size parameter. The quantity

$$\delta_t^Q := R_{t+1} + \gamma \max_{a\in\mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \tag{630}$$

is the *temporal-difference error* for Q-learning, representing the discrepancy between the current estimate and a one-step optimal bootstrap target.

The update rule can be interpreted as a stochastic approximation to the Bellman optimality equation for the action-value function:

$$Q^*(s, a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right]. \tag{631}$$

Defining the optimal Bellman operator T acting on Q-functions by

$$(TQ)(s, a) := R(s, a) + \gamma \int_{\mathcal{S}} \max_{a'} Q(s', a')\, P(ds'|s, a), \tag{632}$$

the Q-learning update can be written as

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \left(\widehat{TQ_t}(s_t, a_t) - Q_t(s_t, a_t) \right), \tag{633}$$

where

$$\widehat{TQ_t}(s_t, a_t) := R_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a') \tag{634}$$

is a single-sample estimate of $(TQ_t)(s_t, a_t)$. Thus, Q-learning performs a stochastic fixed-point iteration for solving $Q = TQ$. A key feature of Q-learning is that the target uses the greedy action $\max_a Q(s_{t+1}, a)$, regardless of how a_t was selected. This enables learning of the optimal policy while following an exploratory behavior policy (e.g., ε-greedy), ensuring sufficient exploration of the state-action space.

Under standard assumptions—finite state-action spaces, bounded rewards, step sizes satisfying

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty, \tag{635}$$

and the condition that each state-action pair (s, a) is visited infinitely often—Q-learning converges almost surely to the optimal action-value function:

$$Q(s, a) \to Q^*(s, a), \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A}. \tag{636}$$

This result is a consequence of stochastic approximation theory combined with the contraction property of the Bellman operator. Once Q^* is obtained, an optimal policy can be extracted via

$$\pi^*(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s, a). \tag{637}$$

From a computational perspective, Q-learning is particularly attractive because it avoids explicit policy representation and value function evaluation; instead, it directly approximates the optimal action-value function. Moreover, it operates online and incrementally, updating only the visited state-action pair at each step.

However, the use of the maximum operator introduces a positive bias in the presence of noise, known as *overestimation bias*. This has led to variants such as Double Q-learning, which mitigate this effect by decoupling action selection and evaluation.

In summary, Q-learning is a foundational model-free reinforcement learning algorithm that combines stochastic approximation with dynamic programming principles. By iteratively updating estimates based on sampled transitions and greedy targets, it converges to the optimal action-value function and enables the derivation of optimal policies in unknown environments.

8.3 Policy Gradient Methods

Policy gradient methods constitute a major class of reinforcement learning algorithms that optimize policies directly in parameter space, and their theoretical and algorithmic foundations were established through the works of Williams (1992) [28], Sutton et al. (1999) [29], and Silver et al. (2014) [30]. Williams (1992) [28] introduced the REINFORCE family of algorithms, providing one of the earliest stochastic gradient-based approaches for reinforcement learning in which policy parameters are updated through unbiased estimates of the gradient of expected return, thereby enabling direct optimization of probabilistic decision policies in stochastic environments. Sutton et al. (1999) [29] subsequently developed a rigorous policy gradient framework for reinforcement learning with function approximation, deriving the policy gradient theorem and establishing actor–critic architectures that combine value-function estimation with gradient-based policy optimization for continuous and high-dimensional control problems. Silver et al. (2014) [30] extended these ideas through deterministic policy gradient methods, demonstrating that deterministic policies can yield more sample-efficient gradient estimation in continuous action spaces and providing scalable algorithms that later became central to deep reinforcement learning and continuous control applications. Together, these works established the mathematical and computational foundations of policy gradient reinforcement learning, showing how optimal behavior can be learned through direct gradient-based optimization of parameterized policies in stochastic and continuous control environments.

Let $\{\pi_\theta\}_{\theta \in \mathbb{R}^d}$ be a family of stochastic policies, where θ denotes the parameter vector. The objective functional is defined as

$$J(\theta) := \mathbb{E}_{\pi_\theta}\left[G_t\right], \tag{638}$$

where the expectation is taken with respect to the trajectory distribution induced by π_θ. More explicitly, under an initial distribution ρ, one may write

$$J(\theta) = \mathbb{E}_\rho^{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]. \tag{639}$$

The central problem is to compute the gradient $\nabla_\theta J(\theta)$ in order to perform gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta J(\theta_k). \tag{640}$$

A key technical difficulty is that the distribution over trajectories depends on θ. To address this, one employs the *likelihood ratio method*. Let $\tau = (s_0, a_0, s_1, a_1, \dots)$ denote a trajectory. The probability density of τ under π_θ is

$$\mathbb{P}_\theta(\tau) = \rho(s_0) \prod_{t=0}^{\infty} \pi_\theta(a_t|s_t)\, P(s_{t+1}|s_t, a_t). \tag{641}$$

Since the environment dynamics P do not depend on θ, the gradient of the log-likelihood is

$$\nabla_\theta \log \mathbb{P}_\theta(\tau) = \sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t). \tag{642}$$

Using this identity, one can differentiate the objective:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \int G_0(\tau)\, \mathbb{P}_\theta(d\tau) \\
&= \int G_0(\tau)\, \nabla_\theta \mathbb{P}_\theta(d\tau) \\
&= \int G_0(\tau)\, \mathbb{P}_\theta(d\tau)\, \nabla_\theta \log \mathbb{P}_\theta(\tau) \\
&= \mathbb{E}_{\pi_\theta}\left[G_0 \sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)\right].
\end{aligned}$$

Rearranging the terms and conditioning on (s_t, a_t) yields

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)\, G_t\right]. \tag{643}$$

Replacing the return G_t with the action-value function $Q^\pi(s_t, a_t)$ gives the *Policy Gradient Theorem*:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\nabla_\theta \log \pi_\theta(a|s)\, Q^\pi(s, a)\right]. \tag{644}$$

An important feature of this result is that it does not involve derivatives of the state distribution, which simplifies computation significantly. The gradient depends only on the policy and the action-value function.

To reduce variance, one often introduces a *baseline* function $b(s)$ and uses the identity

$$\mathbb{E}\left[\nabla_\theta \log \pi_\theta(a|s)\, b(s)\right] = 0. \tag{645}$$

This yields the variance-reduced estimator

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\nabla_\theta \log \pi_\theta(a|s)\, (Q^\pi(s, a) - b(s))\right]. \tag{646}$$

A common choice is $b(s) = V^\pi(s)$, leading to the *advantage function*

$$A^\pi(s, a) := Q^\pi(s, a) - V^\pi(s), \tag{647}$$

and the gradient expression

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\nabla_\theta \log \pi_\theta(a|s)\, A^\pi(s, a)\right]. \tag{648}$$

In practice, the expectations are approximated using sampled trajectories, resulting in stochastic gradient ascent algorithms such as REINFORCE:

$$\theta_{k+1} = \theta_k + \alpha_k \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\, G_t. \tag{649}$$

From a geometric perspective, policy gradient methods perform optimization directly in the parameter space of policies, as opposed to value-based methods which operate in function space. This makes them particularly suitable for high-dimensional or continuous action spaces.

In summary, policy gradient methods provide a principled framework for optimizing parameterized policies via gradient ascent. The policy gradient theorem enables efficient estimation of the gradient using only sampled trajectories, forming the foundation for modern algorithms such as actor-critic methods and proximal policy optimization.

9 Deep Reinforcement Learning

Deep Reinforcement Learning (Deep RL) extends classical reinforcement learning by employing high-capacity function approximators—most notably deep neural networks—to represent value functions, policies, or both. In value-based settings, one approximates the optimal action-value function by a parameterized function $Q(s, a; \theta)$, where $\theta \in \mathbb{R}^d$ denotes the weights of a neural network:

$$Q(s, a; \theta) \approx Q^*(s, a). \tag{650}$$

The objective is to learn θ such that $Q(\cdot, \cdot; \theta)$ satisfies (approximately) the Bellman optimality equation.

A common approach is to minimize the *temporal-difference loss* derived from the Bellman equation. Given a transition $(s_t, a_t, R_{t+1}, s_{t+1})$, define the target

$$y_t := R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta), \tag{651}$$

and consider the squared error loss

$$\mathcal{L}(\theta) := \mathbb{E}\left[\left(Q(s_t, a_t; \theta) - y_t \right)^2 \right]. \tag{652}$$

The parameter update is then performed via stochastic gradient descent:

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta). \tag{653}$$

However, unlike tabular methods, combining function approximation, bootstrapping, and off-policy learning can lead to instability and divergence. To mitigate these issues, several stabilization techniques are employed.

A fundamental technique is *experience replay*. Instead of updating the network using only the most recent transition, one stores past experiences in a dataset (replay buffer)

$$\mathcal{D} := \{(s_i, a_i, r_i, s_i')\}_{i=1}^N, \tag{654}$$

and samples mini-batches uniformly (or according to some priority distribution) to perform updates:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \left[\left(Q(s, a; \theta) - \left(r + \gamma \max_{a'} Q(s', a'; \theta) \right) \right)^2 \right]. \tag{655}$$

This procedure reduces temporal correlations between samples, improves data efficiency by reusing past experiences, and stabilizes gradient estimates.

Another crucial technique is the use of *target networks*. Instead of using the same parameters θ to

compute both the prediction and the target, one introduces a separate set of parameters θ^- (called the target network), which are updated more slowly. The target is then defined as

$$y_t := R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-), \tag{656}$$

and the loss becomes

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left(Q(s_t, a_t; \theta) - \big(R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-)\big)\right)^2\right]. \tag{657}$$

The target parameters are updated periodically or via a soft update:

$$\theta^- \leftarrow \tau\theta + (1-\tau)\theta^-, \quad \tau \ll 1. \tag{658}$$

This decoupling reduces oscillations and prevents harmful feedback loops caused by rapidly changing targets. From an operator-theoretic viewpoint, Deep RL attempts to approximate the fixed point of the Bellman operator within a parameterized function class $\{Q(\cdot,\cdot;\theta)\}$. However, since this class is typically non-linear and non-convex, the projection onto the function space introduces approximation error. Consequently, the iteration can be viewed as

$$Q_{\theta_{k+1}} \approx \Pi\big(TQ_{\theta_k}\big), \tag{659}$$

where Π denotes a projection (induced by the loss function) onto the function class.

In practice, these ideas form the basis of the Deep Q-Network (DQN) algorithm, which successfully combines Q-learning with deep neural networks. Extensions include Double DQN (to reduce overestimation bias), Dueling architectures (to separately estimate value and advantage), and prioritized experience replay.

The three plots (Figure 22a, 22b, and 22c) provide a precise operator-theoretic interpretation of modern reinforcement learning heuristics as approximations to solving a nonlinear fixed-point equation. Recall that the optimal value function is characterized by the Bellman fixed-point equation

$$V^* = TV^*. \tag{660}$$

Each figure visualizes a different aspect of how iterative schemes approximate this equation in practice.

The first plot (Figure 22a) represents the decay of the Bellman residual along the iterates V_k:

$$\mathcal{L}_{\mathrm{TD}}(V_k) = |TV_k - V_k|_\infty. \tag{661}$$

This quantity measures the defect in the fixed-point equation. By the contraction property of T, one has

$$|V_k - V^*| * \infty \leq \frac{1}{1-\gamma} |TV_k - V_k| * \infty, \tag{662}$$

so that vanishing TD loss implies convergence to the optimal value function:

$$\mathcal{L}_{\mathrm{TD}}(V_k) \to 0 \quad \Longleftrightarrow \quad V_k \to V^*. \tag{663}$$

The approximately linear decay in log-scale observed in the figure reflects the geometric contraction:

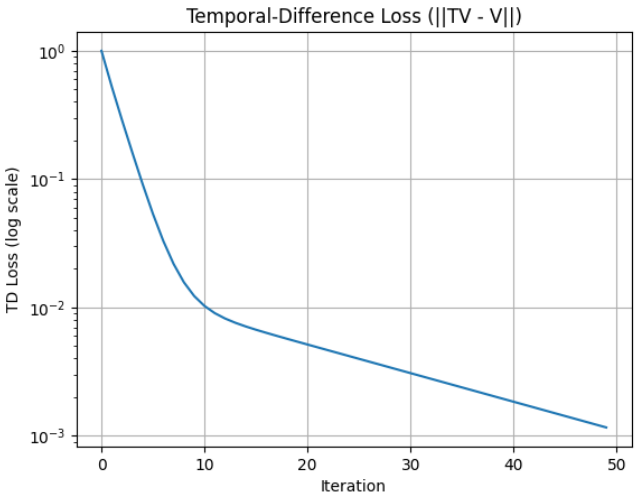
$$|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*|_\infty. \tag{664}$$

The second plot (Figure 22b) compares stochastic approximations of the Bellman update:

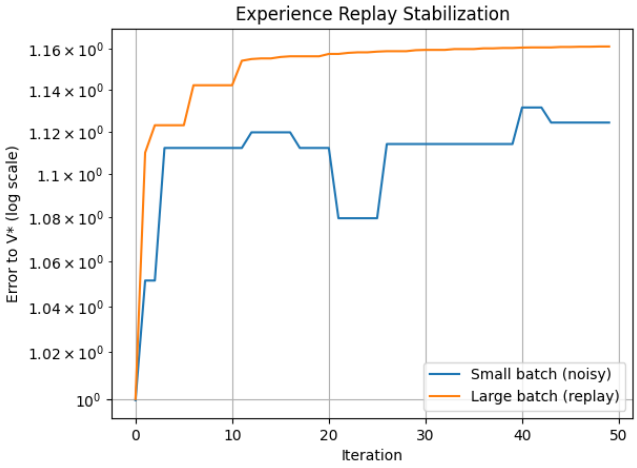
$$V_{k+1} \approx TV_k, \tag{665}$$

where in practice TV_k is replaced by an empirical estimator based on sampled transitions. Denoting a stochastic approximation by $\widehat{T}V_k$, we have

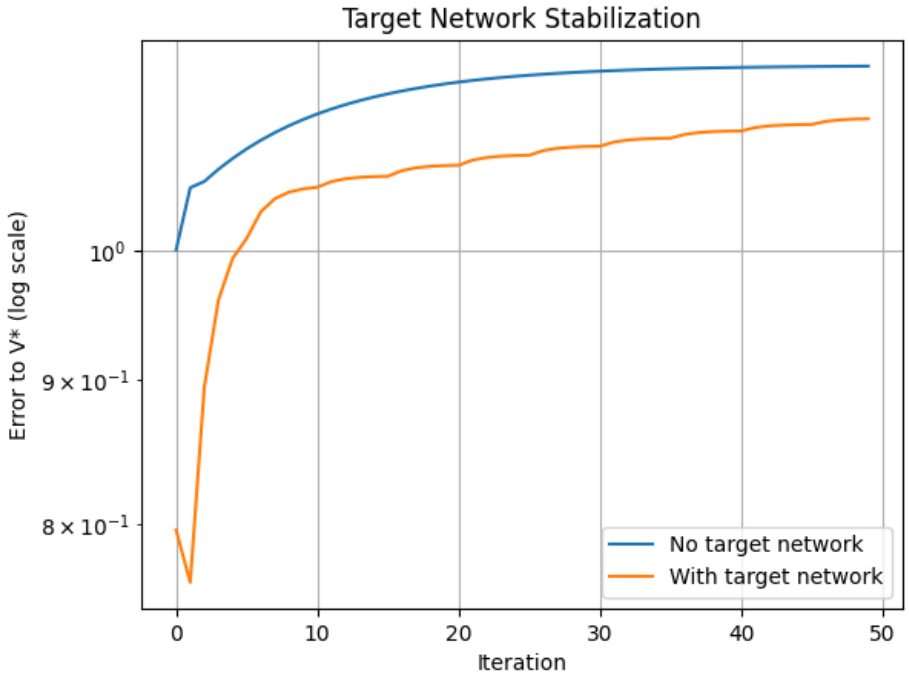
$$\mathbb{E}[\widehat{T}V_k] = TV_k, \tag{666}$$



(a) **Temporal-Difference Loss:** $\|TV - V\|_\infty$



(b) **Experience Replay: Variance Reduction in Stochastic Updates**



(c) **Target Networks: Stabilized Fixed-Point Iteration**

Figure 22: **Visualization of key stabilization mechanisms in deep reinforcement learning.**

but with variance depending on the sample size. Small batches produce high-variance updates:

$$\mathrm{Var}(\widehat{T}V_k) \gg 0, \tag{667}$$

whereas larger replay buffers approximate the expectation more accurately:

$$\widehat{T}V_k \xrightarrow{\text{large batch}} TV_k. \tag{668}$$

Thus, experience replay reduces the variance of the stochastic operator, stabilizing the iterative scheme and making it closer to the deterministic contraction:

$$V_{k+1} = TV_k. \tag{669}$$

The third plot (Figure 22c) illustrates a modified iteration in which the Bellman operator is evaluated using a lagged (target) value function:

$$V_{k+1} = r + \gamma P V_k^{\text{target}}. \tag{670}$$

In contrast to the standard update

$$V_{k+1} = T(V_k), \tag{671}$$

this scheme decouples the evaluation of the operator from the current iterate. Formally, one can interpret this as replacing the nonlinear operator T by a frozen operator:

$$T_k(V) := r + \gamma P V_k^{\text{target}}, \tag{672}$$

which is held fixed over several iterations. This reduces oscillations caused by the feedback loop inherent in repeatedly applying a nonlinear operator with changing parameters. As the target network is periodically updated,

$$V_k^{\text{target}} \to V_k, \tag{673}$$

the method asymptotically recovers the true fixed-point iteration while maintaining intermediate stability.

These three mechanisms correspond to fundamental mathematical structures underlying Bellman equations:

$$\text{Residual:} \quad |TV - V|_\infty, \tag{674}$$

$$\text{Stochastic approximation:} \quad \widehat{T}V \approx TV, \tag{675}$$

$$\text{Stabilized iteration:} \quad V_{k+1} = T_k(V_k). \tag{676}$$

Thus, what appear as algorithmic heuristics in deep reinforcement learning are, in fact, concrete realizations of:

1. residual minimization for fixed-point equations,
2. variance reduction in stochastic operator approximation,
3. and stabilization of nonlinear iterative mappings.

Together, the plots (Figure 22a, 22b, and 22c) demonstrate how modern RL methods approximate and control the convergence to the unique fixed point V^* of a contraction operator in a Banach space.

In summary, Deep RL generalizes classical reinforcement learning to high-dimensional settings by leveraging neural networks as function approximators. Stabilization techniques such as experience replay and target networks are essential to ensure convergence and reliable performance, addressing the challenges introduced by nonlinearity, bootstrapping, and off-policy learning.

9.1 Stabilization Techniques

The development of stabilization techniques in deep reinforcement learning was crucial for enabling neural-network-based agents to learn reliably in high-dimensional environments, and major advances were introduced by Mnih et al. (2015) [31], Lillicrap et al. (2020) [32], Hasselt et al. (2016) [33], and Silver et al. (2016) [34]. Mnih et al. [31] demonstrated that deep neural networks could successfully approximate action-value functions in complex environments through the Deep Q-Network (DQN) framework, introducing key stabilization mechanisms such as experience replay and target networks to reduce correlations in training data and mitigate instabilities arising from nonstationary targets during temporal-difference learning. Lillicrap et al. (2020) [32] extended these ideas to continuous action spaces through Deep Deterministic Policy Gradient (DDPG), combining deterministic policy gradients with replay buffers, target networks, and deep function approximation to achieve stable learning in high-dimensional continuous control tasks. Hasselt et al. (2016) [33] addressed one of the major sources of instability in deep Q-learning by introducing Double DQN, which decouples action selection from target evaluation in the Bellman update, thereby significantly reducing overestimation bias and improving learning stability and performance in deep reinforcement learning systems. A major milestone in stabilization and scalability of deep reinforcement learning was achieved by Silver et al. (2016) [34], where deep neural policy and value networks were successfully integrated with Monte Carlo Tree Search (MCTS) to achieve superhuman performance in the game of Go, demonstrating the effectiveness of stabilized deep reinforcement learning combined with large-scale search and planning mechanisms. Together, these works established the foundational stabilization strategies that enabled deep reinforcement learning algorithms to achieve reliable and scalable performance across discrete and continuous control domains.

Deep reinforcement learning methods, particularly those based on value function approximation such as Deep Q-Networks (DQN), suffer from instability due to the simultaneous use of (i) function approximation, (ii) bootstrapping, and (iii) off-policy data. These factors can lead to divergence or highly oscillatory behavior in the parameter updates. Two fundamental techniques that mitigate these issues are *experience replay* and *target networks*.

Experience Replay. The key idea of experience replay is to break the strong temporal correlations present in sequential data generated by interaction with the environment. Instead of updating the parameters using only the most recent transition $(s_t, a_t, R_{t+1}, s_{t+1})$, one stores past experiences in a replay buffer

$$\mathcal{D} := \{(s_i, a_i, r_i, s_i')\}_{i=1}^{N}. \tag{677}$$

At each iteration, a mini-batch $\{(s_j, a_j, r_j, s_j')\}_{j=1}^{B}$ is sampled (typically uniformly) from $\mathcal{D}$, and the parameters are updated by minimizing the empirical loss

$$\mathcal{L}(\theta) = \frac{1}{B}\sum_{j=1}^{B}\left(Q(s_j, a_j; \theta) - \big(r_j + \gamma \max_{a'} Q(s_j', a'; \theta)\big)\right)^2. \tag{678}$$

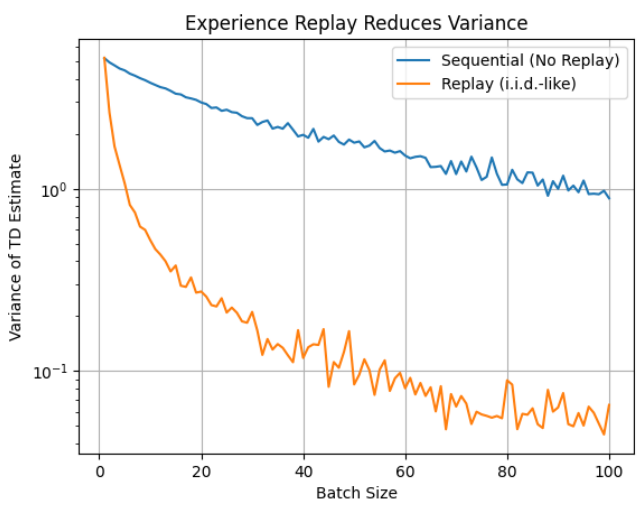
This procedure can be interpreted as replacing the expectation over the on-policy trajectory distribution with an empirical distribution over stored transitions:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mu_{\mathcal{D}}}\left[\left(Q(s, a; \theta) - \big(r + \gamma \max_{a'} Q(s', a'; \theta)\big)\right)^2\right], \tag{679}$$

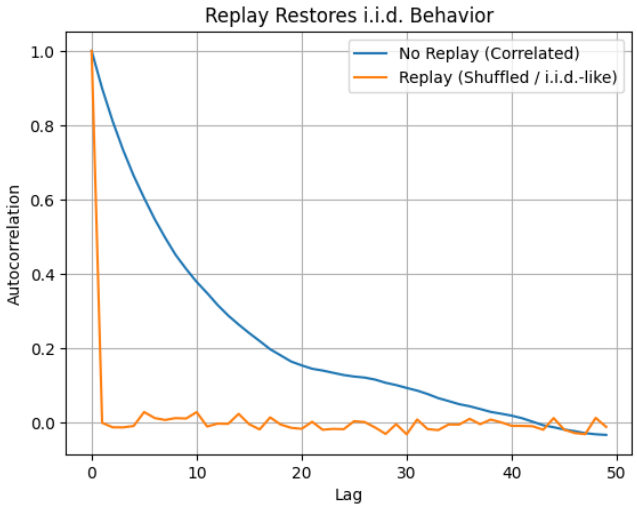
where $\mu_{\mathcal{D}}$ denotes the sampling distribution induced by the replay buffer.

By reusing past experiences, experience replay improves data efficiency and reduces the variance of gradient estimates. Moreover, by sampling transitions approximately independently, it restores the i.i.d. assumption underlying stochastic gradient descent, thereby stabilizing optimization.

Target Networks. A second major source of instability arises from the fact that the target in the temporal-difference update depends on the same parameters θ that are being updated. This creates a moving target problem, which can lead to divergence.



(a) **Variance reduction via experience replay: sampling from a replay buffer approximates i.i.d. transitions and stabilizes gradient estimates.**



(b) **Decorrelation of samples: replay breaks temporal correlations, restoring the i.i.d. assumption underlying stochastic gradient descent.**

Figure 23: **Experience replay improves data efficiency and stabilizes optimization by reducing variance and mitigating temporal correlation in sampled transitions.**

To address this, one introduces a separate set of parameters θ^-, called the *target network*, which are used to compute the target:

$$y_t := R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-). \tag{680}$$

The loss function is then defined as

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left(Q(s_t, a_t; \theta) - \left(R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-)\right)\right)^2\right]. \tag{681}$$

The target parameters θ^- are updated more slowly than the online parameters θ. Two common update schemes are:

- *Periodic update:*
$$\theta^- \leftarrow \theta \quad \text{every } K \text{ steps}, \tag{682}$$
- *Soft (Polyak) update:*
$$\theta^- \leftarrow \tau\theta + (1-\tau)\theta^-, \quad \tau \in (0,1). \tag{683}$$

This decoupling ensures that the target y_t evolves on a slower timescale than the primary network, thereby stabilizing the fixed-point iteration. From an operator-theoretic viewpoint, the update can be interpreted as approximating the solution to

$$Q = TQ, \tag{684}$$

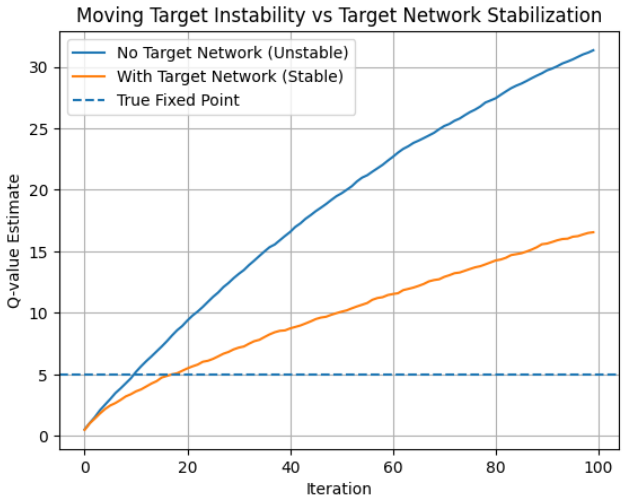
but with a delayed (or lagged) operator:

$$Q_{\theta_{k+1}} \approx \Pi\big(T_{\theta_k^-} Q_{\theta_k}\big), \tag{685}$$

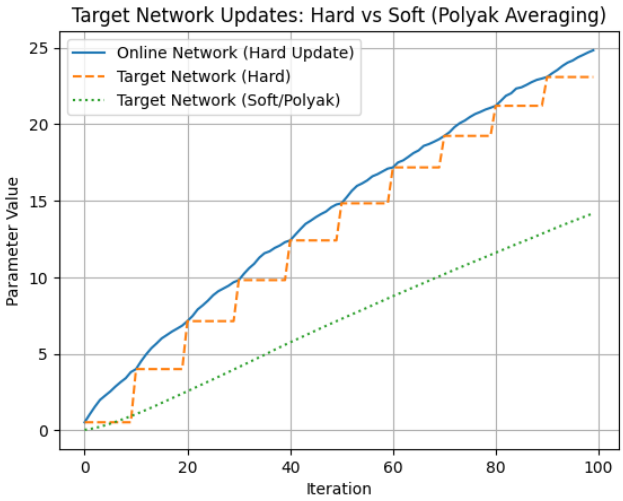
where Π denotes projection onto the function class induced by the neural network.

The two figures (Figure 24a and 24b) provide a rigorous operator-theoretic explanation of how target networks stabilize temporal-difference learning by modifying a nonlinear fixed-point iteration. Recall that the objective is to solve the Bellman equation

$$Q = TQ, \tag{686}$$



(a) **Moving target instability: coupling between $Q(\cdot;\theta)$ and its own bootstrap target leads to oscillations or divergence.**



(b) **Stabilization via target networks: slow update of θ^- yields a quasi-stationary operator, improving convergence.**

Figure 24: **Illustration of target networks in temporal-difference learning. The left plot shows instability caused by a moving target, while the right plot demonstrates stabilization achieved through delayed (lagged) target updates.**

where T is a nonlinear operator (due to the maximization). In standard TD learning without a target network, the update takes the form

$$Q_{k+1} = Q_k + \alpha\big(T(Q_k) - Q_k\big), \tag{687}$$

so that the operator itself depends on the current iterate. As illustrated in Plot 1 (Figure 24a), this induces a moving target problem: the mapping T changes at every step, leading to a non-stationary iteration that can oscillate or diverge. In contrast, introducing a target network replaces the operator by a lagged version, yielding

$$Q_{k+1} = Q_k + \alpha\big(T_{\theta_k^-}(Q_k) - Q_k\big), \tag{688}$$

where θ_k^- evolves on a slower timescale. Thus, the iteration can be interpreted as

$$Q_{\theta_{k+1}} \approx \Pi\big(T_{\theta_k^-} Q_{\theta_k}\big), \tag{689}$$

with Π denoting projection onto the function class. Geometrically as seen in Plot 1 (Figure 24a), this corresponds to replacing a rapidly changing nonlinear operator by an approximately frozen one, thereby restoring stability akin to a contraction mapping.

Plot 2 (Figure 24b) further clarifies how this stabilization is implemented through timescale separation. In the hard update scheme,

$$\theta^- \leftarrow \theta \quad \text{every } K \text{ steps}, \tag{690}$$

the target operator is piecewise constant, producing a sequence of quasi-static fixed-point problems. In the soft (Polyak) update,

$$\theta^- \leftarrow \tau\theta + (1-\tau)\theta^-, \tag{691}$$

the operator evolves smoothly, ensuring gradual adaptation. Both mechanisms enforce that θ_k^- changes more slowly than θ_k, yielding a two-timescale stochastic approximation scheme:

$$\text{fast dynamics: } \theta_k, \qquad \text{slow dynamics: } \theta_k^-. \tag{692}$$

From a deeper perspective, these figures visualize a stabilization of a nonlinear operator iteration. The original problem

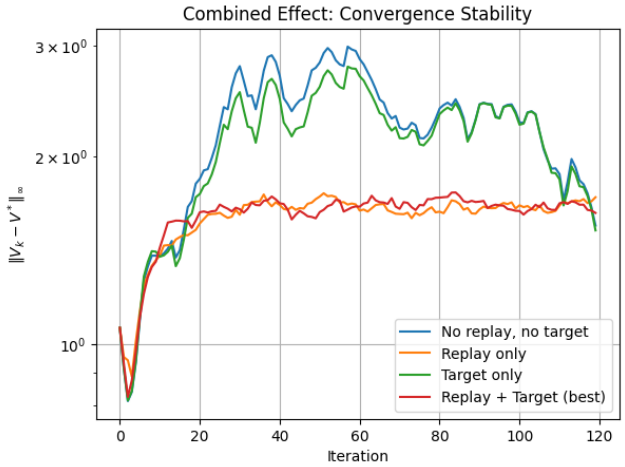
$$Q = TQ \tag{693}$$

is replaced by an approximate iteration with a slowly varying operator,

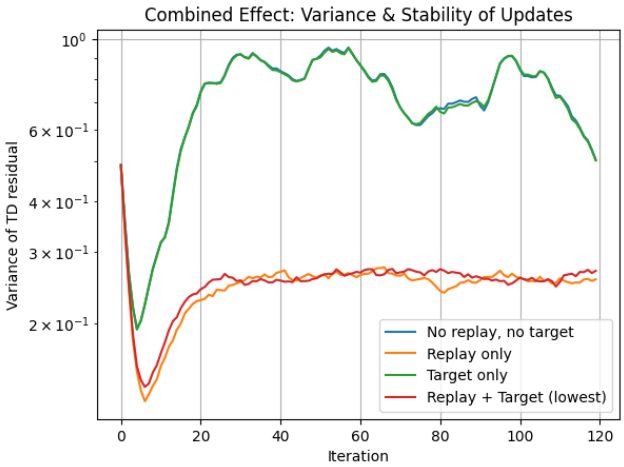
$$Q_{k+1} \approx T_{\text{slow}} Q_k, \tag{694}$$

which behaves much closer to a classical contraction mapping. Consequently, the target network acts as a mechanism for operator freezing, reducing instability caused by feedback, while also implicitly controlling variance and ensuring smoother convergence. In summary, the plots (Figure 24a and 24b) demonstrate that target networks are not merely heuristic modifications, but implement a principled two-timescale stabilization of a nonlinear fixed-point problem, combining ideas from operator theory, stochastic approximation, and dynamical systems.

Combined Effect. When used together, experience replay and target networks significantly improve the stability of deep reinforcement learning algorithms. Experience replay addresses the issue of correlated data and inefficient sample usage, while target networks mitigate the instability caused by rapidly changing bootstrap targets.



(a) **Convergence stability: the error $\|V_k - V^*\|_\infty$ under different settings. The combination of experience replay and target networks yields faster and more stable convergence toward the fixed point.**



(b) **Variance of temporal-difference updates: replay reduces stochastic variance, while target networks mitigate operator drift. Their combination produces the lowest variance and most stable updates.**

Figure 25: **Combined effect of experience replay and target networks. Experience replay improves the approximation $\widehat{T}V_k \approx TV_k$ by reducing variance and restoring approximate independence of samples, while target networks stabilize the iteration by introducing a lagged operator $T_{\theta_k^-}$. Together, they yield a stable and efficient approximation to the fixed-point iteration $V_{k+1} = TV_k$.**

The two figures (Figure 25a and 25b) provide a precise operator-theoretic explanation of how experience replay and target networks jointly stabilize value iteration. In Plot 1 (Figure 25a), you are comparing the convergence behavior of the sequence V_k toward the fixed point V^*, measured by

$$|V_k - V^*|_\infty. \tag{695}$$

In the absence of both mechanisms, the iteration behaves like a stochastic approximation of

$$V_{k+1} \approx V_k + \alpha\big(\widehat{T}_k V_k - V_k\big), \tag{696}$$

where $\widehat{T}_k$ is a noisy, data-dependent operator, leading to oscillatory or unstable behavior. When experience replay alone is used, the operator is better approximated in expectation,

$$\widehat{T}_k V_k \approx TV_k, \tag{697}$$

so that variance is reduced, but the operator itself still changes across iterations, resulting in improved yet imperfect convergence. With target networks alone, the update becomes

$$V_{k+1} \approx V_k + \alpha\big(T * \theta_k^- V_k - V_k\big), \tag{698}$$

which stabilizes the operator but retains stochastic noise, producing a stable yet noisy trajectory. When both mechanisms are combined, the iteration effectively becomes

$$V_{k+1} \approx \Pi\big(\widehat{T}_{\theta_k^-} V_k\big), \tag{699}$$

where Π denotes projection onto the function class. As seen in Plot 1 (Figure 25a), this yields fast and stable convergence, closely resembling a contraction mapping toward V^*.

Plot 2 (Figure 25b) complements this by examining the variability of the temporal-difference updates through

$$\mathrm{Var}\left(TV_k - V_k\right). \tag{700}$$

Experience replay reduces the variance of the stochastic operator,

$$\mathrm{Var}(\widehat{T}_k V_k) \ll \mathrm{Var}(\text{sequential samples}), \tag{701}$$

by approximating independent sampling from the underlying distribution. Target networks, on the other hand, reduce the variability arising from operator drift, effectively replacing a rapidly changing mapping T_k with a slowly evolving one $T_{\theta_k^-}$. The combination therefore yields

$$\text{low variance} + \text{slow operator evolution}, \tag{702}$$

as clearly reflected in the reduced and smoother curves in Plot 2 (Figure 25b).

From a deeper perspective, these figures (Figure 25a and 25b) illustrate that the two techniques address distinct failure modes of stochastic fixed-point iteration. Experience replay ensures that

$$\widehat{T}_k V_k \approx TV_k, \tag{703}$$

thereby reducing stochastic noise and restoring the i.i.d. assumption underlying stochastic approximation. Target networks ensure that

$$T_{\theta_k^-} \approx T_{\text{fixed}}, \tag{704}$$

removing the instability caused by a moving operator. Together, they produce an effective iteration of the form

$$V_{k+1} \approx \Pi\big(T_{\text{stable}} V_k\big), \tag{705}$$

which behaves like a classical contraction mapping. Consequently, the combined method exhibits contraction-like dynamics, reduced variance, and robust convergence.

The key insight, as demonstrated by both plots (Figure 25a and 25b), is that stability in deep reinforcement learning emerges from the interplay

$$\text{Stability} = \underbrace{\text{low variance}}_{\text{experience replay}} + \underbrace{\text{slow operator drift}}_{\text{target network}}, \tag{706}$$

revealing that these techniques are not merely heuristic, but implement a principled variance reduction and operator stabilization framework for nonlinear fixed-point problems.

In summary, these techniques enable the practical application of value-based reinforcement learning with nonlinear function approximators by approximating the contraction properties of the Bellman operator in a stochastic and high-dimensional setting.

9.2 Double DQN

Double Q-learning and Double Deep Q-Networks (Double DQN) were developed to address the overestimation bias inherent in standard Q-learning methods, and their theoretical and algorithmic foundations were established by Hasselt (2010) [35], Hasselt et. al. (2016) [33], and Sutton and Barto (1998) [1]. Hasselt (2010) [35] introduced Double Q-learning as a modification of traditional Q-learning in which action selection and action evaluation are decoupled across separate estimators, thereby significantly reducing the positive bias caused by maximization over noisy value estimates in stochastic environments. Hasselt et. al. (2016) [33] later extended this idea to deep reinforcement learning through the Double DQN framework, combining deep neural-network-based value approximation with the Double Q-learning principle to improve stability, reduce overoptimistic value estimation, and enhance performance in high-dimensional environments such as Atari games. Sutton and Barto (1998) [1] discussed Double Q-learning within the broader reinforcement learning framework, emphasizing the statistical origins of overestimation bias in temporal-difference learning and illustrating how decoupled action evaluation mechanisms improve the reliability and convergence behavior of value-based reinforcement learning algorithms. Together, these works established Double Q-learning and Double DQN as important stabilization techniques in reinforcement learning, providing mathematically grounded approaches for improving value estimation accuracy and learning stability in both tabular and deep reinforcement learning settings.

A well-known limitation of the standard Deep Q-Network (DQN) algorithm is the tendency to overestimate action values due to the maximization step in the Bellman target. In classical DQN, the target for updating the action-value function is given by

$$y_t^{\mathrm{DQN}} = R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-), \tag{707}$$

where θ^- denotes the parameters of the target network. The maximization operator introduces a positive bias because the same function $Q(\cdot;\theta^-)$ is used both to select and to evaluate the maximizing action. Formally, for a noisy estimate $\hat{Q}$ of the true value function,

$$\mathbb{E}\left[\max_a \hat{Q}(s,a)\right] \geq \max_a \mathbb{E}\left[\hat{Q}(s,a)\right], \tag{708}$$

which leads to systematic overestimation.

Double DQN addresses this issue by decoupling the action selection and action evaluation steps. Specifically, the action is selected using the online network parameters θ, while its value is evaluated using the target network parameters θ^-. The modified target becomes

$$y_t^{\mathrm{DDQN}} = R_{t+1} + \gamma\, Q\big(s_{t+1}, \mathrm{argmax}_{a'}\, Q(s_{t+1}, a'; \theta), \theta^-\big)\,. \tag{709}$$

This separation reduces overestimation bias by ensuring that the maximization is performed with respect to one estimator, while evaluation is performed with another. Intuitively, if the estimation errors of the two networks are not perfectly correlated, the bias introduced by the max operator is significantly mitigated.

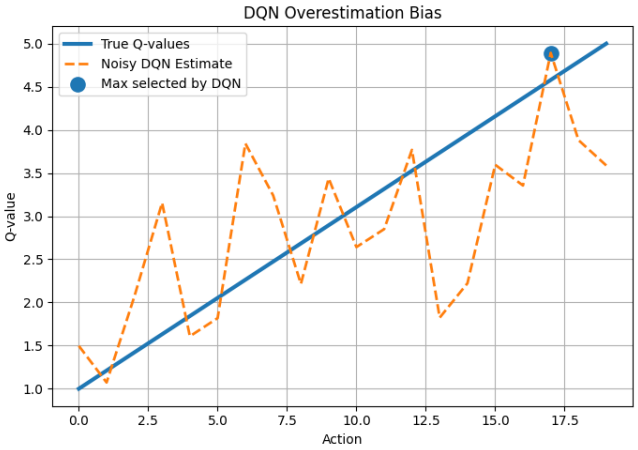
The update rule for the parameters θ is then given by stochastic gradient descent on the squared temporal-difference error:

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta \left(Q(s_t, a_t; \theta_t) - y_t^{\mathrm{DDQN}}\right)^2. \tag{710}$$

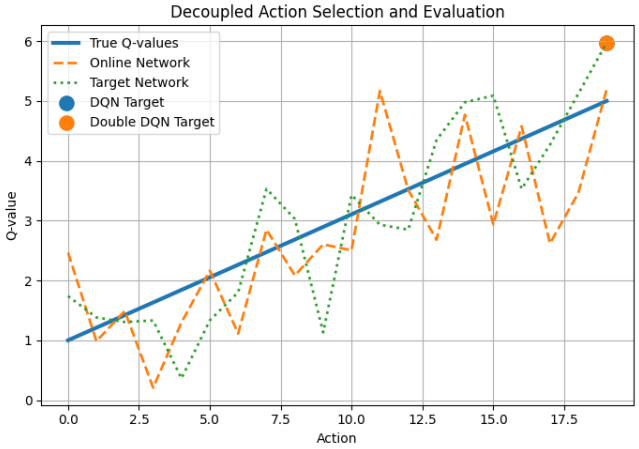
From an operator-theoretic perspective, Double DQN can be interpreted as approximating a modified Bellman operator:

$$(T^{\mathrm{DDQN}}Q)(s,a) = R(s,a) + \gamma\, \mathbb{E}_{s'}\left[Q\big(s', \mathrm{argmax}_{a'}\, Q(s', a'; \theta), \theta^-\big)\right], \tag{711}$$

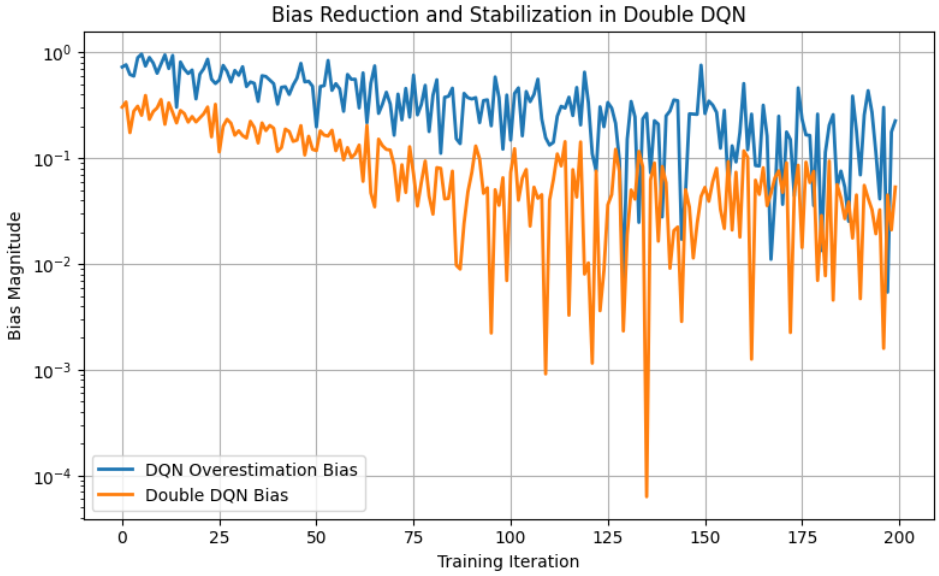
which reduces the upward bias inherent in the standard Bellman optimality operator when approximated with function approximators.



(a) **Overestimation bias in standard DQN. The maximization step $\max_a \hat{Q}(s,a)$ selects positively biased noisy estimates, leading to $\mathbb{E}\left[\max_a \hat{Q}(s,a)\right] \geq \max_a \mathbb{E}[\hat{Q}(s,a)]$.**



(b) **Decoupling of action selection and evaluation in Double DQN. The online network selects the maximizing action, while the target network evaluates it: $y_t^{\text{DDQN}} = R_{t+1} + \gamma Q(s_{t+1}, \text{argmax}_{a'} Q(s_{t+1}, a'; \theta), \theta^-)$. This reduces maximization-induced overestimation.**



(c) **Bias reduction and stabilization in Double DQN.The magnitude of overestimation bias decreases more rapidly and remains more stable compared to standard DQN, illustrating improved behavior of the approximate fixed-point iteration associated with the modified Bellman operator $(T^{\text{DDQN}}Q)(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[Q(s', \text{argmax}_{a'} Q(s', a'; \theta), \theta^-)\right]$.**

Figure 26: **Illustration of Double DQN and its reduction of overestimation bias. The first figure shows the upward bias caused by maximizing noisy value estimates in standard DQN. The second figure illustrates the decoupling of action selection and evaluation in Double DQN. The third figure demonstrates the resulting stabilization and reduction of estimation bias during training.**

In practice, Double DQN retains the stabilization mechanisms of DQN, including experience replay and target networks, but modifies only the target computation. Empirically, this leads to more accurate value estimates, improved stability, and better performance across a wide range of environments.

The three figures (Figure 26a, 26b, and 26c) provide a rigorous visualization of the operator-theoretic motivation behind Double DQN and its reduction of overestimation bias in deep reinforcement learn-

ing. In standard DQN, the Bellman target is computed using the same estimator both to select and to evaluate the maximizing action:

$$y_t^{\mathrm{DQN}} = R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-). \tag{712}$$

As illustrated in Plot 1 (Figure 26a), this creates a systematic upward bias due to noisy maximization. If $\hat{Q}(s,a)$ is a noisy estimator of the true action value, then

$$\mathbb{E}!\left[\max_a \hat{Q}(s,a)\right] \geq \max_a \mathbb{E}[\hat{Q}(s,a)]. \tag{713}$$

The maximization operator preferentially selects actions whose estimation errors happen to be positive, producing persistent overestimation. Geometrically, Plot 1 (Figure 26a) shows that the selected action often lies above the underlying true value function, even when the estimator is unbiased on average.

Plot 2 (Figure 26b) illustrates how Double DQN decouples action selection and action evaluation in order to reduce this bias. Instead of using the same network for both operations, Double DQN computes the target as

$$y_t^{\mathrm{DDQN}} = R_{t+1} + \gamma Q\big(s_{t+1}, \mathrm{argmax}_{a'}\, Q(s_{t+1}, a'; \theta), \theta^-\big)\,. \tag{714}$$

Here, the online network parameters θ are used to determine the maximizing action,

$$a^* = \mathrm{argmax}_{a'}\, Q(s_{t+1}, a'; \theta), \tag{715}$$

while the target network parameters θ^- are used to evaluate that action. The figure visually demonstrates this separation: the online network performs selection, whereas the target network performs evaluation. Since the estimation errors of the two networks are not perfectly correlated, the upward bias introduced by the max operator is significantly reduced.

From an operator-theoretic viewpoint, Double DQN replaces the classical Bellman operator by a modified operator,

$$(T^{\mathrm{DDQN}}Q)(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[Q\big(s', \mathrm{argmax} * a' Q(s', a'; \theta), \theta^-\big)\right]. \tag{716}$$

This modified operator preserves the structure of the Bellman equation while mitigating the bias arising from nonlinear maximization with noisy approximators.

Finally, Plot 3 (Figure 26c) illustrates the stabilization of estimation bias during training. The curves compare the magnitude of overestimation error in DQN and Double DQN across iterations. One observes that standard DQN maintains a substantially larger bias, whereas Double DQN produces smaller and more stable value estimates. The approximately linear decay on the logarithmic scale reflects improved stability of the underlying approximate fixed-point iteration,

$$Q_{k+1} \approx \Pi\big(T^{\mathrm{DDQN}} Q_k\big), \tag{717}$$

where Π denotes projection onto the neural-network function class. Because the modified operator reduces systematic overestimation, the resulting dynamics behave more like a stable contraction mapping.

Taken together, the three figures (Figure 26a, 26b, and 26c) demonstrate that Double DQN is not merely a heuristic modification, but a principled correction of the maximization bias inherent in approximate Bellman operators. By decoupling action selection from evaluation, Double DQN yields more accurate value estimates, reduced bias accumulation, and significantly more stable convergence behavior.

In summary, Double DQN provides a principled correction to the overestimation bias in DQN by decoupling action selection and evaluation, thereby yielding a more reliable approximation of the optimal action-value function.

9.3 Dueling Architectures

The dueling network architecture is a structural modification of the standard Deep Q-Network (DQN) designed to improve the efficiency of value estimation, particularly in states where the choice of action has limited impact on the outcome. The key idea is to decompose the action-value function into two separate components: a *state-value function* and an *advantage function*.

Dueling network architectures were introduced as an important advancement in deep reinforcement learning to improve the efficiency and stability of value-function approximation, building upon the foundations established by Mnih et al. (2015) [31] and later refined alongside Double DQN methods by Hasselt et. al. (2016) [33]. Mnih et al. (2015) [31] demonstrated the effectiveness of deep neural networks for approximating action-value functions through the Deep Q-Network (DQN) framework, combining convolutional neural networks with experience replay and target networks to achieve human-level control in high-dimensional environments. Hasselt et al. (2016) [33] further improved deep Q-learning through the Double DQN approach, reducing overestimation bias by decoupling action selection from action evaluation in the Bellman update and thereby enhancing training stability and performance. Building upon these developments, Wang et al. (2016) [36] introduced the dueling network architecture, in which separate neural network streams are used to estimate the state-value function and the action-dependent advantage function before combining them into a unified action-value estimate, enabling more efficient learning of state representations even when the relative importance of actions is small. Dueling Network Architectures for Deep Reinforcement Learning Together, these works established the foundations of modern deep value-based reinforcement learning architectures, demonstrating how structural modifications to neural network representations can significantly improve learning efficiency, stability, and performance in complex decision-making environments.

Formally, for a given policy, the action-value function can be expressed as

$$Q(s,a) = V(s) + A(s,a), \tag{718}$$

where $V(s)$ represents the value of being in state s, and $A(s,a)$ represents the advantage of taking action a relative to other actions in that state. However, this decomposition is not unique, since one can shift V and A by a constant without changing Q. To ensure identifiability, the dueling architecture imposes a normalization constraint on the advantage function.

In practice, the network is parameterized as follows: a shared feature representation $\phi(s;\theta)$ is first computed, after which two separate streams are used:

$$V(s;\theta_v), \quad A(s,a;\theta_a). \tag{719}$$

These are then combined to produce the action-value estimate. A common aggregation is

$$Q(s,a;\theta) = V(s;\theta_v) + \left(A(s,a;\theta_a) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s,a';\theta_a) \right), \tag{720}$$

which enforces that the mean advantage across actions is zero:

$$\sum_{a} A(s,a) = 0. \tag{721}$$

An alternative formulation uses the maximum instead of the mean:

$$Q(s,a;\theta) = V(s;\theta_v) + \left(A(s,a;\theta_a) - \max_{a'} A(s,a';\theta_a) \right), \tag{722}$$

though the mean-based normalization is more commonly used due to better empirical stability.

The learning procedure remains similar to standard DQN. The parameters $\theta = (\theta_v, \theta_a)$ are updated by minimizing the temporal-difference loss:

$$\mathcal{L}(\theta) = \mathbb{E}\left[(Q(s_t,a_t;\theta) - y_t)^2 \right], \tag{723}$$

where the target y_t is typically computed using a target network:

$$y_t = R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-). \tag{724}$$

The advantage of this architecture becomes apparent in environments where many actions yield similar outcomes. In such cases, learning $V(s)$ directly allows the network to generalize across actions more effectively, while the advantage stream focuses on learning the relative differences between actions. This leads to improved sample efficiency and more stable learning dynamics.

From a functional approximation perspective, the dueling architecture can be viewed as imposing a structured parameterization on $Q(s, a)$, separating global state information from action-specific deviations:

$$Q(s, a; \theta) \in \left\{ V(s) + A(s, a) : \sum_a A(s, a) = 0 \right\}. \tag{725}$$

This decomposition reduces redundancy in representation and enables faster propagation of value information across actions.

The three figures (Figure 27a, 27b, and 27c) provide a geometric and operator-theoretic interpretation of the dueling network architecture, which decomposes the action-value function into separate state-value and advantage components. As illustrated in Figure 1 (Figure 27a), the action-value function is represented as

$$Q(s, a) = V(s) + A(s, a), \tag{726}$$

where $V(s)$ captures the overall quality of the state and $A(s, a)$ measures the relative benefit of choosing action a in that state. Since this decomposition is not unique, the dueling architecture imposes the normalization constraint

$$\sum_a A(s, a) = 0, \tag{727}$$

which leads to the mean-centered aggregation

$$Q(s, a) = V(s) + \left(A(s, a) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a') \right). \tag{728}$$

Figure 1 (Figure 27a) visually demonstrates this decomposition: the horizontal component represents the shared state-value $V(s)$, while the deviations across actions correspond to the normalized advantage values. Geometrically, the advantage stream acts as a fluctuation around the global state baseline.

Figure 2 (Figure 27b) illustrates the principal motivation behind the dueling architecture: improved generalization in states where many actions have similar consequences. In standard DQN, each action-value estimate must be learned independently, even when the state itself is highly informative. By explicitly separating

$$Q(s, a) = V(s) + A(s, a), \tag{729}$$

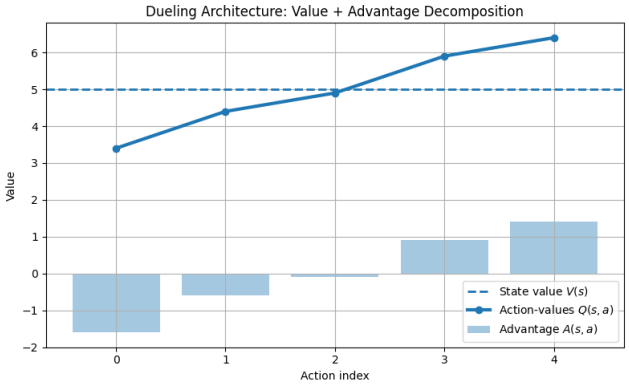
the network can propagate state-value information across all actions simultaneously, while the advantage stream focuses only on learning relative action differences. Consequently, when actions are nearly equivalent,

$$A(s, a) \approx 0, \tag{730}$$

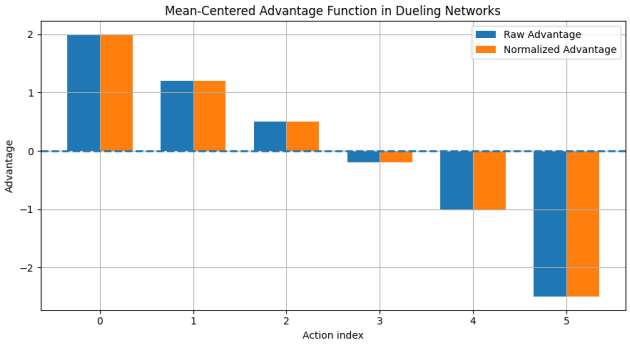
the network can still accurately estimate the value of the state through (V(s)). Figure 2 (Figure 27b) shows that this structured parameterization enables more efficient learning and faster propagation of value information across actions, especially in action-redundant environments.

Finally, Figure 3 (Figure 27c) demonstrates the effect of this decomposition on training dynamics and convergence stability. The parameters are learned by minimizing the temporal-difference loss

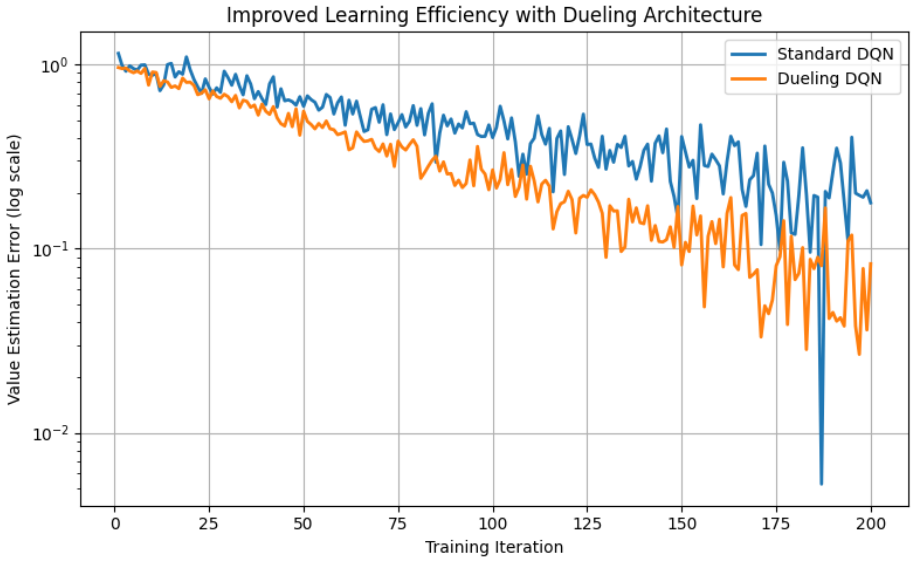
$$\mathcal{L}(\theta) = \mathbb{E}\left[(Q(s_t, a_t; \theta) - y_t)^2 \right], \tag{731}$$



(a) **Decomposition of the action-value function into state-value and advantage components. The figure illustrates $Q(s,a) = V(s) + A(s,a)$, together with the normalized aggregation $Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a')\right)$, which enforces the identifiability condition $\sum_a A(s,a) = 0$.**



(b) **Generalization advantage of the dueling architecture. The shared state-value stream propagates common information across actions, while the advantage stream captures only relative action differences. This structured representation improves sample efficiency in environments where many actions have similar values.**



(c) **Training dynamics of standard DQN versus dueling DQN. The dueling architecture exhibits smoother convergence and reduced variance in temporal-difference updates due to the separation of global state information and action-specific deviations. The resulting approximation space is $Q(s,a;\theta) \in \{V(s) + A(s,a) : \sum_a A(s,a) = 0\}$.**

Figure 27: **Illustration of the dueling network architecture. The first row visualizes the decomposition of the action-value function into value and advantage components and the resulting improvement in cross-action generalization. The second row demonstrates the enhanced stability and learning efficiency obtained through the structured parameterization of the action-value function.**

where the target is typically computed using a target network:

$$y_t = R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-). \tag{732}$$

The figure shows that dueling architectures exhibit smoother and faster convergence than standard DQN. From a functional approximation perspective, the architecture restricts the approximation space to

$$Q(s, a; \theta) \in \{V(s) + A(s, a) : \sum_a A(s, a) = 0\} \tag{733}$$

thereby reducing redundancy in the representation of action values. This structured decomposition lowers variance in temporal-difference updates and improves sample efficiency, producing more stable learning trajectories.

Taken together, the three figures (Figure 27a, 27b, and 27c) show that the dueling architecture is not merely a neural-network modification, but a principled decomposition of the Bellman approximation problem into a global state component and local action-specific deviations. The architecture therefore combines ideas from value decomposition, structured functional approximation, and operator stabilization to achieve more efficient and robust reinforcement learning.

In summary, dueling architectures enhance DQN by explicitly modeling the state-value and advantage components, leading to improved learning efficiency, particularly in high-dimensional or action-redundant environments.

9.4 Prioritized Experience Replay

Prioritized experience replay was developed as a major enhancement to deep reinforcement learning algorithms by improving the efficiency with which agents learn from past experiences, building upon the foundational ideas introduced by Mnih et al. (2015) [31] and conceptually grounded in the reinforcement learning framework presented by Sutton and Barto (1998) [1]. Mnih et al. (2015) [31] introduced the Deep Q-Network (DQN) architecture, where experience replay buffers were used to decorrelate training samples and stabilize neural-network-based temporal-difference learning by randomly sampling past transitions during training. Sutton and Barto (1998) [1] discussed experience replay and temporal-difference learning within the broader context of reinforcement learning, emphasizing the importance of repeated sampling, bootstrapping, and efficient reuse of experience in improving value estimation and policy learning. Building on these ideas, Schaul et al. (2015) [37] introduced prioritized experience replay, in which transitions are sampled according to measures of learning significance such as temporal-difference error magnitude, allowing the agent to focus computational effort on more informative experiences while employing importance-sampling corrections to reduce sampling bias. Together, these works established experience replay as a central mechanism in deep reinforcement learning and demonstrated how prioritization strategies can substantially improve sample efficiency, convergence speed, and learning performance in complex stochastic environments.

Prioritized experience replay is an enhancement of the standard experience replay mechanism used in Deep Q-Network (DQN), designed to improve sample efficiency by biasing the sampling process toward more informative transitions. In standard replay buffers, transitions $(s_t, a_t, r_{t+1}, s_{t+1})$ are sampled uniformly from a dataset

$$\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^N, \tag{734}$$

which treats all experiences as equally important. However, in practice, some transitions—particularly those with large temporal-difference (TD) errors—carry more learning signal.

To formalize this, define the TD error for a transition i as

$$\delta_i = r_i + \gamma \max_{a'} Q(s_i', a'; \theta^-) - Q(s_i, a_i; \theta). \tag{735}$$

Prioritized replay assigns a priority $p_i > 0$ to each transition, typically based on the magnitude of the TD error:

$$p_i = |\delta_i| + \varepsilon, \tag{736}$$

where $\varepsilon > 0$ ensures non-zero probability for all samples. Sampling is then performed according to a probability distribution

$$P(i) = \frac{p_i^\alpha}{\sum_{j=1}^N p_j^\alpha}, \tag{737}$$

where $\alpha \in [0, 1]$ controls the degree of prioritization. When $\alpha = 0$, the scheme reduces to uniform sampling; as $\alpha \to 1$, sampling becomes increasingly focused on high-priority transitions. Because this introduces a sampling bias, importance sampling weights are used to correct the resulting distributional mismatch. For each sampled transition i, define

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)} \right)^\beta, \tag{738}$$

where $\beta \in [0, 1]$ controls the strength of correction. These weights are typically normalized:

$$\tilde{w}_i = \frac{w_i}{\max_j w_j}, \tag{739}$$

to stabilize training.

The loss function is then modified as

$$\mathcal{L}(\theta) = \mathbb{E}_{i \sim P} \left[\tilde{w}_i \left(Q(s_i, a_i; \theta) - y_i \right)^2 \right], \tag{740}$$

where the target y_i is given by

$$y_i = r_i + \gamma \max_{a'} Q(s_i', a'; \theta^-). \tag{741}$$

Two main variants of prioritized replay are commonly used. In *proportional prioritization*, priorities are directly proportional to $|\delta_i|$. In *rank-based prioritization*, transitions are sorted by $|\delta_i|$, and probabilities are assigned based on rank:

$$P(i) \propto \frac{1}{\text{rank}(i)}. \tag{742}$$

The latter is more robust to outliers in TD errors. From an optimization perspective, prioritized replay accelerates convergence by focusing updates on transitions with high Bellman residual:

$$\mathcal{R}(Q) = \| TQ - Q \|_\infty, \tag{743}$$

since large $|\delta_i|$ indicates regions where the current value function is inaccurate. However, excessive prioritization can reduce diversity in the training data and lead to overfitting or instability, which is why annealing β toward 1 over time is commonly employed.

The three figures (Figure 28a, 28b, and 28c) provide a rigorous visualization of the mathematical principles underlying *prioritized experience replay* (PER), which improves the efficiency of Deep Q-Network training by allocating greater sampling probability to transitions carrying larger learning signal. In standard replay buffers, transitions are sampled uniformly from a dataset

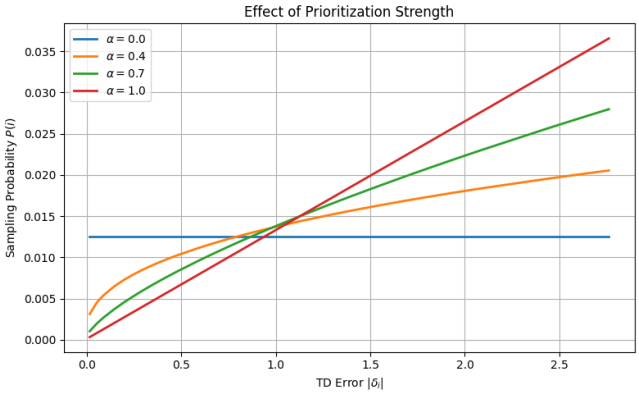
$$\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^N, \tag{744}$$

so that each transition is treated as equally informative. However, transitions with large temporal-difference (TD) error

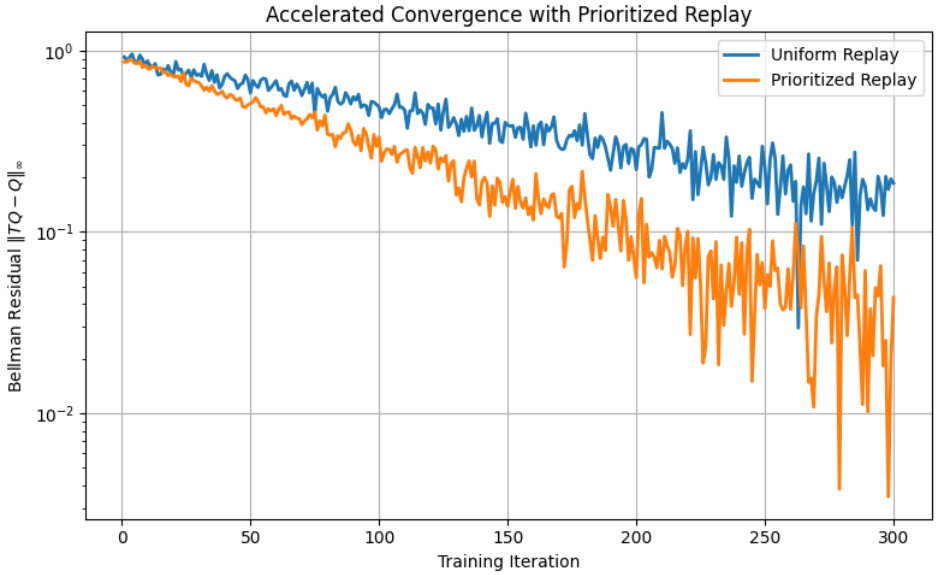
$$\delta_i = r_i + \gamma \max_{a'} Q(s_i', a'; \theta^-) - Q(s_i, a_i; \theta) \tag{745}$$

typically correspond to regions where the current value function approximation is inaccurate and therefore contain stronger optimization signal.

(a) **Comparison between uniform replay and prioritized replay sampling distributions. Under prioritized replay, transitions with larger temporal-difference errors receive higher sampling probability:** $P(i) = \frac{p_i^\alpha}{\sum_{j=1}^N p_j^\alpha}, \qquad p_i = |\delta_i| + \varepsilon.$ **The figure illustrates how prioritization allocates more computational effort to informative transitions.**



(b) **Effect of the prioritization parameter α on the replay distribution. When $\alpha = 0$, sampling reduces to uniform replay, while increasing α increasingly concentrates probability mass on transitions with large TD error:** $\delta_i = r_i + \gamma \max_{a'} Q(s_i', a'; \theta^-) - Q(s_i, a_i; \theta)$.



(c) **Convergence comparison between uniform replay and prioritized replay. Prioritized replay accelerates the reduction of the Bellman residual $\mathcal{R}(Q) = \|TQ - Q\|_\infty$, by focusing updates on transitions with high TD error. The approximately linear decay on the logarithmic scale reflects faster contraction-like convergence behavior.**

Figure 28: **Illustration of prioritized experience replay. The first row visualizes how replay probabilities are redistributed toward informative transitions and how the prioritization parameter controls this behavior. The second row demonstrates the resulting acceleration in Bellman residual reduction and improved learning efficiency compared to uniform replay.**

Figure 28a illustrates the contrast between uniform replay and prioritized replay sampling distributions. In prioritized replay, each transition is assigned a priority

$$p_i = |\delta_i| + \varepsilon, \tag{746}$$

where $\varepsilon > 0$ ensures nonzero sampling probability. Sampling is then performed according to

$$P(i) = \frac{p_i^\alpha}{\sum_{j=1}^N p_j^\alpha}, \tag{747}$$

where $\alpha \in [0, 1]$ controls the strength of prioritization. The figure shows that transitions with larger TD error receive substantially higher sampling probability, thereby concentrating computational effort on the most informative experiences.

Figure 28b visualizes the role of the prioritization parameter α. When

$$\alpha = 0, \tag{748}$$

the distribution reduces to uniform replay:

$$P(i) = \frac{1}{N}. \tag{749}$$

As α increases toward 1, the replay distribution becomes increasingly concentrated around transitions with large TD error. Geometrically, the figure demonstrates how PER interpolates continuously between unbiased uniform sampling and aggressively focused importance-driven sampling. This reflects the trade-off between exploration of the replay buffer and exploitation of high-error transitions.

Because prioritized sampling introduces bias, importance-sampling corrections are required. For each sampled transition, the correction weight is

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)} \right)^\beta, \tag{750}$$

where $\beta \in [0, 1]$ controls the strength of bias correction. These weights are typically normalized:

$$\tilde{w}_i = \frac{w_i}{\max_j w_j}, \tag{751}$$

leading to the weighted loss function

$$\mathcal{L}(\theta) = \mathbb{E}_{i \sim P} \left[\tilde{w}_i \left(Q(s_i, a_i; \theta) - y_i \right)^2 \right]. \tag{752}$$

Finally, Figure 28c demonstrates the practical consequence of prioritized replay on optimization dynamics. The plot compares the decay of the Bellman residual

$$\mathcal{R}(Q) = \|TQ - Q\|_\infty \tag{753}$$

under uniform replay and prioritized replay. The approximately linear decay on the logarithmic scale reflects contraction-like convergence behavior. Prioritized replay exhibits significantly faster residual reduction because updates are concentrated in regions where the Bellman equation is violated most strongly. In operator-theoretic terms, PER adaptively redistributes stochastic approximation effort toward regions with large fixed-point defect:

$$|TQ - Q|. \tag{754}$$

Taken together, the three figures (Figure 28a, 28b, and 28c) show that prioritized experience replay is not merely a heuristic sampling strategy, but a principled mechanism for adaptive stochastic approximation. By emphasizing transitions with large Bellman residual while correcting sampling bias through importance weighting, PER improves sample efficiency, accelerates convergence, and allocates computational resources to the most informative regions of the value function approximation space.

In summary, prioritized experience replay improves the efficiency of DQN by allocating computational resources to the most informative experiences, while using importance sampling to maintain consistency with the underlying objective. This results in faster learning and better utilization of collected data.

10 Exploration-Exploitation Trade-off

A central challenge in reinforcement learning is the exploration-exploitation trade-off: the agent must balance exploiting current knowledge to maximize immediate reward with exploring uncertain actions to improve future performance. Formally, let $Q_t(s, a)$ denote the current estimate of the action-value function. Exploitation corresponds to selecting actions according to

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} Q_t(s_t, a), \tag{755}$$

while exploration requires selecting suboptimal or uncertain actions to refine estimates of Q^*.

This trade-off is inherently sequential and stochastic: insufficient exploration may lead to convergence to suboptimal policies, while excessive exploration degrades cumulative reward. The design of exploration strategies can be viewed as constructing a sampling distribution $\pi_t(a|s)$ that balances these competing objectives.

ϵ-Greedy Exploration. One of the simplest and most widely used strategies is the ϵ-greedy policy. At each time step t, the agent selects an action according to

$$\pi_t(a|s) = \begin{cases} 1 - \epsilon_t + \dfrac{\epsilon_t}{|\mathcal{A}|}, & \text{if } a \in \operatorname{argmax}_{a'} Q_t(s, a'), \\ \dfrac{\epsilon_t}{|\mathcal{A}|}, & \text{otherwise,} \end{cases} \tag{756}$$

where $\epsilon_t \in [0, 1]$ is the exploration parameter. Equivalently, with probability $1 - \epsilon_t$, the agent exploits by choosing a greedy action, and with probability ϵ_t, it explores uniformly at random:

$$a_t = \begin{cases} \operatorname{argmax}_a Q_t(s_t, a), & \text{with probability } 1 - \epsilon_t, \\ \text{Uniform}(\mathcal{A}), & \text{with probability } \epsilon_t. \end{cases} \tag{757}$$

To ensure convergence in stochastic approximation settings, ϵ_t is often decayed over time, e.g.,

$$\epsilon_t = \frac{c}{t}, \quad \text{or} \quad \epsilon_t = \epsilon_0 \cdot \beta^t, \quad \beta \in (0, 1). \tag{758}$$

This guarantees sufficient exploration (infinite visits to all actions) while asymptotically favoring exploitation.

Softmax (Boltzmann) Policies. A more refined exploration strategy assigns probabilities to actions according to a smooth transformation of their estimated values. The softmax (or Boltzmann) policy is defined as

$$\pi_t(a|s) = \frac{\exp\left(Q_t(s, a)/\tau_t\right)}{\sum_{a' \in \mathcal{A}} \exp\left(Q_t(s, a')/\tau_t\right)}, \tag{759}$$

where $\tau_t > 0$ is the temperature parameter. The temperature τ_t controls the degree of exploration:

$$\lim_{\tau_t \to 0} \pi_t(a|s) = \mathbf{1}\{a \in \operatorname{argmax}_{a'} Q_t(s, a')\}, \quad \lim_{\tau_t \to \infty} \pi_t(a|s) = \frac{1}{|\mathcal{A}|}. \tag{760}$$

Thus, small τ_t yields near-greedy behavior, while large τ_t promotes uniform exploration. Annealing schedules such as $\tau_t \to 0$ are commonly used to transition from exploration to exploitation.

Softmax policies can be interpreted as arising from entropy-regularized optimization:

$$\pi_t(\cdot|s) = \operatorname{argmax}_\pi \left\{ \sum_a \pi(a|s) Q_t(s, a) + \tau_t \, \mathcal{H}(\pi(\cdot|s)) \right\}, \tag{761}$$

where

$$\mathcal{H}(\pi) = -\sum_a \pi(a) \log \pi(a) \tag{762}$$

is the Shannon entropy. This formulation highlights the role of entropy in encouraging exploration.

Upper Confidence Bound (UCB). The Upper Confidence Bound (UCB) approach incorporates uncertainty estimates into action selection, favoring actions with high estimated value and high uncertainty. In the multi-armed bandit setting, the UCB rule selects

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} \left[\hat{Q}_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right], \tag{763}$$

where:

- $\hat{Q}_t(a)$ is the empirical mean reward of action a,
- $N_t(a)$ is the number of times action a has been selected up to time t,
- $c > 0$ is a parameter controlling exploration.

The second term represents a confidence interval width derived from concentration inequalities. Actions with fewer samples (small $N_t(a)$) receive larger bonuses, encouraging exploration.

In Markov Decision Processes, UCB ideas are extended to state-action pairs:

$$a_t \in \operatorname{argmax}_a \left[Q_t(s_t, a) + c \sqrt{\frac{\log t}{N_t(s_t, a)}} \right]. \tag{764}$$

UCB methods enjoy strong theoretical guarantees, including logarithmic regret bounds in bandit problems:

$$\text{Regret}(T) = \mathcal{O}(\log T). \tag{765}$$

Unified Perspective. All three methods can be viewed as constructing a policy $\pi_t(a|s)$ that balances estimated value and uncertainty. The ϵ-greedy method injects uniform randomness, softmax policies use a smooth probabilistic weighting, and UCB methods explicitly incorporate confidence bounds.

More generally, one may define exploration strategies via regularized optimization:

$$\pi_t(\cdot|s) = \operatorname{argmax}_\pi \left\{ \sum_a \pi(a|s) Q_t(s,a) + \lambda_t \, \Omega(\pi(\cdot|s)) \right\}, \tag{766}$$

where Ω is a regularization functional (e.g., entropy or confidence penalties). This perspective connects exploration strategies to convex analysis and information theory.

The figures (Figure 29a, 29b, 29c, and 29d) provide a mathematical visualization of the exploration-exploitation trade-off in reinforcement learning, illustrating how different exploration mechanisms balance immediate reward maximization with uncertainty-driven information acquisition. Exploration seeks to improve future decision quality by sampling uncertain actions, whereas exploitation selects actions with the currently highest estimated value:

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} Q_t(s_t, a). \tag{767}$$

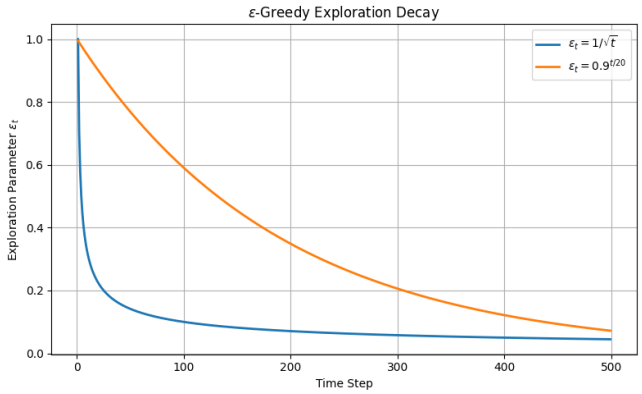
An effective exploration strategy therefore constructs a sampling distribution

$$\pi_t(a|s) \tag{768}$$

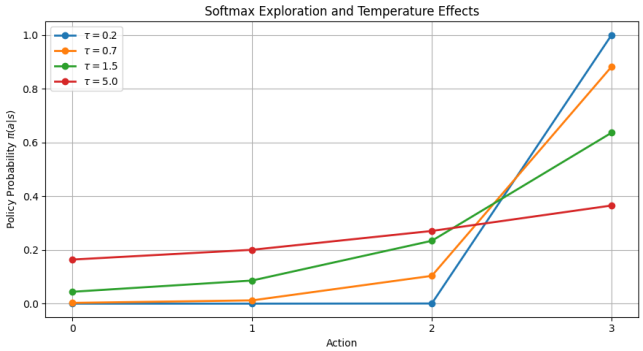
that balances these competing objectives.

Figure 29a illustrates the behavior of the ϵ-greedy exploration mechanism. Under this strategy, the agent follows the policy

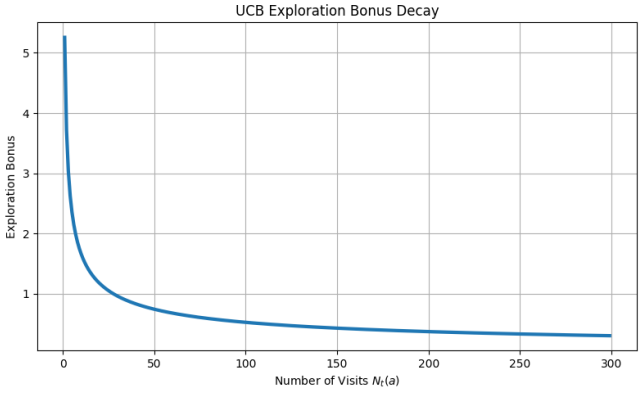
$$\pi_t(a|s) = \begin{cases} 1 - \epsilon_t + \dfrac{\epsilon_t}{|\mathcal{A}|}, & a \in \operatorname{argmax}_{a'} Q_t(s,a'), \\ \dfrac{\epsilon_t}{|\mathcal{A}|}, & \text{otherwise}, \end{cases} \tag{769}$$



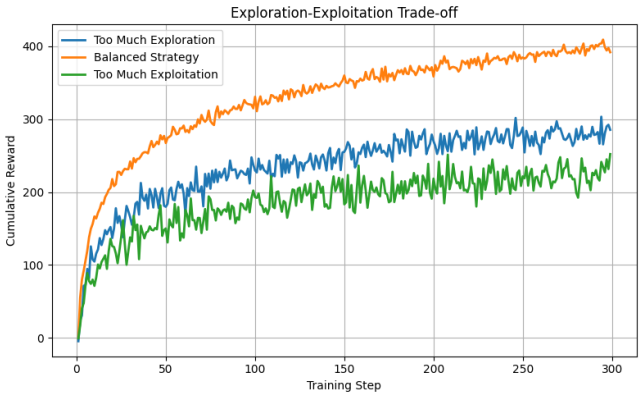
(a) **Decay schedules for the exploration parameter in the ϵ-greedy policy. Exploration is controlled by $\pi_t(a|s) = \begin{cases} 1 - \epsilon_t + \frac{\epsilon_t}{|\mathcal{A}|}, & a \in \text{argmax}_{a'} Q_t(s, a'), \\ \frac{\epsilon_t}{|\mathcal{A}|}, & \text{otherwise.} \end{cases}$ The figure compares inverse-time and exponential decay schedules for ϵ_t.**



(b) **Softmax (Boltzmann) exploration for different temperature parameters τ. Action probabilities are assigned according to $\pi_t(a|s) = \frac{\exp(Q_t(s,a)/\tau_t)}{\sum_{a'} \exp(Q_t(s,a')/\tau_t)}$. Smaller temperatures produce near-greedy behavior, while larger temperatures approach uniform exploration.**



(c) **Exploration bonus in the Upper Confidence Bound (UCB) method: $a_t \in \text{argmax}_a \left[\hat{Q}_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$. The confidence bonus decreases as the number of visits $N_t(a)$ increases, encouraging exploration of undersampled actions.**



(d) **Illustration of the exploration-exploitation trade-off through cumulative reward trajectories. Excessive exploration reduces short-term reward, while excessive exploitation risks convergence to suboptimal policies. Balanced exploration achieves improved long-term performance.**

Figure 29: **Visualization of exploration strategies in reinforcement learning. The first row illustrates ϵ-greedy and softmax exploration mechanisms, while the second row demonstrates uncertainty-driven exploration via UCB and the resulting exploration-exploitation trade-off in cumulative reward performance.**

which means that with probability $1-\epsilon_t$ the agent exploits the greedy action, while with probability ϵ_t it explores uniformly at random:

$$a_t = \begin{cases} \text{argmax}_a \, Q_t(s_t, a), & \text{with probability } 1-\epsilon_t, \\ \text{Uniform}(\mathcal{A}), & \text{with probability } \epsilon_t. \end{cases} \tag{770}$$

The figure compares different decay schedules such as

$$\epsilon_t = \frac{c}{t}, \qquad \epsilon_t = \epsilon_0 \beta^t, \tag{771}$$

demonstrating how exploration gradually diminishes while exploitation increasingly dominates over time.

Figure 29b visualizes softmax (Boltzmann) exploration, where actions are sampled probabilistically according to

$$\pi_t(a|s) = \frac{\exp(Q_t(s,a)/\tau_t)}{\sum_{a'} \exp(Q_t(s,a')/\tau_t)}. \tag{772}$$

The temperature parameter τ_t controls the smoothness of exploration. The figure shows that for large τ_t,

$$\pi_t(a|s) \approx \frac{1}{|\mathcal{A}|}, \tag{773}$$

producing nearly uniform exploration, whereas for small τ_t,

$$\lim_{\tau_t \to 0} \pi_t(a|s) = \mathbf{1}\left\{a \in \text{argmax}_{a'} \, Q_t(s,a')\right\}, \tag{774}$$

yielding near-greedy behavior. From an optimization perspective, this policy arises as the solution of the entropy-regularized problem

$$\pi_t(\cdot|s) = \text{argmax}_\pi \left\{ \sum_a \pi(a|s) Q_t(s,a) + \tau_t \mathcal{H}(\pi(\cdot|s)) \right\}, \tag{775}$$

where

$$\mathcal{H}(\pi) = -\sum_a \pi(a) \log \pi(a) \tag{776}$$

is the Shannon entropy. The figure therefore illustrates how entropy regularization promotes exploration by smoothing the action distribution.

Figure 29c illustrates the Upper Confidence Bound (UCB) exploration mechanism. In UCB methods, actions are selected according to

$$a_t \in \text{argmax}_a \left[\hat{Q}_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right], \tag{777}$$

where $\hat{Q}_t(a)$ denotes the empirical reward estimate and $N_t(a)$ is the number of times action a has been selected. The second term acts as an uncertainty bonus derived from concentration inequalities. The figure shows that actions with small visitation counts receive larger bonuses, thereby encouraging systematic exploration of uncertain actions. As $N_t(a)$ increases, the confidence term decays, leading naturally toward exploitation.

Finally, Figure 29d demonstrates the global effect of exploration strategies on cumulative reward. Excessive exploration produces high uncertainty and reduced short-term performance, whereas excessive exploitation may prematurely commit to suboptimal actions. Balanced exploration yields superior long-term reward accumulation by combining efficient information gathering with value maximization.

Collectively, the figures (Figure 29a, 29b, 29c, and 29d) show that ϵ-greedy methods introduce uniform stochastic perturbations, softmax methods construct entropy-regularized probabilistic policies, and

UCB methods explicitly incorporate uncertainty estimates into the action-selection rule. From a unified perspective, these methods can be interpreted as solving regularized optimization problems of the form

$$\pi_t(\cdot|s) = \text{argmax}_\pi \left\{ \sum_a \pi(a|s) Q_t(s,a) + \lambda_t \Omega(\pi(\cdot|s)) \right\}, \tag{778}$$

where Ω denotes a regularization functional encoding exploration incentives. The figures therefore provide a geometric and probabilistic interpretation of how exploration mechanisms stabilize learning, ensure sufficient coverage of the state-action space, and enable convergence toward optimal policies.

In summary, the exploration-exploitation trade-off is fundamental to reinforcement learning. Effective exploration strategies ensure sufficient coverage of the state-action space while enabling convergence to optimal policies, and their design plays a critical role in both theoretical guarantees and empirical performance.

10.1 Regret Minimization Perspective

The regret minimization perspective has become a central theoretical framework for analyzing sequential decision-making and exploration–exploitation trade-offs in reinforcement learning and bandit problems, with foundational contributions from Lai and Robbins (1985) [38], Bubeck and Cesa-Bianchi (2012) [39], and Lattimore and Szepesvári (2020) [40]. Lai and Robbins (1985) [38] established the asymptotic theory of stochastic multi-armed bandit problems by introducing adaptive allocation strategies and deriving logarithmic lower bounds on cumulative regret, thereby showing the fundamental limits of sequential learning under uncertainty and motivating the development of asymptotically optimal exploration policies. Bubeck and Cesa-Bianchi (2012) [39] provided a comprehensive theoretical treatment of both stochastic and adversarial bandit settings, systematically analyzing regret bounds, upper confidence bound methods, adversarial algorithms, and concentration inequalities that characterize the exploration–exploitation trade-off in online learning systems. Lattimore and Szepesvári (2020) [40] further unified and expanded the modern theory of bandit algorithms by rigorously studying stochastic, contextual, Bayesian, and adversarial bandit frameworks, emphasizing finite-time regret analysis, information-theoretic lower bounds, optimism-based methods, Thompson sampling, and connections to reinforcement learning and sequential optimization. Together, these works established regret minimization as a rigorous quantitative framework for evaluating learning efficiency and exploration strategies in sequential decision-making under uncertainty.

A rigorous formulation of the exploration–exploitation trade-off is provided by the framework of regret minimization, which originates in stochastic bandit theory and extends naturally to reinforcement learning.

Let $\mathcal{A}$ be a finite action set, and for each $a \in \mathcal{A}$, let $\mu(a) := \mathbb{E}[R_t \mid a_t = a]$ denote the (unknown) expected reward of action a. Define the optimal action

$$a^* \in \text{argmax}_{a \in \mathcal{A}} \mu(a), \quad \mu^* := \mu(a^*). \tag{779}$$

At each time t, the learner selects an action a_t based on past observations. The cumulative regret up to time T is defined as

$$\mathcal{R}(T) := \sum_{t=1}^{T} (\mu^* - \mu(a_t)), \tag{780}$$

which quantifies the total loss incurred by not always selecting the optimal action. Equivalently, introducing the suboptimality gap

$$\Delta(a) := \mu^* - \mu(a) \geq 0, \tag{781}$$

one can rewrite regret as

$$\mathcal{R}(T) = \sum_{a \in \mathcal{A}} \Delta(a)\, N_T(a), \tag{782}$$

where

$$N_T(a) := \sum_{t=1}^{T} \mathbf{1}\{a_t = a\} \tag{783}$$

denotes the number of times action a is selected. This decomposition shows that regret is controlled by how often suboptimal actions are chosen. Taking expectations yields

$$\mathbb{E}[\mathcal{R}(T)] = \sum_{a \neq a^*} \Delta(a)\, \mathbb{E}[N_T(a)]. \tag{784}$$

Thus, minimizing regret amounts to controlling $\mathbb{E}[N_T(a)]$ for suboptimal actions, which directly reflects the efficiency of exploration. From another perspective, regret can be expressed as the difference between the reward of an oracle policy that always selects a^* and that of the learning algorithm:

$$\mathbb{E}[\mathcal{R}(T)] = T\mu^* - \mathbb{E}\left[\sum_{t=1}^{T} R_t\right]. \tag{785}$$

Hence, regret minimization is equivalent to maximizing cumulative reward relative to the best fixed action in hindsight.

A central result in bandit theory establishes that any uniformly efficient algorithm must satisfy a lower bound of the form

$$\liminf_{T\to\infty} \frac{\mathbb{E}[N_T(a)]}{\log T} \geq \frac{1}{\mathrm{KL}(\mu(a), \mu^*)}, \tag{786}$$

where $\mathrm{KL}(\cdot,\cdot)$ denotes the Kullback–Leibler divergence between reward distributions. This implies a fundamental lower bound on regret:

$$\mathbb{E}[\mathcal{R}(T)] \geq C \log T, \tag{787}$$

for some constant $C > 0$ depending on the problem instance. Upper Confidence Bound (UCB) algorithms achieve this optimal logarithmic rate by selecting actions according to

$$a_t \in \mathrm{argmax}_{a\in\mathcal{A}} \left[\hat{\mu}_t(a) + c\sqrt{\frac{\log t}{N_t(a)}}\right], \tag{788}$$

where $\hat{\mu}_t(a)$ is the empirical mean reward and the second term is an exploration bonus. One can show that

$$\mathbb{E}[N_T(a)] \leq \frac{C \log T}{\Delta(a)^2} + \mathcal{O}(1), \tag{789}$$

which yields

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T). \tag{790}$$

In reinforcement learning settings with state-dependent decisions, regret is generalized to

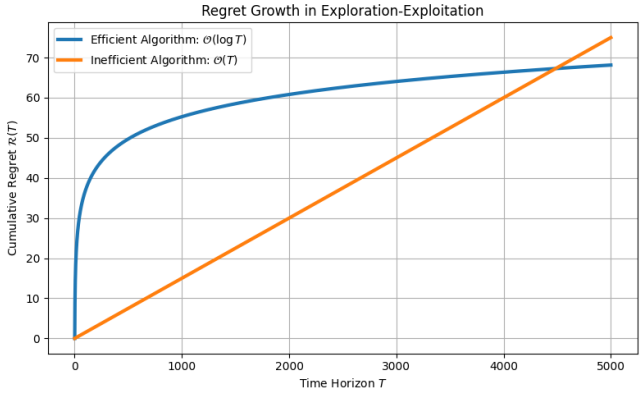
$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(V^*(s_t) - Q^*(s_t, a_t)\right), \tag{791}$$

where V^* and Q^* denote the optimal value and action-value functions. The analysis becomes significantly more complex due to state transitions and delayed rewards, but similar principles apply: efficient exploration aims to minimize visits to suboptimal state-action pairs.

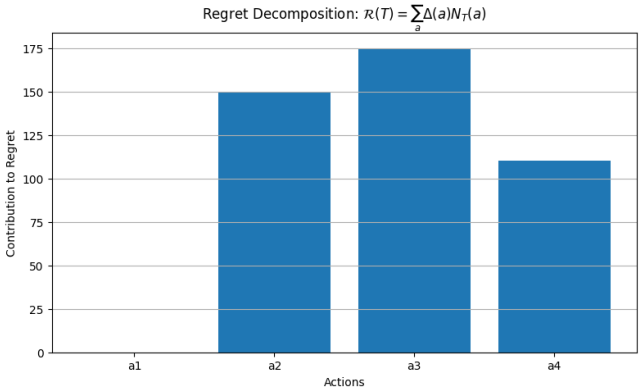
The regret framework provides a precise mathematical lens for understanding exploration. Exploration incurs immediate regret (since suboptimal actions are chosen), but reduces uncertainty, thereby decreasing future regret. Optimal strategies carefully schedule this trade-off so that cumulative regret grows sublinearly:

$$\lim_{T\to\infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0, \tag{792}$$

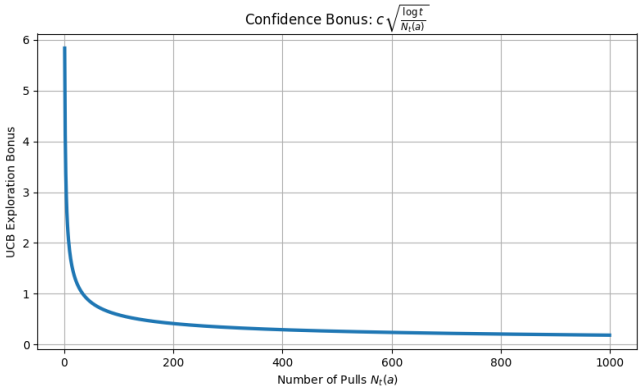
which implies that the average regret vanishes asymptotically.



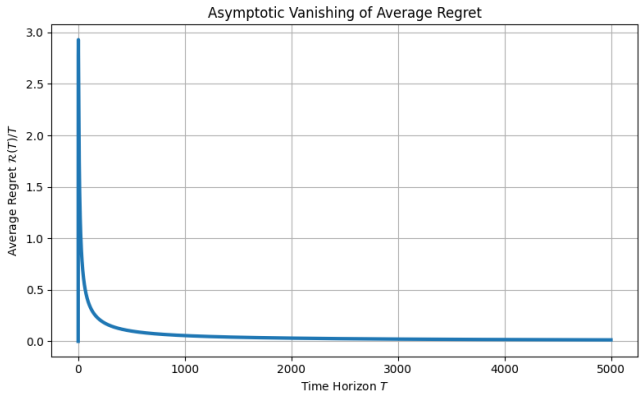
(a) **Comparison between logarithmic and linear regret growth. Efficient exploration strategies achieve $\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T)$, whereas inefficient exploration can lead to linear regret growth: $\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(T)$.**



(b) **Regret decomposition across actions: $\mathcal{R}(T) = \sum_{a \in \mathcal{A}} \Delta(a)\, N_T(a)$, where $\Delta(a) = \mu^* - \mu(a)$ denotes the suboptimality gap and $N_T(a) = \sum_{t=1}^{T} \mathbf{1}\{a_t = a\}$ counts action selections.**



(c) **Upper Confidence Bound (UCB) exploration bonus: $a_t \in \operatorname{argmax}_a \left[\hat{\mu}_t(a) + c\sqrt{\frac{\log t}{N_t(a)}}\right]$. The confidence term decreases as the action is sampled more frequently, balancing exploration and exploitation.**



(d) **Asymptotic decay of average regret: $\lim_{T\to\infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0$. Sublinear regret growth implies that the learning algorithm asymptotically approaches optimal behavior.**

Figure 30: **Visualization of the regret minimization framework in reinforcement learning and stochastic bandits. The figures illustrate cumulative regret growth, regret decomposition across suboptimal actions, uncertainty-driven exploration through UCB bonuses, and the asymptotic vanishing of average regret.**

The figures (Figure 30a, 30b, 30c, and 30d) provide a rigorous visualization of the regret minimization perspective underlying the exploration–exploitation trade-off in reinforcement learning and stochastic bandit problems. Regret measures the cumulative performance loss incurred by a learning agent relative to an oracle policy that always selects the optimal action. Let

$$a^* \in \operatorname{argmax}_{a \in \mathcal{A}} \mu(a), \qquad \mu^* := \mu(a^*), \tag{793}$$

where $\mu(a)$ denotes the expected reward of action a. The cumulative regret up to time horizon T is defined by

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(\mu^* - \mu(a_t) \right), \tag{794}$$

which quantifies the total loss due to selecting suboptimal actions during learning.

Figure 30a illustrates the asymptotic growth behavior of cumulative regret. Efficient exploration strategies produce logarithmic regret growth:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T), \tag{795}$$

whereas poor exploration strategies may yield linear regret:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(T). \tag{796}$$

The logarithmic growth curve demonstrates that exploration is gradually reduced as uncertainty decreases, leading to asymptotically optimal decision-making. In contrast, linear regret indicates persistent suboptimal behavior due to inefficient exploration.

Figure 30b visualizes the decomposition of regret into contributions from suboptimal actions. Defining the suboptimality gap

$$\Delta(a) := \mu^* - \mu(a), \tag{797}$$

and the visitation count

$$N_T(a) = \sum_{t=1}^{T} \mathbf{1}\{a_t = a\}, \tag{798}$$

the regret admits the representation

$$\mathcal{R}(T) = \sum_{a \in \mathcal{A}} \Delta(a)\, N_T(a). \tag{799}$$

Taking expectations yields

$$\mathbb{E}[\mathcal{R}(T)] = \sum_{a \neq a^*} \Delta(a)\, \mathbb{E}[N_T(a)]. \tag{800}$$

The figure shows that regret is fundamentally determined by how frequently the learner selects suboptimal actions. Actions with larger suboptimality gaps contribute disproportionately to cumulative regret, thereby emphasizing the importance of uncertainty-aware exploration.

Figure 30c illustrates the Upper Confidence Bound (UCB) exploration principle. UCB algorithms select actions according to

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} \left[\hat{\mu}_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right], \tag{801}$$

where $\hat{\mu}_t(a)$ is the empirical mean reward and the second term is a confidence bonus derived from concentration inequalities. The figure demonstrates that the exploration bonus decreases as $N_t(a)$ increases. Consequently, poorly explored actions receive larger bonuses, encouraging exploration, while frequently

sampled actions gradually transition toward exploitation. This mechanism yields the logarithmic regret bound

$$\mathbb{E}[N_T(a)] \leq \frac{C \log T}{\Delta(a)^2} + \mathcal{O}(1), \tag{802}$$

which implies

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T). \tag{803}$$

Figure 30d illustrates the asymptotic vanishing of average regret:

$$\lim_{T \to \infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0. \tag{804}$$

This condition characterizes asymptotic optimality, showing that although exploration incurs finite regret initially, its long-term cost becomes negligible relative to the accumulated reward. Geometrically, the decay of average regret reflects the convergence of the learning process toward optimal action selection.

Collectively, the figures (Figure 30a, 30b, 30c, and 30d) illustrate the central principle of regret minimization: exploration introduces temporary performance loss in order to reduce uncertainty and improve future decisions. Efficient algorithms carefully balance this trade-off so that cumulative regret grows sublinearly with time. The regret framework therefore provides a rigorous quantitative foundation for analyzing exploration strategies, deriving performance guarantees, and establishing fundamental limits of learning in reinforcement learning and stochastic control.

In summary, regret minimization formalizes the exploration–exploitation dilemma as a sequential decision problem with performance guarantees. It provides both lower bounds (fundamental limits) and algorithmic upper bounds, thereby serving as a cornerstone of the theoretical analysis of reinforcement learning.

10.2 Optimism in the Face of Uncertainty

The principle of optimism in the face of uncertainty has become one of the most influential theoretical paradigms for balancing exploration and exploitation in reinforcement learning and bandit problems, with major contributions from Auer (2002) [41], Auer et al. (2008) [42], and Lattimore and Szepesvári (2020) [40]. Auer (2002) [41] introduced upper confidence bound (UCB) methods for stochastic multi-armed bandit problems, demonstrating that exploration can be guided by optimistic estimates of uncertain rewards and establishing logarithmic regret guarantees through confidence-based action selection strategies. Auer et al. (2008) [42] extended the optimism principle to reinforcement learning in unknown Markov decision processes through the UCRL framework, deriving near-optimal regret bounds by constructing optimistic models of environment dynamics within statistically plausible confidence sets and planning according to the most favorable admissible model. Lattimore and Szepesvári (2020) [40] further unified and rigorously analyzed optimism-based learning methods within the broader theory of bandit algorithms and sequential decision-making, studying finite-time regret bounds, concentration inequalities, confidence regions, and information-theoretic aspects of exploration strategies in stochastic and adversarial settings. Together, these works established optimism in the face of uncertainty as a mathematically principled approach for efficient exploration, providing strong theoretical guarantees for learning optimal behavior under incomplete information in reinforcement learning and online decision-making problems.

A fundamental principle underlying efficient exploration strategies is *optimism in the face of uncertainty*. The central idea is to bias decision-making toward actions whose value estimates are uncertain, by assigning them artificially inflated (optimistic) values within statistically plausible confidence bounds. This encourages systematic exploration of poorly understood actions while naturally transitioning to exploitation as uncertainty diminishes.

Let $\hat{Q}_t(s,a)$ denote an estimate of the true action-value function $Q^*(s,a)$ based on observations up to time t, and let $b_t(s,a) \geq 0$ denote an uncertainty bonus. The action selection rule is given by

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} \left[\hat{Q}_t(s_t, a) + b_t(s_t, a) \right]. \tag{805}$$

Here, $\hat{Q}_t(s,a) + b_t(s,a)$ can be interpreted as an upper confidence bound on $Q^*(s,a)$.

The construction of the bonus term $b_t(s,a)$ is typically based on concentration inequalities. For example, in the bandit setting with bounded rewards in $[0,1]$, one may use Hoeffding's inequality to define

$$b_t(a) = \sqrt{\frac{2\log t}{N_t(a)}}, \tag{806}$$

where $N_t(a)$ is the number of times action a has been selected up to time t. This yields the classical UCB algorithm:

$$a_t \in \operatorname{argmax}_a \left[\hat{\mu}_t(a) + \sqrt{\frac{2\log t}{N_t(a)}} \right]. \tag{807}$$

The key property of such bonuses is that they decay with increasing sample size:

$$b_t(s,a) \to 0 \quad \text{as } N_t(s,a) \to \infty, \tag{808}$$

ensuring that the algorithm eventually behaves greedily with respect to $\hat{Q}_t$. At the same time, actions with small $N_t(s,a)$ receive larger bonuses, guaranteeing sufficient exploration.

From a probabilistic standpoint, optimism can be formalized via confidence intervals. Let $\mathcal{C}_t(s,a)$ denote a confidence set such that

$$\mathbb{P}\left(Q^*(s,a) \in \mathcal{C}_t(s,a)\right) \geq 1 - \delta. \tag{809}$$

Then an optimistic estimate is given by

$$\hat{Q}_t(s,a) + b_t(s,a) \approx \sup \mathcal{C}_t(s,a). \tag{810}$$

Thus, the agent acts according to the most favorable plausible model consistent with the data. This principle extends naturally to Markov Decision Processes. In that setting, one constructs confidence sets for transition probabilities and rewards, and defines an optimistic MDP (P^+, R^+) such that

$$(P^+, R^+) \in \mathcal{M}_t, \tag{811}$$

where $\mathcal{M}_t$ is a confidence set of plausible models. The agent then computes

$$\pi_t \in \operatorname{argmax}_\pi V^\pi(P^+, R^+), \tag{812}$$

i.e., the optimal policy under the most optimistic model. This approach underlies algorithms such as UCRL.

An alternative realization of optimism is *optimistic initialization*, where value estimates are initialized to high values:

$$Q_0(s,a) = Q_{\text{init}} \gg 0. \tag{813}$$

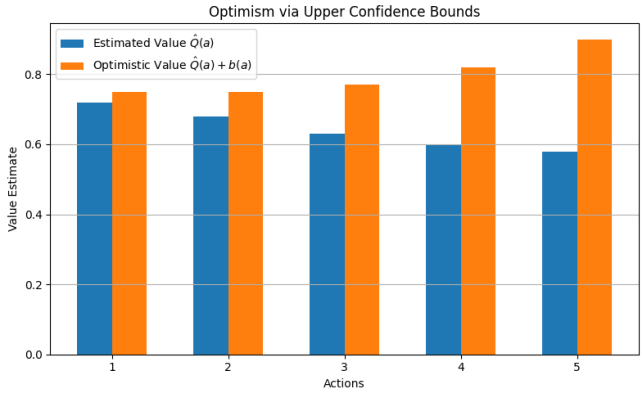
This ensures that initially all actions appear attractive, forcing the agent to explore them. As updates occur, estimates are corrected downward, and the algorithm gradually transitions to exploitation. In modern deep reinforcement learning, explicit confidence bounds are often difficult to compute. Nevertheless, the principle of optimism is approximated through techniques such as:

$$Q(s,a) \approx \mathbb{E}[Q(s,a)] + \text{uncertainty estimate}, \tag{814}$$

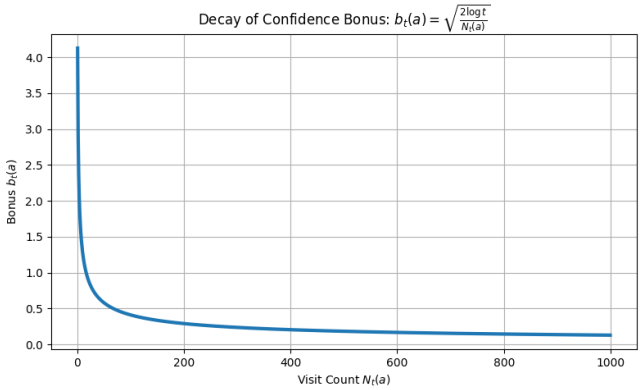
where uncertainty may be captured via ensembles, dropout-based approximations, or Bayesian neural networks. From a theoretical perspective, optimism leads to efficient exploration with provable guarantees. In particular, algorithms based on upper confidence bounds achieve logarithmic regret:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T), \tag{815}$$

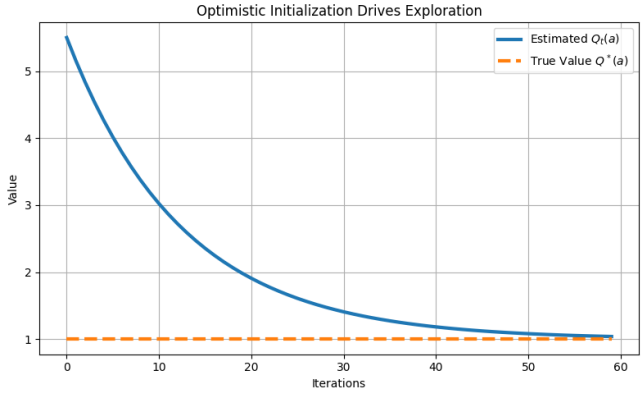
which is optimal up to constant factors.



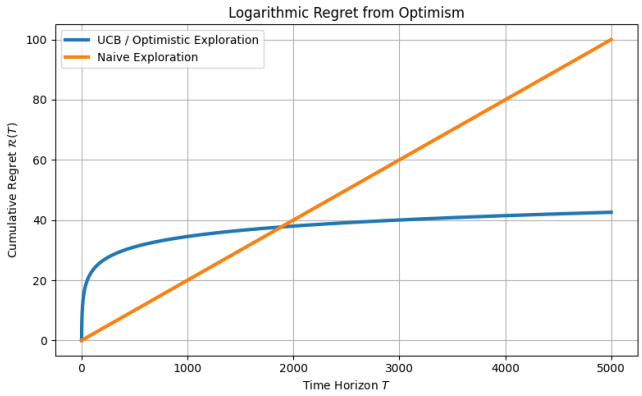
(a) **Optimistic action selection using upper confidence bounds:** $a_t \in \operatorname{argmax}_a \left[\hat{Q}_t(s_t, a) + b_t(s_t, a)\right]$. **The uncertainty bonus encourages exploration of poorly sampled actions.**



(b) **Decay of the exploration bonus:** $b_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}}$. **As the number of visits increases, uncertainty decreases and the algorithm gradually transitions toward exploitation.**



(c) **Optimistic initialization:** $Q_0(s, a) = Q_{\text{init}} \gg 0$. **Initially inflated value estimates force systematic exploration before converging toward the true value function.**



(d) **Regret comparison under optimistic exploration. Efficient uncertainty-driven exploration achieves logarithmic regret:** $\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T)$, **whereas naive exploration strategies may exhibit linear regret growth.**

Figure 31: **Visualization of optimism in the face of uncertainty. The figures illustrate upper confidence bound exploration, decay of uncertainty bonuses, optimistic initialization, and the resulting logarithmic regret guarantees achieved by uncertainty-aware exploration strategies.**

The figures (Figure 31a, 31b, 31c, and 31d) provide a rigorous visualization of the principle of optimism in the face of uncertainty, which forms a fundamental mechanism for efficient exploration in reinforcement learning and stochastic bandit problems. The central idea is to augment estimated action values with uncertainty-dependent bonuses so that actions with insufficient information are assigned optimistically high values. Formally, if

$$\hat{Q}_t(s,a) \tag{816}$$

denotes the current estimate of the optimal action-value function, and

$$b_t(s,a) \geq 0 \tag{817}$$

is an uncertainty bonus, then action selection is performed according to

$$a_t \in \text{argmax}_{a\in\mathcal{A}} \left[\hat{Q}_t(s_t,a) + b_t(s_t,a)\right]. \tag{818}$$

The quantity

$$\hat{Q}_t(s,a) + b_t(s,a) \tag{819}$$

can be interpreted as an upper confidence bound on the unknown optimal value

$$Q^*(s,a). \tag{820}$$

Figure 31a illustrates the effect of optimistic action selection through upper confidence bounds. Actions with large uncertainty receive larger confidence bonuses and therefore appear artificially attractive during decision-making. This mechanism systematically encourages exploration of under-sampled actions while still exploiting actions with large estimated rewards. Geometrically, the figure demonstrates how uncertainty modifies the optimization landscape by shifting the effective value estimates upward according to statistical confidence.

Figure 31b illustrates the decay of uncertainty bonuses with increasing visitation counts. In the classical Upper Confidence Bound (UCB) algorithm, the exploration bonus is given by

$$b_t(a) = \sqrt{\frac{2\log t}{N_t(a)}}, \tag{821}$$

where

$$N_t(a) = \sum_{k=1}^{t} \mathbf{1}\{a_k = a\} \tag{822}$$

denotes the number of times action a has been selected. The figure demonstrates that

$$b_t(a) \to 0 \qquad \text{as} \qquad N_t(a) \to \infty, \tag{823}$$

which implies that uncertainty gradually vanishes as more data are collected. Consequently, the exploration mechanism naturally transitions toward exploitation in the long-time limit. This decay property is essential for achieving asymptotic optimality while still guaranteeing sufficient exploration.

Figure 31c visualizes optimistic initialization, another realization of the optimism principle. In this approach, value estimates are initialized to artificially large values:

$$Q_0(s,a) = Q_{\text{init}} \gg 0. \tag{824}$$

Initially, all actions therefore appear highly rewarding, forcing the agent to explore them. As observations accumulate, the estimates are corrected toward the true value function through temporal-difference updates. The figure shows the gradual convergence of the optimistic estimate toward the true action value:

$$Q_t(s,a) \to Q^*(s,a). \tag{825}$$

This mechanism ensures early-stage exploration without requiring explicit stochastic action perturbations.

Figure 31d illustrates the regret behavior induced by optimistic exploration strategies. The cumulative regret is defined as

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(\mu^* - \mu(a_t)\right), \tag{826}$$

where

$$\mu^* = \max_{a \in \mathcal{A}} \mu(a) \tag{827}$$

is the optimal expected reward. Algorithms based on optimism achieve logarithmic regret:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T), \tag{828}$$

which is asymptotically optimal up to constant factors. In contrast, naive exploration strategies may incur linear regret:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(T). \tag{829}$$

The figure (Figure 31d) demonstrates that uncertainty-aware exploration rapidly reduces long-term performance loss by efficiently allocating exploration effort toward uncertain actions.

From a probabilistic perspective, optimism can be interpreted through confidence sets. Let

$$\mathcal{C}_t(s, a) \tag{830}$$

be a confidence interval satisfying

$$\mathbb{P}\left(Q^*(s, a) \in \mathcal{C}_t(s, a)\right) \geq 1 - \delta. \tag{831}$$

Then the optimistic estimate approximately corresponds to the upper boundary of the confidence set:

$$\hat{Q}_t(s, a) + b_t(s, a) \approx \sup \mathcal{C}_t(s, a). \tag{832}$$

Thus, the agent behaves according to the most favorable statistically plausible model consistent with observed data.

Collectively, the figures (Figure 31a, 31b, 31c, and 31d) demonstrate that optimism in the face of uncertainty provides a principled mechanism for balancing exploration and exploitation. By augmenting value estimates with statistically derived uncertainty bonuses, the algorithm systematically explores uncertain actions while asymptotically converging toward optimal behavior. The resulting dynamics combine probabilistic confidence estimation, sequential decision theory, and regret minimization into a unified framework for efficient reinforcement learning.

In summary, optimism in the face of uncertainty provides a principled and unifying framework for exploration. By augmenting value estimates with uncertainty bonuses derived from statistical confidence, it ensures that the agent systematically explores uncertain actions while converging to optimal behavior as information accumulates.

10.3 Entropy-Regularized Exploration

Entropy-regularized exploration has emerged as a powerful framework for balancing reward maximization and stochastic exploration in reinforcement learning, with important contributions from Ziebart et al. (2008) [43], Haarnoja et al. (2018) [44], and Neu et al. (2017) [45]. Ziebart et al. (2008) [43] introduced the principle of maximum entropy inverse reinforcement learning, proposing that among all policies consistent with observed behavior, the most unbiased distribution should maximize entropy, thereby yielding stochastic policies that naturally capture uncertainty and variability in sequential decision-making processes. Haarnoja et al. (2018) [44] extended entropy-based ideas to deep reinforcement learning through

the Soft Actor-Critic (SAC) algorithm, where policies are optimized not only to maximize expected return but also to maximize entropy, resulting in improved exploration, stability, robustness, and sample efficiency in continuous control tasks. Neu et al. (2017) [45] provided a rigorous theoretical unification of entropy-regularized Markov decision processes by analyzing how entropy terms modify Bellman equations, optimality operators, and policy structures, thereby connecting probabilistic inference, convex optimization, and reinforcement learning within a common mathematical framework. Together, these works established entropy regularization as a principled mechanism for encouraging exploration and robustness in reinforcement learning, while also revealing deep connections between stochastic control, information theory, and probabilistic decision-making.

Entropy-regularized exploration provides a principled and variational approach to the exploration–exploitation trade-off by explicitly favoring stochastic policies. Instead of optimizing only the expected return, one augments the objective with an entropy term that penalizes low-entropy (overly deterministic) policies. The resulting objective functional is

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_t\right] + \lambda\, \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{H}(\pi(\cdot|s_t))\right], \tag{833}$$

where $\lambda > 0$ is a regularization parameter controlling the strength of exploration, and

$$\mathcal{H}(\pi(\cdot|s)) = -\sum_{a\in\mathcal{A}} \pi(a|s) \log \pi(a|s) \tag{834}$$

is the Shannon entropy of the policy at state s. The entropy term encourages dispersion in the action distribution, thereby ensuring that multiple actions are sampled rather than collapsing prematurely to a greedy policy. From a control-theoretic perspective, this corresponds to adding a convex regularizer to the optimization problem, which smooths the objective landscape and improves both exploration and numerical stability.

To analyze this formulation rigorously, define the entropy-regularized value function

$$V_{\lambda}^{\pi}(s) = \mathbb{E}_s^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t, a_t) + \lambda \mathcal{H}(\pi(\cdot|s_t))\right)\right]. \tag{835}$$

The associated Bellman equation becomes

$$V_{\lambda}^{\pi}(s) = \sum_a \pi(a|s)\left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_{\lambda}^{\pi}(s')\right] + \lambda \mathcal{H}(\pi(\cdot|s)). \tag{836}$$

The optimal value function V_{λ}^{*} satisfies the entropy-regularized Bellman optimality equation

$$V_{\lambda}^{*}(s) = \max_{\pi(\cdot|s)} \left\{\sum_a \pi(a|s)\left[Q_{\lambda}^{*}(s,a)\right] + \lambda \mathcal{H}(\pi(\cdot|s))\right\}, \tag{837}$$

where

$$Q_{\lambda}^{*}(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_{\lambda}^{*}(s'). \tag{838}$$

This optimization problem is strictly concave in $\pi(\cdot|s)$ due to the entropy term. Introducing a Lagrange multiplier to enforce the normalization constraint $\sum_a \pi(a|s) = 1$, one obtains the first-order optimality condition

$$Q_{\lambda}^{*}(s,a) - \lambda \log \pi^{*}(a|s) - \lambda = \text{constant}. \tag{839}$$

Solving for $\pi^{*}(a|s)$ yields the Gibbs (Boltzmann) distribution:

$$\pi^{*}(a|s) = \frac{\exp\left(\frac{Q_{\lambda}^{*}(s,a)}{\lambda}\right)}{\sum_{a'} \exp\left(\frac{Q_{\lambda}^{*}(s,a')}{\lambda}\right)}. \tag{840}$$

Substituting back, the optimal value function admits the *log-sum-exp* representation:

$$V_\lambda^*(s) = \lambda \log \left(\sum_{a \in \mathcal{A}} \exp \left(\frac{Q_\lambda^*(s,a)}{\lambda} \right) \right). \tag{841}$$

This expression can be interpreted as a smooth approximation of the maximum operator:

$$\lim_{\lambda \to 0} V_\lambda^*(s) = \max_a Q^*(s,a), \tag{842}$$

showing that the classical Bellman optimality equation is recovered in the zero-temperature limit. From an operator-theoretic viewpoint, the entropy-regularized Bellman operator T_λ is defined as

$$(T_\lambda V)(s) = \lambda \log \left(\sum_a \exp \left(\frac{R(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s')}{\lambda} \right) \right), \tag{843}$$

which is a smooth, contraction mapping under appropriate norms. This regularization improves numerical stability and enables gradient-based optimization methods.

Moreover, entropy regularization admits a dual interpretation in terms of Kullback–Leibler divergence. The policy update can be viewed as solving

$$\pi^*(\cdot|s) = \text{argmax}_\pi \left\{ \sum_a \pi(a|s) Q(s,a) - \lambda \, \text{KL}(\pi(\cdot|s) \| \text{Uniform}) \right\}, \tag{844}$$

which reveals that entropy regularization penalizes deviation from a reference distribution (often uniform).

The figures (Figure 32a, 32b, 32c, 32d) provide a rigorous visualization of entropy-regularized exploration, in which stochasticity is incorporated directly into the reinforcement learning objective through an entropy penalty. Instead of maximizing only the expected cumulative reward, the agent optimizes the entropy-regularized functional

$$J(\pi) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t \right] + \lambda \, \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{H}(\pi(\cdot|s_t)) \right], \tag{845}$$

where

$$\mathcal{H}(\pi(\cdot|s)) = -\sum_{a \in \mathcal{A}} \pi(a|s) \log \pi(a|s) \tag{846}$$

denotes the Shannon entropy of the policy. The regularization parameter $\lambda > 0$ controls the strength of exploration by penalizing low-entropy, highly deterministic policies.

Figure 32a illustrates the Gibbs (Boltzmann) policy induced by entropy regularization. Solving the entropy-regularized Bellman optimization problem yields the optimal stochastic policy

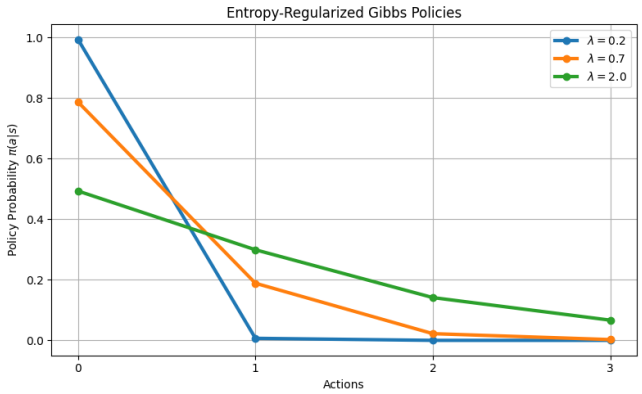
$$\pi^*(a|s) = \frac{\exp\left(\frac{Q_\lambda^*(s,a)}{\lambda}\right)}{\sum_{a'} \exp\left(\frac{Q_\lambda^*(s,a')}{\lambda}\right)}. \tag{847}$$

The figure demonstrates how the regularization parameter governs the concentration of the policy distribution. For small values of λ, the policy becomes sharply concentrated around the maximizing action:

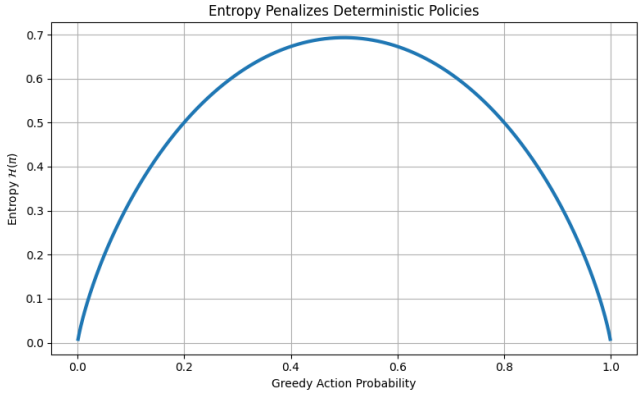
$$\lim_{\lambda \to 0} \pi^*(a|s) = \mathbf{1} \left\{ a \in \text{argmax}_{a'} \, Q^*(s,a') \right\}, \tag{848}$$

whereas larger values of λ produce smoother and more exploratory action distributions.

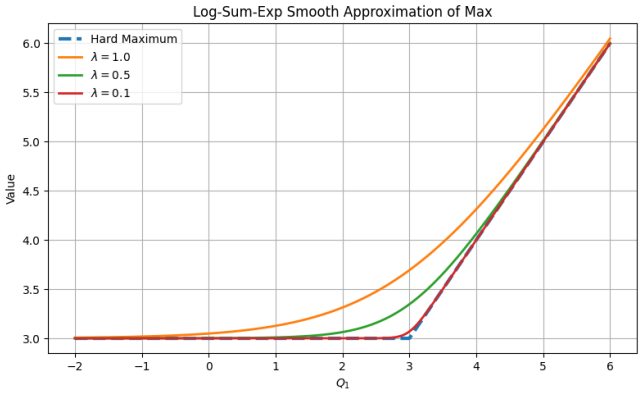
Figure 32b visualizes the entropy functional itself and its role as a convex regularizer. The entropy



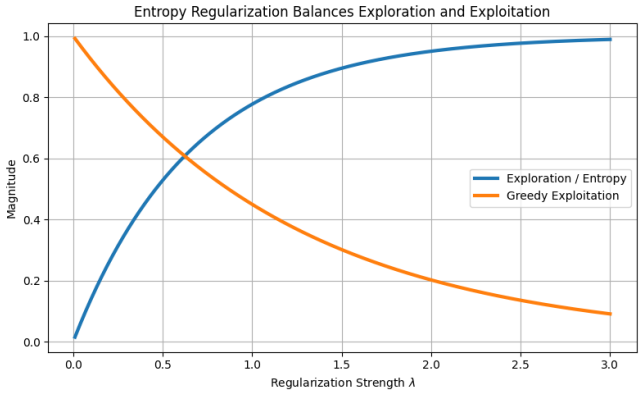
(a) **Gibbs (softmax) policies induced by entropy regularization:** $\pi^*(a|s) = \dfrac{\exp\left(\frac{Q^*_\lambda(s,a)}{\lambda}\right)}{\sum_{a'} \exp\left(\frac{Q^*_\lambda(s,a')}{\lambda}\right)}$. **Larger regularization parameters produce smoother, more exploratory policies.**



(b) **Entropy as a function of policy concentration:** $\mathcal{H}(\pi) = -\sum_a \pi(a) \log \pi(a)$. **Entropy regularization penalizes deterministic policies and encourages exploration.**



(c) **The entropy-regularized Bellman operator produces the log-sum-exp representation:** $V^*_\lambda(s) = \lambda \log\left(\sum_a \exp\left(\frac{Q^*_\lambda(s,a)}{\lambda}\right)\right)$, **which acts as a smooth approximation of the maximum operator.**



(d) **Exploration–exploitation trade-off induced by entropy regularization. Increasing the regularization parameter λ promotes exploration while reducing purely greedy behavior.**

Figure 32: **Visualization of entropy-regularized exploration. The figures illustrate Gibbs policies, entropy penalties, smooth Bellman operators via log-sum-exp regularization, and the resulting balance between exploration and exploitation.**

achieves its maximum for highly dispersed distributions and decreases as the policy becomes deterministic. Thus, entropy regularization penalizes premature collapse to a greedy policy and maintains stochastic exploration throughout learning. From a variational perspective, the policy update solves

$$\pi^*(\cdot|s) = \text{argmax}_\pi \left\{ \sum_a \pi(a|s) Q(s,a) + \lambda \mathcal{H}(\pi(\cdot|s)) \right\}, \tag{849}$$

which represents a convex optimization problem balancing expected reward and entropy.

Figure 32c illustrates the smooth approximation of the maximum operator induced by entropy regularization. Substituting the Gibbs policy into the Bellman equation yields the log-sum-exp representation

$$V_\lambda^*(s) = \lambda \log \left(\sum_{a \in \mathcal{A}} \exp\left(\frac{Q_\lambda^*(s,a)}{\lambda} \right) \right). \tag{850}$$

The figure demonstrates that this expression behaves as a smooth approximation of the classical maximum:

$$\lim_{\lambda \to 0} V_\lambda^*(s) = \max_a Q^*(s,a). \tag{851}$$

Consequently, the entropy-regularized Bellman operator

$$(T_\lambda V)(s) = \lambda \log \left(\sum_a \exp\left(\frac{R(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s')}{\lambda} \right) \right) \tag{852}$$

replaces the non-smooth maximum operator with a differentiable mapping, thereby improving numerical stability and enabling efficient gradient-based optimization.

Figure 32d visualizes the exploration–exploitation trade-off induced by the regularization parameter λ Increasing λ raises policy entropy and encourages broader exploration, while smaller values recover increasingly greedy behavior. The figure therefore illustrates how entropy regularization continuously interpolates between exploratory stochastic control and deterministic exploitation.

From an information-theoretic perspective, entropy regularization admits an equivalent formulation in terms of Kullback–Leibler divergence:

$$\pi^*(\cdot|s) = \text{argmax}_\pi \left\{ \sum_a \pi(a|s) Q(s,a) - \lambda \, \text{KL}\left(\pi(\cdot|s) \| \text{Uniform}\right) \right\}, \tag{853}$$

which shows that the entropy term penalizes deviation from a reference distribution. Thus, entropy-regularized exploration can be interpreted as balancing reward maximization against informational complexity.

Collectively, the figures (Figure 32a, 32b, 32c, 32d) demonstrate that entropy regularization transforms reinforcement learning into a smooth variational optimization problem characterized by stochastic optimal policies, differentiable Bellman operators, and persistent exploration. The resulting framework establishes deep connections between reinforcement learning, convex analysis, information theory, and statistical physics through the emergence of Gibbs distributions and entropy-maximizing dynamics.

In summary, entropy-regularized exploration transforms the reinforcement learning problem into a smooth, convex-analytic optimization problem. It induces stochastic optimal policies, ensures persistent exploration, and provides a bridge between reinforcement learning, statistical physics (via Gibbs measures), and information theory.

10.4 Thompson Sampling and Bayesian Exploration

Thompson sampling and Bayesian exploration constitute a fundamental probabilistic framework for balancing exploration and exploitation in sequential decision-making problems, originating in the pioneering

work of Thompson (1933) [46] and later expanded by Daniel et al. (2018) [47] and by Osband et al. (2016) [48]. Thompson (1933) [46] introduced a Bayesian approach to adaptive decision-making in which actions are selected according to their posterior probability of being optimal, thereby providing one of the earliest principled mechanisms for exploration under uncertainty in stochastic environments. Daniel et al. (2018) [47] later provided a comprehensive theoretical and algorithmic treatment of Thompson sampling, rigorously analyzing Bayesian regret bounds, posterior sampling strategies, information-theoretic properties, and practical implementations across stochastic bandits, contextual learning, and reinforcement learning frameworks. Osband et al. (2016) [48] extended Bayesian exploration ideas to deep reinforcement learning through Bootstrapped DQN, where multiple bootstrapped value-function estimates are used to induce temporally consistent deep exploration, thereby approximating posterior sampling behavior in high-dimensional neural-network-based control problems. Together, these works established Bayesian posterior sampling as a mathematically principled approach for efficient exploration, demonstrating how uncertainty quantification and probabilistic reasoning can guide sequential learning and improve long-term decision-making performance in reinforcement learning systems.

Bayesian approaches provide a principled probabilistic framework for addressing the exploration–exploitation trade-off by explicitly modeling uncertainty over the unknown parameters of the environment. Let $\theta \in \Theta$ denote the latent parameters governing the reward and/or transition dynamics (e.g., reward means in bandits or model parameters in MDPs). Given a prior distribution $p(\theta)$ and observed data $\mathcal{D}_t := \{(s_k, a_k, r_{k+1}, s_{k+1})\}_{k=0}^{t-1}$, the posterior distribution is updated via Bayes' rule:

$$p(\theta \mid \mathcal{D}_t) = \frac{p(\mathcal{D}_t \mid \theta)\, p(\theta)}{\int_\Theta p(\mathcal{D}_t \mid \theta')\, p(\theta')\, d\theta'}. \tag{854}$$

Thompson sampling (also called posterior sampling) selects actions by sampling a parameter from the posterior and acting optimally with respect to that sampled model:

$$\theta_t \sim p(\theta \mid \mathcal{D}_t), \qquad a_t \in \operatorname{argmax}_{a \in \mathcal{A}} Q(s_t, a; \theta_t). \tag{855}$$

Equivalently, the induced policy can be written as a mixture over optimal policies:

$$\pi_t(a \mid s) = \mathbb{P}\big(a \in \operatorname{argmax}_{a'} Q(s, a'; \theta) \,\big|\, \mathcal{D}_t\big) = \int_\Theta \mathbf{1}\big\{a \in \operatorname{argmax}_{a'} Q(s, a'; \theta)\big\}\; p(\theta \mid \mathcal{D}_t)\, d\theta. \tag{856}$$

Thus, the probability of selecting an action equals the posterior probability that it is optimal. This yields a natural and adaptive balance: actions with high posterior uncertainty (and hence non-negligible probability of being optimal) are explored more frequently.

In the stochastic multi-armed bandit setting with Bernoulli rewards, a conjugate Beta prior leads to closed-form updates. If $\mu(a)$ is the success probability of arm a and the prior is $\mu(a) \sim \mathrm{Beta}(\alpha_a, \beta_a)$, then after observing $S_t(a)$ successes and $F_t(a)$ failures,

$$\mu(a) \mid \mathcal{D}_t \sim \mathrm{Beta}\big(\alpha_a + S_t(a),\, \beta_a + F_t(a)\big). \tag{857}$$

Thompson sampling draws $\tilde{\mu}_t(a)$ from these posteriors and selects

$$a_t \in \operatorname{argmax}_a \tilde{\mu}_t(a). \tag{858}$$

Theoretical analysis shows that Thompson sampling achieves near-optimal regret bounds. In many bandit models, one has

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T), \tag{859}$$

matching the minimax lower bounds up to constants. More refined, problem-dependent bounds can be expressed in terms of information-theoretic quantities, reflecting how quickly the posterior concentrates around the true parameter.

From an information-theoretic perspective, Thompson sampling can be interpreted as approximately maximizing expected information gain while optimizing reward. Let $I(\theta; a_t, r_{t+1})$ denote the mutual

information between the parameter and the observation at time t. Then the algorithm implicitly trades off reward and information acquisition, leading to efficient learning dynamics.

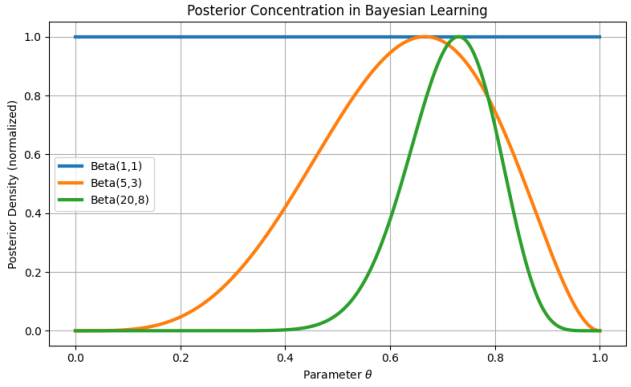
In the context of Markov Decision Processes, Bayesian reinforcement learning extends this idea by maintaining a posterior over transition kernels P and reward functions R. Thompson sampling then proceeds by sampling an MDP (P_t, R_t) from the posterior and computing an optimal policy:

$$(P_t, R_t) \sim p(P, R \mid \mathcal{D}_t), \qquad \pi_t \in \operatorname{argmax}_\pi V^\pi(P_t, R_t). \tag{860}$$

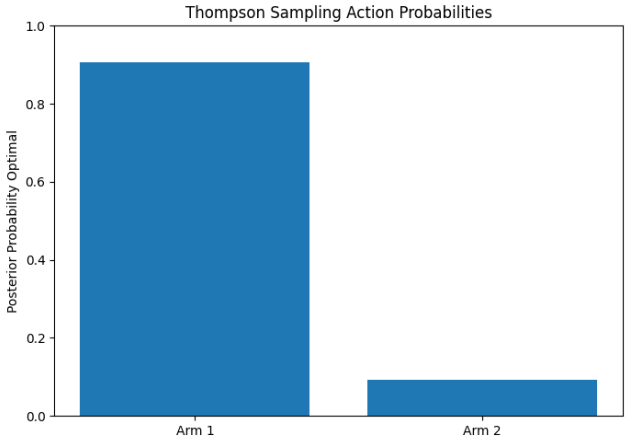
This induces deep exploration, as the sampled model is consistent across time within an episode, encouraging coherent exploratory behavior. Despite its conceptual elegance, exact Bayesian inference is often intractable in large-scale problems. Practical implementations therefore rely on approximations, such as:

$$\theta_t \approx \text{sample from approximate posterior}, \tag{861}$$

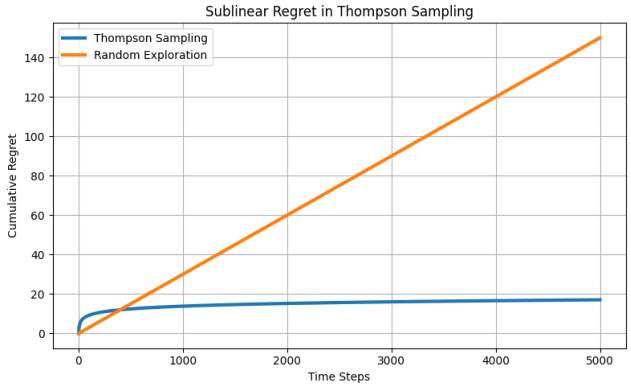
using techniques like variational inference, bootstrap ensembles, or Monte Carlo methods.



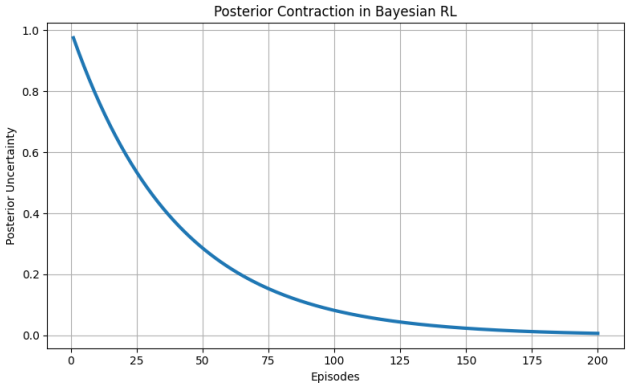
(a) **Posterior concentration under Bayesian inference:** $p(\theta \mid \mathcal{D}_t) = \frac{p(\mathcal{D}_t|\theta)\, p(\theta)}{\int_\Theta p(\mathcal{D}_t|\theta')\, p(\theta')\, d\theta'}$. **As observations accumulate, the posterior contracts around the true parameter.**



(b) **Thompson sampling action selection probabilities. Actions are sampled according to the posterior probability of optimality:** $\pi_t(a|s) = \mathbb{P}\left(a \in \operatorname{argmax}_{a'} Q(s, a'; \theta) \,|\, \mathcal{D}_t\right)$.



(c) **Logarithmic regret growth achieved by Thompson sampling:** $\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T)$. **The sublinear growth implies asymptotically vanishing average regret.**



(d) **Posterior contraction in Bayesian reinforcement learning. Uncertainty over the environment model decreases as more trajectories are observed:** $(P_t, R_t) \sim p(P, R \mid \mathcal{D}_t)$.

Figure 33: **Bayesian exploration and Thompson sampling. The figures illustrate posterior concentration, posterior-based randomized action selection, logarithmic regret growth, and posterior contraction in Bayesian reinforcement learning.**

Figures 33a, 33b, 33c, and 33d illustrate the mathematical structure underlying Thompson sampling

and Bayesian exploration. Figure 33a visualizes posterior concentration in Bayesian inference. Starting from a prior distribution $p(\theta)$ over latent environment parameters, observations $\mathcal{D}_t$ update the posterior according to Bayes' rule:

$$p(\theta \mid \mathcal{D}_t) = \frac{p(\mathcal{D}_t \mid \theta)\, p(\theta)}{\int_\Theta p(\mathcal{D}_t \mid \theta')p(\theta')\, d\theta'}. \tag{862}$$

As additional observations accumulate, the posterior distribution concentrates around the true parameter, reducing uncertainty and sharpening decision-making. The shrinking posterior variance shown in the figure reflects asymptotic learning and posterior consistency.

Figure 33b illustrates the core mechanism of Thompson sampling. At each iteration, the agent samples a parameter realization

$$\theta_t \sim p(\theta \mid \mathcal{D}_t), \tag{863}$$

and selects the action that is optimal under the sampled model:

$$a_t \in \operatorname{argmax}_{a \in \mathcal{A}} Q(s_t, a; \theta_t). \tag{864}$$

Consequently, the probability of selecting an action equals the posterior probability that the action is optimal:

$$\pi_t(a|s) = \mathbb{P}\big(a \in \operatorname{argmax}_{a'} Q(s, a'; \theta) \,\big|\, \mathcal{D}_t\big)\,. \tag{865}$$

The figure demonstrates how uncertainty naturally induces randomized exploration: actions with substantial posterior plausibility are sampled more frequently, while posterior concentration gradually drives exploitation.

Figure 33c compares cumulative regret trajectories and demonstrates the efficiency of Bayesian exploration. The regret is defined by

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(\mu^* - \mu(a_t)\right), \tag{866}$$

where μ^* denotes the optimal expected reward. Thompson sampling achieves logarithmic regret growth:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T), \tag{867}$$

which is reflected in the slowly growing regret curves in the figure. The sublinear growth implies

$$\lim_{T \to \infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0, \tag{868}$$

showing that the average regret vanishes asymptotically.

Figure 33d illustrates Bayesian exploration in Markov Decision Processes. Instead of maintaining uncertainty only over reward means, the agent maintains a posterior over entire environment models:

$$(P_t, R_t) \sim p(P, R \mid \mathcal{D}_t). \tag{869}$$

A policy is then computed by solving the sampled MDP:

$$\pi_t \in \operatorname{argmax}_\pi V^\pi(P_t, R_t). \tag{870}$$

Because the sampled model remains fixed during the episode, exploration becomes temporally coherent, producing *deep exploration*. The figure visualizes how posterior uncertainty over transition dynamics gradually contracts as additional trajectories are observed.

Collectively, the figures (Figures 33a, 33b, 33c, and 33d) demonstrate that Thompson sampling implements exploration through posterior randomization rather than explicit exploration bonuses. From a probabilistic perspective, exploration arises from uncertainty in the posterior distribution itself. As uncertainty decreases, the induced policy automatically transitions from exploratory to exploitative behavior.

This provides a principled Bayesian realization of the exploration–exploitation trade-off, combining statistical inference, stochastic control, and regret minimization into a unified framework.

In summary, Thompson sampling provides a unified and theoretically grounded mechanism for exploration by randomizing over plausible models. By aligning action selection with posterior beliefs about optimality, it achieves an efficient balance between exploration and exploitation, with strong regret guarantees and excellent empirical performance across a wide range of settings.

10.5 Directed vs Undirected Exploration

Exploration strategies in reinforcement learning can be rigorously categorized into *undirected* and *directed* approaches, depending on whether the exploration mechanism is independent of uncertainty estimates or explicitly guided by them. This distinction reflects fundamentally different ways of addressing the exploration–exploitation trade-off.

Undirected exploration introduces stochasticity into the action-selection mechanism without explicitly accounting for uncertainty in value estimates. Formally, actions are sampled according to a stochastic policy:

$$a_t \sim \pi_t(\cdot \mid s_t), \tag{871}$$

where π_t is typically designed to ensure persistent randomness. Examples include ϵ-greedy policies:

$$\pi_t(a|s) = (1-\epsilon_t)\mathbf{1}\{a \in \operatorname{argmax}_{a'} Q_t(s,a')\} + \frac{\epsilon_t}{|\mathcal{A}|}, \tag{872}$$

and softmax (Boltzmann) policies:

$$\pi_t(a|s) = \frac{\exp(Q_t(s,a)/\tau_t)}{\sum_{a'} \exp(Q_t(s,a')/\tau_t)}. \tag{873}$$

In such methods, exploration arises from injected randomness rather than from a structured assessment of uncertainty. Consequently, actions are explored in a relatively uniform or value-smoothed manner, regardless of how well they have been estimated. In contrast, directed exploration explicitly incorporates uncertainty into the decision-making process. The action selection rule takes the form

$$a_t \in \operatorname{argmax}_{a\in\mathcal{A}} \left[Q_t(s_t,a) + b_t(s_t,a) \right], \tag{874}$$

where $b_t(s,a)$ is an exploration bonus reflecting uncertainty, such as variance estimates or confidence intervals. Typically, $b_t(s,a)$ satisfies

$$b_t(s,a) \propto \frac{1}{\sqrt{N_t(s,a)}}, \tag{875}$$

so that less-visited state-action pairs receive larger bonuses.

From a statistical viewpoint, directed exploration aims to minimize uncertainty by prioritizing actions with high variance or limited data. This leads to more efficient sampling. For instance, in bandit problems, one can bound the expected number of times a suboptimal action is selected under directed strategies:

$$\mathbb{E}[N_T(a)] = \mathcal{O}\left(\frac{\log T}{\Delta(a)^2}\right), \tag{876}$$

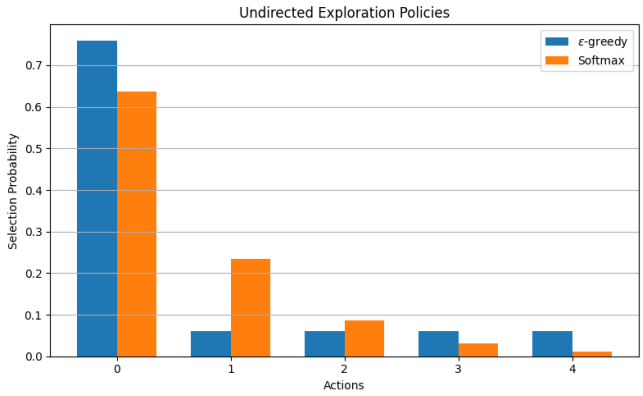
which directly implies logarithmic regret. In contrast, undirected methods often exhibit weaker guarantees unless carefully tuned. A deeper distinction can be made in terms of information acquisition. Let $\mathcal{I}_t(a)$ denote the expected information gain from selecting action a at time t. Directed methods implicitly optimize a criterion of the form

$$a_t \in \operatorname{argmax}_a \left[Q_t(s_t,a) + \beta \mathcal{I}_t(a) \right], \tag{877}$$

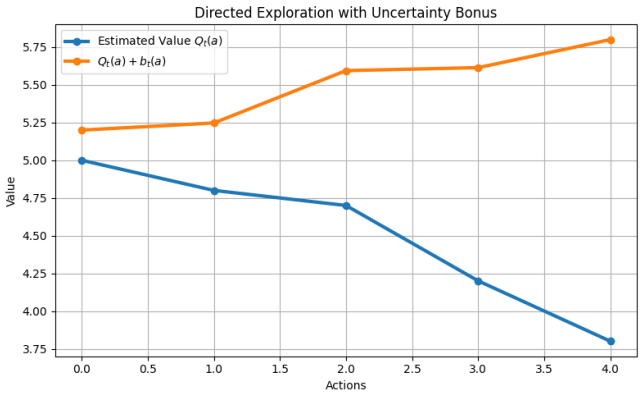
where $\beta > 0$ balances reward and information. Undirected methods, by contrast, do not explicitly account for $\mathcal{I}_t(a)$, and may therefore allocate samples inefficiently.

In reinforcement learning with large or continuous state spaces, the difference becomes even more pronounced. Undirected exploration may fail to adequately cover the state-action space, particularly in sparse-reward environments. Directed exploration methods, such as optimism-based algorithms or posterior sampling, can induce *deep exploration*, where sequences of actions are chosen to reduce long-term uncertainty.

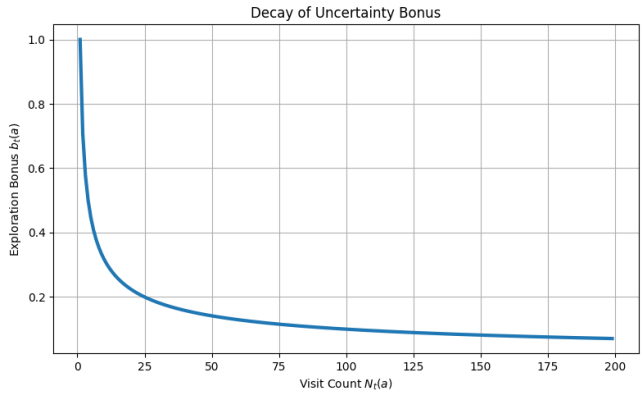
However, directed exploration typically requires additional structure, such as uncertainty estimation or model assumptions, which may be computationally demanding. Undirected methods, while less efficient, are simple to implement and often robust in practice.



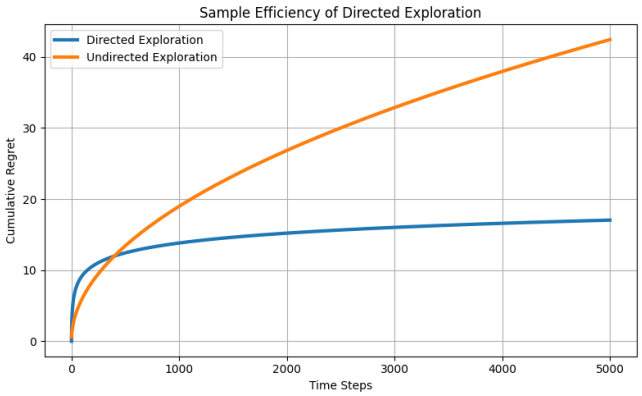
(a) **Undirected exploration via random action selection**



(b) **Directed exploration with uncertainty bonuses**



(c) **Regret comparison between exploration strategies**



(d) **Information gain and deep exploration**

Figure 34: **Illustration of directed versus undirected exploration strategies in reinforcement learning. The figures compare stochastic exploration mechanisms, uncertainty-guided action selection, regret behavior, and information-driven exploration dynamics.**

The figures (Figure 34a, 34b, 34c, and 34d) illustrate the fundamental distinction between undirected and directed exploration strategies in reinforcement learning. Figure 34a visualizes undirected exploration mechanisms, in which stochasticity is injected into the policy independently of uncertainty estimates. In such approaches, actions are sampled according to a randomized policy

$$a_t \sim \pi_t(\cdot|s_t), \tag{878}$$

where exploration arises solely from probabilistic perturbations of the action-selection rule. Typical examples include the ϵ-greedy policy

$$\pi_t(a|s) = (1-\epsilon_t)\mathbf{1}\{a \in \operatorname{argmax}_{a'} Q_t(s,a')\} + \frac{\epsilon_t}{|\mathcal{A}|}, \tag{879}$$

and the softmax (Boltzmann) policy

$$\pi_t(a|s) = \frac{\exp(Q_t(s,a)/\tau_t)}{\sum_{a'} \exp(Q_t(s,a')/\tau_t)}. \tag{880}$$

The figure demonstrates that exploration is distributed relatively uniformly or according to smoothed value estimates, regardless of the uncertainty associated with different actions.

Figure 34b illustrates directed exploration, where uncertainty estimates explicitly influence action selection. The policy selects actions according to

$$a_t \in \text{argmax}_{a \in \mathcal{A}} \left[Q_t(s_t, a) + b_t(s_t, a) \right], \tag{881}$$

where $b_t(s,a)$ is an exploration bonus representing uncertainty or lack of information. Typically, the exploration bonus decays with visitation frequency:

$$b_t(s,a) \propto \frac{1}{\sqrt{N_t(s,a)}}, \tag{882}$$

so that rarely visited state-action pairs receive larger optimism bonuses. The figure visualizes how uncertainty-guided bonuses elevate the apparent value of poorly explored actions, thereby directing exploration toward informative regions of the state-action space.

Figure 34c compares the regret behavior of directed and undirected exploration strategies. Directed methods achieve efficient sampling by systematically reducing uncertainty, yielding bounds of the form

$$\mathbb{E}[N_T(a)] = \mathcal{O}\left(\frac{\log T}{\Delta(a)^2} \right), \tag{883}$$

which implies logarithmic cumulative regret:

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\log T). \tag{884}$$

In contrast, undirected exploration may revisit already well-understood actions excessively, resulting in slower learning and larger cumulative regret. The figure therefore illustrates the superior sample efficiency achieved by uncertainty-aware exploration.

Figure 34d visualizes exploration from an information-theoretic perspective. Directed exploration methods implicitly optimize both reward and information acquisition by selecting actions according to criteria of the form

$$a_t \in \text{argmax}_a \left[Q_t(s_t, a) + \beta \mathcal{I}_t(a) \right], \tag{885}$$

where $\mathcal{I}_t(a)$ denotes the expected information gain associated with action a and $\beta > 0$ balances exploitation and exploration. The figure demonstrates how directed exploration prioritizes informative actions and induces deep exploration, particularly in sparse-reward or high-dimensional environments.

Collectively, the figures (Figure 34a, 34b, 34c, and 34d) show that undirected exploration relies on generic stochasticity to ensure coverage of the action space, whereas directed exploration systematically leverages uncertainty estimates to guide sampling toward informative actions. This distinction leads to fundamentally different learning dynamics: undirected methods emphasize simplicity and robustness, while directed methods achieve improved sample efficiency, lower regret, and stronger theoretical guarantees through principled uncertainty-driven exploration.

In summary, undirected exploration relies on generic stochasticity to ensure coverage of the action space, whereas directed exploration leverages uncertainty estimates to prioritize informative actions. The latter achieves superior sample efficiency and stronger theoretical guarantees, but at the cost of increased computational and modeling complexity.

10.6 Exploration in Continuous Action Spaces

In continuous action spaces, where $\mathcal{A} \subseteq \mathbb{R}^d$ is uncountable, classical discrete exploration strategies (e.g., ϵ-greedy) are not directly applicable. The exploration–exploitation trade-off must instead be handled through perturbations of policies, actions, or parameters in a manner that ensures sufficient coverage of the action space while preserving stability and convergence properties.

A common approach is to employ a stochastic, parameterized policy $\pi_\theta(a|s)$, often chosen from a parametric family such as Gaussian distributions:

$$\pi_\theta(a|s) = \mathcal{N}\big(a \mid \mu_\theta(s), \Sigma_\theta(s)\big), \tag{886}$$

where $\mu_\theta(s) \in \mathbb{R}^d$ is the mean function and $\Sigma_\theta(s)$ is a positive-definite covariance matrix. Actions are then sampled as

$$a_t \sim \pi_\theta(\cdot|s_t), \tag{887}$$

which inherently introduces exploration through stochasticity. The covariance $\Sigma_\theta(s)$ controls the exploration scale, and may be state-dependent or annealed over time:

$$\Sigma_t \to 0 \quad \text{as } t \to \infty, \tag{888}$$

to ensure asymptotic exploitation.

An alternative and widely used method is *action noise injection*, particularly in deterministic policy settings. Let $\mu_\theta : \mathcal{S} \to \mathcal{A}$ be a deterministic policy. Exploration is induced by perturbing the action:

$$a_t = \mu_\theta(s_t) + \epsilon_t, \qquad \epsilon_t \sim \mathcal{N}(0, \Sigma_t). \tag{889}$$

In practice, temporally correlated noise processes such as Ornstein–Uhlenbeck processes are sometimes used:

$$d\epsilon_t = -\beta \epsilon_t \, dt + \sigma \, dW_t, \tag{890}$$

to generate smoother exploration trajectories in physical control tasks. A more structured approach is *parameter space noise*, where randomness is injected directly into the policy parameters rather than the actions. Specifically, one samples

$$\theta' = \theta + \xi, \qquad \xi \sim \mathcal{N}(0, \sigma^2 I), \tag{891}$$

and defines the perturbed policy $\pi_{\theta'}$. Actions are then generated as

$$a_t \sim \pi_{\theta'}(\cdot|s_t). \tag{892}$$

This induces state-dependent, consistent exploration across time steps, since the same perturbed parameters are used over trajectories. From a functional perspective, this corresponds to exploring in the space of policies rather than in the action space.

The distinction between action noise and parameter noise can be formalized by examining the induced distribution over trajectories. Let $\tau = (s_0, a_0, s_1, a_1, \dots)$ denote a trajectory. Under action noise, randomness is injected independently at each time step:

$$\mathbb{P}(\tau) = \prod_t P(s_{t+1}|s_t, a_t)\, \pi(a_t|s_t), \tag{893}$$

whereas under parameter noise, randomness is coupled across time through θ':

$$\mathbb{P}(\tau) = \int \left(\prod_t P(s_{t+1}|s_t, a_t)\, \pi_{\theta'}(a_t|s_t) \right) p(\theta')\, d\theta'. \tag{894}$$

This coupling can lead to more coherent and temporally extended exploration.

In actor-critic methods, exploration is typically integrated into the policy gradient framework. The policy gradient theorem yields

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(a|s)\, Q^{\pi_\theta}(s,a) \right], \tag{895}$$

which naturally accommodates stochastic policies. The variance of the gradient estimator is directly influenced by the exploration distribution π_θ, making the choice of covariance structure critical for efficient learning.

A key challenge in continuous spaces is balancing exploration scale with stability. Excessive noise can destabilize learning, while insufficient noise leads to premature convergence. This trade-off is often managed via adaptive schemes:

$$\Sigma_t = \alpha_t \Sigma_0, \quad \alpha_t \downarrow 0, \tag{896}$$

or via entropy regularization:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_t \gamma^t R_t \right] + \lambda\, \mathbb{E}_{\pi_\theta} \left[\sum_t \gamma^t \mathcal{H}(\pi_\theta(\cdot|s_t)) \right], \tag{897}$$

which encourages sustained stochasticity.

The four figures (Figure 35a, 35b, 35c, and 35d) illustrate the principal mechanisms used for exploration in continuous action spaces. Figure 35a depicts stochastic policy exploration using a Gaussian policy

$$\pi_\theta(a|s) = \mathcal{N}(a \mid \mu_\theta(s), \Sigma_\theta(s)), \tag{898}$$

where exploration is governed by the covariance matrix $\Sigma_\theta(s)$. The spread of the sampled actions around the mean policy $\mu_\theta(s)$ reflects the stochastic coverage of the continuous action space. As training progresses, the covariance is often annealed according to

$$\Sigma_t \to 0, \qquad t \to \infty, \tag{899}$$

which gradually transitions the policy from exploration toward exploitation.

Figure 35b illustrates action-noise exploration for deterministic policies:

$$a_t = \mu_\theta(s_t) + \epsilon_t, \qquad \epsilon_t \sim \mathcal{N}(0, \Sigma_t). \tag{900}$$

The noisy trajectories fluctuate around the deterministic control signal, thereby inducing local exploration in action space. The figure demonstrates how injected Gaussian perturbations broaden the effective action distribution while preserving the underlying policy structure. In physical control tasks, temporally correlated noise processes such as Ornstein–Uhlenbeck noise are frequently used:

$$d\epsilon_t = -\beta \epsilon_t\, dt + \sigma\, dW_t, \tag{901}$$

yielding smoother exploration trajectories.

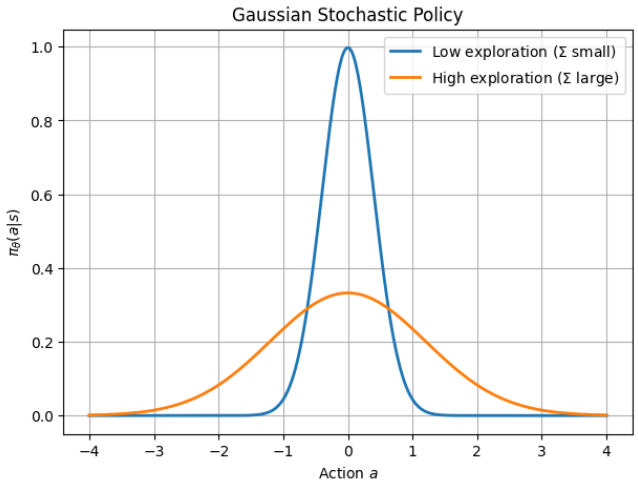
Figure 35c illustrates parameter-space exploration, where perturbations are applied directly to the policy parameters:

$$\theta' = \theta + \xi, \qquad \xi \sim \mathcal{N}(0, \sigma^2 I). \tag{902}$$

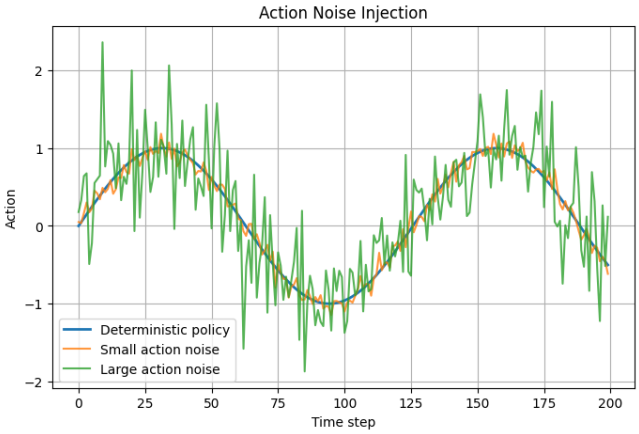
The perturbed policy $\pi_{\theta'}(a|s)$ induces coherent exploration across entire trajectories. Unlike action noise, which perturbs actions independently at each time step, parameter noise generates temporally consistent behavioral deviations. This corresponds to exploration in policy space rather than action space and often produces more structured long-horizon exploration behavior.

Figure 35d illustrates entropy-regularized exploration. The entropy-regularized objective is

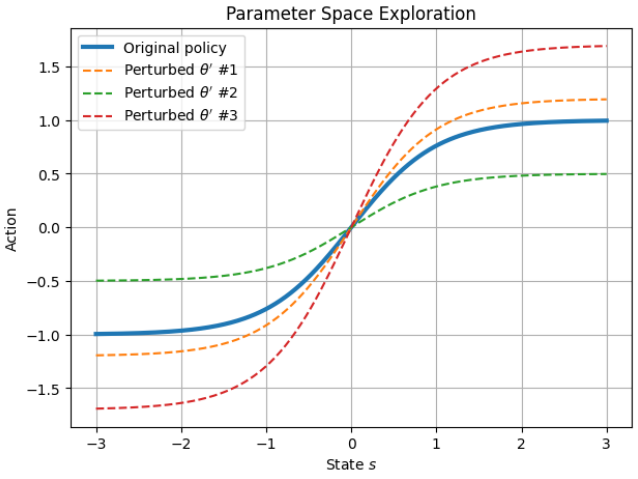
$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_t \gamma^t R_t \right] + \lambda\, \mathbb{E}_{\pi_\theta} \left[\sum_t \gamma^t \mathcal{H}(\pi_\theta(\cdot|s_t)) \right], \tag{903}$$



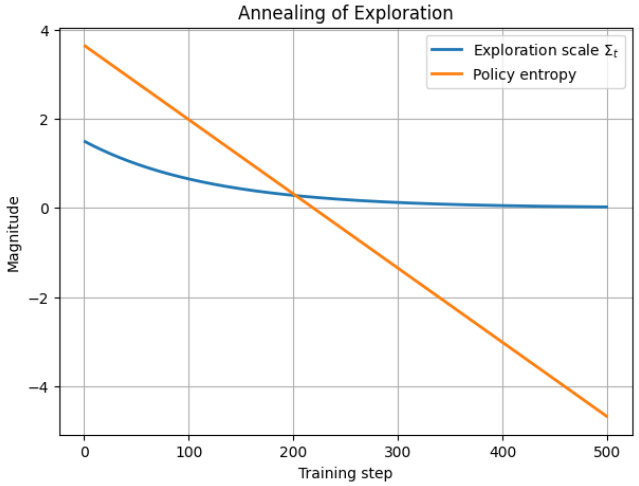
(a) **Gaussian stochastic policy in continuous action spaces.**



(b) **Action-space noise injection for exploration.**



(c) **Parameter-space noise inducing coherent exploration.**



(d) **Entropy-regularized stochastic policy dynamics.**

Figure 35: **Illustration of exploration mechanisms in continuous action spaces. The figures compare stochastic Gaussian policies, action-space noise, parameter-space perturbations, and entropy-regularized exploration.**

where

$$\mathcal{H}(\pi_\theta(\cdot|s)) = -\int \pi_\theta(a|s) \log \pi_\theta(a|s)\, da \tag{904}$$

denotes the differential entropy of the continuous policy distribution. The figure shows how larger entropy regularization maintains broader action distributions and sustained stochasticity, whereas smaller entropy coefficients lead to sharper, near-deterministic policies. From an optimization perspective, entropy regularization smooths the policy landscape and stabilizes gradient-based learning.

Collectively, these figures (Figure 35a, 35b, 35c, and 35d) demonstrate the major mechanisms for exploration in continuous reinforcement learning: stochastic policy sampling, additive action perturbations, parameter-space randomization, and entropy-based regularization. Mathematically, all four approaches aim to ensure sufficient coverage of the continuous action space while balancing exploration scale, stability, and convergence efficiency.

In summary, exploration in continuous action spaces relies on structured stochastic perturbations in action space, policy space, or parameter space. These methods ensure sufficient coverage of $\mathcal{A}$ while enabling gradient-based optimization, and they play a central role in modern reinforcement learning algorithms for continuous control.

10.7 Exploration Challenges in High Dimensions

In high-dimensional state or action spaces, the exploration–exploitation trade-off becomes substantially more challenging due to the curse of dimensionality. Let $\mathcal{S} \subseteq \mathbb{R}^d$ with large d. The volume of the state space grows exponentially with d, and consequently, the probability that a randomly sampled trajectory visits a rewarding region can be exponentially small. Formally, if $\mathcal{S}_{\text{reward}} \subset \mathcal{S}$ denotes the subset of rewarding states, then under naive random exploration,

$$\mathbb{P}\left(s_t \in \mathcal{S}_{\text{reward}}\right) \approx \frac{\text{Vol}(\mathcal{S}_{\text{reward}})}{\text{Vol}(\mathcal{S})} \ll 1, \tag{905}$$

which decays rapidly as $\dim(\mathcal{S})$ increases. This leads to extremely slow learning in sparse-reward environments.

To address this inefficiency, modern exploration strategies incorporate additional structure to guide the agent toward informative or novel states. A prominent approach is *intrinsic motivation*, where the reward function is augmented with an internal signal:

$$R_t^{\text{total}} = R_t^{\text{extrinsic}} + \beta\, R_t^{\text{intrinsic}}, \tag{906}$$

with $\beta > 0$ controlling the exploration strength. The intrinsic reward is typically designed to reflect novelty or prediction error. For example, one may define

$$R_t^{\text{intrinsic}} = \|f_\phi(s_{t+1}) - \hat{f}_\phi(s_t, a_t)\|^2, \tag{907}$$

where f_ϕ is a feature representation and $\hat{f}_\phi$ is a learned predictive model. Large prediction errors indicate unfamiliar regions of the state space, thereby encouraging exploration.

Another principled strategy is *count-based exploration*. In finite state spaces, one may define an exploration bonus

$$b_t(s) = \frac{c}{\sqrt{N_t(s)}}, \tag{908}$$

where $N_t(s)$ is the visitation count. However, in high-dimensional or continuous spaces, exact counts are infeasible. This motivates *pseudo-counts* $\hat{N}_t(s)$ derived from density models. Let $\rho_t(s)$ be a learned density estimator; then pseudo-counts are defined implicitly via

$$\hat{N}_t(s) \approx \frac{\rho_t(s)(1 - \rho'_t(s))}{\rho'_t(s) - \rho_t(s)}, \tag{909}$$

where $\rho'_t(s)$ is the updated density after observing s. The exploration bonus becomes

$$b_t(s) = \frac{c}{\sqrt{\hat{N}_t(s)}}, \tag{910}$$

which generalizes count-based methods to continuous domains.

A third approach is *exploration via representation learning*, where the agent learns a feature map $\phi : \mathcal{S} \to \mathbb{R}^k$ that captures meaningful structure in the environment. Exploration is then performed in the latent space rather than the raw state space. For instance, one may define novelty as

$$\text{Novelty}(s) = \min_{s' \in \mathcal{D}_t} \|\phi(s) - \phi(s')\|, \tag{911}$$

where $\mathcal{D}_t$ is the set of previously visited states. This encourages visiting states that are far from prior experience in representation space. From an information-theoretic viewpoint, these strategies aim to maximize information gain. Let θ denote unknown environment parameters. The intrinsic objective can be expressed as

$$\max \ \mathbb{E}\left[R_t^{\text{extrinsic}} + \beta\, I(\theta; s_{t+1} \mid s_t, a_t)\right], \tag{912}$$

where $I(\cdot;\cdot)$ denotes mutual information. Thus, exploration is guided toward actions that reduce uncertainty about the environment. In high-dimensional settings, *deep exploration* becomes essential. This refers to temporally extended exploration strategies that consider long-term information gain rather than immediate novelty. Formally, one seeks policies that maximize

$$\mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \left(R_t + \beta\, \mathcal{I}_t\right)\right],$$

where $\mathcal{I}_t$ captures multi-step information acquisition.

Despite these advances, exploration in high dimensions remains fundamentally difficult. The effectiveness of exploration depends critically on the quality of learned representations, the design of intrinsic rewards, and the ability to generalize across states. Poorly designed exploration signals may lead to pathological behaviors, such as exploring irrelevant regions or exploiting noise.

The four figures (Figure 36a, 36b, 36c, and 36d) illustrate several fundamental challenges and modern solutions for exploration in high-dimensional reinforcement learning. Figure 36a demonstrates the *curse of dimensionality*: as the dimension d of the state space increases, the probability of randomly visiting rewarding regions decays exponentially,

$$\mathbb{P}(s_t \in \mathcal{S}_{\text{reward}}) \approx \frac{\text{Vol}(\mathcal{S}_{\text{reward}})}{\text{Vol}(\mathcal{S})} \ll 1, \tag{913}$$

which leads to extremely inefficient random exploration in sparse-reward environments. The approximately exponential decay visible in the figure reflects the rapid growth of the ambient state-space volume with dimension.

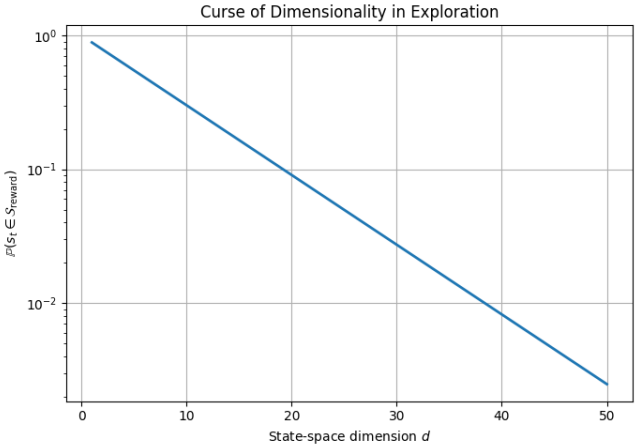
Figure 36b illustrates *intrinsic motivation*, where the total reward signal is augmented by an internal exploration bonus:

$$R_t^{\text{total}} = R_t^{\text{extrinsic}} + \beta R_t^{\text{intrinsic}}. \tag{914}$$

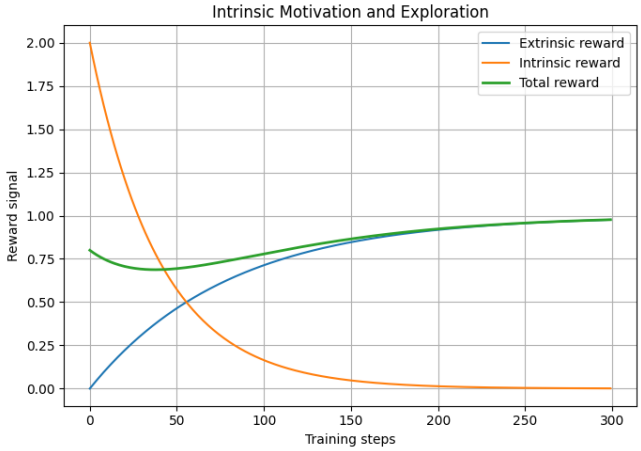
A common intrinsic reward is based on prediction error:

$$R_t^{\text{intrinsic}} = \|f_\phi(s_{t+1}) - \hat{f}_\phi(s_t, a_t)\|^2. \tag{915}$$

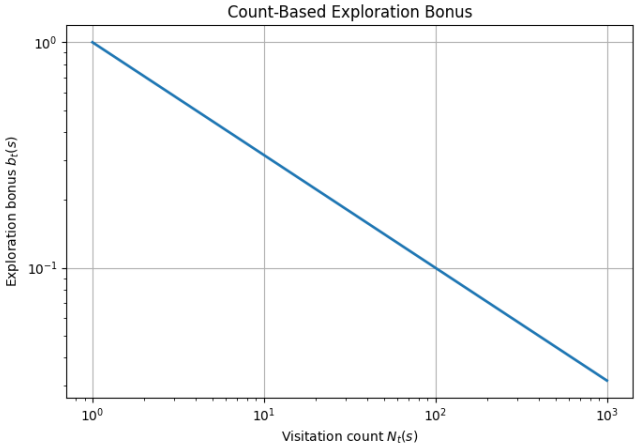
The figure shows that intrinsic rewards are initially large in unfamiliar regions and gradually decay as the environment becomes predictable, while extrinsic reward increases as the agent learns useful behaviors.



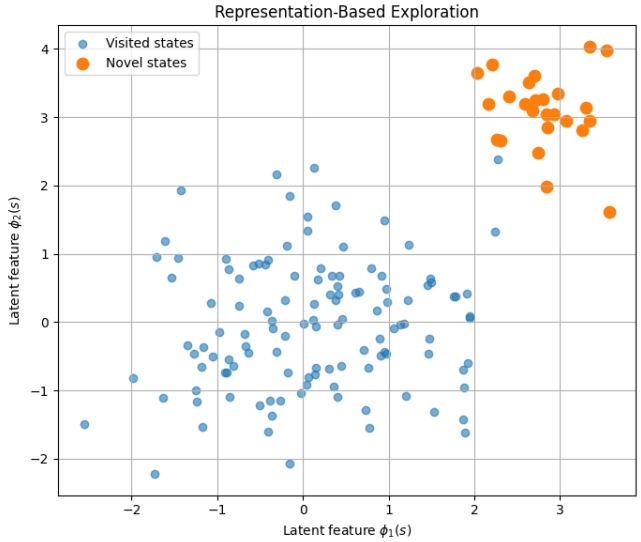
(a) **Exponential decay of random reward discovery probability in high-dimensional state spaces.**



(b) **Intrinsic motivation through prediction-error-based exploration rewards.**



(c) **Decay of count-based exploration bonuses with visitation frequency.**



(d) **Representation-based exploration in latent feature space.**

Figure 36: **Illustration of exploration challenges and structured exploration mechanisms in high-dimensional reinforcement learning. The figures demonstrate the curse of dimensionality, intrinsic motivation through prediction error, count-based exploration bonuses, and representation-learning-based novelty estimation in latent spaces.**

Figure 36c visualizes *count-based exploration*. The exploration bonus decreases with visitation frequency according to

$$b_t(s) = \frac{c}{\sqrt{N_t(s)}}, \tag{916}$$

or, in high-dimensional settings, using pseudo-counts,

$$b_t(s) = \frac{c}{\sqrt{\hat{N}_t(s)}}. \tag{917}$$

The logarithmic decay visible in the figure demonstrates how uncertainty bonuses shrink as states become increasingly well explored, thereby gradually transitioning the agent from exploration toward exploitation.

Figure 36d illustrates *representation-based exploration*, where novelty is measured in a learned latent feature space rather than the raw state space. Given a representation map

$$\phi : \mathcal{S} \to \mathbb{R}^k, \tag{918}$$

novelty may be quantified as

$$\text{Novelty}(s) = \min_{s' \in \mathcal{D}_t} \|\phi(s) - \phi(s')\|. \tag{919}$$

The figure shows previously visited states clustered in latent space, while newly discovered states appear in distant regions, illustrating how representation learning enables structured exploration in environments where raw-state comparisons are ineffective.

Taken together, these figures (Figure 36a, 36b, 36c, and 36d) demonstrate that naive random exploration becomes ineffective in high-dimensional environments, motivating the introduction of structured exploration mechanisms based on prediction error, visitation uncertainty, and learned latent representations. More generally, these approaches can be interpreted through an information-theoretic objective of the form

$$\max \ \mathbb{E}\left[R_t^{\text{extrinsic}} + \beta I(\theta; s_{t+1} \mid s_t, a_t)\right], \tag{920}$$

where $I(\theta; s_{t+1} \mid s_t, a_t)$ denotes information gain about unknown environment parameters. These methods therefore guide exploration toward informative regions of the state space, significantly improving sample efficiency in complex high-dimensional reinforcement learning problems.

In summary, naive random exploration is insufficient in high-dimensional or sparse-reward environments due to exponential scaling. Advanced methods—intrinsic motivation, pseudo-counts, and representation learning—introduce structure that guides exploration toward informative regions of the state space, thereby significantly improving sample efficiency and enabling learning in complex environments.

10.8 Trade-off as a Control Problem

The exploration–exploitation dilemma can be formalized as a higher-level stochastic control problem in which the decision variable is not merely an action a_t, but the policy π_t itself. In this meta-level formulation, the agent dynamically selects a sequence of policies $\{\pi_t\}_{t \geq 0}$ based on accumulated information, with the objective of maximizing cumulative reward:

$$\max_{\{\pi_t\}} \ \mathbb{E}\left[\sum_{t=0}^{T} R_t\right],$$

where the expectation is taken with respect to both the stochastic environment and the randomness induced by the policies.

To make this formulation precise, let θ denote unknown environment parameters (e.g., reward and transition kernels). The agent maintains a belief state $b_t := p(\theta \mid \mathcal{D}_t)$, where $\mathcal{D}_t$ is the history up to

time t. The pair (s_t, b_t) constitutes an augmented state, and the decision problem becomes a partially observable Markov decision process (POMDP) over the belief space. The dynamics evolve as

$$s_{t+1} \sim P(\cdot \mid s_t, a_t, \theta), \qquad b_{t+1} = \mathcal{B}(b_t, s_t, a_t, s_{t+1}), \tag{921}$$

where $\mathcal{B}$ denotes the Bayesian update operator. The optimal strategy solves the Bellman equation on the augmented space:

$$V(s, b) = \sup_{\pi(\cdot|s,b)} \left\{ \mathbb{E}_{a\sim\pi(\cdot|s,b)} \left[\mathbb{E}_{\theta\sim b}[R(s, a; \theta)] + \gamma\, \mathbb{E}_{s',b'}[V(s', b')] \right] \right\}. \tag{922}$$

Here, the expectation over b' reflects the evolution of beliefs, and the policy $\pi(\cdot|s, b)$ explicitly trades off immediate reward against information gain.

This formulation reveals that exploration is not an ad hoc mechanism but arises naturally as optimal control in belief space. Actions that are suboptimal in terms of immediate reward may be selected because they improve the belief b_t, thereby increasing future rewards. This is captured quantitatively through the decomposition

$$\mathbb{E}[R_t + \gamma V(s_{t+1}, b_{t+1})] = \underbrace{\mathbb{E}[R_t]}_{\text{exploitation}} + \underbrace{\gamma \left(\mathbb{E}[V(s_{t+1}, b_{t+1})] - \mathbb{E}[V(s_{t+1}, b_t)] \right)}_{\text{value of information}}. \tag{923}$$

An equivalent information-theoretic formulation augments the objective with an explicit information gain term. Let $I_t := I(\theta; s_{t+1} \mid \mathcal{D}_t, a_t)$ denote the mutual information between the unknown parameter and the next observation. Then one may consider

$$\max_{\{\pi_t\}} \mathbb{E} \left[\sum_{t=0}^{T} (R_t + \beta\, I_t) \right], \tag{924}$$

where $\beta > 0$ controls the exploration intensity. This objective explicitly rewards actions that reduce uncertainty about θ.

From a control-theoretic perspective, the exploration–exploitation trade-off can also be interpreted through dual control, where the control input a_t serves both to regulate the system and to probe it for information. The optimal policy must therefore solve a dual objective:

$$a_t \in \operatorname{argmax}_a \left\{ \mathbb{E}[R_t \mid a] + \gamma\, \mathbb{E}[V(s_{t+1}, b_{t+1}) \mid a] \right\}. \tag{925}$$

The second term depends implicitly on how informative the action a is, leading to a coupling between control and estimation.

Despite its conceptual clarity, solving the exact belief-space control problem is generally intractable due to the infinite-dimensional nature of the belief b_t. Consequently, practical algorithms approximate this framework through heuristics such as optimism, posterior sampling, or entropy regularization, each of which can be viewed as a tractable surrogate for the full control problem.

Figure 37a illustrates exploration in the augmented belief-state space (s_t, b_t), where the belief $b_t = p(\theta \mid \mathcal{D}_t)$ evolves through Bayesian updates:

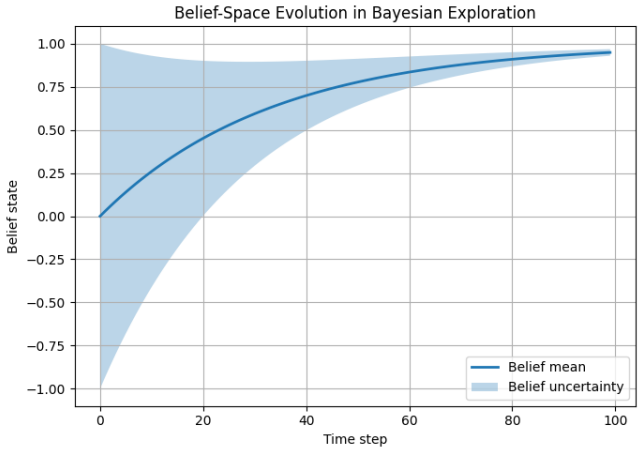
$$b_{t+1} = \mathcal{B}(b_t, s_t, a_t, s_{t+1}). \tag{926}$$

The optimal value function therefore satisfies a Bellman equation on the joint state-belief space:

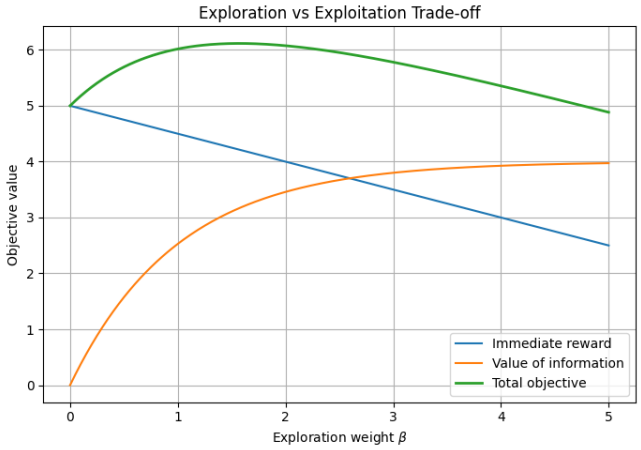
$$V(s, b) = \sup_{\pi} \mathbb{E} \left[R(s, a; \theta) + \gamma V(s', b') \right]. \tag{927}$$

Figure 37b visualizes the decomposition between exploitation and exploration through the value-of-information term:

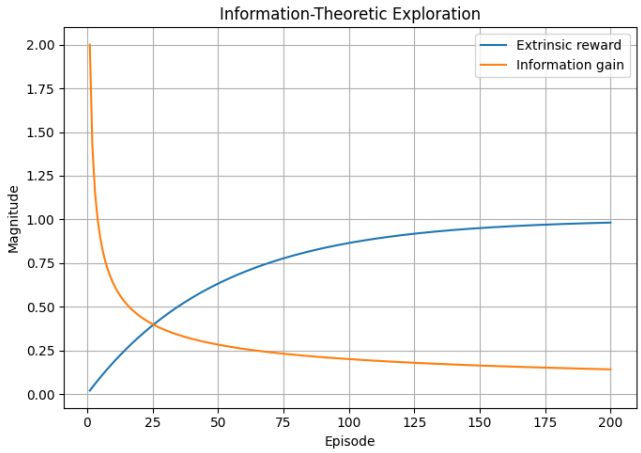
$$\mathbb{E}[R_t + \gamma V(s_{t+1}, b_{t+1})] = \underbrace{\mathbb{E}[R_t]}_{\text{immediate reward}} + \underbrace{\gamma \left(\mathbb{E}[V(s_{t+1}, b_{t+1})] - \mathbb{E}[V(s_{t+1}, b_t)] \right)}_{\text{information value}}. \tag{928}$$



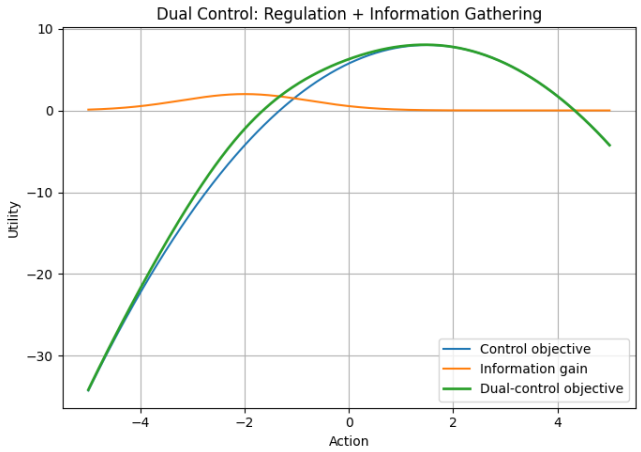
(a) **Belief-space evolution and exploration as Bayesian control.**



(b) **Trade-off between immediate reward and value of information.**



(c) **Information-theoretic exploration objective.**



(d) **Dual control interpretation of exploration and exploitation.**

Figure 37: **Exploration–exploitation viewed as a stochastic control problem.**

Figure 37c illustrates the information-theoretic formulation in which exploration is encouraged through mutual information:

$$\max_{\{\pi_t\}} \mathbb{E}\left[\sum_{t=0}^{T} (R_t + \beta I_t)\right], \qquad I_t = I(\theta; s_{t+1} \mid \mathcal{D}_t, a_t), \tag{929}$$

thereby explicitly rewarding uncertainty reduction. Finally, Figure 37d illustrates the dual-control viewpoint, where actions simultaneously regulate the environment and probe it for information:

$$a_t \in \operatorname{argmax}_a \left\{\mathbb{E}[R_t \mid a] + \gamma \mathbb{E}[V(s_{t+1}, b_{t+1}) \mid a]\right\}. \tag{930}$$

Collectively, the figures (Figure 37a, 37b, 37c, and 37d) demonstrate that exploration emerges naturally as optimal control under uncertainty, unifying reinforcement learning with Bayesian decision theory, adaptive control, and information-theoretic optimization.

In summary, framing the exploration–exploitation trade-off as a control problem reveals that exploration is an inherent component of optimal decision-making under uncertainty. The optimal policy simultaneously maximizes reward and reduces uncertainty, leading to a unified perspective that connects reinforcement learning with Bayesian decision theory, adaptive control, and information theory.

10.9 Summary and Theoretical Insights

The exploration–exploitation trade-off can be understood as a fundamental tension between *reward maximization* and *information acquisition*. At each time step t, the agent selects an action a_t that influences both the immediate reward and the information available for future decisions. This dual role can be formalized through the decomposition

$$\mathbb{E}[R_t + \gamma V_{t+1}] = \underbrace{\mathbb{E}[R_t]}_{\text{exploitation}} + \underbrace{\gamma\,(\mathbb{E}[V_{t+1}] - V_t)}_{\text{value of information}}, \tag{931}$$

where the second term captures the expected improvement in future value due to learning.

A central mathematical framework for analyzing this trade-off is *regret minimization*. The cumulative regret

$$\mathcal{R}(T) = \sum_{t=1}^{T} (\mu^* - \mu(a_t)) \tag{932}$$

quantifies the cost of exploration. Efficient algorithms ensure sublinear regret:

$$\lim_{T\to\infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0, \tag{933}$$

which implies asymptotic optimality. Lower bounds show that, in general,

$$\mathbb{E}[\mathcal{R}(T)] \geq \Omega(\log T), \tag{934}$$

highlighting that some level of exploration (and hence regret) is unavoidable. Uncertainty quantification plays a crucial role in guiding exploration. Confidence-based methods construct upper bounds

$$Q_t(s,a) \leq \hat{Q}_t(s,a) + b_t(s,a), \tag{935}$$

where $b_t(s,a)$ is chosen such that

$$\mathbb{P}\big(Q^*(s,a) \leq \hat{Q}_t(s,a) + b_t(s,a)\big) \geq 1 - \delta. \tag{936}$$

This leads to optimistic action selection:

$$a_t \in \operatorname{argmax}_a \left[\hat{Q}_t(s_t,a) + b_t(s_t,a)\right], \tag{937}$$

which ensures that uncertain actions are explored sufficiently.

An alternative perspective is provided by entropy-based methods, which introduce stochasticity directly into the policy. The entropy-regularized objective

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_t\right] + \lambda\, \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{H}(\pi(\cdot|s_t))\right] \tag{938}$$

encourages policies with higher entropy:

$$\mathcal{H}(\pi(\cdot|s)) = -\sum_{a} \pi(a|s) \log \pi(a|s). \tag{939}$$

The resulting optimal policy takes the form

$$\pi^*(a|s) \propto \exp\left(\frac{Q^*(s,a)}{\lambda}\right), \tag{940}$$

which smoothly interpolates between exploration and exploitation via the temperature parameter λ. Bayesian approaches provide a probabilistic treatment of uncertainty. Let θ denote unknown environment parameters with posterior $p(\theta|\mathcal{D}_t)$. Thompson sampling selects actions according to

$$\theta_t \sim p(\theta|\mathcal{D}_t), \quad a_t \in \operatorname{argmax}_a Q(s_t, a; \theta_t), \tag{941}$$

which induces a policy

$$\pi_t(a|s) = \mathbb{P}\big(a \in \operatorname{argmax}_{a'} Q(s,a';\theta) \mid \mathcal{D}_t\big)\,. \tag{942}$$

Thus, exploration arises naturally from posterior uncertainty, with actions chosen in proportion to their probability of being optimal. From an information-theoretic standpoint, exploration can be interpreted as maximizing mutual information. Let $I(\theta; s_{t+1} \mid s_t, a_t)$ denote the information gained about the environment. One may consider objectives of the form

$$\max\ \mathbb{E}\left[\sum_{t=0}^{T} \left(R_t + \beta\, I(\theta; s_{t+1} \mid s_t, a_t)\right)\right], \tag{943}$$

which explicitly reward informative actions.

These perspectives are unified by the principle that optimal exploration must balance immediate reward with uncertainty reduction. However, no single exploration strategy is universally optimal. The effectiveness of a method depends on several structural properties of the problem:

$$\text{Performance} = f\big(\dim(\mathcal{S},\mathcal{A}), \text{reward sparsity}, \text{model structure}, \text{prior information}\big). \tag{944}$$

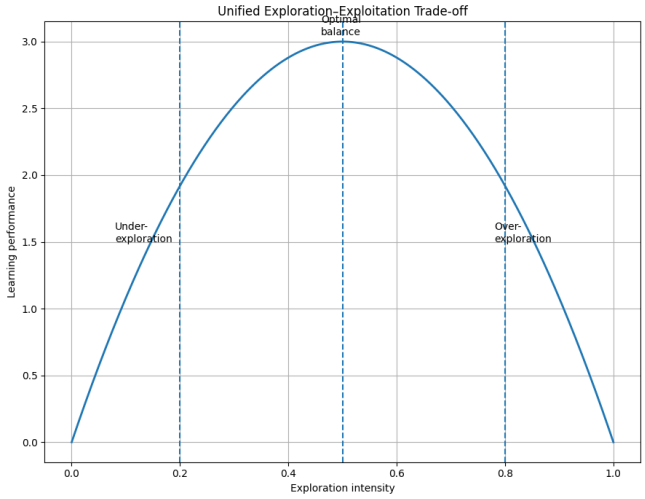
In high-dimensional or structured environments, methods leveraging representation learning or Bayesian structure are often necessary, while in simpler settings, confidence-based or entropy-based approaches may suffice.

The six figures (Figure 38a, 38b, 38c, 38d, 38e, and 38f) collectively illustrate the major theoretical formulations of the exploration–exploitation trade-off in reinforcement learning. Figure 38a visualizes the fundamental decomposition

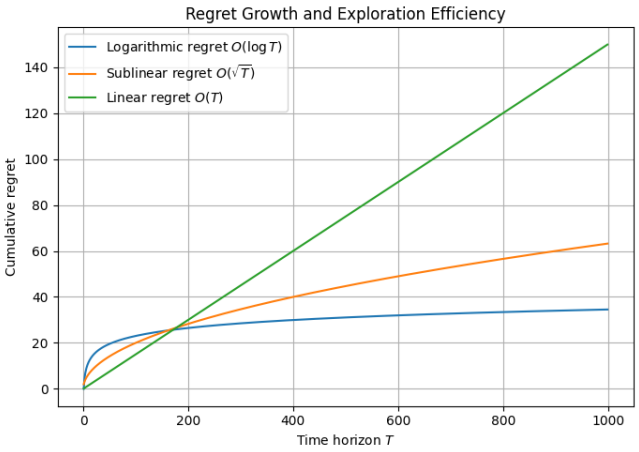
$$\mathbb{E}[R_t + \gamma V_{t+1}] = \underbrace{\mathbb{E}[R_t]}_{\text{exploitation}} + \underbrace{\gamma\,(\mathbb{E}[V_{t+1}] - V_t)}_{\text{value of information}}, \tag{945}$$

showing that optimal decision-making balances immediate reward maximization with long-term information acquisition. Figure 38b illustrates cumulative regret growth,

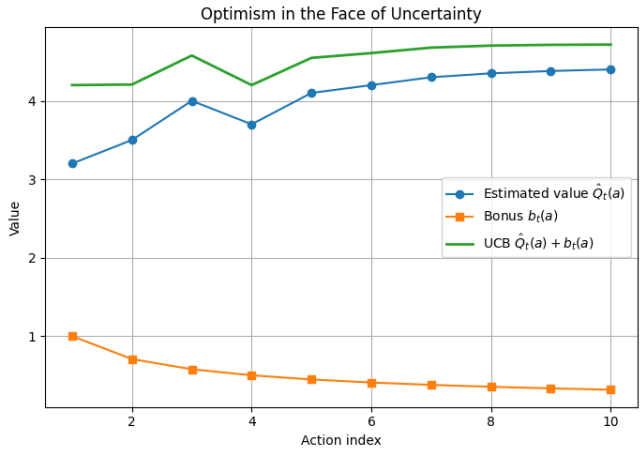
$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(\mu^* - \mu(a_t)\right), \tag{946}$$



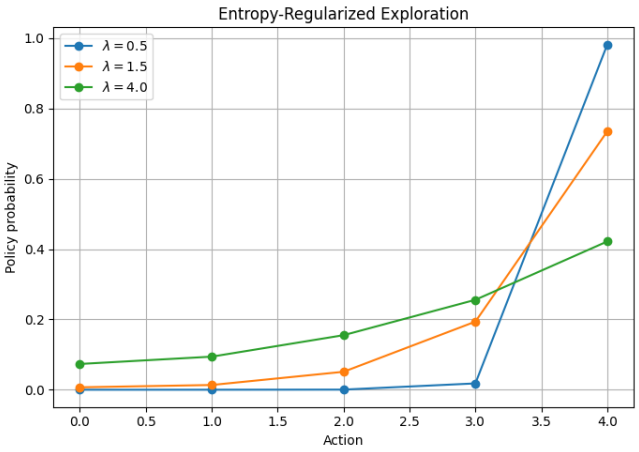
(a) **Reward maximization vs information acquisition**



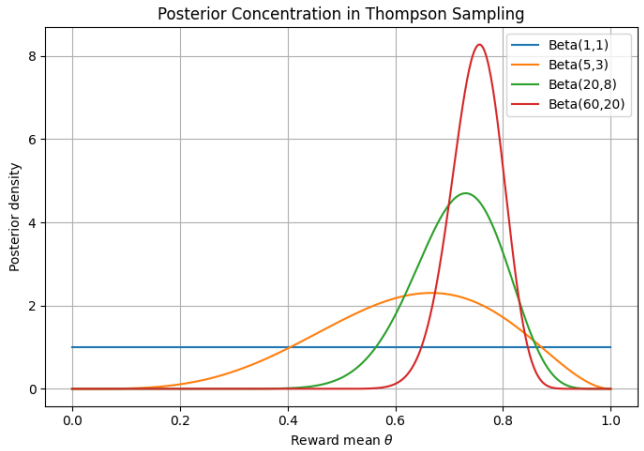
(b) **Sublinear regret growth**



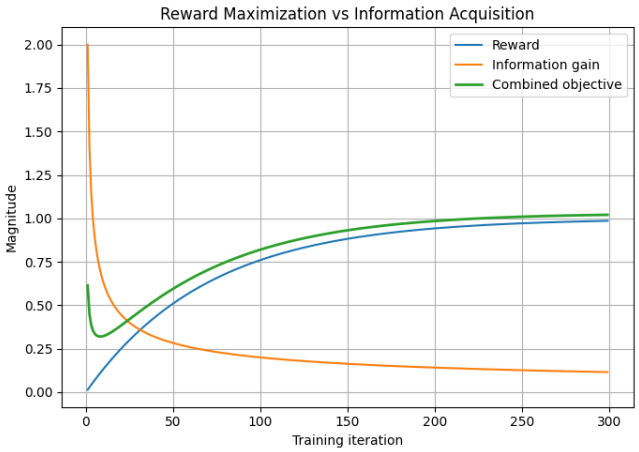
(c) **Optimism via confidence bounds**



(d) **Entropy-regularized stochastic policy**



(e) **Posterior sampling and Bayesian exploration**



(f) **Information-theoretic exploration**

Figure 38: **Illustration of major theoretical perspectives on the exploration–exploitation trade-off: reward-information decomposition, regret minimization, optimism under uncertainty, entropy-regularized exploration, Bayesian posterior sampling, and information-theoretic exploration.**

demonstrating that efficient exploration strategies achieve sublinear regret,

$$\lim_{T\to\infty} \frac{\mathbb{E}[\mathcal{R}(T)]}{T} = 0, \tag{947}$$

whereas inefficient exploration produces nearly linear regret accumulation. Figure 38c depicts optimism in the face of uncertainty through confidence-based exploration:

$$a_t \in \operatorname{argmax}_a \left[\hat{Q}_t(s_t, a) + b_t(s_t, a)\right], \tag{948}$$

where the uncertainty bonus

$$b_t(s, a) \propto \frac{1}{\sqrt{N_t(s, a)}} \tag{949}$$

encourages exploration of poorly sampled actions. Figure 38d illustrates entropy-regularized exploration, where the policy is obtained by maximizing

$$J(\pi) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t\right] + \lambda \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{H}(\pi(\cdot|s_t))\right], \tag{950}$$

leading to the Gibbs policy

$$\pi^*(a|s) \propto \exp\left(\frac{Q^*(s, a)}{\lambda}\right). \tag{951}$$

The figure shows how the temperature parameter λ controls the transition between deterministic exploitation and stochastic exploration. Figure 38e visualizes Bayesian exploration through posterior concentration in Thompson sampling:

$$\theta_t \sim p(\theta|\mathcal{D}_t), \qquad a_t \in \operatorname{argmax}_a Q(s_t, a; \theta_t), \tag{952}$$

where exploration naturally decreases as the posterior distribution concentrates around the true parameter. Finally, Figure 38f illustrates the information-theoretic perspective in which exploration maximizes both reward and information gain:

$$\max\ \mathbb{E}\left[\sum_{t=0}^{T} \left(R_t + \beta I(\theta; s_{t+1}|s_t, a_t)\right)\right]. \tag{953}$$

Collectively, these figures (Figure 38a, 38b, 38c, 38d, 38e, and 38f) demonstrate that exploration can be interpreted through multiple complementary viewpoints—regret minimization, uncertainty quantification, entropy regularization, Bayesian inference, and information theory—all unified by the principle that efficient learning requires balancing reward optimization with uncertainty reduction.

In summary, the exploration–exploitation trade-off admits multiple rigorous formulations—regret-theoretic, probabilistic, variational, and information-theoretic—all of which capture the same underlying principle: efficient learning requires systematically balancing reward optimization with uncertainty reduction.

11 Challenges

Reinforcement learning, while theoretically elegant, faces several fundamental challenges that arise from the interplay between stochasticity, function approximation, and sequential decision-making. These challenges are not merely engineering difficulties but reflect deep mathematical and statistical limitations inherent to learning in unknown environments.

11.1 Sample Inefficiency

Sample inefficiency remains one of the central challenges in reinforcement learning, referring to the large amount of interaction data often required for agents to learn effective policies, and this issue has been studied extensively by Kearns and Singh (2002) [49], Agarwal et al. (2020) [50], and Sutton and Barto (1998) [1]. Sutton and Barto (1998) [1] discussed sample inefficiency as a fundamental limitation of many reinforcement learning algorithms, particularly in high-dimensional environments where learning accurate value functions and policies from sparse rewards and noisy transitions may require extensive exploration and repeated interactions with the environment. Kearns and Singh (2002) [49] addressed this issue from a theoretical perspective by developing reinforcement learning algorithms with provable polynomial-time sample complexity guarantees, showing that near-optimal behavior can be achieved with a finite and theoretically bounded number of samples in Markov decision processes. Agarwal et al. (2020) [50] further investigated sample efficiency in the context of offline reinforcement learning, introducing optimistic approaches that leverage fixed datasets without additional environment interaction while analyzing the statistical and computational trade-offs involved in learning robust policies from previously collected experience. Together, these works established both the practical and theoretical dimensions of sample inefficiency in reinforcement learning, motivating the development of algorithms that improve data efficiency, reduce exploration costs, and enable reliable learning under limited interaction budgets.

A central and pervasive challenge in reinforcement learning is *sample inefficiency*, i.e., the requirement of a very large number of environment interactions to learn a near-optimal policy. This difficulty stems from the need to simultaneously estimate transition dynamics, reward structure, and long-term value functions under partial and noisy observations.

Formally, let π_t denote the policy after t interactions, and let V^* be the optimal value function. A natural measure of learning efficiency is the sample complexity

$$N(\varepsilon, \delta) := \inf \left\{ T : \mathbb{P}\left(\|V^{\pi_T} - V^*\|_\infty \le \varepsilon \right) \ge 1 - \delta \right\}, \tag{954}$$

which quantifies the number of samples required to achieve ε-optimality with high probability. Even in the tabular setting, lower bounds show that

$$N(\varepsilon, \delta) = \Omega\left(\frac{|\mathcal{S}||\mathcal{A}|}{\varepsilon^2 (1-\gamma)^3} \log \frac{1}{\delta} \right), \tag{955}$$

indicating a polynomial dependence on the size of the state-action space and a strong dependence on the discount factor γ.

The origin of this inefficiency can be traced to the need for accurate estimation of value functions. For example, the Bellman equation

$$V^\pi(s) = \mathbb{E}\left[R(s,a) + \gamma V^\pi(s') \mid s \right] \tag{956}$$

involves expectations over unknown transition kernels. Estimating these expectations from data introduces statistical error. If $\hat{P}$ and $\hat{R}$ denote empirical estimates, then the induced value function $\hat{V}^\pi$ satisfies

$$\|\hat{V}^\pi - V^\pi\|_\infty \le \frac{1}{1-\gamma} \left(\|\hat{R} - R\|_\infty + \gamma \sup_{s,a} \|\hat{P}(\cdot|s,a) - P(\cdot|s,a)\|_1 \|V^\pi\|_\infty \right), \tag{957}$$

showing that estimation errors are amplified by a factor $(1-\gamma)^{-1}$. From a regret perspective, one evaluates performance through

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(\mu^* - \mu(a_t) \right), \quad \mathbb{E}[\mathcal{R}(T)] = o(T). \tag{958}$$

Achieving sublinear regret requires sufficient exploration of all relevant state-action pairs. In particular, for many algorithms,

$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}\left(\sqrt{|\mathcal{S}||\mathcal{A}|T} \right), \tag{959}$$

which implies that a large number of exploratory steps are necessary before exploitation dominates.

A key difficulty is the *exploration burden*. To reliably estimate $Q^*(s,a)$, each state-action pair must be visited sufficiently often:

$$N_t(s,a) \to \infty \quad \text{as } t \to \infty. \tag{960}$$

However, in large or continuous spaces, uniform exploration is infeasible. Consequently, the agent must balance targeted exploration with limited data, which slows convergence. Another perspective arises from concentration inequalities. Let $\hat{Q}_t(s,a)$ be an empirical estimate of $Q^*(s,a)$. Then, with high probability,

$$|\hat{Q}_t(s,a) - Q^*(s,a)| \leq \mathcal{O}\left(\sqrt{\frac{\log t}{N_t(s,a)}}\right). \tag{961}$$

Thus, achieving a uniform error ε requires

$$N_t(s,a) = \Omega\left(\frac{\log t}{\varepsilon^2}\right), \tag{962}$$

for each (s,a), leading to overall sample complexity scaling with $|\mathcal{S}||\mathcal{A}|$.

In practical domains such as robotics or healthcare, where each interaction may be expensive or risky, this inefficiency becomes a critical limitation. As a result, significant research focuses on improving sample efficiency through mechanisms such as model-based learning, off-policy evaluation, experience replay, and transfer learning.

The six figures (Figure 39a, 39b, 39c, 39d, 39e, and 39f) collectively illustrate the fundamental sources and consequences of sample inefficiency in reinforcement learning. Figure 39a depicts the rapid growth of the sample complexity

$$N(\varepsilon,\delta) = \inf\left\{T : \mathbb{P}\left(\|V^{\pi_T} - V^*\|_\infty \leq \varepsilon\right) \geq 1-\delta\right\}, \tag{963}$$

as the target accuracy ε decreases. The approximately inverse-square scaling

$$N(\varepsilon,\delta) = \Omega\left(\frac{|\mathcal{S}||\mathcal{A}|}{\varepsilon^2(1-\gamma)^3}\log\frac{1}{\delta}\right) \tag{964}$$

illustrates that achieving highly accurate value estimation requires a very large number of interactions. Figure 39b visualizes the amplification factor

$$\frac{1}{1-\gamma}, \tag{965}$$

which appears in error propagation bounds such as

$$\|\hat{V}^\pi - V^\pi\|_\infty \leq \frac{1}{1-\gamma}\left(\|\hat{R} - R\|_\infty + \gamma \sup_{s,a}\|\hat{P}(\cdot|s,a) - P(\cdot|s,a)\|_1 \|V^\pi\|_\infty\right). \tag{966}$$

As $\gamma \to 1$, small estimation errors accumulate over long horizons and become significantly magnified. Figure 39c compares sublinear and linear regret growth,

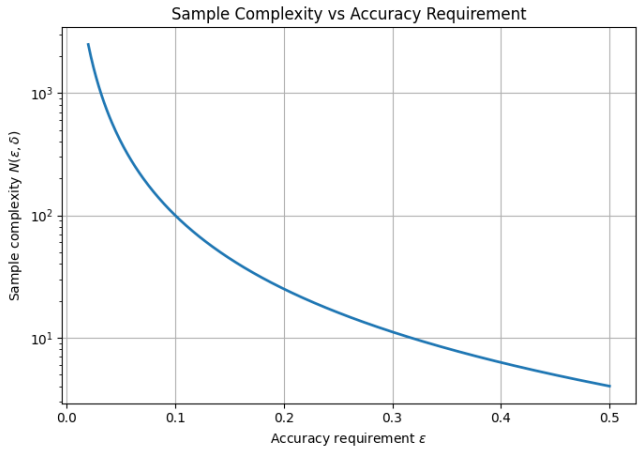
$$\mathcal{R}(T) = \sum_{t=1}^{T}\left(\mu^* - \mu(a_t)\right), \tag{967}$$

demonstrating that efficient exploration strategies aim to achieve

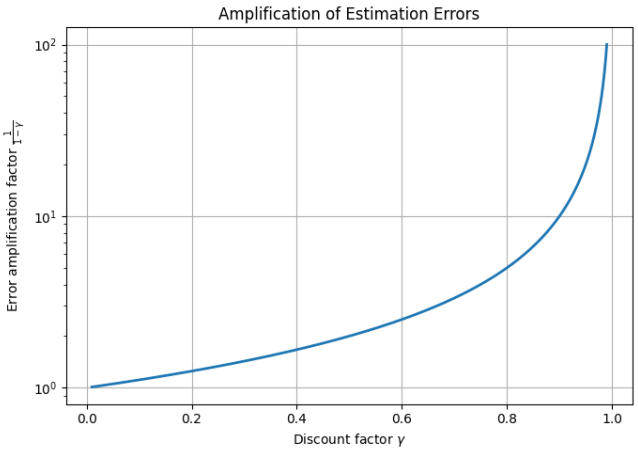
$$\mathbb{E}[\mathcal{R}(T)] = \mathcal{O}(\sqrt{T}), \tag{968}$$

or similarly sublinear scaling, whereas poor exploration leads to persistent linear regret accumulation. Figure 39d illustrates statistical concentration behavior governing value estimation:

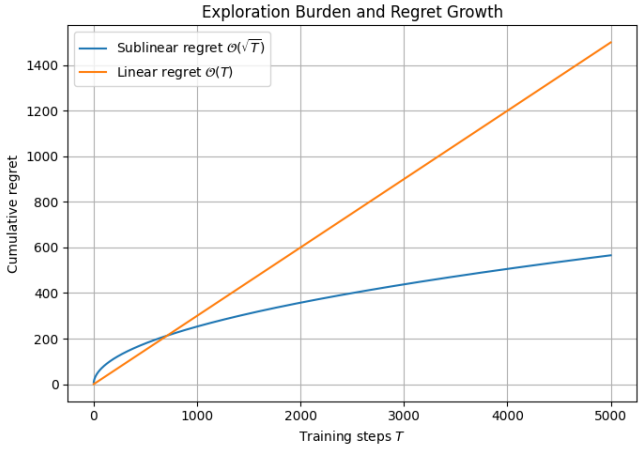
$$|\hat{Q}_t(s,a) - Q^*(s,a)| \leq \mathcal{O}\left(\sqrt{\frac{\log t}{N_t(s,a)}}\right). \tag{969}$$



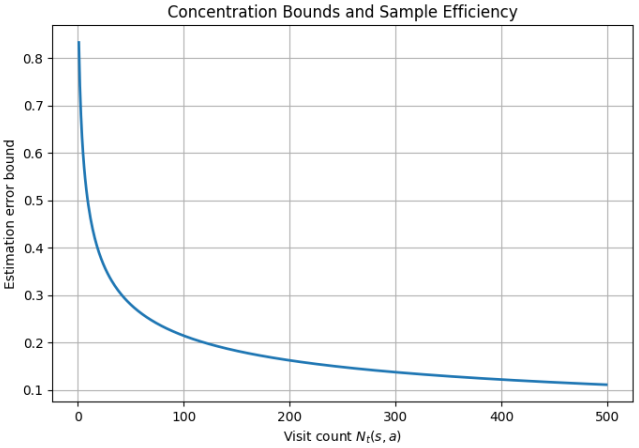
(a) **Sample complexity versus accuracy requirement**



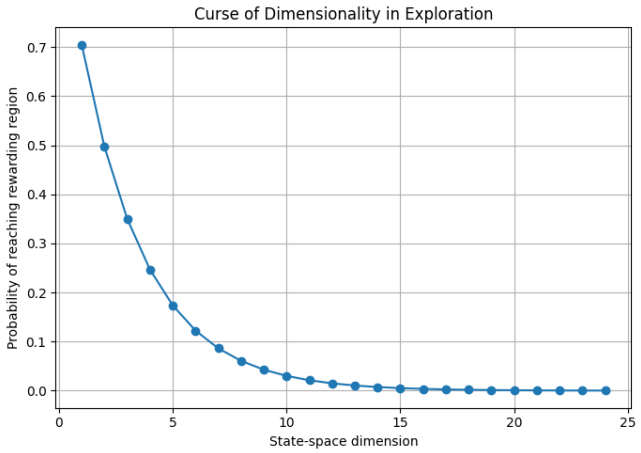
(b) **Error amplification due to discounting**



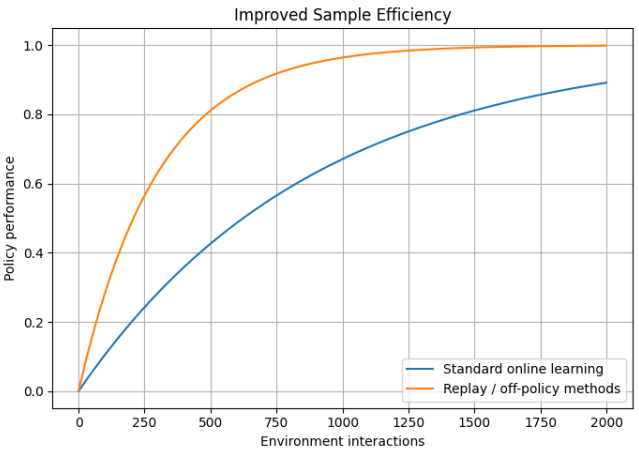
(c) **Regret growth during exploration**



(d) **Concentration bounds and estimation accuracy**



(e) **Curse of dimensionality in exploration**



(f) **Improved sample efficiency using replay methods**

Figure 39: **Illustration of sample inefficiency in reinforcement learning: dependence of sample complexity on accuracy requirements, amplification of estimation errors through discounting, regret accumulation during exploration, concentration bounds governing value estimation, the curse of dimensionality in exploration, and improvements obtained through replay and off-policy learning methods.**

The decay of the confidence interval with increasing visitation count $N_t(s, a)$ demonstrates why repeated sampling of each state-action pair is required for accurate learning. Figure 39e visualizes the curse of dimensionality in exploration. The probability of randomly reaching informative or rewarding states decreases exponentially with dimension:

$$\mathbb{P}(s_t \in \mathcal{S}_{\text{reward}}) \approx \frac{\text{Vol}(\mathcal{S}_{\text{reward}})}{\text{Vol}(\mathcal{S})} \ll 1. \tag{970}$$

This explains why naive exploration becomes ineffective in large state spaces. Finally, Figure 39f illustrates how mechanisms such as experience replay and off-policy learning accelerate policy improvement by reusing past transitions more effectively. These approaches reduce the effective sample complexity and improve learning speed compared with purely online learning.

Collectively, the figures (Figure 39a, 39b, 39c, 39d, 39e, and 39f) demonstrate that sample inefficiency arises from statistical estimation error, long-horizon error propagation, exploration requirements, concentration phenomena, and the exponential scaling of high-dimensional spaces. They also illustrate how modern reinforcement learning methods attempt to mitigate these limitations through improved data reuse and structured exploration strategies.

In summary, sample inefficiency arises from the need to accurately estimate long-term value functions under uncertainty, the requirement of sufficient exploration across large state-action spaces, and the amplification of estimation errors through temporal propagation. It remains one of the most fundamental obstacles to deploying reinforcement learning in real-world systems.

11.2 Instability in Function Approximation

Instability in function approximation is one of the fundamental theoretical and practical challenges in reinforcement learning, particularly when value functions are represented using parameterized approximators such as neural networks, and this issue has been extensively studied by Baird (1995) [51], Sutton and Barto (1998) [1], Tsitsiklis and Van Roy (1996) [25], and Ghosh (2025) [22]. Sutton and Barto (1998) [1] discussed how the combination of bootstrapping, off-policy learning, and function approximation can lead to divergence and unstable learning dynamics in temporal-difference methods, highlighting the difficulties that arise when reinforcement learning algorithms operate in large or continuous state spaces. Baird (1995) [51] addressed these challenges through the development of residual algorithms, proposing methods that minimize Bellman residual errors more directly in order to improve convergence behavior and reduce instability associated with approximate dynamic programming and temporal-difference learning. Tsitsiklis and Van Roy (1996) [25] provided rigorous mathematical analyses of temporal-difference learning with linear function approximation, establishing convergence properties under certain conditions while also demonstrating situations in which instability and divergence can occur due to the interaction between stochastic approximation dynamics and approximate value representations. Theoretical aspects of instability arising from nonlinear function approximation and deep neural optimization are also examined in Ghosh (2025) [22], particularly in relation to convergence dynamics, approximation sensitivity, and training behavior in deep learning models. Together, these works established the theoretical foundations for understanding instability in approximate reinforcement learning, motivating the development of more stable optimization methods, residual minimization techniques, and modern stabilization strategies used in deep reinforcement learning.

When value functions or policies are represented using parameterized function classes, such as linear architectures or deep neural networks, the resulting learning dynamics are governed by coupled, nonlinear stochastic recursions. This introduces significant sources of instability that are absent in classical tabular settings.

Let $V(s; \theta)$ denote a parameterized value function. A typical temporal-difference (TD) update takes the form

$$\theta_{t+1} = \theta_t + \alpha_t \, \delta_t \, \nabla_\theta V(s_t; \theta_t), \tag{971}$$

where the TD error is

$$\delta_t = R_{t+1} + \gamma V(s_{t+1}; \theta_t) - V(s_t; \theta_t). \tag{972}$$

This update can be interpreted as a stochastic gradient step on a *bootstrapped* objective, where the target itself depends on the current parameter θ_t. Unlike supervised learning, where targets are fixed, here the target

$$y_t(\theta_t) := R_{t+1} + \gamma V(s_{t+1}; \theta_t) \tag{973}$$

moves as θ_t evolves, leading to a nonstationary optimization problem.

More generally, the update can be written as a stochastic approximation:

$$\theta_{t+1} = \theta_t + \alpha_t \, F(\theta_t, \xi_t), \tag{974}$$

where ξ_t captures randomness from the Markov process and F encodes the update direction. The expected dynamics are governed by

$$\bar{F}(\theta) := \mathbb{E}[F(\theta, \xi)], \tag{975}$$

and the asymptotic behavior is often analyzed via the limiting ordinary differential equation (ODE)

$$\dot{\theta}(t) = \bar{F}(\theta(t)). \tag{976}$$

A key difficulty is that $\bar{F}$ is generally nonlinear and may not be a gradient field, i.e., there may not exist a scalar objective $L(\theta)$ such that

$$\bar{F}(\theta) = -\nabla_\theta L(\theta). \tag{977}$$

Consequently, standard convergence guarantees from convex optimization do not apply. Fixed points θ^* satisfying

$$\bar{F}(\theta^*) = 0 \tag{978}$$

may fail to correspond to optimal value functions and can even be unstable equilibria:

$$\exists\, \theta^* \text{ such that } \nabla \bar{F}(\theta^*) \text{ has eigenvalues with positive real part.} \tag{979}$$

Instability is particularly severe in the *off-policy* setting, where updates are performed using data generated from a behavior policy $\mu \neq \pi$. In this case, the expected update involves a mismatch:

$$\bar{F}(\theta) = \mathbb{E}_\mu \left[(R_{t+1} + \gamma V(s_{t+1}; \theta) - V(s_t; \theta)) \, \nabla_\theta V(s_t; \theta) \right], \tag{980}$$

which need not be a contraction. In fact, even for linear function approximation

$$V(s; \theta) = \phi(s)^\top \theta, \tag{981}$$

the update reduces to

$$\theta_{t+1} = \theta_t + \alpha_t \left(b - A\theta_t \right), \tag{982}$$

where

$$A = \mathbb{E}_\mu \left[\phi(s_t)(\phi(s_t) - \gamma \phi(s_{t+1}))^\top \right], \quad b = \mathbb{E}_\mu \left[R_{t+1} \phi(s_t) \right]. \tag{983}$$

If A is not positive definite, the iteration may diverge:

$$\|\theta_t\| \to \infty \quad \text{as } t \to \infty. \tag{984}$$

Another source of instability is the *deadly triad*, consisting of:

$$\text{(i) function approximation,} \quad \text{(ii) bootstrapping,} \quad \text{(iii) off-policy learning.} \tag{985}$$

When all three are present, divergence can occur even in simple problems.

From a dynamical systems viewpoint, the learning process evolves on multiple time scales. For instance, in actor-critic methods,

$$\theta_{t+1} = \theta_t + \alpha_t \nabla_\theta J(\theta_t, w_t), \quad w_{t+1} = w_t + \beta_t \, G(w_t, \theta_t), \tag{986}$$

with $\beta_t \gg \alpha_t$. The coupled ODE system

$$\dot{\theta}(t) = \bar{g}(\theta(t), w(t)), \quad \dot{w}(t) = \bar{h}(w(t), \theta(t)) \tag{987}$$

may exhibit complex behaviors, including limit cycles or instability, unless carefully controlled. To mitigate these issues, practical algorithms introduce stabilization mechanisms such as:

$$\text{(i) target networks: } \theta^- \approx \theta \text{ (slow updates)}, \tag{988}$$

$$\text{(ii) experience replay: decorrelation of samples}, \tag{989}$$

$$\text{(iii) gradient clipping or regularization.} \tag{990}$$

These heuristics can be interpreted as modifying the effective operator to restore approximate contraction properties.

The six figures (Figure 40a, 40b, 40c, 40d, 40e, and 40f) collectively illustrate the major sources of instability that arise in reinforcement learning with function approximation. Figure 40a visualizes the nonstationary nature of temporal-difference learning targets. Unlike supervised learning, the target itself depends on the evolving parameter vector:

$$y_t(\theta_t) = R_{t+1} + \gamma V(s_{t+1}; \theta_t), \tag{991}$$

leading to updates of the form

$$\theta_{t+1} = \theta_t + \alpha_t \delta_t \nabla_\theta V(s_t; \theta_t), \tag{992}$$

where

$$\delta_t = R_{t+1} + \gamma V(s_{t+1}; \theta_t) - V(s_t; \theta_t). \tag{993}$$

The oscillatory mismatch between the value estimate and the moving target demonstrates how bootstrapping creates a continuously shifting optimization landscape. Figure 40b illustrates the ordinary differential equation (ODE) viewpoint of stochastic approximation:

$$\dot{\theta}(t) = \bar{F}(\theta(t)), \tag{994}$$

where

$$\bar{F}(\theta) = \mathbb{E}[F(\theta, \xi)]. \tag{995}$$

Stable trajectories converge toward equilibrium points, whereas unstable trajectories diverge exponentially. Instability occurs when the Jacobian

$$\nabla \bar{F}(\theta^*) \tag{996}$$

possesses eigenvalues with positive real parts. Figure 40c demonstrates divergence caused by off-policy learning. For linear function approximation,

$$V(s; \theta) = \phi(s)^\top \theta, \tag{997}$$

the expected update becomes

$$\theta_{t+1} = \theta_t + \alpha_t (b - A\theta_t), \tag{998}$$

with

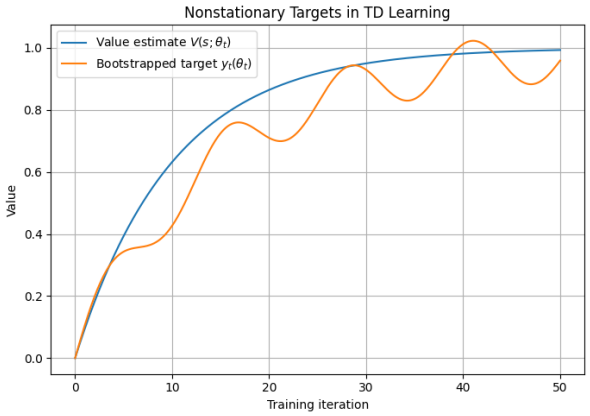
$$A = \mathbb{E}_\mu \left[\phi(s_t)(\phi(s_t) - \gamma \phi(s_{t+1}))^\top \right]. \tag{999}$$

If A is not positive definite, the parameter norm may diverge:

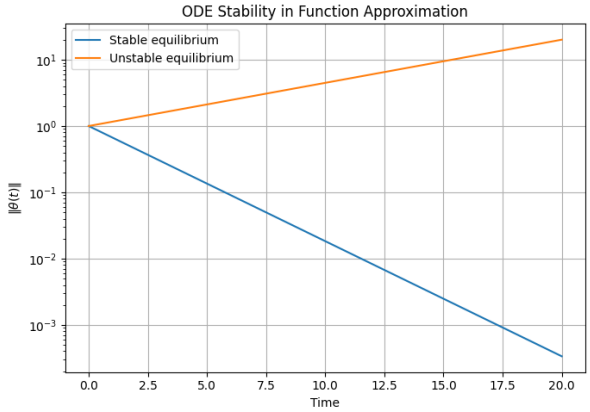
$$\|\theta_t\| \to \infty. \tag{1000}$$

The figure contrasts stable on-policy convergence with unstable off-policy growth. Figure 40d illustrates the interaction of the deadly triad:

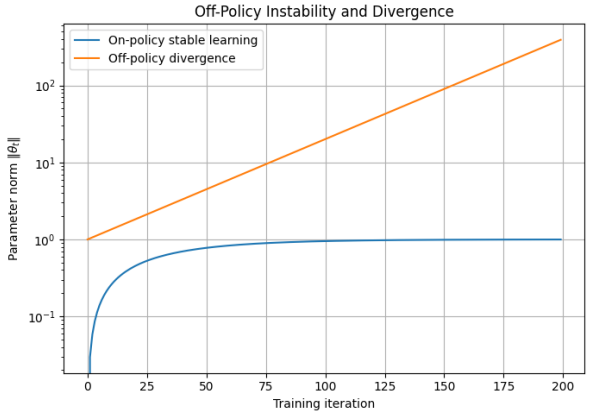
$$\text{(i) function approximation}, \quad \text{(ii) bootstrapping}, \quad \text{(iii) off-policy learning}. \tag{1001}$$



(a) **Nonstationary bootstrapped targets in TD learning**

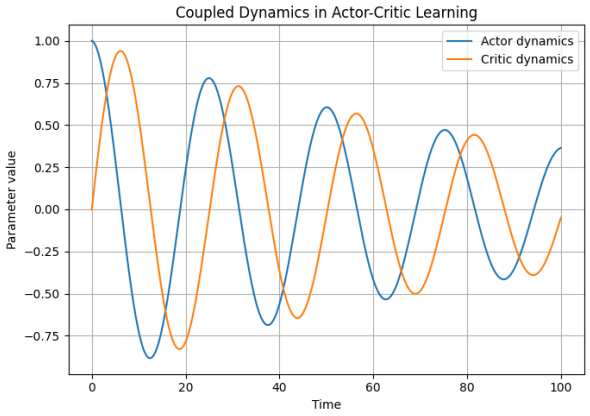


(b) **Stable and unstable ODE dynamics**

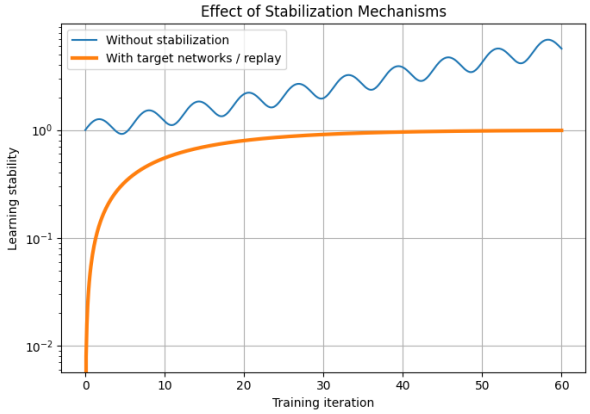


(c) **Off-policy divergence in function approximation**

(d) **Interaction of the deadly triad**



(e) **Coupled actor–critic dynamics**



(f) **Stabilization using replay and target networks**

Figure 40: **Illustration of instability phenomena arising in reinforcement learning with function approximation, including nonstationary targets, unstable stochastic approximation dynamics, off-policy divergence, the deadly triad, oscillatory actor–critic behavior, and stabilization mechanisms such as target networks and experience replay.**

Individually, each component may be manageable, but their combined interaction can amplify instability and produce divergence. Figure 40e depicts oscillatory behavior in coupled actor–critic systems governed by multi-timescale stochastic dynamics:

$$\theta_{t+1} = \theta_t + \alpha_t \nabla_\theta J(\theta_t, w_t), \tag{1002}$$

$$w_{t+1} = w_t + \beta_t G(w_t, \theta_t), \qquad \beta_t \gg \alpha_t. \tag{1003}$$

The associated coupled ODE system

$$\dot{\theta}(t) = \bar{g}(\theta(t), w(t)), \qquad \dot{w}(t) = \bar{h}(w(t), \theta(t)) \tag{1004}$$

may exhibit persistent oscillations or limit-cycle behavior due to interactions between actor and critic updates. Finally, Figure 40f demonstrates how stabilization techniques mitigate divergence. Mechanisms such as target networks,

$$\theta^- \approx \theta, \tag{1005}$$

experience replay, and gradient regularization reduce oscillations and restore approximate contraction behavior in the effective update operator. The comparison between stabilized and unstabilized learning trajectories highlights the crucial role of these methods in practical deep reinforcement learning.

Collectively, the figures (Figure 40a, 40b, 40c, 40d, 40e, and 40f) demonstrate that instability in reinforcement learning emerges from nonlinear stochastic approximation, moving targets, lack of contraction, off-policy mismatch, and coupled dynamical interactions. They also illustrate how modern stabilization techniques modify the effective learning dynamics to improve convergence and robustness.

In summary, instability in function approximation arises from the interaction between nonstationary targets, stochastic approximation, and lack of contraction in the underlying operators. The resulting dynamics are governed by nonlinear systems that may fail to converge, making stability analysis and algorithm design a central challenge in modern reinforcement learning.

11.3 Reward Shaping

Reward shaping has become an important technique in reinforcement learning for accelerating learning and guiding exploration through modified reward signals, and its theoretical foundations were established by Ng et al. (1999) [52], further discussed by Sutton and Barto (1998) [1], and extended by Wiewiora (2003) [53]. Sutton and Barto (1998) [1] described reward shaping as a mechanism for incorporating prior knowledge into reinforcement learning systems by augmenting environmental rewards in ways that encourage desirable behaviors and improve learning efficiency while preserving long-term optimization objectives. Ng et al. (1999) [52] provided the foundational theoretical analysis of reward transformations by proving the policy invariance theorem for potential-based shaping functions, demonstrating that appropriately constructed shaping rewards preserve optimal policies while significantly improving learning speed and exploration efficiency. Wiewiora (2003) [53] further expanded these ideas by establishing the equivalence between potential-based reward shaping and specific forms of Q-value initialization, thereby clarifying the relationship between reward modification, prior knowledge incorporation, and accelerated value-function learning in temporal-difference reinforcement learning methods. Together, these works established reward shaping as a mathematically principled method for improving reinforcement learning efficiency while maintaining policy optimality, providing important theoretical insights into guided exploration and accelerated sequential decision-making.

The specification of the reward function is a central modeling decision in reinforcement learning, as it defines the optimization objective implicitly through the return

$$J(\pi) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{1006}$$

In principle, $R(s, a)$ should faithfully encode the true task objective; however, in practice it is often a proxy signal that is easier to specify or compute. This mismatch gives rise to the problem of *reward*

design, where improperly specified rewards can induce suboptimal or even pathological behaviors.

A principled approach to modifying rewards is *potential-based reward shaping*. Given a potential function $F : \mathcal{S} \to \mathbb{R}$, define the shaped reward

$$\tilde{R}(s, a, s') = R(s, a) + \gamma F(s') - F(s). \tag{1007}$$

This formulation ensures policy invariance. Indeed, for any policy π, the corresponding action-value functions satisfy

$$Q^{\pi}_{\tilde{R}}(s, a) = Q^{\pi}_{R}(s, a) + F(s), \tag{1008}$$

and hence the optimal action-value functions obey

$$Q^{*}_{\tilde{R}}(s, a) = Q^{*}_{R}(s, a) + F(s). \tag{1009}$$

Since the additive term $F(s)$ does not depend on a, the set of optimal actions is preserved:

$$\operatorname{argmax}_a Q^{*}_{\tilde{R}}(s, a) = \operatorname{argmax}_a Q^{*}_{R}(s, a). \tag{1010}$$

Thus, potential-based shaping modifies the learning dynamics without altering the optimal policy. From a telescoping sum argument, the cumulative shaped return satisfies

$$\begin{aligned}\sum_{t=0}^{T} \gamma^t \tilde{R}(s_t, a_t, s_{t+1}) &= \sum_{t=0}^{T} \gamma^t R(s_t, a_t) + \sum_{t=0}^{T} \gamma^t \big(\gamma F(s_{t+1}) - F(s_t)\big) \\ &= \sum_{t=0}^{T} \gamma^t R(s_t, a_t) - F(s_0) + \gamma^{T+1} F(s_{T+1}),\end{aligned}$$

which differs from the original return only by boundary terms. As $T \to \infty$ and $\gamma < 1$, the terminal term vanishes under boundedness assumptions, preserving optimality.

Despite this theoretical guarantee, reward shaping introduces several challenges. If the shaping function does not satisfy the potential-based structure, then policy invariance may fail:

$$\operatorname{argmax}_a Q^{*}_{\tilde{R}}(s, a) \neq \operatorname{argmax}_a Q^{*}_{R}(s, a), \tag{1011}$$

leading to biased solutions. Moreover, even potential-based shaping can distort the learning landscape by altering gradient magnitudes and exploration behavior. A major difficulty arises in *sparse reward* settings, where rewards are observed only rarely. For example,

$$R_t = \mathbf{1}\{s_t \in \mathcal{S}_{\text{goal}}\}, \tag{1012}$$

yields nonzero feedback only upon reaching a goal state. In such cases, the policy gradient estimator

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R_t \, \nabla_\theta \log \pi_\theta(a_t|s_t)\right] \tag{1013}$$

suffers from high variance and vanishing signal, since most trajectories contribute zero reward:

$$\mathbb{P}(R_t \neq 0) \ll 1. \tag{1014}$$

This leads to extremely slow learning and poor credit assignment. The *credit assignment problem* itself can be formalized as identifying the contribution of each action to the final return. In long-horizon problems,

$$G_0 = \sum_{t=0}^{\infty} \gamma^t R_t, \tag{1015}$$

the influence of early actions is mediated through many intermediate transitions, making it difficult to estimate gradients accurately.

To mitigate these issues, various techniques are employed. Baseline subtraction reduces variance:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t (R_t - b(s_t))\, \nabla_\theta \log \pi_\theta(a_t|s_t)\right], \tag{1016}$$

where $b(s)$ is a value function approximation. Intrinsic reward signals can also be introduced:

$$R_t^{\text{total}} = R_t^{\text{extrinsic}} + \beta R_t^{\text{intrinsic}}, \tag{1017}$$

to encourage exploration in sparse environments. Nevertheless, reward shaping remains a delicate process. Poorly designed rewards may incentivize unintended behaviors, a phenomenon often referred to as *reward hacking*. Formally, if $\tilde{R}$ is misaligned with the true objective R, then maximizing

$$\mathbb{E}^\pi\left[\sum_t \gamma^t \tilde{R}_t\right] \tag{1018}$$

may yield policies that perform poorly with respect to the true task.

The phenomena illustrated in Figures 41a, 41b, 41c, 41d, 41e, and 41f collectively demonstrate the theoretical and practical implications of reward shaping in reinforcement learning. Figure 41a illustrates potential-based reward shaping, where the modified reward

$$\tilde{R}(s, a, s') = R(s, a) + \gamma F(s') - F(s) \tag{1019}$$

augments the original reward by the difference of a potential function. As shown in the figure, the shaping term provides additional intermediate guidance while preserving the optimal policy. This invariance property follows from

$$Q^*_{\tilde{R}}(s, a) = Q^*_R(s, a) + F(s), \tag{1020}$$

which implies

$$\operatorname{argmax}_a Q^*_{\tilde{R}}(s, a) = \operatorname{argmax}_a Q^*_R(s, a). \tag{1021}$$

Figure 41b visualizes the telescoping structure of the cumulative shaped return:

$$\sum_{t=0}^{T} \gamma^t \tilde{R}(s_t, a_t, s_{t+1}) = \sum_{t=0}^{T} \gamma^t R(s_t, a_t) + \sum_{t=0}^{T} \gamma^t \left(\gamma F(s_{t+1}) - F(s_t)\right) \tag{1022}$$

$$= \sum_{t=0}^{T} \gamma^t R(s_t, a_t) - F(s_0) + \gamma^{T+1} F(s_{T+1}), \tag{1023}$$

showing that the shaping contributions collapse into boundary terms. Consequently, as $T \to \infty$ and $\gamma < 1$, the asymptotic optimal policy remains unchanged.

Figures 41c and 41d illustrate the challenges associated with sparse rewards and high-variance gradient estimation. In sparse-reward environments, rewards occur only rarely:

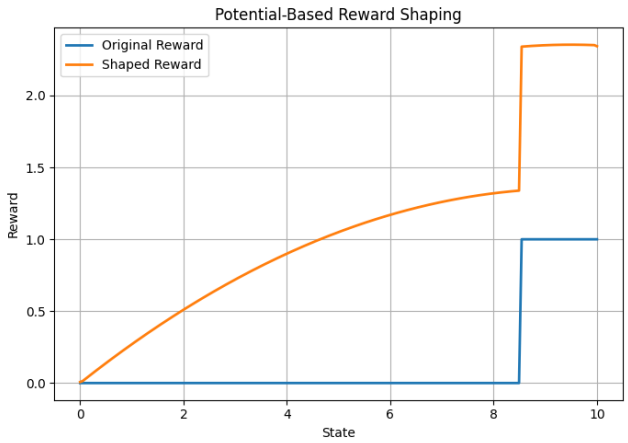
$$R_t = \mathbf{1}\{s_t \in \mathcal{S}_{\text{goal}}\}, \tag{1024}$$

so that

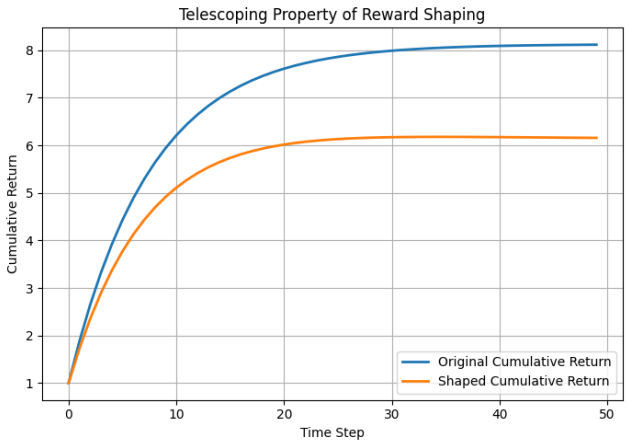
$$\mathbb{P}(R_t \neq 0) \ll 1. \tag{1025}$$

As depicted in Figure 41c, the agent may traverse long trajectories without receiving informative feedback, leading to extremely slow learning. This directly affects the policy gradient estimator shown in Figure 41d:

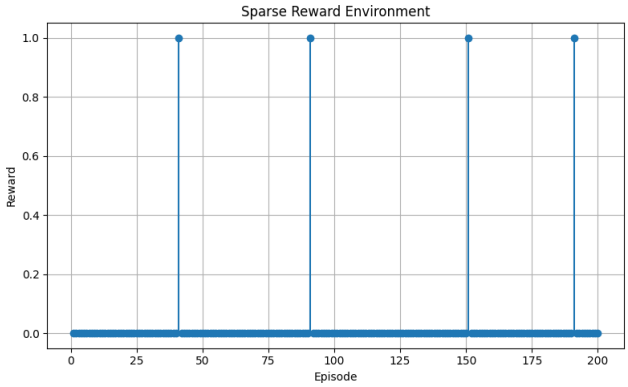
$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R_t \nabla_\theta \log \pi_\theta(a_t|s_t)\right], \tag{1026}$$



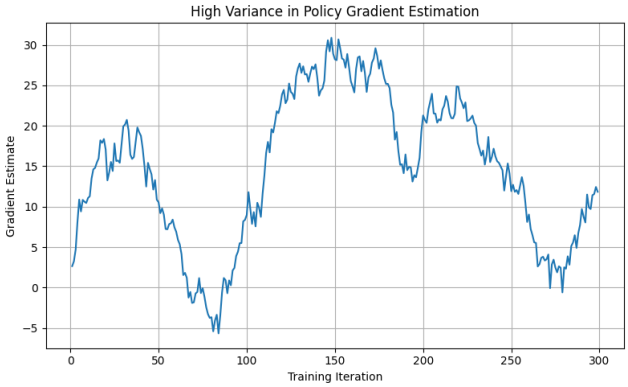
(a) **Potential-based reward shaping**



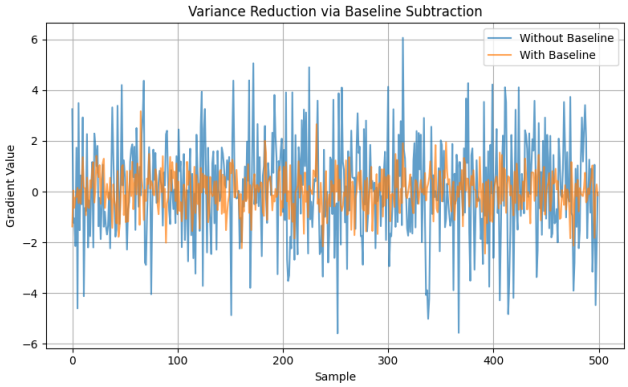
(b) **Telescoping property of shaped returns**



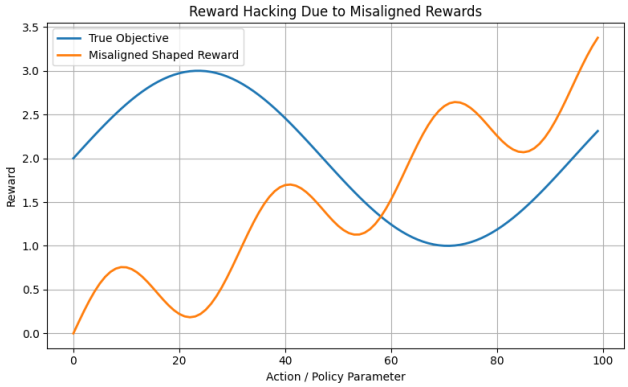
(c) **Sparse reward environment**



(d) **High-variance policy gradient estimation**



(e) **Variance reduction using baselines**



(f) **Reward hacking due to misaligned rewards**

Figure 41: **Illustrations of reward shaping, sparse rewards, variance reduction, and reward misalignment in reinforcement learning.**

whose variance becomes large because most sampled trajectories contribute negligible reward signal. The resulting noisy gradient estimates can destabilize optimization and hinder convergence.

Figure 41e demonstrates variance reduction through baseline subtraction. Introducing a baseline $b(s_t)$ modifies the gradient estimator to

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \big(R_t - b(s_t)\big) \nabla_\theta \log \pi_\theta(a_t|s_t)\right], \tag{1027}$$

which preserves unbiasedness while reducing estimator variance. As shown in the figure, the variance-reduced gradient exhibits significantly smoother optimization trajectories and more stable convergence behavior.

Finally, Figure 41f illustrates reward hacking caused by misaligned reward functions. If the shaped reward $\tilde{R}$ does not accurately reflect the true objective R, then optimizing

$$\max_\pi \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t \tilde{R}_t\right] \tag{1028}$$

may produce policies that achieve high proxy reward while performing poorly on the intended task. The figure demonstrates how an agent can exploit loopholes in the reward specification, thereby emphasizing that reward shaping must carefully balance informativeness with alignment to the true objective.

Taken together, Figures 41a, 41b, 41c, 41d, 41e, and 41f illustrate that reward shaping fundamentally modifies the optimization landscape of reinforcement learning. Properly designed shaping terms can accelerate learning and improve exploration while preserving optimality, whereas poorly specified rewards can induce instability, inefficient learning, or pathological behaviors.

In summary, reward shaping is a powerful but subtle tool. While potential-based shaping provides a theoretically sound mechanism for accelerating learning without altering optimal policies, practical implementations must carefully balance informativeness, stability, and alignment with the true objective to avoid bias and unintended consequences.

11.4 Curse of Dimensionality

The curse of dimensionality is one of the central challenges in dynamic programming and reinforcement learning, referring to the exponential growth in computational and storage complexity as the dimensionality of the state and action spaces increases, and this issue has been extensively studied by Bellman (1957) [15], Bertsekas and Tsitsiklis (1996) [2], Powell (2007) [54] and Ghosh (2025) [22]. Bellman (1957) [15] originally introduced the term "curse of dimensionality" while developing dynamic programming, observing that exact recursive optimization methods become computationally infeasible for high-dimensional sequential decision problems because the number of states and decision variables grows exponentially with system complexity. Bertsekas and Tsitsiklis (1996) [2] further analyzed this challenge within the framework of neuro-dynamic programming and approximate reinforcement learning, emphasizing the need for function approximation, asynchronous computation, simulation-based learning, and approximate value iteration techniques to overcome the intractability of exact dynamic programming in large-scale stochastic systems. Powell (2007) [54] expanded these ideas through the theory of approximate dynamic programming, developing computational methods based on value-function approximation, policy approximation, simulation optimization, and decomposition techniques for solving high-dimensional sequential optimization problems arising in engineering, operations research, and reinforcement learning. The challenges posed by high-dimensional representation spaces and the role of deep architectures in mitigating the curse of dimensionality are discussed in Ghosh (2025) [22], highlighting the mathematical principles that enable scalable approximation and hierarchical feature learning in complex systems. Together, these works established the curse of dimensionality as a fundamental limitation of exact stochastic optimization methods while also laying the theoretical and computational foundations for approximate reinforcement learning and scalable dynamic programming techniques in complex

high-dimensional environments.

The *curse of dimensionality* refers to the exponential growth of computational, statistical, and representational complexity as the dimension of the state or action space increases. Let $\mathcal{S} \subseteq \mathbb{R}^d$ denote the state space. A naive discretization with n grid points per dimension yields

$$|\mathcal{S}| = n^d, \tag{1029}$$

which becomes prohibitively large even for moderate d. This exponential scaling directly impacts dynamic programming methods, whose computational cost typically scales at least linearly with $|\mathcal{S}||\mathcal{A}|$.

From a statistical perspective, the curse manifests in the sample complexity required for accurate estimation. Suppose one seeks to approximate the value function V^* uniformly within error ε. In the absence of additional structure (e.g., smoothness or low-dimensional embeddings), nonparametric estimation theory implies that the required number of samples scales as

$$N(\varepsilon) = \Omega\left(\varepsilon^{-d}\right), \tag{1030}$$

reflecting the fact that high-dimensional spaces require exponentially many samples to achieve uniform coverage. This phenomenon can be understood geometrically. Let $\mathcal{S}_{\text{useful}} \subset \mathcal{S}$ denote the subset of states that yield informative or high-reward outcomes. If exploration is approximately uniform, then the probability of visiting such a region is

$$\mathbb{P}(s_t \in \mathcal{S}_{\text{useful}}) \approx \frac{\text{Vol}(\mathcal{S}_{\text{useful}})}{\text{Vol}(\mathcal{S})}. \tag{1031}$$

In high dimensions, volume concentrates in unintuitive ways, and typically

$$\frac{\text{Vol}(\mathcal{S}_{\text{useful}})}{\text{Vol}(\mathcal{S})} \to 0 \quad \text{as } d \to \infty, \tag{1032}$$

rendering random exploration ineffective.

The curse also affects optimization. Consider value iteration:

$$V_{k+1}(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V_k(s') P(ds'|s,a) \right]. \tag{1033}$$

Each update requires integration over $\mathcal{S}$, which becomes computationally expensive as d increases. Similarly, in continuous action spaces $\mathcal{A} \subseteq \mathbb{R}^m$, the maximization step

$$\sup_{a \in \mathcal{A}} Q(s,a) \tag{1034}$$

becomes a high-dimensional optimization problem. Function approximation is a primary tool for mitigating the curse by exploiting structure in the problem. For instance, linear architectures assume

$$V(s;\theta) = \phi(s)^\top \theta, \tag{1035}$$

where $\phi : \mathcal{S} \to \mathbb{R}^k$ is a feature map with $k \ll |\mathcal{S}|$. More generally, deep neural networks approximate

$$V(s;\theta) = f_\theta(s), \tag{1036}$$

where f_θ is a compositional function capable of representing complex patterns. Under favorable conditions (e.g., low intrinsic dimensionality or smoothness), such models can achieve approximation error

$$\|V^* - V(\cdot;\theta)\| \leq \varepsilon \tag{1037}$$

with sample complexity depending on the effective dimension rather than d.

However, function approximation introduces new challenges. Generalization error becomes a central concern:

$$\mathbb{E}\left[|V(s;\theta) - V^*(s)|\right] \tag{1038}$$

depends on both the expressiveness of the model and the distribution of training data. Moreover, optimization in high-dimensional parameter spaces is itself nontrivial, often involving nonconvex objectives:

$$\min_{\theta} \mathbb{E}\left[\left(V(s;\theta) - y(s)\right)^2\right], \tag{1039}$$

where $y(s)$ is a bootstrapped target. Another mitigating strategy is to exploit structural assumptions such as sparsity, compositionality, or low-rank structure. For example, if the value function decomposes as

$$V(s) = \sum_{i=1}^{k} V_i(s_i), \tag{1040}$$

with $k \ll d$, then the effective dimensionality is reduced. Similarly, manifold assumptions posit that states lie near a lower-dimensional subset $\mathcal{M} \subset \mathbb{R}^d$:

$$\dim(\mathcal{M}) \ll d, \tag{1041}$$

allowing more efficient learning.

Despite these advances, the curse of dimensionality remains a fundamental limitation. It affects exploration, estimation, and optimization simultaneously, and often interacts with other challenges such as sample inefficiency and instability.

Figures 42a, 42b, 42c, 42d, 42e and 42f illustrate several fundamental manifestations of the curse of dimensionality in reinforcement learning. Figure 39e(a) demonstrates the exponential growth of the state space cardinality

$$|\mathcal{S}| = n^d, \tag{1042}$$

showing that even a modest discretization resolution n leads to an enormous number of states as the dimension d increases. This exponential scaling severely affects the computational complexity of dynamic programming algorithms. Correspondingly, Figure 42b illustrates the rapid growth of sample complexity,

$$N(\varepsilon) = \Omega(\varepsilon^{-d}), \tag{1043}$$

which indicates that achieving a fixed approximation accuracy ε requires exponentially many samples in high-dimensional spaces. Figure 42c depicts the decay of the probability of visiting informative or rewarding regions:

$$\mathbb{P}(s_t \in \mathcal{S}_{\text{useful}}) \approx \frac{\text{Vol}(\mathcal{S}_{\text{useful}})}{\text{Vol}(\mathcal{S})}, \tag{1044}$$

where the useful volume becomes negligibly small relative to the total volume as d increases. This explains why naive random exploration becomes ineffective in high-dimensional environments. Figure 42d further shows the computational explosion associated with Bellman updates:

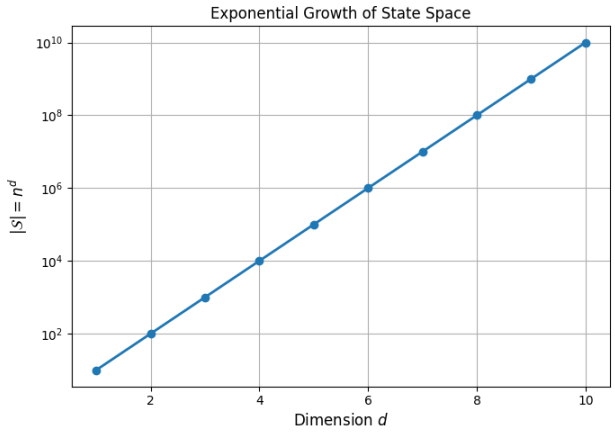
$$V_{k+1}(s) = \sup_{a \in \mathcal{A}} \left[R(s,a) + \gamma \int_{\mathcal{S}} V_k(s') P(ds'|s,a)\right], \tag{1045}$$

where both integration and maximization become increasingly expensive with dimensionality. Figure 42e demonstrates how function approximation mitigates this scaling problem by replacing tabular representations with compact parameterized models:

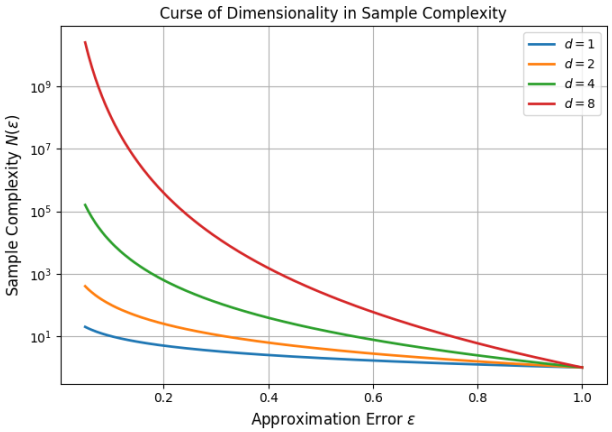
$$V(s;\theta) = f_\theta(s), \tag{1046}$$

thereby reducing the effective complexity from exponential growth to a more manageable scaling determined by the representation dimension. Finally, Figure 42f illustrates the manifold hypothesis, where high-dimensional observations lie near a lower-dimensional manifold $\mathcal{M} \subset \mathbb{R}^d$ satisfying

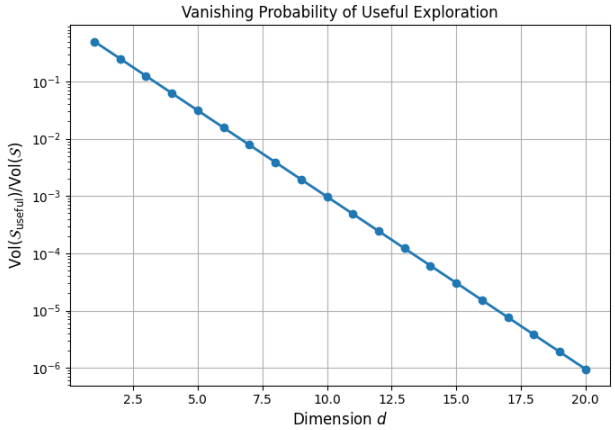
$$\dim(\mathcal{M}) \ll d. \tag{1047}$$



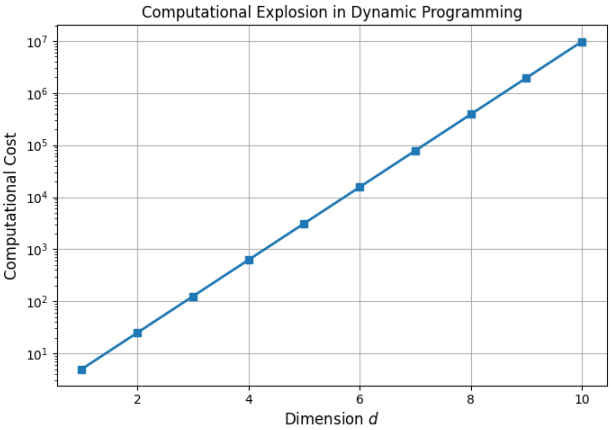
(a) **Exponential growth of the state space $|\mathcal{S}| = n^d$.**



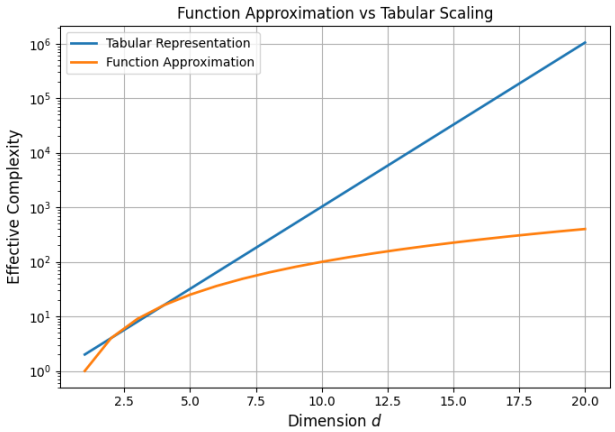
(b) **Sample complexity scaling with ε^{-d}.**



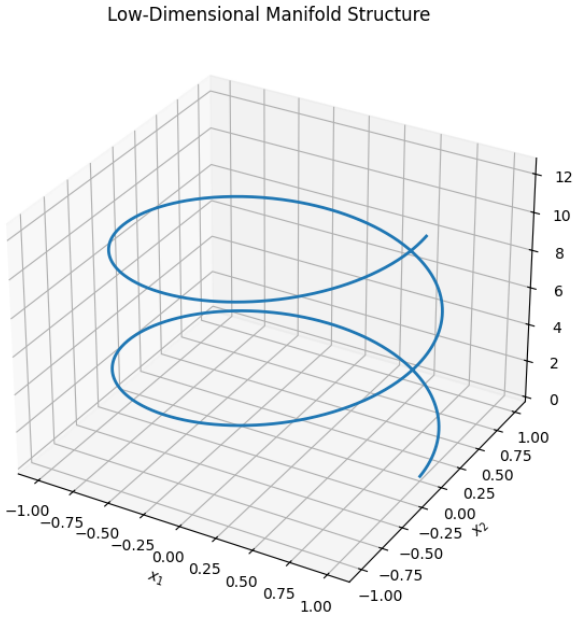
(c) **Vanishing probability of exploring useful regions in high dimensions.**



(d) **Computational explosion in dynamic programming methods.**



(e) **Function approximation mitigating dimensional scaling.**



(f) **Low-dimensional manifold structure in high-dimensional spaces.**

Figure 42: **Illustrations of the curse of dimensionality in reinforcement learning and the role of structured representations in mitigating high-dimensional complexity.**

This structural assumption enables representation learning methods to exploit intrinsic low-dimensional geometry, thereby alleviating the curse of dimensionality and improving both learning efficiency and generalization performance.

In summary, the exponential scaling induced by high-dimensional state and action spaces poses severe obstacles to both theoretical analysis and practical implementation. Overcoming these difficulties requires leveraging structure through function approximation, representation learning, and principled exploration, while carefully managing the trade-offs introduced by these approximations.

12 Future Directions

Reinforcement learning (RL) has achieved remarkable empirical success, yet several foundational challenges remain open. These challenges motivate a number of active research directions that aim to improve data efficiency, robustness, safety, and theoretical understanding. We outline some of the most important directions below.

12.1 Offline Reinforcement Learning

Offline reinforcement learning has emerged as a major research direction aimed at learning effective decision-making policies entirely from previously collected datasets without requiring additional interaction with the environment, and important advances in this area were made by Levine et al. (2020) [55], Kumar et al. (2020) [56], and Fujimoto et al. (2019) [57]. Levine et al. (2020) [55] provided a comprehensive tutorial and survey of offline reinforcement learning, systematically analyzing the challenges associated with distributional shift, extrapolation error, dataset coverage, and policy evaluation while reviewing theoretical foundations and practical algorithmic approaches for learning from fixed behavioral datasets. Fujimoto et al. (2019) [57] introduced methods for off-policy deep reinforcement learning without online exploration, demonstrating how policies can be learned robustly from static datasets while mitigating overestimation and instability caused by evaluating actions insufficiently represented in the training data. Kumar et al. (2020) [56] further advanced the field through Conservative Q-Learning (CQL), which incorporates conservative value estimation principles to prevent unrealistic overestimation of out-of-distribution actions, thereby improving stability and robustness in offline reinforcement learning settings. Together, these works established many of the theoretical and algorithmic foundations of offline reinforcement learning, showing how robust policy optimization can be achieved from fixed datasets while addressing the fundamental challenges of uncertainty, distributional mismatch, and extrapolation error in data-driven sequential decision-making.

Offline (or batch) reinforcement learning considers the setting in which an agent must learn entirely from a fixed dataset

$$\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^{N}, \tag{1048}$$

generated by a behavior policy μ, without any additional interaction with the environment. The objective remains to compute a policy π that maximizes the expected discounted return

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_t\right], \tag{1049}$$

but now under the severe constraint that all information must be extracted from $\mathcal{D}$.

A central challenge in this setting is the mismatch between the data-generating distribution and the distribution induced by the learned policy. Let $d^{\mu}(s, a)$ denote the stationary distribution of state-action pairs under μ, and $d^{\pi}(s, a)$ the corresponding distribution under π. Then, in general,

$$d^{\pi}(s, a) \neq d^{\mu}(s, a), \tag{1050}$$

and learning requires extrapolation from d^{μ} to d^{π}. This discrepancy is often formalized as *distributional shift*. In particular, the learned policy may assign non-negligible probability to actions that are rarely or never observed in the dataset:

$$\pi(a|s) > 0 \quad \text{while} \quad d^{\mu}(s, a) \approx 0. \tag{1051}$$

In such regions, value estimates rely on extrapolation and can incur arbitrarily large errors:

$$|\hat{Q}(s,a) - Q^{\pi}(s,a)| \gg 0 \quad \text{if } (s,a) \notin \text{supp}(\mathcal{D}). \tag{1052}$$

This issue is exacerbated by the bootstrapping nature of temporal-difference learning. Consider a fitted Q-iteration update:

$$\hat{Q}_{k+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a'} \hat{Q}_k(s',a') \right]. \tag{1053}$$

If $\hat{Q}_k(s',a')$ is overestimated in out-of-distribution regions, the maximization operator propagates and amplifies this error, leading to systematic overestimation:

$$\hat{Q}(s,a) \geq Q^{\pi}(s,a). \tag{1054}$$

To mitigate distributional shift, many offline RL methods impose constraints that keep the learned policy close to the behavior policy. A common approach is to solve a constrained optimization problem:

$$\max_{\pi} \; J(\pi) \quad \text{subject to} \quad D\big(\pi(\cdot|s) \,\|\, \mu(\cdot|s)\big) \leq \epsilon, \quad \forall s \in \mathcal{S}, \tag{1055}$$

where D is a divergence measure such as the Kullback–Leibler divergence:

$$D_{\text{KL}}(\pi \,\|\, \mu) = \sum_{a} \pi(a|s) \log \frac{\pi(a|s)}{\mu(a|s)}. \tag{1056}$$

This constraint ensures that π remains within the support of the dataset, reducing extrapolation error. An alternative approach is *conservative value estimation*, where the learning objective penalizes Q-values for unsupported actions. For example, one may solve

$$\min_{\hat{Q}} \; \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[\left(\hat{Q}(s,a) - y(s,a) \right)^2 \right] + \alpha \left(\mathbb{E}_{a\sim\pi(\cdot|s)}[\hat{Q}(s,a)] - \mathbb{E}_{a\sim\mu(\cdot|s)}[\hat{Q}(s,a)] \right), \tag{1057}$$

where $y(s,a)$ is a Bellman target. The regularization term enforces

$$\hat{Q}(s,a) \leq Q^{\pi}(s,a), \tag{1058}$$

thereby counteracting overestimation. Another important technique is importance sampling for off-policy evaluation. For a trajectory $(s_0, a_0, \ldots, s_T, a_T)$, define the importance weight

$$w_t = \prod_{k=0}^{t} \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}. \tag{1059}$$

Then,

$$J(\pi) = \mathbb{E}^{\mu} \left[\sum_{t=0}^{\infty} \gamma^t w_t R_t \right]. \tag{1060}$$

However, these weights can have high variance:

$$\text{Var}(w_t) \to \infty \quad \text{as } t \to \infty, \tag{1061}$$

making naive importance sampling impractical.

From a theoretical perspective, performance bounds in offline RL typically depend on a *concentrability coefficient*:

$$C := \sup_{s,a} \frac{d^{\pi}(s,a)}{d^{\mu}(s,a)}, \tag{1062}$$

which measures how well the dataset covers the state-action space relevant to π. Large values of C indicate severe distributional shift and lead to weak guarantees:

$$\|V^{\pi} - \hat{V}^{\pi}\| \leq \mathcal{O}\left(\frac{C}{\sqrt{N}} \right). \tag{1063}$$

Offline RL is particularly important in domains where data collection is expensive, risky, or ethically constrained. However, its success critically depends on dataset quality. If the dataset lacks sufficient coverage of high-reward regions, no algorithm can reliably recover an optimal policy:

$$\sup_{\pi} J(\pi) \quad \text{is unattainable if } \operatorname{supp}(\mathcal{D}) \cap \operatorname{supp}(d^{\pi^*}) = \varnothing. \tag{1064}$$

In summary, offline reinforcement learning introduces a fundamentally different regime in which exploration is impossible and generalization beyond the dataset is required.

Figures 43a, 43b, 43c, 43d, 43e, and 43f illustrate several fundamental phenomena arising in offline reinforcement learning. Figure 43a depicts the distributional mismatch between the behavior policy distribution $d^{\mu}(s,a)$ and the learned policy distribution $d^{\pi}(s,a)$, highlighting the central issue of distributional shift:

$$d^{\pi}(s,a) \neq d^{\mu}(s,a). \tag{1065}$$

The shaded out-of-distribution regions correspond to state-action pairs for which the dataset provides little or no coverage, causing the learned policy to extrapolate beyond the support of the data. Consequently, as shown in Figure 43b, extrapolation errors grow rapidly with increasing distance from the dataset support:

$$|\hat{Q}(s,a) - Q^{\pi}(s,a)| \gg 0 \quad \text{if } (s,a) \notin \operatorname{supp}(\mathcal{D}). \tag{1066}$$

This effect becomes particularly problematic under bootstrapping updates. Figure 43c illustrates the propagation and amplification of overestimated Q-values through fitted Q-iteration:

$$\hat{Q}_{k+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a'} \hat{Q}_k(s',a') \right], \tag{1067}$$

where errors in unsupported regions are recursively reinforced by the maximization operator, yielding

$$\hat{Q}(s,a) \geq Q^{\pi}(s,a). \tag{1068}$$

To mitigate this instability, offline reinforcement learning methods often constrain the learned policy to remain close to the behavior policy. Figure 43d demonstrates such KL-constrained optimization:

$$\max_{\pi} J(\pi) \quad \text{subject to} \quad D_{\mathrm{KL}}(\pi \,\|\, \mu) \leq \epsilon, \tag{1069}$$

where

$$D_{\mathrm{KL}}(\pi \,\|\, \mu) = \sum_{a} \pi(a|s) \log \frac{\pi(a|s)}{\mu(a|s)}. \tag{1070}$$

This constraint prevents the learned policy from assigning large probability mass to unsupported actions. Figure 43e illustrates another major challenge: the variance explosion of importance sampling weights,

$$w_t = \prod_{k=0}^{t} \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}, \tag{1071}$$

whose variance may grow exponentially:

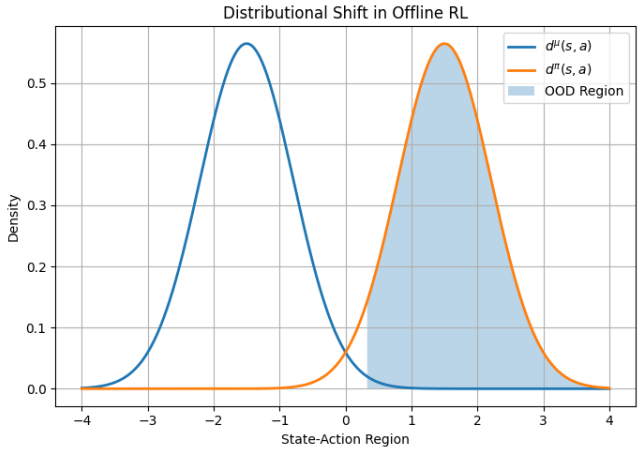
$$\operatorname{Var}(w_t) \to \infty \quad \text{as } t \to \infty. \tag{1072}$$

This instability limits the practical applicability of naive off-policy evaluation methods. Finally, Figure 43f shows the dependence of theoretical error bounds on the concentrability coefficient

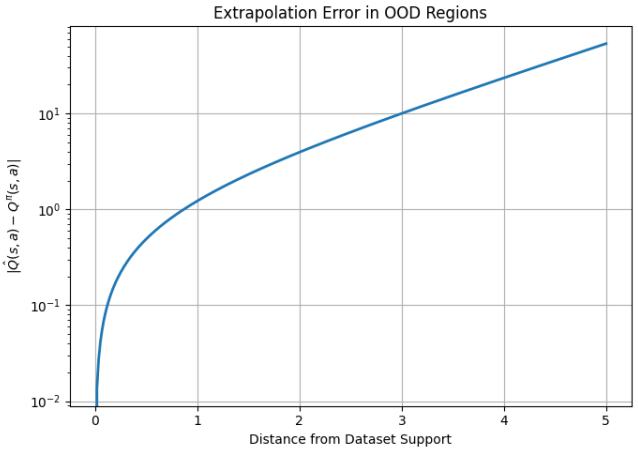
$$C = \sup_{s,a} \frac{d^{\pi}(s,a)}{d^{\mu}(s,a)}, \tag{1073}$$

which quantifies the severity of distributional shift. The corresponding estimation error scales as

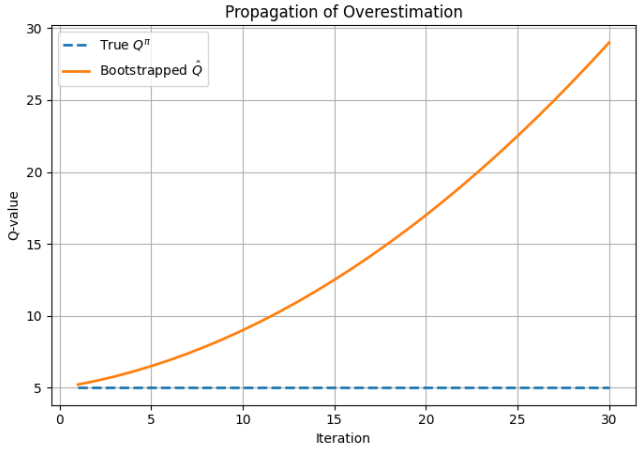
$$\|V^{\pi} - \hat{V}^{\pi}\| \leq \mathcal{O}\left(\frac{C}{\sqrt{N}} \right), \tag{1074}$$



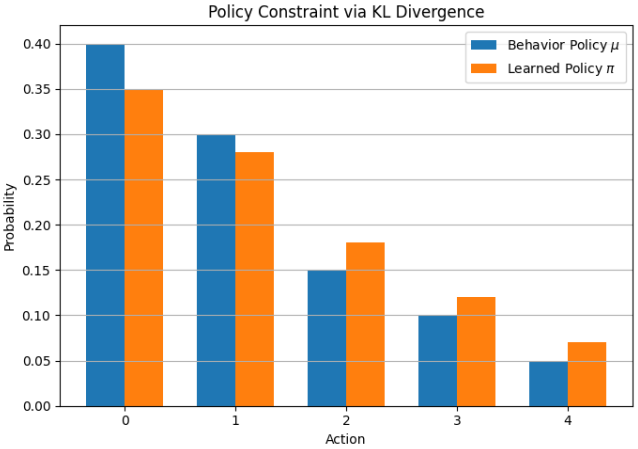
(a) **Distributional shift between $d^{\mu}(s,a)$ and $d^{\pi}(s,a)$.**



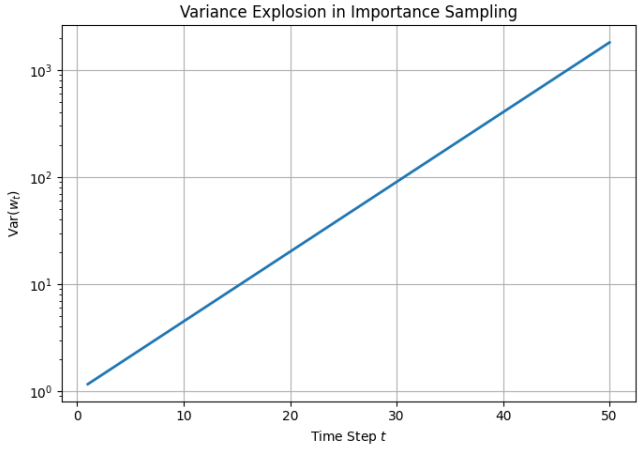
(b) **Growth of extrapolation error outside dataset support.**



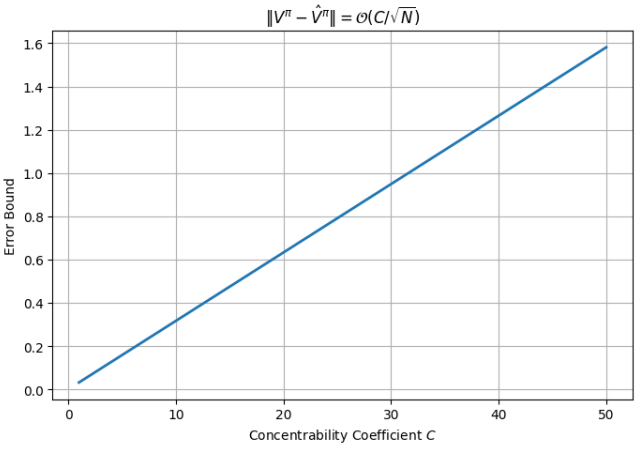
(c) **Propagation of overestimation through bootstrapping.**



(d) **KL-constrained policy optimization in offline RL.**



(e) **Variance explosion in importance sampling weights.**



(f) **Dependence of error bounds on the concentrability coefficient.**

Figure 43: **Illustrations of key challenges and theoretical concepts in offline reinforcement learning, including distributional shift, extrapolation error, bootstrapping instability, constrained optimization, importance sampling variance, and concentrability-dependent error bounds.**

demonstrating that poor dataset coverage leads directly to weaker guarantees and larger approximation errors. Collectively, Figures 43a, 43b, 43c, 43d, 43e, and 43f illustrate the fundamental interplay between distributional shift, extrapolation error, bootstrapping instability, conservative policy optimization, importance sampling variance, and dataset coverage in offline reinforcement learning.

The interplay between distributional shift, bootstrapping, and function approximation creates unique challenges that necessitate constrained optimization, conservative estimation, and careful statistical analysis.

12.2 Multi-Agent Systems

Multi-agent systems and multi-agent reinforcement learning study how multiple interacting agents learn and make decisions within shared environments, and the theoretical and algorithmic foundations of this field have been significantly developed by Busoniu et al. (2010) [58], Shoham and Leyton-Brown (2008) [59], and Lowe et al. (2017) [60]. Busoniu et al. (2010) [58] provided a comprehensive treatment of multi-agent reinforcement learning, analyzing cooperative, competitive, and mixed-agent settings while discussing learning dynamics, coordination mechanisms, communication strategies, decentralized control, and convergence challenges in environments with interacting adaptive agents. Shoham and Leyton-Brown (2008) [59] developed the broader theoretical foundations of multiagent systems by integrating concepts from game theory, distributed artificial intelligence, mechanism design, and strategic reasoning, thereby providing rigorous mathematical frameworks for understanding equilibrium behavior, cooperation, and competition among rational agents. Lowe et al. (2017) [60] extended these ideas to deep reinforcement learning through the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) framework, introducing actor–critic architectures with centralized training and decentralized execution to address nonstationarity and coordination challenges in complex multi-agent environments. Multi-Agent Actor-Critic Together, these works established the modern foundations of multi-agent reinforcement learning, demonstrating how interacting intelligent agents can learn coordinated or competitive behaviors in stochastic and dynamic environments through distributed sequential decision-making mechanisms.

Multi-agent reinforcement learning (MARL) generalizes the single-agent framework to settings involving a collection of interacting decision-makers. Let $\mathcal{N} = \{1, \dots, N\}$ denote the set of agents. At each time t, agent $i \in \mathcal{N}$ selects an action a_t^i according to a policy $\pi^i(\cdot|s_t)$, and the joint action is denoted by

$$a_t = (a_t^1, \dots, a_t^N) \in \mathcal{A} := \prod_{i=1}^{N} \mathcal{A}^i. \tag{1075}$$

The environment evolves according to a controlled transition kernel

$$s_{t+1} \sim P(\cdot \mid s_t, a_t), \tag{1076}$$

and each agent receives an individual reward $R^i(s_t, a_t)$, leading to returns

$$J^i(\pi^1, \dots, \pi^N) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R^i(s_t, a_t)\right]. \tag{1077}$$

Depending on the structure of the reward functions, one obtains different classes of problems. In fully cooperative settings, all agents share a common objective:

$$R^1 = R^2 = \cdots = R^N = R, \quad \max_{\pi^1, \dots, \pi^N} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right], \tag{1078}$$

which can be viewed as a decentralized control problem. In contrast, competitive or general-sum settings give rise to stochastic games, where agents have potentially conflicting objectives.

A central difficulty in MARL is the induced non-stationarity. From the perspective of a single agent i, the effective transition dynamics depend on the policies of all other agents:

$$P^{\pi^{-i}}(s'|s,a^i) = \sum_{a^{-i}\in\mathcal{A}^{-i}} P(s'|s,a^i,a^{-i})\,\pi^{-i}(a^{-i}|s), \tag{1079}$$

where $\pi^{-i} = (\pi^j)_{j\neq i}$. Since π^{-i} evolves during learning, the induced transition kernel $P^{\pi^{-i}}$ is time-dependent:

$$P^{\pi_t^{-i}}(s'|s,a^i) \neq P^{\pi_{t+1}^{-i}}(s'|s,a^i), \tag{1080}$$

violating the stationarity assumptions underlying standard RL algorithms. This leads to instability and complicates convergence analysis.To address this, one often considers centralized training with decentralized execution. During training, agents may access global information and learn a joint action-value function:

$$Q(s,a^1,\dots,a^N), \tag{1081}$$

while at execution time each agent uses a decentralized policy $\pi^i(a^i|o^i)$ based on local observations o^i. Factorization methods aim to decompose the joint value function:

$$Q(s,a^1,\dots,a^N) \approx \sum_{i=1}^{N} Q^i(s,a^i), \tag{1082}$$

or more generally via monotonic mixing networks to ensure tractability. In competitive or general-sum settings, equilibrium concepts play a fundamental role. A policy profile $(\pi^1,\dots,\pi^N)$ is a Nash equilibrium if

$$\pi^i \in \mathrm{argmax}_{\tilde{\pi}^i}\, J^i(\tilde{\pi}^i,\pi^{-i}), \quad \forall i\in\mathcal{N}. \tag{1083}$$

Equivalently, no agent can unilaterally improve its expected return. In zero-sum two-player games, the problem reduces to finding a saddle point:

$$\max_{\pi^1}\min_{\pi^2} J^1(\pi^1,\pi^2). \tag{1084}$$

However, computing Nash equilibria in high-dimensional stochastic games is computationally challenging, and convergence guarantees for learning dynamics are limited.

Another important challenge is scalability. The joint action space grows exponentially:

$$|\mathcal{A}| = \prod_{i=1}^{N} |\mathcal{A}^i|, \tag{1085}$$

making naive value-based methods intractable. Similarly, communication constraints and partial observability lead to decentralized partially observable Markov decision processes (Dec-POMDPs), where each agent observes only $o_t^i \sim O^i(\cdot|s_t)$, further complicating learning. Recent research directions focus on improving coordination and credit assignment among agents. For example, one may define a difference reward for agent i:

$$D^i(s,a) = R(s,a) - R(s,a^{-i},c^i), \tag{1086}$$

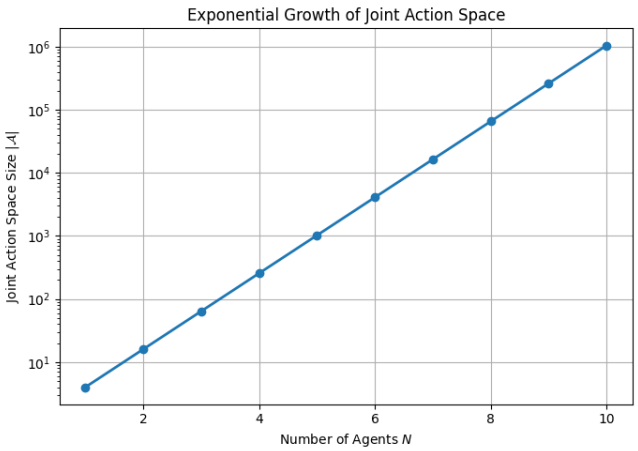
where c^i is a baseline action, to isolate the contribution of agent i. Alternatively, counterfactual reasoning is used:

$$A^i(s,a^i) = Q(s,a) - \mathbb{E}_{\tilde{a}^i\sim\pi^i}[Q(s,(\tilde{a}^i,a^{-i}))], \tag{1087}$$

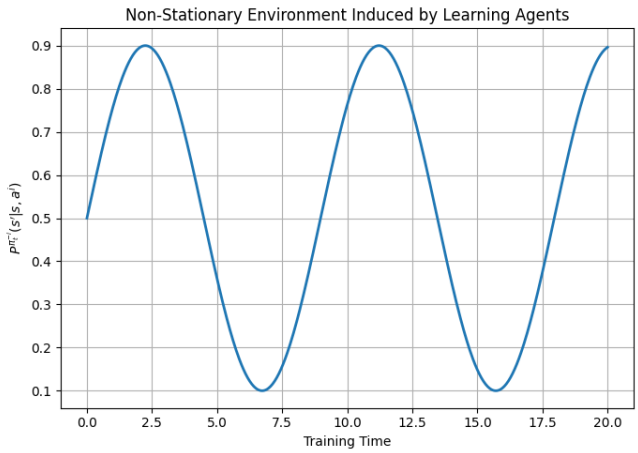
to reduce variance in policy gradients.

Figures 44a, 44b, 44c, 44d, 44e, and 44f illustrate several central concepts and challenges in multi-agent reinforcement learning (MARL). Figure 44a demonstrates the exponential growth of the joint action space as the number of agents increases. Since the joint action space is given by

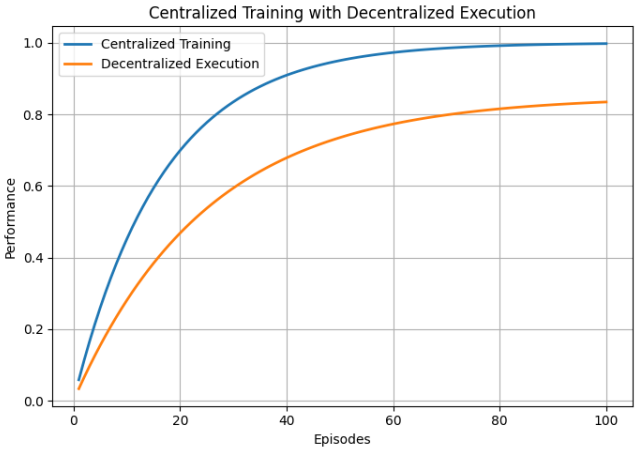
$$\mathcal{A} = \prod_{i=1}^{N} \mathcal{A}^i, \tag{1088}$$



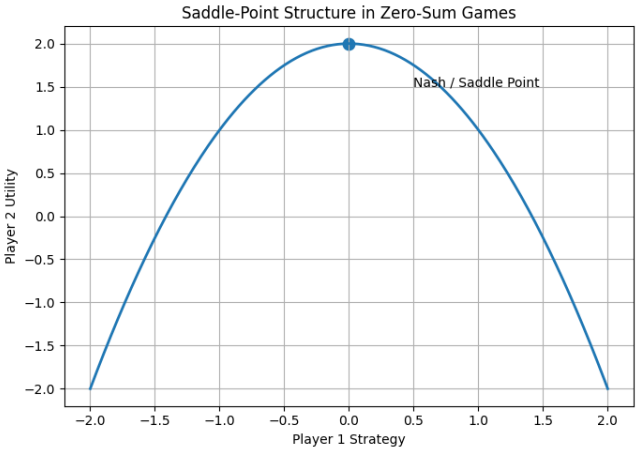
(a) **Exponential growth of the joint action space in multi-agent systems.**



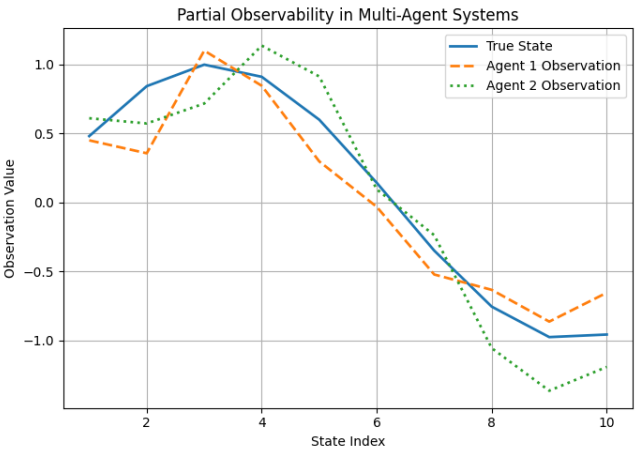
(b) **Non-stationary transition dynamics induced by evolving agent policies.**



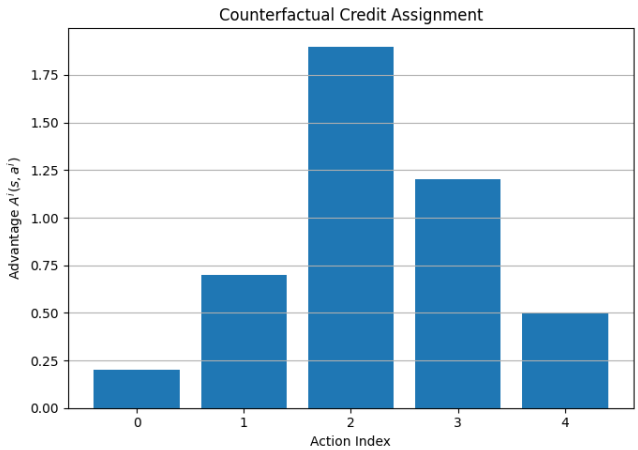
(c) **Centralized training with decentralized execution.**



(d) **Saddle-point structure corresponding to Nash equilibrium in zero-sum games.**



(e) **Partial observability in decentralized partially observable environments.**



(f) **Counterfactual credit assignment and advantage estimation.**

Figure 44: **Illustrations of major theoretical and computational challenges in multi-agent reinforcement learning, including combinatorial growth of joint action spaces, non-stationarity, centralized training paradigms, equilibrium structures, partial observability, and counterfactual credit assignment mechanisms.**

its cardinality scales as

$$|\mathcal{A}| = \prod_{i=1}^{N} |\mathcal{A}^i|, \tag{1089}$$

leading to severe combinatorial complexity even when each individual agent possesses only a small action set. This exponential scaling substantially increases the difficulty of exploration, planning, and value estimation.

Figure 44b illustrates the non-stationarity induced by simultaneously learning agents. From the perspective of agent i, the effective environment dynamics depend on the policies of all other agents:

$$P^{\pi^{-i}}(s'|s, a^i) = \sum_{a^{-i}} P(s'|s, a^i, a^{-i})\, \pi^{-i}(a^{-i}|s), \tag{1090}$$

where π^{-i} denotes the joint policy of the remaining agents. Since these policies evolve during training,

$$P^{\pi_t^{-i}}(s'|s, a^i) \neq P^{\pi_{t+1}^{-i}}(s'|s, a^i), \tag{1091}$$

the induced transition kernel becomes time-dependent, violating the stationarity assumptions underlying classical reinforcement learning theory.

Figure 44c depicts the paradigm of centralized training with decentralized execution (CTDE). During training, agents may access global state information and learn a joint value function

$$Q(s, a^1, \ldots, a^N), \tag{1092}$$

while execution remains decentralized through local policies

$$\pi^i(a^i|o^i), \tag{1093}$$

where o^i denotes the local observation available to agent i. To reduce complexity, factorization methods approximate the joint value function as

$$Q(s, a^1, \ldots, a^N) \approx \sum_{i=1}^{N} Q^i(s, a^i), \tag{1094}$$

or through more general monotonic mixing architectures.

Figure 44d illustrates the saddle-point structure associated with Nash equilibria in competitive or zero-sum games. A policy profile $(\pi^1, \ldots, \pi^N)$ constitutes a Nash equilibrium if

$$\pi^i \in \operatorname{argmax}_{\tilde{\pi}^i} J^i(\tilde{\pi}^i, \pi^{-i}), \qquad \forall i \in \mathcal{N}, \tag{1095}$$

meaning that no agent can improve its expected return by unilateral deviation. In two-player zero-sum games, the equilibrium problem reduces to

$$\max_{\pi^1} \min_{\pi^2} J^1(\pi^1, \pi^2), \tag{1096}$$

which geometrically corresponds to a saddle point in the value landscape.

Figure 44e demonstrates partial observability in decentralized partially observable Markov decision processes (Dec-POMDPs). Each agent observes only a local signal

$$o_t^i \sim O^i(\cdot|s_t), \tag{1097}$$

rather than the full environment state. Consequently, coordination becomes substantially more difficult, since agents must infer latent global information from incomplete and noisy observations.

Finally, Figure 44f illustrates counterfactual credit assignment, which seeks to isolate the contribution of individual agents to collective performance. A counterfactual advantage estimator is defined as

$$A^i(s, a^i) = Q(s, a) - \mathbb{E}_{\tilde{a}^i \sim \pi^i} \left[Q(s, (\tilde{a}^i, a^{-i})) \right], \tag{1098}$$

which measures the improvement attributable specifically to the chosen action of agent i. Similarly, difference rewards may be defined as

$$D^i(s, a) = R(s, a) - R(s, a^{-i}, c^i), \tag{1099}$$

where c^i is a baseline action. These methods reduce variance and improve coordination by providing more informative learning signals. Collectively, Figures 44a, 44b, 44c, 44d, 44e, and 44f illustrate the major theoretical and computational challenges of MARL, including combinatorial scaling, non-stationarity, decentralized coordination, equilibrium computation, partial observability, and multi-agent credit assignment.

In summary, multi-agent systems introduce a rich and challenging extension of reinforcement learning, characterized by strategic interactions, non-stationarity, and combinatorial complexity. Future progress requires advances in equilibrium computation, scalable representations, decentralized learning, and theoretical guarantees for convergence and generalization in multi-agent environments.

12.3 Safe Reinforcement Learning

Safe reinforcement learning focuses on developing learning algorithms that optimize long-term performance while simultaneously satisfying safety, reliability, and risk constraints, and important contributions to this field were made by García and Fernández (2015) [61], Achiam et al. (2017) [62], and Chow et al. (2018) [63]. García and Fernández (2015) [61] provided a comprehensive survey of safe reinforcement learning methods, systematically categorizing approaches based on risk-sensitive criteria, constrained optimization, safe exploration, and robustness considerations, while highlighting the importance of ensuring reliable behavior during both learning and deployment in uncertain environments. Achiam et al. (2017) [62] introduced Constrained Policy Optimization (CPO), a policy-gradient-based framework that incorporates explicit safety constraints into reinforcement learning through trust-region optimization methods, enabling agents to improve performance while maintaining probabilistic guarantees on constraint satisfaction throughout training. Chow et al. (2018) [63] further developed the theoretical foundations of risk-constrained reinforcement learning by formulating reinforcement learning problems under coherent risk measures and probabilistic safety constraints, providing rigorous analyses and algorithms for optimizing policies while controlling exposure to high-risk outcomes in stochastic environments. Together, these works established the modern framework of safe reinforcement learning, demonstrating how safety guarantees, risk management, and constrained optimization principles can be systematically integrated into sequential decision-making and reinforcement learning systems.

Safe reinforcement learning (Safe RL) extends the standard reinforcement learning framework by incorporating explicit constraints on system behavior, ensuring that learning and execution respect safety requirements. Let $R(s, a)$ denote the reward function and $C(s, a)$ a cost function capturing undesirable outcomes (e.g., risk, constraint violations). The objective is to solve the constrained optimization problem

$$\max_{\pi} \; J(\pi) \quad \text{subject to} \quad J_C(\pi) := \mathbb{E}^{\pi} \left[\sum_{t=0}^{\infty} \gamma^t C(s_t, a_t) \right] \leq d, \tag{1100}$$

where d is a prescribed safety threshold. This formulation defines a *constrained Markov decision process* (CMDP).

A classical approach to solving CMDPs is via Lagrangian relaxation. Introducing a dual variable $\lambda \geq 0$, one defines the Lagrangian

$$\mathcal{L}(\pi, \lambda) = J(\pi) - \lambda \left(J_C(\pi) - d \right), \tag{1101}$$

and seeks a saddle point:

$$\min_{\lambda \geq 0} \max_{\pi} \mathcal{L}(\pi, \lambda). \tag{1102}$$

Under suitable regularity conditions, strong duality holds, and the constrained problem can be solved by iteratively updating the policy and the multiplier:

$$\pi_{k+1} \approx \text{argmax}_\pi \, \mathcal{L}(\pi, \lambda_k), \quad \lambda_{k+1} = [\lambda_k + \eta \, (J_C(\pi_k) - d)]_+ . \tag{1103}$$

The Bellman equations also admit constrained counterparts. Define the cost value function

$$V_C^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t C(s_t, a_t) \mid s_0 = s \right]. \tag{1104}$$

Then the constraint can be expressed pointwise or in expectation:

$$V_C^\pi(s) \le d(s), \quad \text{or} \quad \mathbb{E}_{s\sim\rho}[V_C^\pi(s)] \le d. \tag{1105}$$

The associated Bellman operator for the Lagrangian becomes

$$(T_\lambda^\pi V)(s) = \int_{\mathcal{A}} \left[R(s,a) - \lambda C(s,a) + \gamma \int_{\mathcal{S}} V(s') P(ds'|s,a) \right] \pi(da|s), \tag{1106}$$

highlighting that safety constraints effectively modify the reward structure.

A major challenge in Safe RL is *safe exploration*. Standard RL methods may violate constraints during learning, even if the final policy is safe. Formally, one may require high-probability guarantees:

$$\mathbb{P}\left(C(s_t, a_t) \le c_{\max} \; \forall t\right) \ge 1 - \delta, \tag{1107}$$

or cumulative constraints:

$$\mathbb{P}\left(\sum_{t=0}^{T} C(s_t, a_t) \le D \right) \ge 1 - \delta. \tag{1108}$$

Ensuring such guarantees is difficult because the agent must balance exploration (to learn the environment) with constraint satisfaction (to remain safe). One approach is to restrict policies to a *safe set*. Let Π_{safe} denote the set of policies satisfying

$$J_C(\pi) \le d. \tag{1109}$$

Then the optimization becomes

$$\max_{\pi \in \Pi_{\text{safe}}} J(\pi). \tag{1110}$$

Projection-based methods update policies and then project back onto Π_{safe}, though computing this projection is generally intractable.

Another approach is the use of *shielding* or *control barrier functions*, where unsafe actions are overridden by a safety mechanism. Let $\mathcal{A}_{\text{safe}}(s)$ denote the set of admissible actions:

$$\mathcal{A}_{\text{safe}}(s) = \{a \in \mathcal{A} : C(s,a) \le c_{\max}\}. \tag{1111}$$

Then the executed action is

$$\tilde{a}_t = \begin{cases} a_t, & \text{if } a_t \in \mathcal{A}_{\text{safe}}(s_t), \\ a_t^{\text{safe}}, & \text{otherwise.} \end{cases} \tag{1112}$$

From a probabilistic perspective, Safe RL can also be framed in terms of risk-sensitive objectives. For example, one may minimize the variance of returns:

$$\text{Var}^\pi(G_0), \tag{1113}$$

or optimize coherent risk measures such as Conditional Value-at-Risk (CVaR):

$$\text{CVaR}_\alpha(G_0) = \mathbb{E}[G_0 \mid G_0 \le q_\alpha], \tag{1114}$$

where q_α is the α-quantile. The objective then becomes

$$\max_{\pi} \ \mathrm{CVaR}_\alpha(G_0), \tag{1115}$$

providing robustness against worst-case outcomes. Despite these approaches, Safe RL remains a challenging research area. Ensuring constraint satisfaction under function approximation, partial observability, and limited data is difficult. Moreover, there is often a trade-off between performance and safety:

$$\max_{\pi} J(\pi) \quad \text{vs} \quad \min_{\pi} J_C(\pi), \tag{1116}$$

which must be carefully balanced.

Figures 45a, 45b, 45c, 45d, 45e, and 45f illustrate several fundamental concepts in safe reinforcement learning, where the objective is to optimize long-term performance while simultaneously enforcing safety constraints. Figure 45a depicts the trade-off between expected reward and accumulated safety cost. In constrained reinforcement learning, the policy optimization problem is formulated as

$$\max_{\pi} \ J(\pi) \quad \text{subject to} \quad J_C(\pi) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t C(s_t, a_t) \right] \leq d, \tag{1117}$$

where $J(\pi)$ denotes the expected discounted reward and $J_C(\pi)$ represents the cumulative safety cost. The figure illustrates that increasing performance often incurs higher risk or constraint violations, thereby producing an inherent reward–safety trade-off.

Figure 45b illustrates the Lagrangian formulation used to solve constrained Markov decision processes (CMDPs). Introducing a dual variable $\lambda \geq 0$, one defines the Lagrangian

$$\mathcal{L}(\pi, \lambda) = J(\pi) - \lambda \left(J_C(\pi) - d \right), \tag{1118}$$

and seeks the saddle-point solution

$$\min_{\lambda \geq 0} \max_{\pi} \mathcal{L}(\pi, \lambda). \tag{1119}$$

The saddle structure shown in the figure reflects the simultaneous maximization over policies and minimization over constraint penalties, balancing performance and safety.

Figure 45c compares safe and unsafe exploration trajectories. Standard exploration methods may temporarily violate constraints during learning, even if the final learned policy is safe. To guarantee safe exploration, one may require

$$\mathbb{P}(C(s_t, a_t) \leq c_{\max}, \ \forall t) \geq 1 - \delta, \tag{1120}$$

or cumulative guarantees of the form

$$\mathbb{P}\left(\sum_{t=0}^{T} C(s_t, a_t) \leq D \right) \geq 1 - \delta. \tag{1121}$$

The unsafe trajectory in the figure exceeds the safety threshold, whereas the safe trajectory remains within acceptable bounds throughout learning.

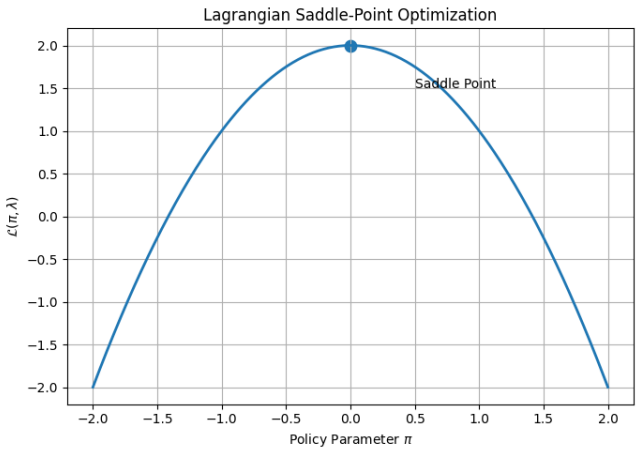
Figure 45d illustrates shielding and safe action projection. Let

$$\mathcal{A}_{\text{safe}}(s) = \{ a \in \mathcal{A} : C(s, a) \leq c_{\max} \} \tag{1122}$$

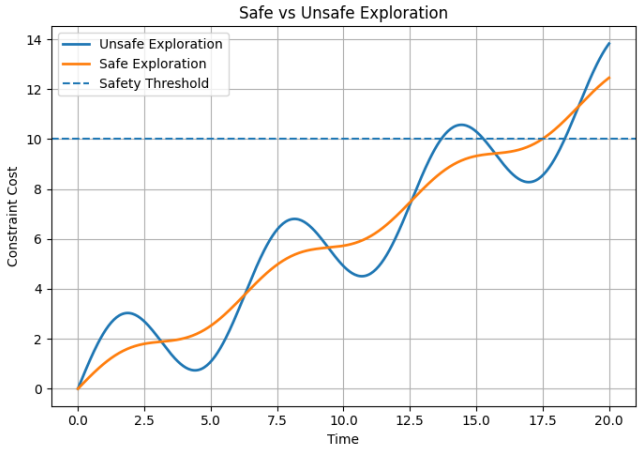
denote the admissible action set. The executed control action is then modified according to

$$\tilde{a}_t = \begin{cases} a_t, & a_t \in \mathcal{A}_{\text{safe}}(s_t), \\ a_t^{\text{safe}}, & \text{otherwise}, \end{cases} \tag{1123}$$

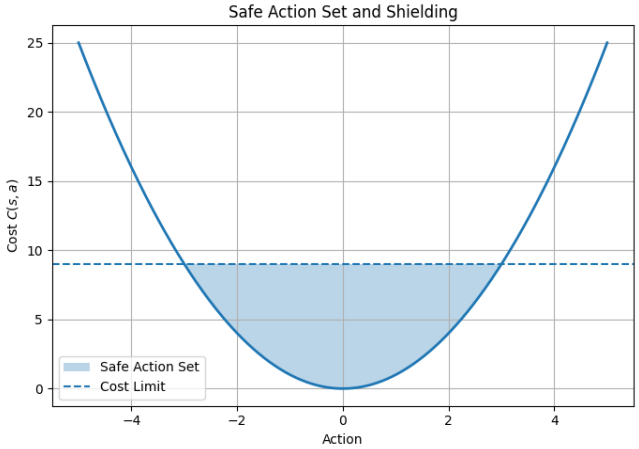
(a) Reward-safety trade-off in constrained reinforcement learning.



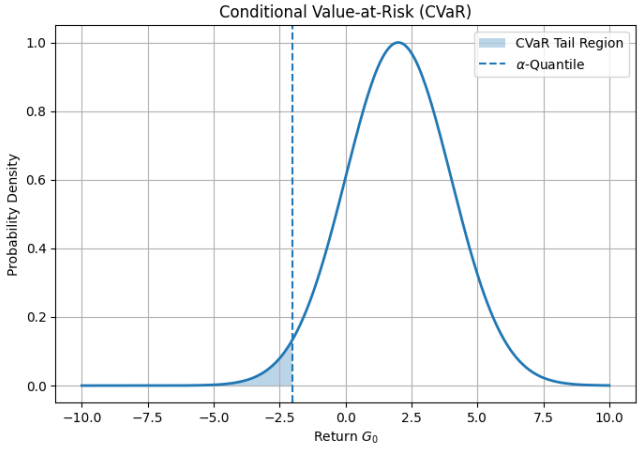
(b) Lagrangian saddle-point optimization for constrained policies.



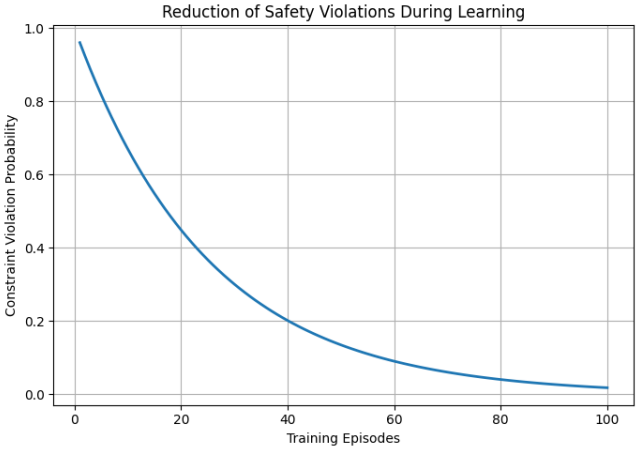
(c) Comparison between safe and unsafe exploration trajectories.



(d) Safe action set and shielding mechanism.



(e) Conditional Value-at-Risk (CVaR) and tail-risk minimization.



(f) Reduction in constraint violation probability during training.

Figure 45: **Illustrations of major concepts in safe reinforcement learning, including reward–safety trade-offs, constrained optimization via Lagrangian methods, safe exploration, shielding mechanisms, risk-sensitive objectives, and probabilistic safety guarantees.**

where unsafe actions are replaced by certified safe alternatives. The shaded region in the figure represents the feasible safe-action domain satisfying the safety constraints.

Figure 45e demonstrates risk-sensitive optimization through Conditional Value-at-Risk (CVaR). Rather than maximizing expected return alone, one may optimize a coherent risk measure:

$$\mathrm{CVaR}_\alpha(G_0) = \mathbb{E}[G_0 \mid G_0 \le q_\alpha], \tag{1124}$$

where q_α denotes the α-quantile of the return distribution. The shaded tail region in the figure corresponds to low-return catastrophic outcomes, which CVaR-based methods explicitly penalize in order to improve robustness against worst-case events.

Finally, Figure 45f illustrates the reduction of safety-violation probability during training. As the learning algorithm improves its estimation of safe policies, the probability of violating constraints decreases over time:

$$\mathbb{P}(C(s_t, a_t) > c_{\max}) \to 0 \quad \text{as } t \to \infty. \tag{1125}$$

This reflects the convergence of the policy toward safer operating regimes while maintaining task performance. Collectively, Figures 45a, 45b, 45c, 45d, 45e, and 45f illustrate the central mechanisms of safe reinforcement learning, including constrained optimization, duality, safe exploration, shielding, risk-sensitive control, and probabilistic safety guarantees.

In summary, safe reinforcement learning introduces additional layers of complexity by requiring guarantees beyond expected performance. It combines elements of stochastic control, optimization under constraints, and risk theory. Future progress hinges on developing algorithms that provide provable safety guarantees during both learning and deployment, while remaining computationally tractable in high-dimensional settings.

12.4 Theoretical Generalization Guarantees

Theoretical generalization guarantees in reinforcement learning aim to understand how accurately learned policies and value functions transfer beyond observed data and finite samples, and major contributions in this direction were made by Jiang and Li (2016) [64], Farahmand et al. (2010) [65], and Agarwal et al. (2022) [66]. Farahmand et al. (2010) [65] developed foundational analyses of error propagation in approximate policy iteration and value iteration, rigorously characterizing how approximation errors accumulate through successive Bellman updates and how concentrability conditions influence the stability and performance of approximate reinforcement learning algorithms. Jiang and Li (2016) [64] advanced the theory of off-policy evaluation through doubly robust estimators, combining importance sampling and value-function approximation to obtain statistically efficient and low-variance estimates of policy performance from previously collected data, thereby improving the reliability and generalization behavior of reinforcement learning systems under distributional mismatch. Agarwal et al. (2022) [66] further unified and expanded the theoretical foundations of reinforcement learning by systematically analyzing sample complexity, regret bounds, function approximation, exploration, generalization error, and convergence guarantees for modern reinforcement learning algorithms in both tabular and approximate settings. Together, these works established key theoretical frameworks for understanding generalization, approximation error, and statistical reliability in reinforcement learning, thereby providing rigorous guarantees for the performance and robustness of learning algorithms in uncertain and high-dimensional environments.

Understanding generalization in reinforcement learning (RL) requires extending classical statistical learning theory to sequential, dependent, and policy-dependent data. Unlike supervised learning, where data are assumed i.i.d., RL data are generated by a trajectory

$$(s_0, a_0, s_1, a_1, \dots), \tag{1126}$$

whose distribution depends on the policy π and the transition kernel P. Let $\mathcal{D}_{\text{train}}$ denote the empirical distribution induced by a behavior policy μ, and let $\mathcal{D}_\pi$ denote the (unknown) distribution induced by

the learned policy π. The generalization gap can be expressed as

$$\mathcal{E}(\pi) = \left| \mathbb{E}_{\mathcal{D}_\pi}[G_0] - \mathbb{E}_{\mathcal{D}_{\text{train}}}[G_0] \right|, \tag{1127}$$

which captures both statistical estimation error and distributional mismatch. A key difficulty arises from *distributional shift*. Even if a value function $\hat{Q}$ is accurate under $\mathcal{D}_{\text{train}}$, it may be inaccurate under $\mathcal{D}_\pi$. This mismatch can be quantified via density ratios:

$$w(s,a) = \frac{d^\pi(s,a)}{d^\mu(s,a)}, \tag{1128}$$

where d^π and d^μ denote occupancy measures. Then, for any function f,

$$\mathbb{E}_{\mathcal{D}_\pi}[f(s,a)] = \mathbb{E}_{\mathcal{D}_{\text{train}}}\left[w(s,a) f(s,a) \right], \tag{1129}$$

and large values of $w(s,a)$ amplify estimation errors.

From a statistical learning perspective, suppose $\hat{Q} \in \mathcal{F}$ is learned from N samples. Then, under appropriate regularity conditions, one can derive bounds of the form

$$\sup_{f \in \mathcal{F}} \left| \mathbb{E}_{\mathcal{D}_{\text{train}}}[f] - \frac{1}{N} \sum_{i=1}^{N} f(s_i, a_i) \right| \leq \mathcal{O}\left(\frac{\text{Complexity}(\mathcal{F})}{\sqrt{N}} \right), \tag{1130}$$

where $\text{Complexity}(\mathcal{F})$ may be measured via VC-dimension, Rademacher complexity, or covering numbers. However, translating such bounds to performance guarantees requires accounting for compounding errors through the Bellman operator:

$$\hat{Q}(s,a) = R(s,a) + \gamma \mathbb{E}_{s'} \left[\max_{a'} \hat{Q}(s', a') \right]. \tag{1131}$$

This leads to error propagation phenomena. If the Bellman error is bounded:

$$\| \hat{Q} - T\hat{Q} \|_\infty \leq \varepsilon, \tag{1132}$$

then one obtains performance bounds of the form

$$\| \hat{Q} - Q^* \|_\infty \leq \frac{\varepsilon}{1-\gamma}. \tag{1133}$$

Thus, even small approximation errors can be amplified by a factor $(1-\gamma)^{-1}$.

Another central notion is *uniform convergence* over the state-action space:

$$\sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} |\hat{Q}(s,a) - Q^*(s,a)| \leq \varepsilon. \tag{1134}$$

Achieving such guarantees is difficult in high-dimensional or continuous spaces without structural assumptions. Recent advances impose conditions such as *linear MDPs*, where

$$P(s'|s,a) = \phi(s,a)^\top \psi(s'), \quad R(s,a) = \phi(s,a)^\top \theta, \tag{1135}$$

with feature dimension $d \ll |\mathcal{S}||\mathcal{A}|$, enabling sample complexity bounds of the form

$$N = \tilde{\mathcal{O}}\left(\frac{d}{(1-\gamma)^3 \varepsilon^2} \right). \tag{1136}$$

Another important concept is the *Bellman rank*, which captures the intrinsic complexity of the MDP. Under low Bellman rank assumptions, one can derive finite-sample guarantees even in large state spaces. Similarly, notions such as eluder dimension and witness rank provide alternative measures of complexity

governing exploration and generalization. In addition, stability and generalization of policy gradient methods depend on controlling the variance of gradient estimates:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t G_t \, \nabla_\theta \log \pi_\theta(a_t|s_t)\right], \tag{1137}$$

where high variance can degrade sample efficiency and lead to poor generalization.

A further challenge is *off-policy evaluation*. Given data from μ, one seeks to estimate $J(\pi)$ for a different policy π. Importance sampling yields

$$J(\pi) = \mathbb{E}^\mu\left[\sum_{t=0}^{\infty} \gamma^t w_t R_t\right], \quad w_t = \prod_{k=0}^{t} \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}, \tag{1138}$$

but the variance of w_t can grow exponentially with t, limiting practical applicability.

Figures 46a, 46b, 46c, 46d, 46e, and 46f illustrate several fundamental aspects of theoretical generalization in reinforcement learning. Figure 46a depicts the generalization gap between the empirical training distribution and the distribution induced by the learned policy. In reinforcement learning, the training data are generated by trajectories

$$(s_0, a_0, s_1, a_1, \dots), \tag{1139}$$

whose distribution depends on both the environment dynamics and the behavior policy. Consequently, the discrepancy between training and deployment performance can be quantified through

$$\mathcal{E}(\pi) = \left|\mathbb{E}_{\mathcal{D}_\pi}[G_0] - \mathbb{E}_{\mathcal{D}_{\text{train}}}[G_0]\right|, \tag{1140}$$

where $\mathcal{D}_{\text{train}}$ denotes the empirical data distribution and $\mathcal{D}_\pi$ denotes the occupancy distribution induced by the learned policy. The divergence between the two curves in Figure 46a illustrates how policy deployment may produce performance degradation due to distributional mismatch.

Figure 46b illustrates the phenomenon of distributional shift through occupancy measures. Let

$$w(s,a) = \frac{d^\pi(s,a)}{d^\mu(s,a)}, \tag{1141}$$

where d^π and d^μ denote the occupancy distributions under the target policy π and behavior policy μ, respectively. Expectations under the target distribution satisfy

$$\mathbb{E}_{\mathcal{D}_\pi}[f(s,a)] = \mathbb{E}_{\mathcal{D}_{\text{train}}}\left[w(s,a) f(s,a)\right]. \tag{1142}$$

The figure demonstrates that large deviations between d^π and d^μ lead to large density ratios, thereby amplifying approximation and estimation errors in poorly covered regions of the state-action space.

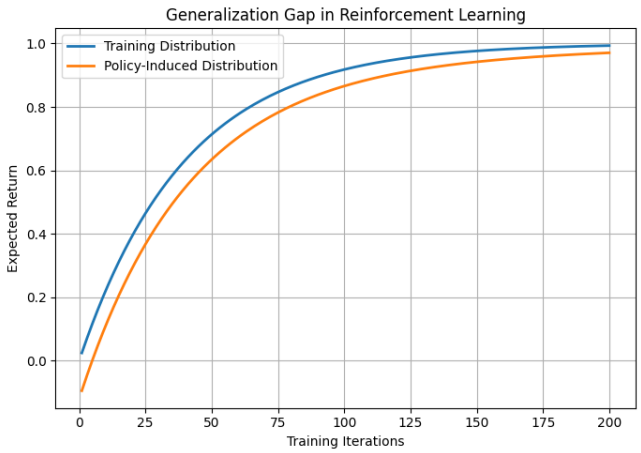
Figure 46c shows the decay of statistical learning bounds as the number of samples increases. For a hypothesis class $\mathcal{F}$, uniform convergence theory yields bounds of the form

$$\sup_{f \in \mathcal{F}} \left| \mathbb{E}[f] - \frac{1}{N}\sum_{i=1}^{N} f(s_i, a_i) \right| \le \mathcal{O}\left(\frac{\text{Complexity}(\mathcal{F})}{\sqrt{N}}\right), \tag{1143}$$

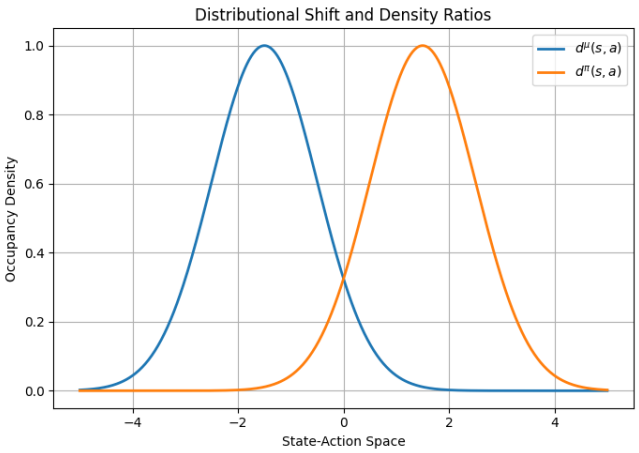
where Complexity($\mathcal{F}$) may correspond to VC-dimension, Rademacher complexity, or covering numbers. The inverse square-root decay shown in the figure reflects the classical statistical convergence rate associated with empirical risk minimization.

Figure 46d illustrates Bellman error amplification. If the Bellman residual satisfies

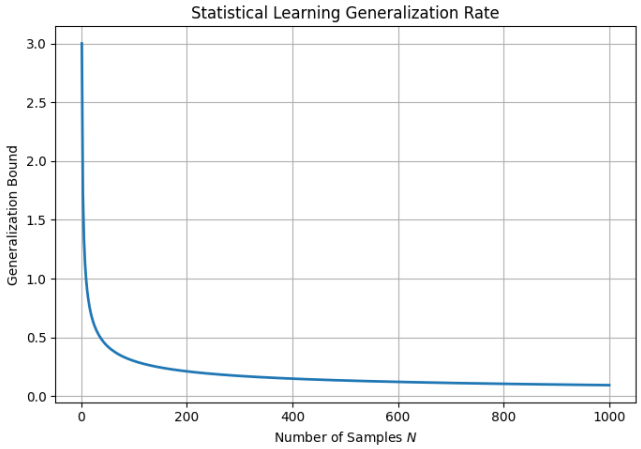
$$\|\hat{Q} - T\hat{Q}\|_\infty \le \varepsilon, \tag{1144}$$



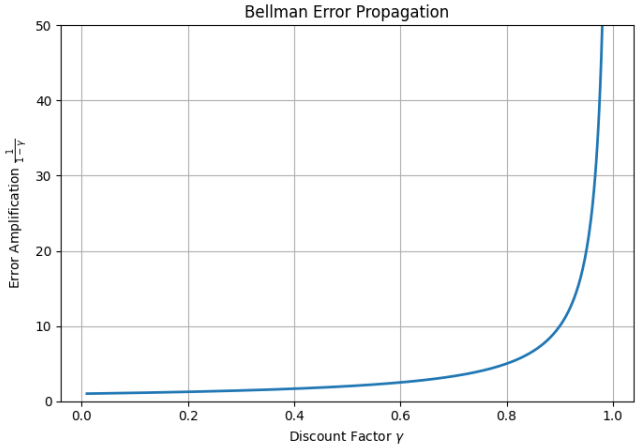
(a) **Generalization gap between training and policy-induced distributions.**



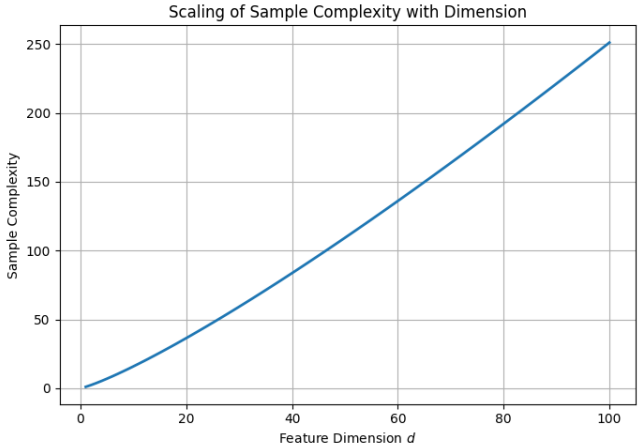
(b) **Distributional shift and density ratio mismatch between d^{μ} and d^{π}.**



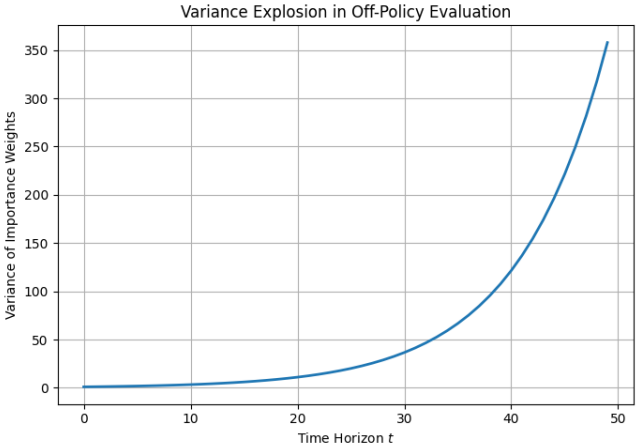
(c) **Decay of statistical generalization bounds with increasing sample size.**



(d) **Bellman error amplification as $\gamma \to 1$.**



(e) **Growth of sample complexity with feature-space dimension.**



(f) **Exponential growth of importance sampling variance in off-policy evaluation.**

Figure 46: **Illustrations of theoretical generalization guarantees and challenges in reinforcement learning, including generalization gaps, distributional shift, statistical learning bounds, Bellman error propagation, dimensional sample complexity scaling, and variance explosion in off-policy evaluation.**

then the induced performance error obeys

$$\|\hat{Q} - Q^*\|_\infty \leq \frac{\varepsilon}{1-\gamma}. \tag{1145}$$

As the discount factor $\gamma \to 1$, the amplification factor

$$\frac{1}{1-\gamma} \tag{1146}$$

grows rapidly, as shown in Figure 46d. This demonstrates that even small approximation errors can propagate through recursive Bellman updates and produce substantial value estimation errors.

Figure 46e illustrates the growth of sample complexity with feature-space dimension. Under structured assumptions such as linear MDPs,

$$P(s'|s,a) = \phi(s,a)^\top \psi(s'), \qquad R(s,a) = \phi(s,a)^\top \theta, \tag{1147}$$

one obtains finite-sample guarantees of the form

$$N = \tilde{\mathcal{O}}\left(\frac{d}{(1-\gamma)^3 \varepsilon^2}\right), \tag{1148}$$

where d denotes the intrinsic feature dimension. The figure illustrates that increasing representational complexity generally requires significantly larger datasets to achieve reliable generalization performance.

Finally, Figure 46f illustrates the variance explosion problem in off-policy evaluation. Importance sampling estimates the value of a target policy π using trajectories generated by a behavior policy μ:

$$J(\pi) = \mathbb{E}^\mu \left[\sum_{t=0}^{\infty} \gamma^t w_t R_t\right], \tag{1149}$$

where the importance weight is

$$w_t = \prod_{k=0}^{t} \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}. \tag{1150}$$

Because the product accumulates multiplicatively over time, the variance satisfies

$$\mathrm{Var}(w_t) \to \infty \quad \text{as } t \to \infty, \tag{1151}$$

as reflected by the exponential growth shown in Figure 46f. This phenomenon represents one of the major obstacles to stable off-policy generalization in reinforcement learning.

Collectively, Figures 46a, 46b, 46c, 46d, 46e, and 46f illustrate the central theoretical mechanisms governing generalization in reinforcement learning, including distributional mismatch, statistical complexity, recursive Bellman error propagation, dimensional sample complexity growth, and instability in off-policy evaluation.

In summary, theoretical generalization in reinforcement learning requires integrating tools from statistical learning theory, stochastic processes, and control theory. The key challenges arise from temporal dependence, distributional shift, and error propagation through recursive structures. Future progress depends on identifying appropriate complexity measures, deriving tight finite-sample bounds, and developing algorithms whose empirical success can be matched by rigorous guarantees in high-dimensional, function approximation regimes.

13 Conclusion

Reinforcement Learning offers a mathematically elegant framework for sequential decision-making. Despite strong theoretical guarantees, scaling RL to real-world problems remains an open challenge.

References

[1] Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction (Vol. 1, No. 1, pp. 9-11). Cambridge: MIT press.

[2] Bertsekas, D. P., & Tsitsiklis, J. N. (1996). Neuro-dynamic programming (1st ed.). Athena Scientific.

[3] Puterman, M. L. (2014). Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons.

[4] Bertsekas, D., & Shreve, S. E. (1996). Stochastic optimal control: the discrete-time case (Vol. 5). Athena Scientific.

[5] Hernández-Lerma, O., & Lasserre, J. B. (2012). Discrete-time Markov control processes: basic optimality criteria (Vol. 30). Springer Science & Business Media.

[6] Cohn, D. L. (2013). Measure theory (Vol. 2). New York: Birkhäuser.

[7] Feinberg, E. A., & Shwartz, A. (Eds.). (2012). Handbook of Markov decision processes: methods and applications (Vol. 40). Springer Science & Business Media.

[8] Hernández-Lerma, O., & Lasserre, J. B. (2012). Further topics on discrete-time Markov control processes (Vol. 42). Springer Science & Business Media.

[9] Bertsekas, D. (2012). Dynamic programming and optimal control: Volume I (Vol. 4). Athena scientific.

[10] Norris, J. R. (1998). Markov chains (No. 2). Cambridge university press.

[11] Revuz, D. (2008). Markov chains (Vol. 11). Elsevier.

[12] Dynkin, E. B., & Yushkevich, A. A. (1979). Controlled markov processes (Vol. 69). New York: Springer-Verlag.

[13] Szepesvári, C. (2022). Algorithms for reinforcement learning. Springer nature.

[14] Watkins, C. J. C. H. (1989). Learning from delayed rewards.

[15] Bellman, R. I. C. H. A. R. D. (1957). Dynamic programming, princeton univ. Press Princeton, New Jersey, 39.

[16] Howard, R. A. (1960). Dynamic programming and markov processes.

[17] Kreyszig, E. (1991). Introductory functional analysis with applications. John Wiley & Sons.

[18] Munos, R. (2003, August). Error bounds for approximate policy iteration. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning (pp. 560-567).

[19] Meyer, C. D. (2023). Matrix analysis and applied linear algebra. Society for Industrial and Applied Mathematics.

[20] Horn, R. A., & Johnson, C. R. (2012). Matrix analysis. Cambridge university press.

[21] Ortega, J. M., & Rheinboldt, W. C. (2000). Iterative solution of nonlinear equations in several variables. Society for Industrial and Applied Mathematics.

[22] Ghosh, S. (2025). Mathematical Foundations of Deep Learning.

[23] Scherrer, B. (2014, June). Approximate policy iteration schemes: A comparison. In International Conference on Machine Learning (pp. 1314-1322). PMLR.

[24] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. Machine learning, 3(1), 9-44.

[25] Tsitsiklis, J., & Van Roy, B. (1996). Analysis of temporal-diffference learning with function approximation. Advances in neural information processing systems, 9.

[26] Watkins, C. J., & Dayan, P. (1992). Q-learning. Machine learning, 8(3), 279-292.

[27] Jaakkola, T., Jordan, M., & Singh, S. (1993). Convergence of stochastic iterative dynamic programming algorithms. Advances in neural information processing systems, 6.

[28] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3), 229-256.

[29] Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. Advances in neural information processing systems, 12.

[30] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014, January). Deterministic policy gradient algorithms. In International conference on machine learning (pp. 387-395). Pmlr.

[31] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

[32] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N. M. O., Erez, T., Tassa, Y., ... & Wierstra, D. P. (2020). U.S. Patent No. 10,776,692. Washington, DC: U.S. Patent and Trademark Office.

[33] Van Hasselt, H., Guez, A., & Silver, D. (2016, March). Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 30, No. 1).

[34] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), 484-489.

[35] Hasselt, H. (2010). Double Q-learning. Advances in neural information processing systems, 23.

[36] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016, June). Dueling network architectures for deep reinforcement learning. In International conference on machine learning (pp. 1995-2003). PMLR.

[37] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv preprint arXiv:1511.05952.

[38] Lai, T. L., & Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. Advances in applied mathematics, 6(1), 4-22.

[39] Bubeck, S., & Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. arXiv preprint arXiv:1204.5721.

[40] Lattimore, T., & Szepesvári, C. (2020). Bandit algorithms. Cambridge University Press.

[41] Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. Journal of machine learning research, 3(Nov), 397-422.

[42] Auer, P., Jaksch, T., & Ortner, R. (2008). Near-optimal regret bounds for reinforcement learning. Advances in neural information processing systems, 21.

[43] Ziebart, B. D., Maas, A. L., Bagnell, J. A., & Dey, A. K. (2008, July). Maximum entropy inverse reinforcement learning. In Aaai (Vol. 8, pp. 1433-1438).

[44] Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., ... & Levine, S. (2018). Soft actor-critic algorithms and applications. arXiv preprint arXiv:1812.05905.

[45] Neu, G., Jonsson, A., & Gómez, V. (2017). A unified view of entropy-regularized markov decision processes. arXiv preprint arXiv:1705.07798.

[46] Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika, 25(3/4), 285-294.

[47] Daniel, J. R., Benjamin, V. R., Abbas, K., Ian, O., & Zheng, W. (2018). A tutorial on thompson sampling. Foundations and Trends® in Machine Learning, 11(1), 1-99.

[48] Osband, I., Blundell, C., Pritzel, A., & Van Roy, B. (2016). Deep exploration via bootstrapped DQN. Advances in neural information processing systems, 29.

[49] Kearns, M., & Singh, S. (2002). Near-optimal reinforcement learning in polynomial time. Machine learning, 49(2), 209-232.

[50] Agarwal, R., Schuurmans, D., & Norouzi, M. (2020, November). An optimistic perspective on offline reinforcement learning. In International conference on machine learning (pp. 104-114). PMLR.

[51] Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In Machine learning proceedings 1995 (pp. 30-37). Morgan Kaufmann.

[52] Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In Icml (Vol. 99, pp. 278-287).

[53] Wiewiora, E. (2003). Potential-based shaping and Q-value initialization are equivalent. Journal of Artificial Intelligence Research, 19, 205-208.

[54] Powell, W. B. (2007). Approximate Dynamic Programming: Solving the curses of dimensionality (Vol. 703). John Wiley & Sons.

[55] Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643.

[56] Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative q-learning for offline reinforcement learning. Advances in neural information processing systems, 33, 1179-1191.

[57] Fujimoto, S., Meger, D., & Precup, D. (2019, May). Off-policy deep reinforcement learning without exploration. In International conference on machine learning (pp. 2052-2062). PMLR.

[58] Buşoniu, L., Babuška, R., & De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. Innovations in multi-agent systems and applications-1, 183-221.

[59] Shoham, Y., & Leyton-Brown, K. (2008). Multiagent systems: Algorithmic, game-theoretic, and logical foundations. Cambridge University Press.

[60] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. Advances in neural information processing systems, 30.

[61] Garcıa, J., & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. Journal of Machine Learning Research, 16(1), 1437-1480.

[62] Achiam, J., Held, D., Tamar, A., & Abbeel, P. (2017, July). Constrained policy optimization. In International conference on machine learning (pp. 22-31). Pmlr.

[63] Chow, Y., Ghavamzadeh, M., Janson, L., & Pavone, M. (2018). Risk-constrained reinforcement learning with percentile risk criteria. Journal of Machine Learning Research, 18(167), 1-51.

[64] Jiang, N., & Li, L. (2016, June). Doubly robust off-policy value evaluation for reinforcement learning. In International conference on machine learning (pp. 652-661). PMLR.

[65] Farahmand, A. M., Szepesvári, C., & Munos, R. (2010). Error propagation for approximate policy and value iteration. Advances in neural information processing systems, 23.

[66] Agarwal, A., Jiang, N., Kakade, S. M., & Sun, W. (2022). Reinforcement learning: Theory and algorithms. 2021. URL https://rltheorybook. github. io, 1.