November 2015

# Safe Reinforcement Learning

Philip S. Thomas

# SAFE REINFORCEMENT LEARNING

A Dissertation Presented

by

PHILIP S. THOMAS

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2015

College of Information and Computer Sciences

# SAFE REINFORCEMENT LEARNING

A Dissertation Presented

by

PHILIP S. THOMAS

Approved as to style and content by:

---

Andrew G. Barto, Chair

---

Sridhar Mahadevan, Member

---

Shlomo Zilberstein, Member

---

Weibo Gong, Member

---

James Allan, Chair
College of Information and Computer Sciences

# ABSTRACT

## SAFE REINFORCEMENT LEARNING

SEPTEMBER 2015

PHILIP S. THOMAS

B.Sc., CASE WESTERN RESERVE UNIVERSITY

M.Sc., CASE WESTERN RESERVE UNIVERSITY

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Andrew G. Barto

This dissertation proposes and presents solutions to two new problems that fall within the broad scope of *reinforcement learning* (RL) research. The first problem, *high confidence off-policy evaluation* (HCOPE), requires an algorithm to use historical data from one or more behavior policies to compute a high confidence lower bound on the performance of an evaluation policy. This allows us to, for the first time, provide the user of any RL algorithm with confidence that a newly proposed policy (which has never actually been used) will perform well.

The second problem is to construct what we call a *safe reinforcement learning algorithm*—an algorithm that searches for new and improved policies, while ensuring that the probability that a "bad" policy is proposed is low. Importantly, the user of the RL algorithm may tune the meaning of "bad" (in terms of a desired performance baseline) and how low the probability of a bad policy being deployed should be, in order to capture the level of risk that is acceptable for the application at hand.

We show empirically that our solutions to these two critical problems require surprisingly little data, making them practical for real problems. While our methods allow us to, for the first time, produce convincing statistical guarantees about the performance of a policy without requiring its execution, the primary contribution of this dissertation is *not* the methods that we propose. The primary contribution of this dissertation is a compelling argument that these two problems, HCOPE and safe reinforcement learning, which at first may seem out of reach, are actually tractable. We hope that this will inspire researchers to propose their own methods, which improve upon our own, and that the development of increasingly data-efficient safe reinforcement learning algorithms will catalyze the widespread adoption of reinforcement learning algorithms for suitable real-world problems.

# TABLE OF CONTENTS

# LIST OF TABLES

**Table** **Page**

# LIST OF FIGURES

**Figure** **Page**

# CHAPTER 1

# INTRODUCTION

As our ability to construct intelligent machines improves, so too must our ability to ensure that they are safe. Advances in artificial intelligence over the past few decades have resulted in an abundance of proposed and real applications that are both increasingly beneficial and risky. Consider, for example, self driving cars (Kim et al., 2013), machines that guide medical policy and practice (Thapa et al., 2005), and general purpose robotic workers (Haddadin et al., 2011), all of which have the potential to revolutionize our lives for the better. The large positive impacts of these applications comes with an increased necessity for safety measures since the failure of any of these intelligent systems could be catastrophic.

This dissertation focuses on constructing safe intelligent machines that learn how to make sequences of decisions that result in desirable outcomes. For example, the decisions could encode how much energy to give to a motor in a robot at each moment to keep the robot balanced (Kuindersma et al., 2013), the sequence of interventions that best control the spread of forest fires (Houtman et al., 2013), or how much of a drug to give patients to optimize their prognosis (Moore et al., 2010). *Reinforcement learning* (RL) research (Sutton and Barto, 1998) studies such (safe or unsafe) intelligent decision-making machines, called *agents*, which learn from their experiences.

This dissertation focuses on RL applications where some decision-making mechanism, called a *policy*, is already in use. This policy makes decisions, and the outcomes of those decisions are observed and recorded. RL algorithms can analyze this historical data to propose a new policy that may be better than the policy currently being

used. In order to define what makes an RL algorithm *safe* in this context, consider a motivating example: optimization of a policy for the treatment of a disease that is sometimes terminal. Doctors currently implement a closed-loop treatment policy when they decide how to treat a patient. Analysis of data from previous cases could be used to propose a new and improved treatment policy. In this example, what would make a policy "safe" to deploy? We contend that the new policy should be deemed safe if and only if it is guaranteed with high confidence to be at least as "good" as the current policy, where the definition of "good" will be formalized later.

This definition of safety extends beyond this one example to any other application where some policy is currently being used and the deployment of a worse policy could be costly or dangerous. For the remainder of this dissertation, unless otherwise specified, **we call an RL method *safe* if it guarantees with high confidence that every change that it makes to the decision-making mechanism (policy) will be an improvement.**

There are many other notions of safety in RL research. For example, even though our methods give probabilistic guarantees that a new policy is an improvement over the current policy, how can we ensure that our formal definition of what makes a policy "good" does not result in an "improved" policy that has less desirable behavior? We discuss this and other notions of safety in Section 2.6.

## 1.1 Contributions

This dissertation has three chapters that contain contributions: Chapters 3, 4, and 5. The primary contributions of Chapter 3 are the derivation of the *consistent weighted per-decision importance sampling* (CWPDIS) estimator for off-policy evaluation, and proofs that several similar estimators are strongly consistent estimators of the performance of the evaluation policy even if there are multiple behavior policies (but if an additional restriction is satisfied).

The primary contributions of Chapter 4 are the formalization of the exact and approximate HCOPE problems, the construction of exact and approximate HCOPE methods, and the derivation of a concentration inequality, called the *collapsed upper tail* (CUT) inequality, that is particularly well suited to HCOPE.

The primary contributions of Chapter 5 are the definition of *safe reinforcement learning algorithms*, and the construction of the first safe batch and incremental reinforcement learning algorithms.

## 1.2 Layout

The remainder of this dissertation is structured as follows:

- Chapter 2 (Background and Related Work): This chapter provides background information that is relevant to all subsequent chapters, and a review of work that is related to the overarching concept of ensuring safety in reinforcement learning research.

- Chapter 3 (Importance Sampling for Off-Policy Evaluation): This chapter discusses different estimators of the performance of an evaluation policy using historical data from one or more behavior policies. This is the most technical chapter, and although the methods it derives improve the performance of subsequent algorithms, the subsequent chapters remain coherent if this chapter is skipped (if the reader is familiar with ordinary importance sampling for reinforcement learning).

- Chapter 4 (High Confidence Off-Policy Evaluation, HCOPE): This chapter presents and proposes a solution to the HCOPE problem, and is the heart of this dissertation.

- Chapter 5 (Safe Policy Improvement): This chapter formalizes the notion of *safe reinforcement learning* and proposes batch and incremental safe reinforcement

learning algorithms. These algorithms build upon the foundation laid in the previous chapter.

- Chapter 6 (Conclusion and Future Work): This brief chapter summarizes the work and proposes avenues for future work.

# CHAPTER 2

# BACKGROUND AND RELATED WORK

This chapter presents background and discussion of related work that is relevant to all of the chapters of this dissertation and its overarching message of safety in RL. Several later chapters contain additional specialized background and related work sections.

## 2.1 Notation

The notation is this dissertation is standardized as follows. Sets are written in calligraphic capital letters, e.g., $\mathcal{X}$. Elements of sets are lowercase letters that are typically similar to the set they belong to, e.g., $x \in \mathcal{X}$. We write $\{x \in \mathcal{X} : A(x)\}$ to denote the set of elements in $\mathcal{X}$ that satisfy some Boolean statement $A(x)$. When it is inconvenient to specify the set $\mathcal{X}$ (but its definition is clear), we sometimes use the shorthand $\{x : A(x)\}$. If $\mathcal{X} \subseteq \mathcal{Z}$ and $\mathcal{Y} \subseteq \mathcal{Z}$, then we write $\mathcal{X} \setminus \mathcal{Y}$ to denote $\{z \in \mathcal{Z} : z \in \mathcal{X}, z \notin \mathcal{Y}\}$, where we use a comma to denote the Boolean operator AND. We also write $\{X_i\}_{i=1}^n$ to denote the set $\{X_1, \ldots, X_n\}$.

Whereas we use brackets to denote a set, we use parentheses to denote ordered collections of objects (sequences or tuples). For example, $(x, y)$ is the ordered pair with the first element equal to $x$ and the second element equal to $y$. We write $(X_i)_{i=1}^n$ to denote $(X_1, \ldots, X_n)$.

Random variables are denoted by capital letters, e.g., $X$, and their instantiations by lowercase letters, e.g., $x$. We denote by $\operatorname{supp} p$ the support of a probability mass function $p$, i.e., $\{x : p(x) \neq 0\}$.

Also, in some cases we write $\sum_{x \in \mathcal{X}} \ldots$ to denote the sum where $x$ takes every value in the set $\mathcal{X}$. However, we also sometimes use the shorthand $\sum_x$ to denote the same thing when the set $\mathcal{X}$ is clear from context but inconvenient to specify.

Lastly, we use $A \coloneqq B$ to denote that $A$ is defined to be equal to $B$. If an equation is too long for one line we split it across two lines. We use $\times$ to denote that scalar multiplication has been split across two lines.

## 2.2 Environments of Interest

In this section we discuss our decision to model the environment as a *partially observable Markov decision process* (POMDP) rather than as a *Markov decision process* (MDP), even though we are primarily interested in MDPs. Both MDPs and POMDPs are defined formally in the following section (Section 2.3).

Real world problems that are not MDPs are often shoehorned into the MDP formulation (by ignoring partial observability and non-Markovian dynamics) to allow for the application of methods designed for MDPs (Thomas, 2009, pages 26–27). When presenting safe algorithms, if we adopt the MDP formulation, this could be a source of error that is not accounted for by our analysis. Even though we are primarily interested in problems that *approximately* fit the MDP framework, we must therefore derive our methods and analyze them using the more general POMDP formulation. This shows that our methods, which are intended for MDPs, are valid even when the real problem being solved does not precisely fit the definition of an MDP, but does fit the more general POMDP formulation.

## 2.3 Partially Observable Markov Decision Processes (POMDPs) and Markov Decision Processes (MDPs)

We are interested in *sequential decision problems*: problems where an agent must optimize a sequence of decisions that it makes. A diagram of this paradigm is provided

in Figure 2.1. In this diagram the agent is the machine or algorithm that we design, the environment is the subset of the universe that is relevant to the agent, and the sensor is a mechanism that allows the agent to measure properties of the environment.

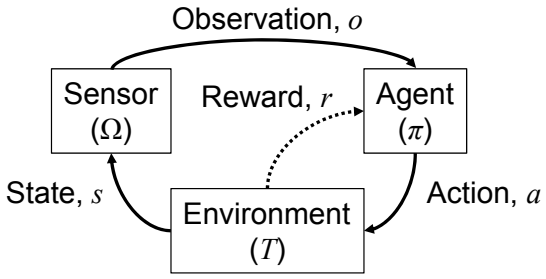


**Figure 2.1.** Diagram of how the agent and environment interact.

The process begins when the agent makes an observation, $o$, about the state, $s$, of the environment using some sort of sensor. The workings of this sensor will later be described using the symbol $\Omega$. For example, this observation could be a force reading and image from a force sensor and camera on a robot, or the description of the symptoms of a patient as reported by a doctor or nurse. The agent then makes a decision about which action, $a$, to select (e.g., what ability of the robot to execute next, or what treatment to recommend for the patient). Later we will use the symbol $\pi$ to denote the agent's mechanism for making this decision. The action causes the environment to transition to some new state according to dynamics that we will later associate with the symbol $T$. This transition also results in a scalar reward, $r$, that is some measure of how "good" the current state of the environment is.

This process repeats some finite number of times, after which it terminates. We call the sequence of states, observations, actions, and rewards that result from repeated interaction of the agent with the environment a *trajectory*. The agent's goal is to determine how to select actions so as to maximize the expected total reward on any future trajectories.

We formalize this notion of a sequential decision problem as a *partially observable Markov decision process* (POMDP). Let $t$ denote the iteration or *time step*, which begins with $t = 1$. Formally, we define a POMDP to be a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, d_{S_1}, \Omega, \mathcal{R})$, where:

1. $\mathcal{S}$ is the finite set of possible states that the environment can be in, and is called the *state set*. We write $s$ to denote an element of $\mathcal{S}$ and $S_t$ to denote the state that occurs at time $t$, which is a random variable. The agent is *not* provided with $\mathcal{S}$ (it does not know what states are possible *a priori*), nor is it provided with $S_t$.

2. $\mathcal{A}$ is the finite set of actions that the agent can select from, and is called the *action set*. We write $a$ to denote an element of $\mathcal{A}$ and $A_t$ to denote the action that occurs at time $t$, which is a random variable. We assume that the agent is provided with $\mathcal{A}$ (it knows what actions are possible *a priori*) and it is aware of which action, $A_t$, it has chosen at each time step.

3. $\mathcal{O}$ is the finite set of possible observations that the agent can make about the environment, and is called the *observation set*. We write $o$ to denote an element of $\mathcal{O}$ and $O_t$ to denote the observation that occurs at time $t$, which is a random variable. The agent is provided with $\mathcal{O}$ and also observes $O_t$.

4. We write $H_t$ to denote the complete history of states, observations, actions, and scalar rewards up until (and including) time $t$:

$$H_t \coloneqq (S_1, O_1, A_1, R_1, S_2, O_2, A_2, R_2, \ldots, S_t, O_t, A_t, R_t).$$

   Let $\mathcal{H}_t$ be the set of all possible histories of length $t$. For brevity we also define $\Xi_t$ to be the data at time $t$:

$$\Xi_t \coloneqq (S_t, O_t, A_t, R_t), \tag{2.1}$$

so that $H_t = (\Xi_i)_{i=1}^t$. We write $\xi_t \coloneqq (s_t, o_t, a_t, r_t)$ to denote a sample of $\Xi_t$ and $h_t \coloneqq (s_i, o_i, a_i, r_i)_{i=1}^t = (\xi_i)_{i=1}^t$ to denote a sample of $H_t$. So, whenever $h_t$ is defined in an equation, we have implicitly defined $s_i, o_i, a_i$, and $r_i$ for all $1 \leq i \leq t$. Also, to simplify later math, we define $H_0 \coloneqq \emptyset$.

5. $\mathcal{R}$ is called the *reward function*, and it governs how the bounded scalar reward, $R_t \in [r_{\text{lb}}, r_{\text{ub}}]$, is produced at time $t$.[1] Here we introduce the first Markov assumption—the reward at time $t$, $R_t$, depends only on $S_t, O_t, A_t$, and $S_{t+1}$:

$$\begin{aligned}\Pr\Big(R_t = r_t \Big| H_{t-1} = h_{t-1}, S_t = s_t, O_t = o_t, A_t = a_t, S_{t+1} = s_{t+1}\Big) \\ = \Pr\Big(R_t = r_t \Big| S_t = s_t, O_t = o_t, A_t = a_t, S_{t+1} = s_{t+1}\Big),\end{aligned}$$

for all $t \geq 1, r_t, h_{t-1}, s_t, o_t, a_t$, and $s_{t+1}$. Furthermore, the distribution over $R_t$ is independent of the time, $t$, given $S_t, O_t, A_t$, and $S_{t+1}$, i.e., for all $t \geq 1$ and $\hat{t} \geq 1$,

$$\begin{aligned}\Pr\Big(R_t = r \Big| S_t = s, O_t = o, A_t = a, S_{t+1} = s'\Big) \\ = \Pr\Big(R_{\hat{t}} = r \Big| S_{\hat{t}} = s, O_{\hat{t}} = o, A_{\hat{t}} = a, S_{\hat{t}+1} = s'\Big),\end{aligned}$$

for all $r, s, o, a$, and $s'$. So, we define $\mathcal{R}$ to be

$$\mathcal{R}(r|s, o, a, s') \coloneqq \Pr\Big(R_t = r \Big| S_t = s, O_t = o, A_t = a, S_{t+1} = s'\Big),$$

for all $s, o, a, r, s'$ and any any $t \geq 1$. The agent is provided with the reward signal, $R_t$.

---

[1] Notice that we have broken our notational convention by using a calligraphic letter, $\mathcal{R}$, to denote a function rather than a set—this is to differentiate the reward function (which is traditionally denoted by the letter $R$) from the rewards, $R_t$.

6. We call $T$ the *transition function* since it governs how the environment transitions between states. Here we introduce another Markov assumption: the distribution over $S_{t+1}$ depends only on $S_t$ and $A_t$, i.e.,

$$\Pr\Big(S_{t+1}=s_{t+1}\Big|H_t=h_t\Big)=\Pr\Big(S_{t+1}=s_{t+1}\Big|S_t=s_t,A_t=a_t\Big),$$

for all $t\geq 1, s_{t+1}$, and $h_t$ (recall that $s_t$ and $a_t$ are specified by $h_t$). Furthermore, the transition probabilities are independent of the time, $t$. So, for any $t\geq 1$ and $\hat{t}\geq 1$:

$$\Pr\Big(S_{t+1}=s'\Big|S_t=s,A_t=a\Big)=\Pr\Big(S_{\hat{t}+1}=s'\Big|S_{\hat{t}}=s,A_{\hat{t}}=a\Big),$$

for all $s, a$, and $s'$. So, we define $T(s'|s,a)$ to be the probability of the environment entering state $s'$ if action $a$ is taken in state $s$:

$$T(s'|s,a)\coloneqq\Pr\Big(S_{t+1}=s'\Big|S_t=s,A_t=a\Big),$$

for all $t\geq 1, s, a$, and $s'$. The agent is *not* provided with $T$—it can only learn about $T$ from its experiences.

7. We call $d_{S_1}$ the *initial state distribution* since it is the distribution over $S_1$: $d_{S_1}(s)=\Pr(S_1=s)$. Like $T$, $d_{S_1}$ is not provided to the agent—the agent can only learn about $d_{S_1}$ from its experiences.

8. We call $\Omega$ the *observation function* since it governs how the sensor produces observations. We introduce yet another Markov assumption: that the observation at time $t$ depends only on the state at time $t$, i.e.,

$$\Pr\Big(O_t=o_t\Big|H_{t-1}=h_{t-1},S_t=s_t\Big)=\Pr\Big(O_t=o_t\Big|S_t=s_t\Big),$$

for all $t \geq 1, h_{t-1}, s_t$, and $o_t$. We also assume that the observations do not depend directly on the time step $t$. So, for any $t \geq 1$ and $\hat{t} \geq 1$ we have

$$\Pr\left(O_t = o \middle| S_t = s\right) = \Pr\left(O_{\hat{t}} = o \middle| S_{\hat{t}} = s\right),$$

for all $o$ and $s$. So, we define $\Omega(o|s)$ to be the probability that the sensor produces observation $o$ when the agent is in state $s$. That is, for any $t \geq 1$

$$\Omega(o|s) \coloneqq \Pr\left(O_t = o \middle| S_t = s\right).$$

The agent is *not* provided with $\Omega$.

*Markov decision processes* (MDPs) are POMDPs with the additional requirement that the agent can observe the full state. Formally, this means that $\mathcal{O} = \mathcal{S}$ and $O_t = S_t$ always.

For simplicity, we have formalized POMDPs using finite state, action, and observation sets. However, the analyses and methods in this dissertation extend to the settings where these sets are countably or uncountably infinite and where the probability distributions thereover may or may not have probability density functions (Thomas et al., 2015b).

The agent's mechanism for selecting actions is called the *policy*, and is typically denoted by $\pi$. Here we introduce our final Markov assumption—that the policy is memoryless and stationary. We call the policy *memoryless* because the action at time $t$ depends only on the observation at time $t$:

$$\Pr\left(A_t = a_t \middle| H_{t-1} = h_{t-1}, S_t = s_t, O_t = o_t\right) = \Pr\left(A_t = a_t \middle| S_t = s_t, O_t = o_t\right),$$

for all $t \geq 1, h_{t-1}, s_t, o_t$, and $a_t$. Memoryless policies for POMDPs are sometimes also referred to as *state-free* policies (Kaelbling et al., 1996, Sections 7.1 and 7.2). Our

use of memoryless policies places the onus of state estimation on the mechanism, $\Omega$, which produces observations. That is, any inference about the true underlying state should be performed by the mechanism, $\Omega$, that produces the observations, since the distribution over $A_t$ depends only on $O_t$. We call the policy *stationary* because the action probabilities are independent of the time step, $t$, given $O_t$, i.e., for all $t \geq 1$ and $\hat{t} \geq 1$:

$$\Pr\Big(A_t = a \Big| S_t = s, O_t = o\Big) = \Pr\Big(A_{\hat{t}} = a \Big| S_{\hat{t}} = s, O_{\hat{t}} = o\Big),$$

for all $s, o$, and $a$. Let $\pi(\cdot|o)$ denote the conditional distribution over the action set given that the agent last observed $o$. That is, for any $t \geq 1$,

$$\pi(a|o) \coloneqq \Pr\Big(A_t = a \Big| O_t = o\Big).$$

A *parameterized policy*, $\pi_\theta$, is a policy that is parameterized by a *parameter vector*, $\theta$, containing $n_\theta$ real-valued parameters: $\theta \in \mathbb{R}^{n_\theta}$. As $\theta$ changes, so to does the distribution over actions chosen by the agent. We write $\Pi$ to denote the set of all possible policies.

There are at most $L$ actions taken by the agent before the process terminates. Each such sequence of $L$ interactions between the agent and environment from $t = 1$ to $L$ is called an *episode*. We call the history of an entire episode, $H_L$, a *trajectory*. When the process terminates, the agent enters a special state, $\overset{\infty}{s}$, called the *absorbing state*. The absorbing state only has a single admissible action (so every policy is the same in the absorbing state), it always transitions to itself, and transitioning to it always produces a reward of zero. So, once the agent enters the absorbing state there are no more decisions to be made or rewards to be obtained. By padding episodes with absorbing states, we may sum from $t = 1$ to $L$ for all trajectories when working with $S_t, O_t, A_t$, or $R_t$.

Since each policy produces a distribution over histories, we will abuse notation and sometimes treat $\pi$ as a distribution over histories so that we can write $H_t \sim \pi$ to denote that the history through time $t$, $H_t$, was sampled by executing the policy $\pi$ on the POMDP. We also write $\text{supp}_\text{t}\, \pi$ to denote the set of histories of length $t$ that have non-zero probability of occuring when using the policy $\pi$, i.e., $\text{supp}_t\, \pi \coloneqq \{h_t : \Pr(H_t = h_t|\pi) \neq 0\}$.

We write $G(H_L) \in [0, 1]$ to denote some quantification of how good the trajectory $H_L$ is, and we call $G(H_L)$ the *return of trajectory* $H_L$. Notice that this differs from the standard definition in RL literature, which does not require the return to be normalized to the range $[0, 1]$. After each episode completes, the agent is provided with $G(H_L)$. Although our algorithms and analyses are agnostic to the precise definition of $G(H_L)$, in our experiments we use the standard *normalized discounted return*:

$$G(H_L) \propto \sum_{t=1}^{L} \gamma^{t-1} R_t, \tag{2.2}$$

where $\gamma \in [0, 1]$ is a parameter that discounts rewards based on how late in the trajectory they occur. Notice the $\propto$ in (2.2), which denotes that the discounted sum of rewards must be normalized to ensure that $G(H_L) \in [0, 1]$. If no domain specific knowledge about the possible magnitude of the discounted sum of rewards is available, then $G(H_L)$ can be normalized as follows. Notice that

$$G_\text{ub} \coloneqq \begin{cases} r_\text{ub} \frac{1-\gamma^L}{1-\gamma} & \text{if } \gamma < 1 \\ L r_\text{ub} & \text{otherwise,} \end{cases} \tag{2.3}$$

and

$$G_\text{lb} \coloneqq \begin{cases} r_\text{lb} \frac{1-\gamma^L}{1-\gamma} & \text{if } \gamma < 1 \\ L r_\text{lb} & \text{otherwise,} \end{cases} \tag{2.4}$$

are upper and lower bounds on $\sum_{t=1}^{L} \gamma^{t-1} R_t$. We can therefore write (2.2) without the $\propto$ as:

$$G(H_L) \coloneqq \frac{\left(\sum_{t=1}^{T} \gamma^{t-1} R_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}. \tag{2.5}$$

The agent's goal is to find, over the course of multiple episodes, a policy that results in it obtaining the largest returns possible, on average. This notion is formalized by the agent's objective function $\rho : \Pi \to \mathbb{R}$, which gives the expected return when using the specified policy:

$$\rho(\pi) \coloneqq \mathrm{E}\left[G(H_L) | H_L \sim \pi\right] \in [0, 1].$$

## 2.4 Limitations of the POMDP Framework

The POMDP framework as we have defined it suffers from two primary drawbacks. First, we have adopted the *finite horizon* setting by requiring all trajectories to terminate within $L$ time steps. This precludes indefinite horizon environments where the length of each episode is unbounded (but not infinite) and infinite horizon environments where episodes never terminate.

Second, we have made an implicit stationarity assumption: the POMDP does not change between episodes. That is, $\mathcal{S}, \mathcal{A}, \mathcal{O}, T, d_{S_1}, \Omega$, and $\mathcal{R}$ are the same for every episode. This means that the POMDP cannot account for changes in the system that occur across episodes. This assumption means that our safety guarantees will not account for some real factors. For example:

1. Consider the optimization of a controller for a robot. If data were collected in the past and used on a robot after it has suffered from additional wear, the dynamics of the robot may have changed, and this change will not be accounted for in our analysis.

2. Consider the optimization of a system that suggests medical treatments. If historical data were collected prior to the mutation of some bacteria to become

resistant to a specific treatment, our approach will be overly optimistic about the performance of that treatment (since our approach uses the past data).

Intuitively, a stationarity assumption of some sort is necessary. Our methods use the data that have been collected in the past to determine how to act in the future. Without some assumption that the future will resemble the past, the data that have been collected cannot be used to inform future decisions. Such assumptions of stationarity, although varying in their restrictiveness, are therefore common across all of machine learning research.

## 2.5 Gridworld

To ground the subsequent theoretical discussion, we present an example POMDP that we refer to throughout our derivations. Since this example will assist with our presentation, it is meant to be representative of many properties common to real world POMDPs, even though it is *not* an impressive or motivating example. We call this POMDP the *gridworld*, and it is depicted in Figure 2.2. In the gridworld the agent resides on a $4 \times 4$ grid, and the agent correctly observes its position. So, the observation set is $\mathcal{O} = \{1, 2, 3, 4\} \times \{1, 2, 3, 4\} \cup \left\{\overset{\infty}{s}\right\}$. Let $(x, y) \in \mathcal{O}$ denote the Cartesian coordinates of the agent, with $x$ denoting its horizontal position (larger $x$ move the agent to the right) and $y$ denoting its vertical position (larger $y$ move the agent down). The agent always begins in the top left position, $(1, 1)$. The agent has four available actions: $\mathcal{A} = \{\text{up, down, left, right}\}$, which deterministically move the agent one square along the grid in the specified direction. So, if the agent is in position $(2, 2)$ and it selects the action $A_t = \text{down}$, then it will move to the position $(2, 3)$ with probability one. If the agent selects an action that would cause it to move outside of the grid, then the agent stays in the same position.

Each episode terminates if the agent reaches the bottom right position, $(4, 4)$, or if 100 time steps have passed, i.e., $L = 100$. So, while $S_t = \overset{\infty}{s}$ if $t > L$, it can occur that

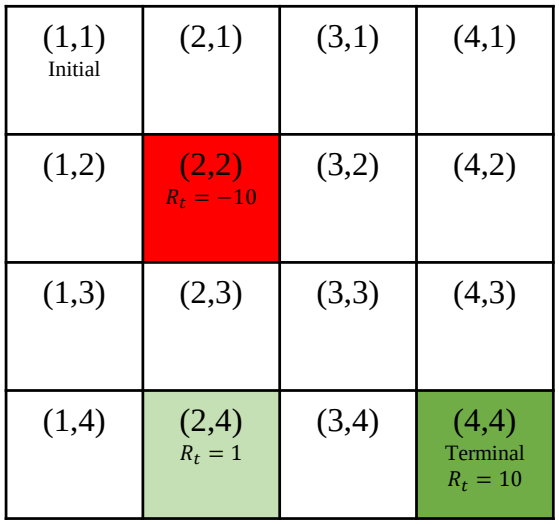


**Figure 2.2.** Graphical depiction of the gridworld. Each square is a possible observation, and the initial and terminal positions are marked. The reward for entering each state is $-1$ except for when the agent enters the positions that are otherwise labeled. The colors represent positions that result in larger and smaller than normal rewards.

$S_L \neq^{\infty} s$ if the agent does not reach the bottom right corner before time runs out. In order for the transition function to cause a transition to the absorbing state after 100 time steps, the underlying state includes an encoding of the current time step, which is unobserved to the agent. So, $\mathcal{S} = \left( \underbrace{\{1,2,3,4\} \times \{1,2,3,4\}}_{\text{position}} \times \underbrace{\{1,\ldots,100\}}_{\text{time}} \right) \cup \left\{ \overset{\infty}{s} \right\}$. The state $S_t = (x, y, t)$ encodes the position of the agent and the current time step, $t$, unless $S_t =^{\infty} s$.

The reward for each transition depends only on the position that the agent enters. A reward of $-1$ is given for entering any position except for three. Entering the position $(2, 2)$ causes a large penalty of $R_t = -10$, which the agent should attempt to avoid. Entering the position $(2, 4)$ causes a small positive reward of $R_t = 1$, and so the agent should tend towards this position. Entering the terminal position $(4, 4)$, provides a large reward of $R_t = 10$. Notice that rewards are provided based on the state that is *entered*, regardless of where it was entered from. For example, the

transition from $(2,2)$ to any adjacent position results in a reward of $R_t = -1$ while the transition to $(2,2)$ from any position results in a reward of $R_t = -10$.

For this POMDP and all others in this dissertation, we select $\gamma = 1$ and use the normalized discounted return defined in (2.5) for $G(H_L)$. Although we do not use $\gamma$ (we set it to one), we include it in our derivations in case a reader is faced with an application where the use of $\gamma < 1$ is desired. For normalization, $r_{\text{lb}} = -10$ and $r_{\text{ub}} = 10$. Also, the smallest possible return is bounded below by $G_{\text{lb}} = -10L$ (the worst state happens at every step) and the largest possible return is bounded above by $G_{\text{ub}} = 1(L-1) + 10$ (the small positive reward happens at every step except for the last, where the large reward just prior to terminating is received). Notice that these bounds are loose—we could use more knowledge about the domain to tighten them. We use loose bounds for this example since tight bounds may, in the absence of domain-specific knowledge, not be available for many real world applications.

Notice also that the optimal policy is not deterministic. Ideally, the agent would select the "down" action three times to move to $(1,4)$, followed by the "right" action once to move to $(2,4)$. It would then select the action "down" repeatedly, causing it to remain in the same position and collect several $+1$ rewards. When $t = 98$ and $t = 99$ the agent would select the action to move right so that it reaches the terminal position, $(4,4)$ just in time to receive the $+10$ reward before the trajectory terminates due to time running out. However, since the agent must select its actions without observing the time step, it cannot implement this deterministic policy.

So, any optimal policy (which does not depend on the unobserved time step) must be stochastic. It causes movement to $(2,4)$, where it chooses randomly (with some probability) between the "down" and "right" actions. Once it reaches $(3,4)$, it then deterministically moves to $(4,4)$ causing the episode to terminate. The challenging question is to determine what the action probabilities in $(2,4)$ should be for an optimal policy. If the probability of "down" is too small, then the agent will not remain in

that position very long in most trajectories. If it is too large, then the agent will rarely collect the reward of +10 in position $(4, 4)$ before the time limit runs out.

When we represent policies for the gridworld, we use tabular softmax action selection (Sutton and Barto, 1998).

## 2.6 Related Work

As evidenced by a recent open letter, organized by the Future Life Institute and signed by many prominent artificial intelligence researchers (Future Life Institute, 2015), ensuring safety is an important current issue across artificial intelligence research. The type of safe algorithms presented in this dissertation are a small part of this larger effort to ensure safety in artificial intelligence research. This section reviews some other approaches to ensuring different types of safety in sequential decision problems (a review of safety related work in other branches of artificial intelligence is beyond the scope of this work).

### 2.6.1 Control Theoretic Approaches

Like RL research, control theoretic research focuses on how to select actions to control a system. Control research, including *optimal control* (Bertsekas, 1976, Bertsekas and Shreve, 2007, Khalil, 2001, Lubin and Athans, 1996, Stengel, 1986), *adaptive control* (Ioannou, 2012), and particularly *robust control* (Zhou et al., 1995), deals with constructing controllers (policies) that ensure various forms of optimality and safety, ranging from guaranteed stability of a system to guarantees that regions of the state space will never be visited.

The distinction between control theoretic research and reinforcement learning research is tenuous. However, control theoretic methods typically assume that a significant amount is known about the transition function, e.g., that the transitions are described by a differential equation or that an approximation to the transition

function is provided and the error in the approximation bounded. By contrast, the methods proposed in this dissertation assume that little is known about the transition function and do not attempt to model it.

This is both a benefit and a hindrance. It means that the methods we propose are applicable to a broader class of problems, including ones that include interactions with humans that would be challenging to model (e.g., the digital marketing example discussed later). However, if information about the transition function is known, then control theoretic approaches that leverage this knowledge may require less historical data to find good policies while enforcing various safety guarantees (Akametalu et al., 2014, Kretchmar et al., 2001, Perkins and Barto, 2003).

### 2.6.2 Constraints on Policy Space

Another form of safety that can be ensured is that the policy will never enter a dangerous region of *policy* space. Consider, for example, a control problem where the policy is represented as a *proportional-integral-derivative* (PID) controller with gains (policy parameters) $\theta$. PID controllers are the most widely used control algorithms in industry, and have been studied in depth (Åström and Hägglund, 1995, O'Dwyer, 2003). Techniques exist for determining the set of stable gains (non-dangerous policy parameters) when an approximate model of the system is available—see for example the work of Söylemez et al. (2003). An RL algorithm should search the space of policy parameters, $\theta$, for those that optimize $\rho(\pi_\theta)$ while ensuring that the policy parameters never enter the dangerous region of policy space—that the gains of the PID controller remain within the stable region. Although such policy search methods have been investigated (Bhatnagar et al., 2009, Thomas et al., 2013), like the control-theoretic approaches, they require an approximate model of the system in order to be applicable (to determine the safe region of policy space).

### 2.6.3 Unintended Consequences of Goal-Directed Behavior

In his 1964 book, Norbert Wiener gave early warnings regarding intelligent machines (Wiener, 1964, page 59):

> [...] if it grants you anything at all it grants you what you ask for, not what you should have asked for or what you intend.

and (Wiener, 1964, page 63):

> A goal-seeking mechanism will not necessarily seek *our* goals unless we design it for that purpose, and in that design we must foresee all steps of the process for which it is designed, [...]. The penalties for errors of foresight, great as they are now, will be enormously increased as automatization comes into its full use.

Nick Bostrom recently echoed these warnings that what seems like a reasonable goal for an agent can result in undesirable behavior that was not anticipated by the person or mechanism that originally specified the goal (Bostrom, 2014).

In the context of RL, the creator of the agent must select the definition of rewards so as to cause the agent to produce desirable behavior. However, as emphasized by Wiener, for complicated tasks it can be challenging to design a single scalar reward signal that will result in the desired behavior. Often the reward function is tweaked over several trials to find one that results in the desired behavior. Researchers have shown that often the most obvious reward function for a problem is not the most appropriate (Singh et al., 2009, Sorg et al., 2010).

This results in a safety concern—even if the intelligent machine correctly optimizes the specified objective function (in RL, the expected return), the designer of the objective function may not foresee undesirable ways that the objective function can be optimized until after the agent has produced undesirable behavior. Although this problem has been noticed, we are not aware of any existing literature that suggests a specific solution other than optimizing the reward function by trial and error (Niekum et al., 2010).

### 2.6.4 Probably Approximately Correct (PAC) Algorithms

This dissertation focuses on potential applications where the deployment of a single bad policy can be costly or dangerous, and so guarantees about the performance of each new policy are required before the policy can be used. Other research has focused on RL algorithms that have various guarantees about their convergence. Here we review one such effort that can be thought of as providing a safety guarantee.

An RL algorithm is called *probably approximately correct in Markov decision processes* (PAC-MDP, or PAC) if it guarantees that its expected return is within $\epsilon$ of optimal with probability $1 - \delta$ after a fixed number of time steps that is less than some polynomial function of $|\mathcal{S}|, |\mathcal{A}|, 1/\epsilon, 1/\delta$, and $1/(1-\gamma)$. The polynomial function that specifies the number of samples needed by the algorithm is called its *sample complexity* (Kakade, 2003). There are many PAC RL algorithms and lower bounds on the sample complexity of an arbitrary RL algorithm (Auer et al., 2010, Brunskill and Li, 2014, Guo and Brunskill, 2015, Strehl and Littman, 2005, 2008, Strehl et al., 2009, 2006, Szita and Szepesvári, 2010).

The primary drawback of PAC RL algorithms, which precludes their practical use for ensuring safety, is that the number of samples (time steps) required to ensure that the policy's expected return is within $\epsilon$ of optimal with probability $1 - \delta$ is usually prohibitively large. For example, a recent PAC RL algorithm requires approximately $10^{17}$ time steps to guarantee that the policy for a gridworld similar to the one we described in Section 2.5 is probably approximately correct (Lattimore and Hutter, 2012). By comparison, our methods will provide a high confidence improvement to the policy after approximately $10^3$ time steps.

# CHAPTER 3

# IMPORTANCE SAMPLING FOR OFF-POLICY EVALUATION

This chapter contains technical details that may distract the reader from the overarching messages regarding safe reinforcement learning. **If this is your first time reading this dissertation, then we recommend skipping much of this chapter:**

1. Read the first paragraph below this warning.
2. Skip to Section 3.2, which defines the problem that this chapter attempts to solve. Stop before reading Assumption 1.
3. Skip to Section 3.4. Read Sections 3.4 and 3.5, skipping over Subsection 3.5.2. The reader may then skip the remainder of this chapter. For subsequent chapters, the reader need only know that NPDIS and CWPDIS are alternatives to IS (presented in Section 3.5). NPDIS produces unbiased estimates of $\rho(\pi_e)$. CWPDIS produces estimates of $\rho(\pi_e)$ that tend to be more accurate, but which are not unbiased.

---

This chapter presents mechanisms for *off-policy evaluation*—estimating the performance of one policy, called the *evaluation policy*, using historical data (trajectories) from previous policies, called the *behavior policies*. Although this chapter does not provide confidence bounds for the estimates produced by its methods, it will serve as the foundation for the remainder of this dissertation.

We present six methods, *importance sampling* (IS), *per-decision importance sampling* (PDIS), *normalized per-decision importance sampling* (NPDIS), *weighted importance sampling* (WIS), *weighted per-decision importance sampling* (WPDIS), and *consistent weighted per-decision importance sampling* (CWPDIS). Of these, only NPDIS and CWPDIS are novel contributions. However, it is important that we can precisely specify the different properties of each of these approaches—particularly whether they are unbiased and/or consistent estimators.

It is well known that IS and PDIS are unbiased estimators and that IS, PDIS, and WIS are consistent estimators if using historical data from a single behavior policy (Powell and Swann, 1966, Precup, 2000, Rubinstein and Kroese, 2007). However, we are not aware of any previous proofs that they are consistent if, as we propose in later chapters, many behavior policies are used. In this chapter we therefore provide proofs that each of these approaches is consistent even if multiple behavior policies are used (these proofs require an additional technical assumption that is discussed later).

We also propose NPDIS, which is a variant of PDIS that will be better suited to our later uses. We show that it is unbiased and consistent. We then show that WPDIS is *not* a consistent estimator as claimed in the literature (Precup, 2000), and derive CWPDIS, which we show is a consistent alternative to WPDIS. Lastly, we show that even if a behavior policy is deterministic, IS and NPDIS will produce useful estimates of the performance of the evaluation policy.

The contributions of this chapter are:

1. Derivation of the NPDIS estimator, which is a variant of PDIS that is better suited to our later uses.

2. Proof that the WPDIS estimator is not a consistent estimator, despite claims otherwise.

3. Derivation of the CWPDIS estimator, an alternative to WPDIS that is consistent.

4. Proofs that IS, PDIS, NPDIS, WIS, and CWPDIS are consistent even if there are multiple behavior policies (as long as the probability of each action is either zero or bounded away from zero for every behavior policy).

5. Proofs that the expected values of the IS and NPDIS estimators are never larger than the true performance of the evaluation policy, regardless of the support of the evaluation and behavior policies.

## 3.1 Background

This section provides a review of what it means for a sequence to converge almost surely and what it means for estimators to be unbiased and consistent. It also provides two forms of the law of large numbers that can be used to show that an estimator is consistent.

### 3.1.1 Almost Sure Convergence

**Definition 1** (Almost Sure Convergence)**.** *A sequence of random variables,* $(X_i)_{i=1}^{\infty}$*, converges almost surely to the random variable* $X$ *if for all* $\epsilon > 0$

$$\Pr\left(\lim_{n\to\infty} X_n = X\right) = 1.$$

We write $X_i \xrightarrow{\text{a.s.}} X$ to denote that $(X_i)_{i=1}^{\infty}$ convergences almost surely to $X$. For brevity, we often say that $X_i$ (rather than $(X_i)_{i=1}^{\infty}$) converges almost surely to $X$. A few properties of almost sure convergence that we will use are:

**Property 1.** $X_i \xrightarrow{a.s.} X$ *implies that* $f(X_i) \xrightarrow{a.s.} f(X)$ *for every continuous function* $f$ *(Jiang, 2010, Section 2.8, Exercise 14).*

**Property 2.** *If* $X_i \xrightarrow{a.s.} X$ *and* $Y_i \xrightarrow{a.s.} Y$*, where* $X$ *and* $Y$ *are real numbers (not random variables), and* $Y \neq 0$*, then* $\frac{X_i}{Y_i} \xrightarrow{a.s.} \frac{X}{Y}$*.*

*Proof.*

$$\begin{aligned}\Pr\left(\lim_{n\to\infty}\frac{X_n}{Y_n}=\frac{X}{Y}\right) \geq& \Pr\left(\left(\lim_{n\to\infty}X_n=X\right),\left(\lim_{n\to\infty}Y_n=Y\right)\right)\\ \geq& 1-\Pr\left(\lim_{n\to\infty}X_n\neq X\right)-\Pr\left(\lim_{n\to\infty}Y_n\neq Y\right)\\ =& 1.\end{aligned}$$

■

**Property 3.** *If* $\{X_i^j\}_{j=1}^m$ *are* $m<\infty$ *sequences of random variables such that* $X_i^j \xrightarrow{a.s.} X^j$ *for all* $j \in \{1,\ldots,m\}$*, where* $X^j$ *is a random variable, then* $\sum_{j=1}^m X_i^j \xrightarrow{a.s.} \sum_{j=1}^m X^j$*.*

*Proof.*

$$\begin{aligned}\Pr\left(\lim_{n\to\infty}\sum_{j=1}^m X_n^j=\sum_{j=1}^m X^j\right) \geq& \Pr\left(\bigcap_{j=1}^m \lim_{n\to\infty}X_n^j=X^j\right)\\ \geq& 1-\sum_{j=1}^m\Pr\left(\lim_{n\to\infty}X_n^j\neq X^j\right)\\ =& 1.\end{aligned}$$

■

### 3.1.2 Unbiased and Consistent Estimators

In the subsequent sections we present different estimators of $\rho(\pi)$ and we discuss whether or not they are unbiased and/or consistent. In this section we review these two properties. The notation of this section is independent of the remainder of this dissertation in order to allow for the use of common notation when defining unbiased

and consistent estimators (e.g., here $\theta$ is a real number unrelated to the parameters of a policy).

**Definition 2.** *Let $\theta$ be a real number and $\hat{\theta}$ be a random variable. We call $\hat{\theta}$ an* **unbiased estimator** *of $\theta$ if and only if*

$$\mathbf{E}\left[\hat{\theta}\right] = \theta.$$

**Definition 3.** *Let $\theta$ be a real number and $(\hat{\theta}_n)_{n=1}^{\infty}$ be an infinite sequence of random variables. We call $\hat{\theta}_n$, a (strongly)* **consistent estimator** *of $\theta$ if and only if $\hat{\theta}_n \xrightarrow{a.s.} \theta$.*

To help understand the notation in Definition 3, consider a simple example. Let $\{X_i\}_{i=1}^{\infty}$ be identically distributed random variables and $\theta = \mathbf{E}[X_1]$ be their expected value. Then $\hat{\theta}_n \coloneqq \frac{1}{n}\sum_{i=1}^{n} X_i$ is an unbiased estimator of $\theta$ for any $n \geq 1$. Also, $\hat{\theta}_n$ is a consistent estimator of $\theta$. Notice that we specified the values of $n$ when saying that $\hat{\theta}_n$ is unbiased, but not when saying that $\hat{\theta}_n$ is consistent. This is because each single estimator is called *unbiased* while consistency is a property of the entire sequence of estimators. When we say that $\hat{\theta}_n$ is a consistent estimator, we do not specify $n$ and are implicitly talking about the entire sequence $(\hat{\theta}_n)_{n=1}^{\infty}$.

Notice that estimators can be any combination of biased/unbiased and consistent/inconsistent. Below we provide four specific examples:

- Unbiased and consistent: The sample mean of $n$ samples from a normal distribution is an unbiased and consistent estimator of the mean of the normal distribution.

- Unbiased and inconsistent: The first of $n$ samples from a normal distribution is an unbiased estimator of the mean of the normal distribution, however it is *not* a consistent estimator of the mean of the normal distribution. It is not consistent because it only uses the first sample, so as $n \to \infty$, it does not converge almost surely to the true mean.

- Biased and consistent: If $\{X_i\}_{i=1}^n$ are samples from a normal distribution, then $\frac{1}{n} + \frac{1}{n}\sum_{i=1}^n X_i$ is a biased but consistent estimator of the mean of the normal distribution. It is biased because, for any finite $n$, its expected value is $\theta + \frac{1}{n} \neq \theta$. It is consistent because as $n \to \infty$ the $\frac{1}{n}$ term goes to zero.

- Biased and inconsistent: If $\{X_i\}_{i=1}^n$ are samples from a normal distribution, then $3 + \frac{1}{n}\sum_{i=1}^n X_i$ is a biased and inconsistent estimator of $\theta$.

### 3.1.3 Laws of Large Numbers

We present two versions of the strong law of large numbers. The first, the *Khintchine strong law of large numbers*, applies to random variables that are identically distributed, but which may have infinite variance. The second, the *Kolmogorov strong law of large numbers*, applies to random variables that are not necessarily identically distributed, but which have finite variances. The forms that we present for these two laws are simplified special cases of the more general theorems provided in the references.

**Theorem 1** (Khintchine Strong Law of Large Numbers)**.** *Let $\{X_i\}_{i=1}^\infty$ be independent and identically distributed random variables. Then $(\frac{1}{n}\sum_{i=1}^n X_i)_{n=1}^\infty$ is a sequence of random variables that converges almost surely to $\mathbf{E}[X_1]$, i.e., $\frac{1}{n}\sum_{i=1}^n X_i \xrightarrow{a.s.} \mathbf{E}[X_1]$.*

*Proof.* See the work of Sen and Singer (1993, Theorem 2.3.13). ■

**Theorem 2** (Kolmogorov Strong Law of Large Numbers)**.** *Let $\{X_i\}_{i=1}^\infty$ be independent (not necessarily identically distributed) random variables. If all $X_i$ have the same mean and bounded variance (i.e., there is a finite constant $b$ such that for all $i \geq 1$, $\mathrm{Var}(X_i) \leq b$), then $(\frac{1}{n}\sum_{i=1}^n X_i)_{n=1}^\infty$ is a sequence of random variables that converges almost surely to $\mathbf{E}[X_1]$, i.e., $\frac{1}{n}\sum_{i=1}^n X_i \xrightarrow{a.s.} \mathbf{E}[X_1]$*

*Proof.* See the work of Sen and Singer (1993, Theorem 2.3.10 with Proposition 2.3.10). ■

These two laws can be used to show that different estimators are consistent. We present two corollaries that will assist in these proofs. The first, Corollary 1, applies when the random variables are identically distributed. The second, Corollary 2, applies when the random variables are not necessarily identically distributed, but have bounded variance.

**Corollary 1.** *Let $\{X_i\}_{i=1}^{\infty}$ be independent and identically distributed random variables with the same mean, $\mu$. Then $\frac{1}{n}\sum_{i=1}^{n} X_i$ is a consistent estimator of $\mu$.*

*Proof.* By the Khintchine strong law of large numbers we have that $\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{\text{a.s.}} \mu$, and thus that $\frac{1}{n}\sum_{i=1}^{n} X_i$ is a consistent estimator of $\mu$. ■

**Corollary 2.** *Let $\{X_i\}_{i=1}^{\infty}$ be independent random variables with the same mean, $\mu$, and bounded variance (i.e., there is a finite constant $b$ such that for all $i \geq 1$, $\text{Var}(X_i) \leq b$). Then $\frac{1}{n}\sum_{i=1}^{n} X_i$ is a consistent estimator of $\mu$.*

*Proof.* By the Kolmogorov strong law of large numbers we have that $\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{\text{a.s.}} \mu$, and thus that $\frac{1}{n}\sum_{i=1}^{n} X_i$ is a consistent estimator of $\mu$. ■

## 3.2 Problem Description

We assume that we are given a data set, $\mathcal{D}$, that consists of $n_{\mathcal{D}}$ trajectories, $\{H_L^i\}_{i=1}^{n_{\mathcal{D}}}$, each labeled by the policy that generated it, $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$, i.e.,

$$\mathcal{D} = \left\{ \left(H_L^i, \pi_i\right) : i \in \{1, \ldots, n_{\mathcal{D}}\}, H_L^i \sim \pi_i \right\}. \tag{3.1}$$

Note that $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$ are *behavior policies*—those that generated the batch of data (trajectories). Finally, we denote by $\pi_e$ the *evaluation policy*—the newly proposed policy that should be evaluated using the data set $\mathcal{D}$. Although some trajectories in $\mathcal{D}$ may have been generated using the evaluation policy, we are particularly interested in the setting where some or all of the behavior policies are different from the evaluation

policy. Our goal is to present a mechanism that takes as input a single trajectory, $H_L$, and the policy, $\pi_b$ that generated it (or a batch of trajectories and the policies that generated them, $\mathcal{D}$) and outputs an estimate of $\rho(\pi_e)$.

We will sometimes have equations that deal with multiple trajectories from $\mathcal{D}$. We use an index in the exponent of a symbol to denote the trajectory that it came from. For example, $S_t^i$ denotes the random variable for the state that occurs at time $t$ in the $i^{\text{th}}$ trajectory in $\mathcal{D}$.

Initially we will include the following assumption, which is discussed in detail in Section 3.11.

**Assumption 1.** *If $\pi_e(a|o) \neq 0$ then $\pi_i(a|o) \neq 0$ for all $i \in \{1, \ldots, n\}$, $a \in \mathcal{A}$, and $o \in \mathcal{O}$.*

**Corollary 3.** *For all $t \geq 1$, if Assumption 1 holds and $\Pr(H_t = h_t|\pi_e) = 0$, then for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$, $h_t$, and $t \geq 1$, we have that $\Pr(H_t = h_t|\pi_i) = 0$.*

*Proof.* Notice that

$$
\begin{aligned}
\Pr(H_t = h_t|\pi) &\overset{\text{(a)}}{=} \Pr(S_1 = s_1) \Pr(O_1 = o_1|S_1 = s_1) \Pr(A_1 = a_1|S_1 = s_1, O_1 = o_1) \\
&\quad \times \prod_{i=2}^{t} \Big( \Pr(R_{i-1} = r_{i-1}|H_{i-2} = h_{i-2}, S_{i-1} = s_{i-1}, O_{i-1} = o_{i-1}, A_{i-1} = a_{i-1}, S_i = s_i) \\
&\quad \times \Pr(S_i = s_i|H_{i-1} = h_{i-1}) \Pr(O_i = o_i|H_{i-1} = H_{i-1}, S_i = s_i) \\
&\quad \times \Pr(A_i = a_i|H_{i-1} = h_{i-1}, S_i = s_i, O_i = o_i) \Big) \\
&\overset{\text{(b)}}{=} \Pr(S_1 = s_1) \Pr(O_1 = o_1|S_1 = s_1) \Pr(A_1 = a_1|O_1 = o_1) \\
&\quad \times \prod_{i=2}^{t} \Big( \Pr(R_{i-1} = r_{i-1}|S_{i-1} = s_{i-1}, A_{i-1} = a_{i-1}, S_i = s_i) \\
&\quad \times \Pr(S_i = s_i|S_{i-1} = s_{i-1}, A_{i-1} = a_{i-1}) \Pr(O_i = o_i|S_i = s_i) \\
&\quad \times \Pr(A_i = a_i|O_i = o_i) \Big) \\
&\overset{\text{(c)}}{=} d_{S_1}(s_1)\Omega(o_1|s_1)\pi(a_1|o_1) \\
&\quad \times \prod_{i=2}^{t} \mathcal{R}(r_{i-1}|s_{i-1}, o_{i-1}, a_{i-1}) T(s_i|s_{i-1}, a_{i-1}) \Omega(o_i|s_i) \pi(a_i|o_i). \qquad (3.2)
\end{aligned}
$$

where **(a)** comes from repeated application of the rule that, for any random variables $X$ and $Y$, $\Pr(X=x, Y=y) = \Pr(X=x)\Pr(Y=y|X=x)$ **(b)** comes from the Markov assumptions in the definition of a POMDP, and **(c)** comes from the definitions of $d_{S_1}, \Omega, \pi, R$ and $T$.

If $\Pr(H_t = h_t|\pi_e) = 0$, then one of the terms in the product above (using $\pi_e$ for $\pi$) must be zero, which means that one of the terms in the expansion of $\Pr(H_t = h_t|\pi_i)$ must also be zero. ■

## 3.3 Lemmas and Corollaries

In this section we collect lemmas and corollaries that are used repeatedly throughout this chapter. First, we show that $\Pr(H_L=h_L|\pi_e)/\Pr(H_L=h_L|\pi_b)$ can be written in a way that does not depend on the unknown parameters of the POMDP like the transition function. Although this has been shown before for the MDP setting (Precup, 2000), we prove it again for completeness.

**Lemma 1.** *Let $\pi_e$ and $\pi_b$ be any two policies and $t \geq 1$. Let $h_t$ be any history of length $t$ that has non-zero probability under $\pi_b$, i.e., $\Pr(H_t=h_t|\pi_b) \neq 0$. Then*

$$\frac{\Pr(H_t=h_t|\pi_e)}{\Pr(H_t=h_t|\pi_b)} = \prod_{i=1}^{t-1} \frac{\pi_e(a_i|o_i)}{\pi_e(a_i|o_i)}.$$

*Proof.* By expanding the probability of a history given a policy, we have from (3.2) that

$$\begin{aligned}\frac{\Pr(H_t=h_t|\pi_e)}{\Pr(H_t=h_t|\pi_b)} =& \frac{d_{S_1}(s_1)\Omega(o_1|s_1)\pi_e(a_1|o_1)}{d_{S_1}(s_1)\Omega(o_1|s_1)\pi_b(a_1|o_1)} \\ &\times \frac{\prod_{i=1}^{t}\mathcal{R}(r_{i-1}|s_{i-1},o_{i-1},a_{i-1})T(s_i|s_{i-1},a_{i-1})\Omega(o_i|s_i)\pi_e(a_i|o_i)}{\prod_{i=1}^{t}\mathcal{R}(r_{i-1}|s_{i-1},o_{i-1},a_{i-1})T(s_i|s_{i-1},a_{i-1})\Omega(o_i|s_i)\pi_b(a_i|o_i)} \\ =& \prod_{i=1}^{t}\frac{\pi_e(a_i|o_i)}{\pi_b(a_i|o_i)}.\end{aligned}$$

■

Next we introduce a lemma to show that some expected values that are conditioned on trajectories coming from a policy can be expressed by conditioning on only the beginning of the trajectory. Recall that $\xi_t$ is defined in (2.1).

**Lemma 2.** *Let $f : \mathcal{H}_t \to \mathbb{R}$. Then*

$$\mathbf{E}[f(\Xi_1, \ldots, \Xi_t)|H_L \sim \pi] = \mathbf{E}[f(\Xi_1, \ldots, \Xi_t)|H_t \sim \pi].$$

*Proof.*

$$\begin{aligned}
\mathbf{E}[f(\Xi_1, \ldots, \Xi_t)|H_L \sim \pi] &= \sum_{H_L} \Pr(H_L = h_L|\pi) f(\xi_1, \ldots, \xi_t) \\
&\overset{\textbf{(a)}}{=} \sum_{\xi_1, \ldots \xi_L} \Pr(\Xi_1 = \xi_1, \ldots, \Xi_L = \xi_L|\pi) f(\xi_1, \ldots, \xi_t) \\
&\overset{\textbf{(b)}}{=} \sum_{\xi_1, \ldots, \xi_L} \Pr(\Xi_1 = \xi_1, \ldots, \Xi_t = \xi_t|\pi) \times \\
&\quad \Pr(\Xi_{t+1} = \xi_{t+1}, \ldots, \Xi_L = \xi_L|\pi, \xi_1, \ldots, \xi_t) f(\xi_1, \ldots, \xi_t) \\
&\overset{\textbf{(c)}}{=} \sum_{\xi_1, \ldots, \xi_t} \Pr(\Xi_1 = \xi_1, \ldots, \Xi_t = \xi_t|\pi) f(\xi_1, \ldots, \xi_t) \\
&\quad \times \underbrace{\sum_{\xi_{t+1}, \ldots, \xi_L} \Pr(\Xi_{t+1} = \xi_{t+1}, \ldots, \Xi_L = \xi_L|\xi_1, \ldots, \xi_t)}_{=1} \\
&= \sum_{\xi_1, \ldots, \xi_t} \Pr(\underbrace{\Xi_1 = \xi_1, \ldots, \Xi_t = \xi_t}_{\iff H_t = h_t}|\pi) f(\xi_1, \ldots, \xi_t) \\
&= \sum_{h_t} \Pr(H_t = h_t|\pi) f(\xi_1, \ldots, \xi_t) \\
&= \mathbf{E}[f(\Xi_1, \ldots, \Xi_t)|H_t \sim \pi],
\end{aligned}$$

where **(a)** comes from the definition of $H_L$, **(b)** holds because for any random variables $X$ and $Y$, $\Pr(X = x, Y = y) = \Pr(X = x) \Pr(Y = y|X = x)$, and **(c)** comes from reordering terms. ■

The next lemma establishes that the expected likelihood ratio that will be used in importance sampling is one.

**Lemma 3.** *Let $\pi_e$ and $\pi_b$ be any policies such that Assumption 1 holds. Then for any constant integer $k \geq 1$,*

$$\mathbf{E}\left[\prod_{t=1}^{k}\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)}\middle| H_L \sim \pi_b\right] = 1.$$

*Proof.*

$$\begin{aligned}
\mathbf{E}\left[\prod_{t=1}^{k}\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)}\middle| H_L \sim \pi_b\right] &\overset{\textbf{(a)}}{=} \mathbf{E}\left[\prod_{t=1}^{k}\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)}\middle| H_k \sim \pi_b\right] \\
&\overset{\textbf{(b)}}{=} \mathbf{E}\left[\frac{\Pr\left(H_k = h_k|\pi_e\right)}{\Pr\left(H_k = h_k|\pi_b\right)}\middle| H_k \sim \pi_b\right] \\
&= \sum_{\text{supp}_k\, \pi_b} \Pr\left(H_k = h_k|\pi_b\right)\frac{\Pr\left(H_k = h_k|\pi_e\right)}{\Pr\left(H_k = h_k|\pi_b\right)} \\
&= \sum_{\text{supp}_k\, \pi_b} \Pr\left(H_k = h_k|\pi_e\right) \qquad (3.3) \\
&\overset{\textbf{(c)}}{=} \sum_{\text{supp}_k\, \pi_e} \Pr\left(H_k = h_k|\pi_e\right) \\
&= 1,
\end{aligned}$$

where **(a)** comes from Lemma 2, **(b)** comes from Corollary 1, and **(c)** comes from Corollary 3, which relies on Assumption 1. ■

Similarly, if Assumption 1 does not necessarily hold, then the expected likelihood ratio is no more than one:

**Lemma 4.** *Let $\pi_e$ and $\pi_b$ be any policies. Then for any constant integer $k \geq 1$,*

$$\mathbf{E}\left[\prod_{t=1}^{k}\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)}\middle| H_L \sim \pi_b\right] \leq 1.$$

*Proof.*

$$\begin{aligned}\mathbf{E}\left[\prod_{t=1}^{k}\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)}\middle|H_L\sim\pi_b\right] &\overset{\textbf{(a)}}{=}\sum_{\text{supp}_k\,\pi_b}\Pr\left(H_k=h_k|\pi_e\right)\\ &\overset{\textbf{(b)}}{\leq}\sum_{\text{supp}_k\,\pi_e}\Pr\left(H_k=h_k|\pi_e\right)\\ &=1,\end{aligned}$$

where **(a)** comes from (3.3) and **(b)** holds because probabilities are nonnegative. ■

## 3.4 Overview of Importance Sampling Approaches

The following sections present several different forms of importance sampling. Each can be used to take a trajectory, $H_L$, generated by the behavior policy, $\pi_b$, or a set of trajectories and behavior policies, $\mathcal{D}$, and will produce an estimate of the performance, $\rho(\pi_e)$, of the evaluation policy, $\pi_e$. These estimates are denoted by $\hat{\rho}^{\dagger}(\pi_e|H_L,\pi_b)$ for a single trajectory and $\dagger(\pi_e|\mathcal{D})$ for a set of trajectories, where $\dagger$ is replaced by an abbreviation for the name of the specific estimator used, i.e.,

$$\dagger \in \{\text{IS, PDIS, NPDIS, WIS, WPDIS, CWPDIS}\}.$$

The computation of these estimates will be the first step in our proposed approach to high confidence off-policy evaluation in the next chapter, and the various properties of the different estimators will be crucial to understanding when they are and are not applicable.

Some of the methods in later chapters assume that upper and lower bounds on $\hat{\rho}^{\dagger}(\pi_e|H_L,\pi_b)$ can be computed given $\pi_e$ and $\pi_b$. The upper bound is denoted by $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e,\pi_b)$, and the lower bound (when not zero) is $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e,\pi_b)$. The bounds, $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e,\pi_b)$ and $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e,\pi_b)$, should be the least upper bound and greatest lower bound that are known for $\hat{\rho}^{\dagger}(\pi_e|H_L,\pi_b)$.

Table 3.1 presents a summary of the relevant information about each importance sampling method that we discuss. The meanings of the columns are:

1. **Name**: Specifies the name of the estimator (the value of †).

2. **Eq.**: Gives the equation where the estimator, $\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)$, is presented. Some methods are only meant to be applied to batches of trajectories, and so they have the value "N/A" listed for this column.

3. **Batch Eq.**: Gives the equation where the batch version of the estimator is presented. The batch version can be applied to a set of trajectories rather than just a single one.

4. **Unbiased**: Specifies whether the estimator is an unbiased estimator of $\rho(\pi_e)$, i.e., $\mathbf{E}\left[\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)\middle|H_L \sim \pi_b\right] = \rho(\pi_e)$.

5. **Consistent**: Specifies whether the batch estimator, $\dagger(\pi_e|\mathcal{D})$, is a consistent estimator of $\rho(\pi_e)$. See the relevant sections for descriptions of the requirements to ensure consistency.

6. $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e, \pi_b)$: Either the equation where the lower bound is specified, or a constant value if the bound is independent of $\pi_e$ and $\pi_b$.

7. $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e, \pi_b)$: Either the equation where the upper bound is specified, or a constant value if the bound is independent of $\pi_e$ and $\pi_b$.

| Name (†) | Eq. | Batch Eq. | Unbiased | Consistent | $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e, \pi_b)$ | $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e, \pi_b)$ |
|---|---|---|---|---|---|---|
| IS | (3.6) | (3.7) | Yes | Yes | 0 | (3.8) |
| PDIS | (3.11) | (3.12) | Yes | Yes | (3.15) | (3.16) |
| NPDIS | (3.17) | (3.18) | Yes | Yes | 0 | (3.19) |
| WIS | N/A | (3.21) | No | Yes | 0 | 1 |
| WPDIS | N/A | (3.25) | No | No | N/A | N/A |
| CWPDIS | N/A | (3.26) | No | Yes | 0 | 1 |

**Table 3.1.** Summary of properties of the importance sampling methods that we discuss.

## 3.5 Importance Sampling (IS)

If the historical data contains only *on-policy* trajectories—if each trajectory, $H_L^i$, in $\mathcal{D}$ was generated by $\pi_i = \pi_e$, then $G(H_L^i)$ is an unbiased estimator of $\rho(\pi_e)$ for each $i \in \{1, \ldots, n_{\mathcal{D}}\}$. However, the historical data usually contains at least some *off-policy* trajectories, which means that $\pi_i \neq \pi_e$. So, the trajectories that we collected came from a different distribution than they would if they were produced by $\pi_e$. We require a method for correcting for this difference in sampling and evaluation distributions when estimating the expected value of a function (in this case $G(H_L)$).

To visualize this problem, consider Figure 3.1, which depicts the distribution of returns under an evaluation and behavior policy. In the gridworld problem, the behavior policy randomly selects actions with equal probability regardless of which state it is in. This results in an expected return of approximately $-72.377$, which means that (after normalizing the expected return) $\rho(\pi_b) \approx 0.836$. The evaluation policy is a significantly better policy, although it is still far from optimal (it learns to reach the goal while avoiding the large penalty, but it does not remain in the position $(2, 4)$ very long. Its expected return is approximately $-2.219$, which means that $\rho(\pi_e) \approx 0.900$.

One method for estimating the expected value of a function when samples come from a distribution that is different from the desired distribution is *importance sampling* (IS). Let $p$ and $q$ be two distributions (probability mass functions) over some

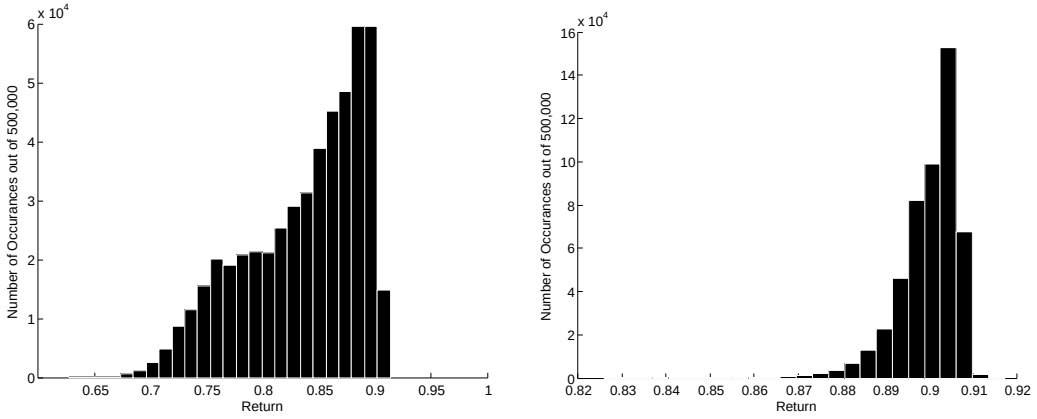


**Figure 3.1.** Empirical estimate of the return distributions of the behavior (left) and evaluation (right) policies for the gridworld. These estimates were formed by sampling 500,000 on-policy trajectories and plotting a histogram of their returns. The problem that we are faced with is to estimate the expected value of the random variable on the right given samples of the random variable on the left.

set, $\mathcal{X}$, and let $f : \mathcal{X} \to \mathbb{R}$. Assume that if $q(x) = 0$ then $f(x)p(x) = 0$ (or, more restrictively, assume that $\operatorname{supp} p \subseteq \operatorname{supp} q$). We wish to estimate $\mathbf{E}[f(X)|X \sim p]$, however, we can only sample $x$ from the *sampling distribution*, $q$.

By writing out the definition of expected value and multiplying by $1 = \frac{q(x)}{q(x)}$, it is straightforward to derive the IS estimator:

$$\begin{aligned}
\mathbf{E}[f(X)|X \sim p] &= \sum_{x \in \operatorname{supp} p} p(x) f(x) \\
&\overset{\textbf{(a)}}{=} \sum_{x \in \operatorname{supp} q} p(x) \frac{q(x)}{q(x)} f(x) \\
&= \sum_{x \in \operatorname{supp} q} q(x) \frac{p(x)}{q(x)} f(x) \\
&= \mathbf{E}\left[ \frac{p(X)}{q(X)} f(X) \middle| X \sim q \right], \qquad (3.4)
\end{aligned}$$

where **(a)** holds because we assumed that if $q(x) = 0$ then $f(x)p(x) = 0$.

So, if $X \sim q$, then the estimator $\frac{p(X)}{f(X)} f(X)$ is an unbiased estimator of $\mathbf{E}[f(X)|X \sim p]$. The ratio, $\frac{p(X)}{q(X)}$, is called the *importance weight* or *likelihood ratio*. Intuitively, if

a specific $x$ is more likely under the sampling distribution, $q$, than under the target distribution, $p$, then sampling $q$ will generate that $x$ too many times, and so it should be given a smaller weight to simulate it occurring less frequently. Similarly, if a specific $x$ is less likely under the sampling distribution, $q$, than the target distribution, $p$, then sampling $q$ will generate that $x$ too few times, and so it should be given a larger weight when it does occur to simulate it occurring more frequently. This reweighting is exactly what the importance weight does.

In the context of RL, $\mathcal{X}$ will be the set of possible trajectories, $\mathcal{H}_L$, $p$ is the distribution over trajectories when using the evaluation policy, $\pi_e$, $q$ is the distribution over trajectories when using some other behavior policy, $\pi_b$, and $f(X)$ is $G(H_L)$. The assumption that $f(x)p(x) = 0$ if $q(x) = 0$ is ensured by Assumption 1.

So, if $H_L \sim \pi_b$, Assumption 1 holds, and

$$\hat{\rho}^{\text{IS}}(\pi_e|h_L, \pi_b) \coloneqq \underbrace{\frac{\Pr(H_L = h_L|\pi_e)}{\Pr(H_L = h_L|\pi_b)}}_{\text{importance weight}} \underbrace{G(H_L)}_{\text{return}}, \tag{3.5}$$

then $\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ and is called the *importance weighted return*. Using Lemma 1, we can simplify (3.5) to get an alternate definition of the importance weighted return:

$$\begin{aligned}\hat{\rho}^{\text{IS}}(\pi_e|h_L, \pi_b) &\coloneqq \frac{\Pr(H_L = h_L|\pi_e)}{\Pr(H_L = h_L|\pi_b)} G(H_L) \\ &= \underbrace{\prod_{t=1}^{L} \frac{\pi_e(a_t|o_t)}{\pi_b(a_t|o_t)}}_{\text{importance weight}} \underbrace{G(H_L)}_{\text{return}}.\end{aligned} \tag{3.6}$$

Importance sampling has been well known for a long time, and its use for RL in this form is also not new (Precup, 2000).

Recall that $\mathcal{D}$ is a set of $n_{\mathcal{D}}$ trajectories and the policies that produced them, as defined in (5.1). We define the IS estimator for all of $\mathcal{D}$ (as opposed to a single trajectory) to be the mean of the individual IS estimators for each trajectory:

$$\text{IS}(\pi_e|\mathcal{D}) \coloneqq \frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i). \tag{3.7}$$

Since each $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i)$ is an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds, $\text{IS}(\pi_e|\mathcal{D})$ is also an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds:

$$\begin{aligned}
\mathbf{E}[\text{IS}(\pi_e|\mathcal{D})|H_L^i \sim \pi_i, \forall i \in \{1, \ldots, n_{\mathcal{D}}\}] =& \mathbf{E}\left[\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i) \middle| H_L^i \sim \pi_i, \forall i \in \{1, \ldots, n_{\mathcal{D}}\}\right] \\
=& \frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \mathbf{E}\left[\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i) \middle| H_L^i \sim \pi_i\right] \\
=& \frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \rho(\pi_e) \\
=& \rho(\pi_e).
\end{aligned}$$

### 3.5.1 Upper and Lower Bounds on the IS Estimator

Later it will be important that we can provide upper and lower bounds on the IS estimator, $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i)$. The importance weights are always nonnegative, i.e., if $H_L \sim \pi_b$ then

$$\prod_{t=1}^{L} \frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)} \geq 0,$$

for all policies $\pi_e$ and $\pi_b$. The returns, $G(H_L)$ are also always nonnegative since we assumed that they are normalized to $G(H_L) \in [0, 1]$. So, we have a trivial lower bound, that if $H_L \sim \pi_b$ then

$$\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b) = \prod_{t=1}^{L} \frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)} G(H_L) \geq 0,$$

for all policies $\pi_e$ and $\pi_b$.

Without domain specific knowledge, we usually cannot produce tight upper bounds on $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i)$. However, we can always compute an (often loose) upper bound by assuming that the worst possible ratio of action probabilities occurs at every time step. That is, if $H_L \sim \pi_b$ then

$$\hat{\rho}^{\text{IS}}_{\text{ub}}(\pi_e, \pi_b) \coloneqq \left(\max_{a,o} \frac{\pi_e(a|o)}{\pi_b(a|o)}\right)^L \geq \hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b), \tag{3.8}$$

for all $\pi_e, \pi_b$, and $H_L$. In all of the experiments in this dissertation, we solve (approximately) for this global maximum using the black-box optimization algorithm *covariance matrix adaptation evolution strategy* (CMA-ES) (Hansen, 2006).[1]

### 3.5.2 Consistency of IS Estimator

Our derivation of the IS estimator made it clear that if Assumption 1 holds, then $\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ if $H_L \sim \pi_b$ and $\text{IS}(\pi_e|\mathcal{D})$ is an unbiased estimator of $\rho(\pi_e)$ regardless of how many trajectories are in $\mathcal{D}$. Here we present two different sets of assumptions that ensure that the batch IS estimator is consistent.

First we show that the batch IS estimator is consistent if there is only one behavior policy:

**Theorem 3.** *If Assumption 1 holds and there is only one behavior policy, i.e., $\pi_i = \pi_b$ for all $i$, then IS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.*

*Proof.* This follows from Corollary 1 since **(1)** $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ due to Assumption 1 and **(2)** the random variables $\{\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_b)\}_{i=1}^{n_{\mathcal{D}}}$ are identically distributed since the trajectories are all generated by the same behavior policy. ■

[1] CMA-ES only produces estimates of the globally optimal value. This can introduce some error into our analyses, however, as we will see later, this error can only bias our results in favor of the existing methods that we compete with.

Next we show that the IS estimator is consistent if there are many behavior policies, all of which are bounded away from being deterministic when the evaluation policy is not deterministic.

**Theorem 4.** *If Assumption 1 holds and there exists a constant* $\epsilon > 0$ *such that* $\pi_i(a|o) \geq \epsilon$ *for all* $i \in \{1, \ldots, n_{\mathcal{D}}\}$ *and* $(a, o)$ *where* $\pi_e(a|o) \neq 0$*, then IS*$(\pi_e|\mathcal{D})$ *is a consistent estimator of* $\rho(\pi_e)$ *even if there are multiple behavior policies.*

*Proof.* This follows from Corollary 2 since **(1)** $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ due to Assumption 1 and **(2)** $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_b) \in \left[0, \frac{1}{\epsilon^L}\right]$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$, and so $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_b)$ has bounded variance. ■

### 3.5.3 Example: Gridworld

Perhaps the biggest challenge that we must overcome in the following chapter arises due to the large possible range of the IS estimator and its high variance. To make this clear, consider the application of IS to the behavior and evaluation policies from the gridworld. We will use the exact same trajectories that were used to produce the left plot in Figure 3.1 (the empirical distribution of returns under the behavior policy) to produce an estimate of the performance of the evaluation policy. That is, we compute $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i)$, for each of the 500,000 trajectories ($i \in \{1, \ldots, 500{,}000\}$) that were generated by the behavior policy. The resulting distribution of the 500,000 estimators, $\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_i)$, is depicted in Figure 3.2.

The mean importance weighted return (i.e., IS$(\pi_e|\mathcal{D})$) was 0.900, which is equal to the true expected performance, $\rho(\pi_e)$, to all three decimal places. The largest observed importance weighted return was 68.261 and the smallest was approximately 0. The first column contains 357,160 samples (it goes off the top of the plot), which means that approximately 70% of the trajectories resulted in an importance weighted return that was approximately zero. In practice this means that the IS estimator is often an underestimate when using only a few trajectories. This is not at odds

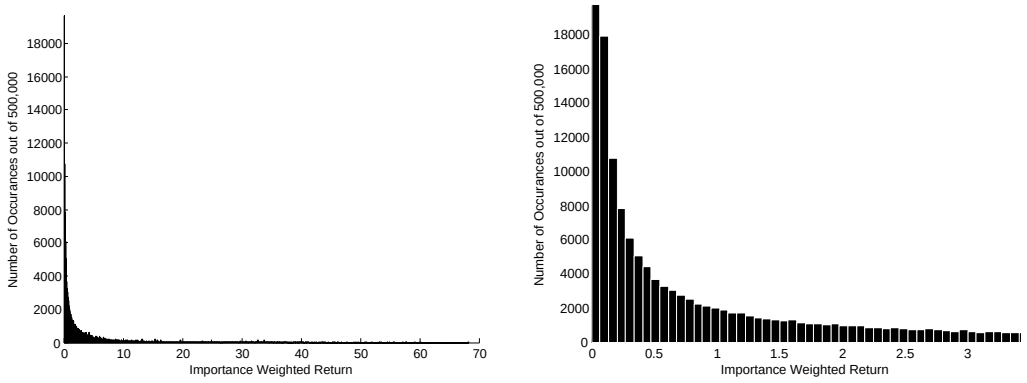


**Figure 3.2.** Empirical distribution of importance weighted returns when evaluating the evaluation policy for the gridworld using 500,000 trajectories from the behavior policy. The plot on the right is zoomed in to provide more detail around the region containing most of the probability mass. The first column is cut off by the limit of the vertical axis—it dwarfs the others with a value of 357,160.

with it being an unbiased estimator. Consider what happens when using only a single trajectory. From Figure 3.2, 70% of the time the importance weighted return is approximately zero, and so at least 70% of the time it is an underestimate of the true performance ($\rho(\pi_e) \approx 0.9$). However, when it is an overestimate (which is rare), it is often a significant over-estimate—in one case it was 68.261. This averages out so that the expected value is the true performance.

The upper bound on $\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b)$ is $\hat{\rho}^{\text{IS}}_{\text{ub}}(\pi_e, \pi_b) = 8.889 \times 10^{59}$. To see where this upper bound comes from, notice that the behavior policy draws actions from a uniform distribution, so the denominator of $\frac{\pi_e(a|o)}{\pi_b(a|o)}$ is always 0.25 (there are four possible actions). The evaluation policy has some observations for which it almost deterministically selects a specific action (e.g., when adjacent to the terminal state, $(4, 4)$, it may know to move to it). This means that there are some observations where the numerator is near one. Our upper bound assumes that the largest possible ratio, in this case $1/0.25$, always occurs for all $L = 100$ time steps. So, the upper bound would be approximately $4^{100} \approx 1.6 \times 10^{60}$. The upper bound that we computed was

slightly lower than this, which means that the evaluation policy was never perfectly deterministic about its actions and so the numerator was slightly smaller than one.

This massive upper bound will cause our approach to high confidence off-policy evaluation, proposed in the next chapter, to fail when using most existing concentration inequalities. One approach to improving performance would be to try to find a tighter upper bound on the importance weighted returns. However, it may not be possible to improve much upon the upper bound that we proposed. Consider what would happen if the evaluation policy quickly moved the agent to the position $(2, 4)$ and then deterministically chose the "down" action until time ran out. This would result in an expected return of approximately 94 (less, depending on how long it takes the agent to reach $(2, 4)$), which is nearly optimal, so $G(H_L)$ would be close to one. Also, the trajectory most often produced by the evaluation policy could occur under the random behavior policy. If it does, the importance weighted return will be massive since approximately 96 time steps will have the action "down" chosen in the position $(2, 4)$, which results in an importance weight of approximately $4^{96}$ for those transitions (this will be multiplied by the importance weight from the time steps during which the agent moved to $(2, 4)$, which might make the importance weight larger still).

## 3.6 Per-Decision Importance Sampling

*Per-decision importance sampling* (PDIS) is an approach specific to sequential systems like ours. It often has lower variance than the IS estimator, but is still an unbiased estimator of $\rho(\pi_e)$. PDIS is only applicable for certain definitions of $G(H_L)$, which includes all affine functions of the rewards like the normalized discounted return defined in (2.5). Intuitively, the PDIS estimator uses a different importance weight for each reward rather than one importance weight for the entire return.

The PDIS estimator can be derived as follows if $G(H_L)$ is the normalized discounted return defined in (2.5):

$$\begin{aligned}\rho(\pi_e) =& \frac{\mathbf{E}\left[\gamma^0 R_1 + \gamma^1 R_2 + \ldots + \gamma^{L-1} R_L \middle| H_L \sim \pi_e\right] - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1} R_t | H_L \sim \pi_e\right]\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}.\end{aligned} \tag{3.9}$$

By Lemma 2 we have that $\mathbf{E}[\gamma^{t-1} R_t | H_L \sim \pi_e] = \mathbf{E}[\gamma^{t-1} R_t | H_t \sim \pi_e]$. This means that each term only depends on the history up until the relevant reward. So, continuing Equation (3.9), we can use ordinary importance sampling to estimate each expected value:

$$\begin{aligned}\rho(\pi_e) =& \frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1} R_t | H_t \sim \pi_e\right]\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \frac{\left(\sum_{t=1}^{L} \sum_{\text{supp}_t \pi_e} \Pr(H_t = h_t | \pi_e) \gamma^{t-1} r_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ \overset{\textbf{(a)}}{=}& \frac{\left(\sum_{t=1}^{L} \sum_{\text{supp}_t \pi_b} \Pr(H_t = h_t | \pi_e) \gamma^{t-1} r_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \frac{\left(\sum_{t=1}^{L} \sum_{\text{supp}_t \pi_b} \frac{\Pr(H_t = h_t | \pi_b)}{\Pr(H_t = h_t | \pi_b)} \Pr(H_t = h_t | \pi_e) \gamma^{t-1} r_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \frac{\left(\sum_{t=1}^{L} \sum_{\text{supp}_t \pi_b} \frac{\Pr(H_t = h_t | \pi_e)}{\Pr(H_t = h_t | \pi_b)} \Pr(H_t = h_t | \pi_b) \gamma^{t-1} r_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ \overset{\textbf{(b)}}{=}& \frac{\left(\sum_{t=1}^{L} \sum_{\text{supp}_t \pi_b} \left(\prod_{i=1}^{t} \frac{\pi_e(a_i|o_i)}{\pi_b(a_i|o_i)}\right) \Pr(H_t = h_t | \pi_b) \gamma^{t-1} r_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1} R_t \prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)} \middle| H_t \sim \pi_b\right]\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\ =& \mathbf{E}\left[\frac{\left(\sum_{t=1}^{L} \gamma^{t-1} R_t \prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \middle| H_t \sim \pi_b\right],\end{aligned} \tag{3.10}$$

where **(a)** holds by Assumption 1 due to Corollary 3 and **(b)** comes from Lemma 1. We define the PDIS estimator to be:

$$\hat{\rho}^{\text{PDIS}}(\pi_e|H_L,\pi_b) \coloneqq \frac{\left(\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right)-G_{\text{lb}}}{G_{\text{ub}}-G_{\text{lb}}}. \tag{3.11}$$

From its derivation, it is clear that $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L,\pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds and $H_L \sim \pi_b$. Notice also that PDIS is equivalent to IS if the rewards are all zero except for the final reward. PDIS was first derived for the MDP setting by Precup (2000). Here we have extended their result to the full POMDP setting. Also notice that, while the IS estimator, $\hat{\rho}^{\text{IS}}(\pi_e|H_L,\pi_b)$, is never negative, the PDIS estimator, $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L,\pi_b)$, can be negative if any of the individual rewards are negative.

As with the IS estimator, if we have a set, $\mathcal{D}$, of trajectories and the policies that generated them, we define the PDIS estimator to be the mean of the individual PDIS estimators for each trajectory. So, by substituting in the definition of $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i,\pi_i)$ and reordering terms we have that:[2]

$$\begin{aligned}\text{PDIS}(\pi_e|\mathcal{D}) &\coloneqq \frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i,\pi_i)\\ &= \left(\frac{1}{G_{\text{ub}}-G_{\text{lb}}}\right)\left(\left(\sum_{t=1}^{L}\gamma^{t-1}\underbrace{\frac{\sum_{i=1}^{n_{\mathcal{D}}}R_t\prod_{j=1}^{t}\frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}{n_{\mathcal{D}}}}_{\text{IS estimate of }\mathbf{E}[R_t|H_t\sim\pi_e]}\right)-G_{\text{lb}}\right).\end{aligned} \tag{3.12}$$

From (3.12) it clear what PDIS is doing—for each time, $t$, it uses ordinary importance sampling to estimate the expected discounted *reward* at time $t$. Lastly, notice that the batch PDIS estimator is an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds:

[2]Recall that $A_j^i$ denotes the $j^{\text{th}}$ action in the $i^{\text{th}}$ trajectory in $\mathcal{D}$.

$$
\begin{aligned}
\mathbf{E}\left[\mathrm{PDIS}(\pi_e|\mathcal{D})\right] =& \mathbf{E}\left[\frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\hat{\rho}^{\mathrm{PDIS}}(\pi_e|H_L^i,\pi_i)\right] \\
=& \frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\mathbf{E}\left[\hat{\rho}^{\mathrm{PDIS}}(\pi_e|H_L^i,\pi_i)\right] \\
=& \frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\rho(\pi_e) \\
=& \rho(\pi_e). \qquad (3.13)
\end{aligned}
$$

### 3.6.1 Upper and Lower Bounds on the PDIS Estimator

To bound the PDIS estimator we will assume that the worst possible likelihood ratio occurs at every time step. Let

$$
\varsigma \coloneqq \max_{a,o}\frac{\pi_e(a|o)}{\pi_b(a|o)}, \qquad (3.14)
$$

We can then bound the PDIS estimator by

$$
\hat{\rho}^{\mathrm{PDIS}}_{\mathrm{ub}}(\pi_e,\pi_b) \coloneqq \begin{cases} \frac{\left(\sum_{t=1}^{L}\gamma^{t-1}r_{\mathrm{ub}}\varsigma^t\right)-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}} & \text{if } r_{\mathrm{ub}} \geq 0 \\ \frac{-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}} & \text{otherwise,} \end{cases} \qquad (3.15)
$$

and

$$
\hat{\rho}^{\mathrm{PDIS}}_{\mathrm{lb}}(\pi_e,\pi_b) \coloneqq \begin{cases} \frac{\left(\sum_{t=1}^{L}\gamma^{t-1}r_{\mathrm{lb}}\varsigma^t\right)-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}} & \text{if } r_{\mathrm{lb}} \leq 0 \\ \frac{-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}} & \text{otherwise.} \end{cases} \qquad (3.16)
$$

### 3.6.2 Consistency of PDIS Estimator

Our derivation of the PDIS estimator made it clear that $\hat{\rho}^{\mathrm{PDIS}}(\pi_e|H_L,\pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ if $H_L \sim \pi_b$ and Assumption 1 holds, and that $\mathrm{PDIS}(\pi_e|\mathcal{D})$ is an unbiased estimator of $\rho(\pi_e)$ regardless of how many trajectories are in $\mathcal{D}$ if Assumption 1 holds. Here we present two different sets of assumptions that ensure that the batch PDIS estimator is consistent.

First we show that the batch PDIS estimator is consistent if there is only one behavior policy:

**Theorem 5.** *If Assumption 1 holds and there is only one behavior policy, i.e., $\pi_i = \pi_b$ for all $i$, then PDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.*

*Proof.* This follows from Corollary 1 since **(1)** $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and **(2)** the random variables $\{\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_b)\}_{i=1}^{n_{\mathcal{D}}}$ are identically distributed since the trajectories are all generated by the same behavior policy. ■

Next we show that the PDIS estimator is consistent if there are many behavior policies, all of which are bounded away from being deterministic when the evaluation policy is not deterministic.

**Theorem 6.** *If Assumption 1 holds and there exists a constant $\epsilon > 0$ such that $\pi_i(a|o) \geq \epsilon$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and $(a, o)$ where $\pi_e(a|o) \neq 0$, then PDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$ even if there are multiple behavior policies.*

*Proof.* This follows from Corollary 2 since **(1)** $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and **(2)**

$$\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_b) \in \left[\frac{\min\{\frac{1}{\epsilon^L} r_{\text{lb}}, 0\} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}, \frac{\max\{0, \frac{1}{\epsilon^L} r_{\text{ub}}\} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}\right],$$

and so $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_b)$ has bounded variance. ■

### 3.6.3 Example: Gridworld

Consider the application of PDIS to the behavior and evaluation policies from the gridworld. We will use the exact same trajectories that were used to produce the left plot in Figure 3.1 (the empirical distribution of returns under the behavior policy) to produce an estimate of the performance of the evaluation policy. That is, we

compute $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L^i, \pi_i)$, for each of the 500,000 trajectories ($i \in \{1, \ldots, 500{,}000\}$) that were generated by the behavior policy. The resulting distribution of the 500,000 estimators, $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L, \pi_i)$, is depicted in Figure 3.4.

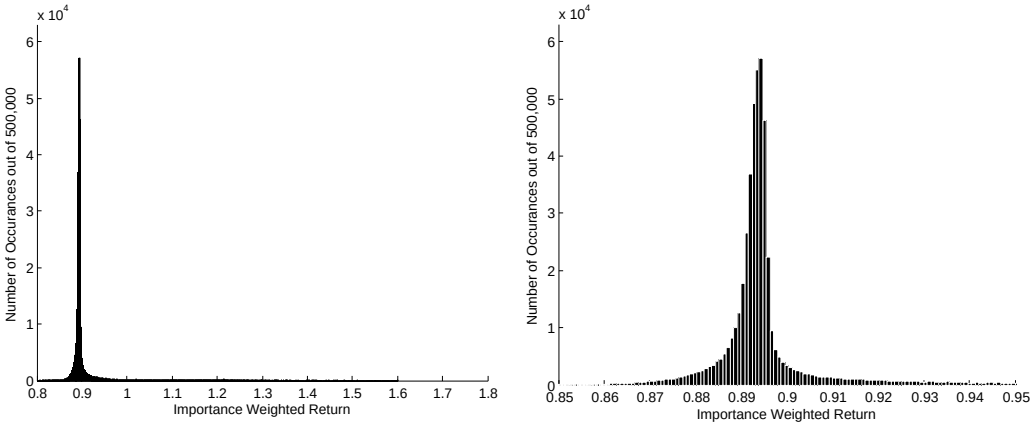


**Figure 3.3.** Empirical distribution of PDIS estimators when evaluating the evaluation policy for the gridworld using 500,000 trajectories from the behavior policy. The plot on the right is zoomed in to provide more detail around the region containing most of the probability mass. The tallest column is **not** cut off in this plot and has a value of 57,167.

The mean PDIS estimator (i.e., $\text{PDIS}(\pi_e|\mathcal{D})$) was 0.900, which is equal to the true expected performance, $\rho(\pi_e)$, to all three decimal places. The largest observed PDIS estimator was 1.601 and the smallest was approximately 0.800. The upper and lower bounds on $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L, \pi_b)$ are $\hat{\rho}^{\text{PDIS}}_{\text{ub}}(\pi_e, \pi_b) = 1.071 \times 10^{58}$ and $\hat{\rho}^{\text{PDIS}}_{\text{lb}}(\pi_e, \pi_b) = -1.071 \times 10^{58}$.

Notice that the distribution of $\hat{\rho}^{\text{PDIS}}(\pi_e|H_L, \pi_b)$ is much more desirable than the distribution of $\hat{\rho}^{\text{IS}}(\pi_e|H_L, \pi_b)$—it has much less variance. It is also approximately symmetric, which means that it does not tend to under or over-estimate $\rho(\pi_e)$ when using only a few trajectories.

However, PDIS is poorly suited to producing high confidence lower bounds on $\rho(\pi_e)$. The methods that we will present in the next chapter perform poorly if $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e, \pi_b)$, the lower bound on $\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)$, is far away from $\rho(\pi_e)$. The IS estima-

tor's lower bound was zero, which is much closer to $\rho(\pi_e) \in [0, 1]$ than $\hat{\rho}^{\text{PDIS}}_{\text{lb}}(\pi_e, \pi_b) = -1.071 \times 10^{58}$. In the next section we propose a normalized form of the PDIS estimator that, like the IS estimator, is lower bounded by zero.

## 3.7 Normalized Per-Decision Importance Sampling (NPDIS) Estimator

To make the PDIS estimator more suitable for our use, we must make it so that it is lower bounded by zero. We achieve this by subtracting $r_{\text{lb}}$ from every reward. This makes the rewards nonnegative. We then correct for the bias that this introduces by adding back $r_{\text{lb}} \sum_{t=1}^{L} \gamma^{t-1} = r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1}$ prior to normalization.[3] We call this estimator the *normalized per-decision importance sampling* (NPDIS) estimator, and it is given by

$$\hat{\rho}^{\text{NPDIS}}(\pi_e | H_L, \pi_b) \coloneqq \frac{\left( \sum_{t=1}^{L} \gamma^{t-1} (R_t - r_{\text{lb}}) \prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)} \right) + r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}, \quad (3.17)$$

and in batch form:

$$\text{NPDIS}(\pi_e | \mathcal{D}) \coloneqq \frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \hat{\rho}^{\text{NPDIS}}(\pi_e | H_L^i, \pi_i), \quad (3.18)$$

In Theorem 7 we show that $\hat{\rho}^{\text{NPDIS}}(\pi_e | H_L, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ and in Theorem 8 we show that the batch NPDIS estimator is an unbiased estimator of $\rho(\pi_e)$ regardless of how many trajectories are in $\mathcal{D}$.

**Theorem 7.** *The NPDIS estimator, $\hat{\rho}^{NPDIS}(\pi_e | H_L, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds and $H_L \sim \pi_b$.*

[3] If $\gamma = 1$, then this term should be replaced with $L r_{\text{lb}}$.

*Proof.*

$$
\begin{aligned}
&\mathbf{E}\left[\hat{\rho}^{\mathrm{NPDIS}}(\pi_e|H_L,\pi_b)\middle|H_L\sim\pi_b\right]\\
=&\mathbf{E}\left[\frac{\left(\sum_{t=1}^{L}\gamma^{t-1}(R_t-r_{\mathrm{lb}})\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right)+r_{\mathrm{lb}}\frac{\gamma^L-1}{\gamma-1}-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}}\middle|H_L\sim\pi_b\right]\\
=&\mathbf{E}\left[\frac{\left(\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right)-\left(\sum_{t=1}^{L}\gamma^{t-1}r_{\mathrm{lb}}\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right)+r_{\mathrm{lb}}\frac{\gamma^L-1}{\gamma-1}-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}}\middle|H_L\sim\pi_b\right]\\
=&\frac{\mathbf{E}\left[\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\middle|H_L\sim\pi_b\right]-r_{\mathrm{lb}}\sum_{t=1}^{L}\gamma^{t-1}\mathbf{E}\left[\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\middle|H_L\sim\pi_b\right]+r_{\mathrm{lb}}\frac{\gamma^L-1}{\gamma-1}-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}}\\
\overset{\textbf{(a)}}{=}&\frac{\mathbf{E}\left[\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\middle|H_L\sim\pi_b\right]-r_{\mathrm{lb}}\sum_{t=1}^{L}\gamma^{t-1}+r_{\mathrm{lb}}\frac{\gamma^L-1}{\gamma-1}-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}}\\
=&\frac{\mathbf{E}\left[\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{i=1}^{t}\frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\middle|H_L\sim\pi_b\right]-G_{\mathrm{lb}}}{G_{\mathrm{ub}}-G_{\mathrm{lb}}}\\
=&\mathbf{E}\left[\hat{\rho}^{\mathrm{PDIS}}(\pi_e|H_L,\pi_b)\right]\\
\overset{\textbf{(b)}}{=}&\rho(\pi_e),
\end{aligned}
$$

where **(a)** uses Lemma 3 (which uses Assumption 1) and **(b)** uses (3.13) (which also relies on Assumption 1). ■

**Theorem 8.** *The NPDIS batch estimator, NPDIS($\pi_e|\mathcal{D}$) is an unbiased estimator of $\rho(\pi_e)$ if Assumption 1 holds.*

*Proof.*

$$
\begin{aligned}
\mathbf{E}\left[\mathrm{NPDIS}(\pi_e|\mathcal{D})\right]=&\mathbf{E}\left[\frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\hat{\rho}^{\mathrm{NPDIS}}(\pi_e|H_L^i,\pi_i)\right]\\
=&\frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\mathbf{E}\left[\hat{\rho}^{\mathrm{NPDIS}}(\pi_e|H_L^i,\pi_i)\right]\\
\overset{\textbf{(a)}}{=}&\frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}}\rho(\pi_e)\\
=&\rho(\pi_e),
\end{aligned}
$$

where **(a)** holds by Theorem 7 and Assumption 1. ■

### 3.7.1 Upper and Lower Bounds on the NPDIS Estimator

We can bound the NPDIS estimator by

$$\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_b) \geq 0.$$

and

$$\hat{\rho}^{\text{NPDIS}}_{\text{ub}}(\pi_e, \pi_b) \coloneqq \frac{\left(\sum_{t=1}^{L} \gamma^{t-1}(r_{\text{ub}} - r_{\text{lb}})\varsigma^t\right) + r_{\text{lb}}\frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}, \tag{3.19}$$

where $\varsigma$ is defined in (3.14).

### 3.7.2 Consistency of NPDIS Estimator

We showed that $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ if $H_L \sim \pi_b$, and that $\text{NPDIS}(\pi_e|\mathcal{D})$ is an unbiased estimator of $\rho(\pi_e)$ regardless of how many trajectories are in $\mathcal{D}$. Here we present two different sets of assumptions that ensure that the batch NPDIS estimator is consistent.

First we show that the batch NPDIS estimator is consistent if there is only one behavior policy:

**Theorem 9.** *If Assumption 1 holds and there is only one behavior policy, i.e., $\pi_i = \pi_b$ for all $i$, then NPDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.*

*Proof.* This follows from Corollary 1 since **(1)** $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and **(2)** the random variables $\{\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_b)\}_{i=1}^{n_{\mathcal{D}}}$ are identically distributed since the trajectories are all generated by the same behavior policy. ■

Next we show that the NPDIS estimator is consistent if there are many behavior policies, all of which are bounded away from being deterministic when the evaluation policy is not deterministic.

**Theorem 10.** *If Assumption 1 holds and there exists a constant $\epsilon > 0$ such that $\pi_i(a|o) \geq \epsilon$ for $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and $(a, o)$ where $\pi_e(a|o) \neq 0$, then PDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$ even if there are multiple behavior policies.*

*Proof.* This follows from Corollary 2 since **(1)** $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_b)$ is an unbiased estimator of $\rho(\pi_e)$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and **(2)** for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$

$$\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_b) \in \left[0, \frac{\frac{L}{\epsilon^L}(r_{\text{ub}} - r_{\text{lb}}) + r_{\text{lb}}\frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}\right],$$

and so $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_b)$ has bounded variance. ■

### 3.7.3 Example: Gridworld

Consider the application of NPDIS to the behavior and evaluation policies from the gridworld. We will use the exact same trajectories that were used to produce the left plot in Figure 3.1 (the empirical distribution of returns under the behavior policy) to produce an estimate of the performance of the behavior policy. That is, we compute $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L^i, \pi_i)$, for each of the 500,000 trajectories ($i \in \{1, \ldots, 500{,}000\}$) that were generated by the behavior policy. The resulting distribution of the 500,000 estimators, $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_i)$, is depicted in Figure 3.4.

The mean NPDIS estimator (i.e., PDIS$(\pi_e|\mathcal{D})$) was 0.900, which is equal to the true expected performance, $\rho(\pi_e)$, to all three decimal places. The largest observed PDIS estimator was 60.494 and the smallest was approximately 0.011. The upper bound on $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_b)$ is $\hat{\rho}_{\text{ub}}^{\text{PDIS}}(\pi_e, \pi_b) = 2.1421.071 \times 10^{58}$. Notice that the first column extends beyond the top of the plot to a value of 212,131. Also notice that the NPDIS estimator is distributed much like the IS estimator, although not identically. This is because the evaluation policy tends to reach the terminal state quickly. The terminal state has the largest reward, which makes up much of the return, and its importance weight is the same as the importance weight used by the IS estimator.

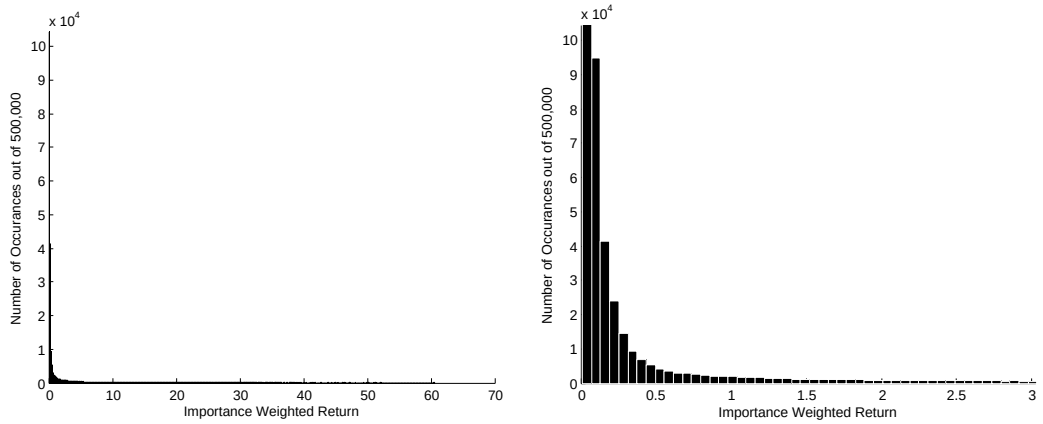


**Figure 3.4.** Empirical distribution of NPDIS estimators when evaluating the evaluation policy for the gridworld using 500,000 trajectories from the behavior policy. The plot on the right is zoomed in to provide more detail around the region containing most of the probability mass. The first column is cut off by the limit of the vertical axis—it dwarfs the others with a value of 212,131.

Still, the NPDIS estimator is preferable to the IS estimator since it has slightly lower variance and a lower upper bound, and it will be preferable to the PDIS estimator in the next chapter because its lower bound is 0, which is closer to $\rho(\pi_e)$ than $\hat{\rho}_{\text{lb}}^{\text{PDIS}}(\pi_e, \pi_b)$.

## 3.8 Weighted Importance Sampling (WIS) Estimator

The primary drawback of IS, PDIS, and NPDIS is their large possible range, which results in high variance for IS and NPDIS. Notice, for example, that even though the returns, $G(H_L)$, and expected returns, $\rho(\pi)$, are between 0 and 1, the estimators, $\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)$ for $\dagger \in \{\text{IS, PDIS, NPDIS}\}$, can be much larger than 1 (see for example $\hat{\rho}_{\text{ub}}^{\dagger}(\pi_e, \pi_b)$ or the largest observed estimator values for the gridworld example for each estimator). *Weighted importance sampling* (WIS) is an alternative that is usually biased but which has a much smaller range (it is always in $[0, 1]$, like the true expected return) and it has much lower variance than IS and NPDIS (it often performs similarly to PDIS).

When there are few samples available, the lower variance of WIS may produce a larger reduction in expected square error than the additional error incurred due to the bias (Thomas et al., 2015a), making WIS preferable to IS and NPDIS even though it is a biased estimator of $\rho(\pi_e)$. Furthermore, although WIS is a biased estimator, we will show that it is consistent, which means that it becomes less biased as the number of samples increases.

WIS is intended for the batch setting where many samples are available. Let $\{q_i\}_{i=1}^n$ be $n$ probability mass functions. Let $p$ be another probability mass function. The average of the IS estimator over all $q_i$ produces an unbiased estimator of $\mathbf{E}[f(X)|X \sim p]$:

$$\mathbf{E}[f(X)|X \sim p] = \mathbf{E}\left[\underbrace{\frac{\sum_{i=1}^n \frac{p(X_i)}{q_i(X_i)} f(X_i)}{n}}_{\text{mean IS estimator}} \middle| X_1 \sim q_1, \dots X_n \sim q_n\right].$$

The WIS estimator divides by the sum of the importance weights, rather than $n$:

$$\mathbf{E}[f(X)|X \sim p] \approx \mathbf{E}\left[\underbrace{\frac{\sum_{i=1}^n \frac{p(X_i)}{q_i(X_i)} f(X_i)}{\sum_{i=1}^n \frac{p(X_i)}{q_i(X_i)}}}_{\text{WIS estimator}} \middle| X_1 \sim q_1, X_n \sim q_n\right]. \tag{3.20}$$

In our setting, the WIS estimator is given by

$$\text{WIS}(\pi_e|\mathcal{D}) \coloneqq \frac{\sum_{i=1}^{n_\mathcal{D}} \prod_{t=1}^L \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)} G(H_L^i)}{\sum_{i=1}^{n_\mathcal{D}} \prod_{t=1}^L \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)}}. \tag{3.21}$$

In our case where $G(H_L) \in [0, 1]$, dividing by the sum of the importance weights ensures that the WIS estimator is also bounded within $[0, 1]$. We saw that the IS estimator was often an underestimate if only a few samples are available because most samples have small (near zero) importance weights. The WIS estimator is less

sensitive to this because, if every sample has a small (near zero) importance weight, the denominator will be much less than $n$, which increases the value of the WIS estimator relative to the IS estimator. Similarly, if a massive importance weight occurs, the IS estimator tends to greatly overestimate the target value. However, the WIS estimator will divide by the massive importance weight causing the estimator to maintain a reasonable value.

To better see what WIS is doing and why it is consistent, consider the case where there is only a single sample in (3.20), i.e., $n = 1$. The importance weights in WIS cancel from the numerator and denominator and the WIS estimator is merely $f(X_1)$, where $X_1 \sim q_1$, which can be a biased estimator of $\mathbf{E}[f(X)|X \sim p]$. However, notice from Lemma 3 that as the number of samples increases, the denominator of the WIS estimator tends towards $n$ (if it is $n$, then WIS is equivalent to IS).

The WIS estimator is trivially bounded within $[0, 1]$, and so $\hat{\rho}^{\text{WIS}}_{\text{lb}}(\pi_e, \pi_b) = 0$ and $\hat{\rho}^{\text{WIS}}_{\text{ub}}(\pi_e, \pi_b) = 1$ for all $\pi_e$ and $\pi_b$. Below we show that WIS is a consistent estimator of $\rho(\pi_e)$ if there is a single behavior policy (Theorem 11) or if there are multiple behavior policies that satisfy a technical requirement (Theorem 12).

**Theorem 11.** *If Assumption 1 holds and there is only one behavior policy, i.e., $\pi_i = \pi_b$ for all $i$, then WIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.*

*Proof.* First, we rewrite the WIS estimator by multiplying the numerator and denominator both by $\frac{1}{n_{\mathcal{D}}}$:

$$\text{WIS}(\pi_e|\mathcal{D}) = \frac{\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)} G(H_L^i)}{\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)}}. \tag{3.22}$$

This proof then proceeds by first showing that the numerator of (3.22) converges almost surely to $\rho(\pi_e)$ and then that the denominator converges almost surely to 1. By Property 2, this means that WIS$(\pi_e|\mathcal{D})$ also converges almost surely to $\rho(\pi_e)$, and so WIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.

**Numerator:** The numerator is equal to $\text{IS}(\pi_e|\mathcal{D})$. Therefore, by Theorem 3 and Assumption 1, the numerator converges almost surely to $\rho(\pi_e)$.

**Denominator:** By Lemma 3 we have that for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$:

$$\mathbf{E}\left[\prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)} \middle| H_L^i \sim \pi_i\right] = 1. \tag{3.23}$$

Furthermore, each term, $\prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)}$ is identically distributed for each $i \in \{1, \ldots, n_{\mathcal{D}}\}$ since there is only one behavior policy. So, by the Khintchine strong law of large numbers, we have that

$$\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)} \xrightarrow{\text{a.s.}} 1. \tag{3.24}$$

■

**Theorem 12.** *If Assumption 1 holds and there exists a constant $\epsilon > 0$ such that $\pi_i(a|o) \geq \epsilon$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and $(a, o)$ where $\pi_e(a|o) \neq 0$, then WIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$ even if there are multiple behavior policies.*

*Proof.* This proof has the same form as the proof of Theorem 11—we show that the numerator and denominator of (3.22) converge almost surely to 1 and $\rho(\pi_e)$ respectively.

**Numerator:** The numerator is equal to $\text{IS}(\pi_e|\mathcal{D})$. Therefore, by Theorem 4 and Assumption 1, the numerator converges almost surely to $\rho(\pi_e)$.

**Denominator:** Again, by Lemma 3 we have that (3.23) holds. Furthermore, each term, $\prod_{t=1}^{L} \frac{\pi_e(A_t^i|O_t^i)}{\pi_i(A_t^i|O_t^i)} \in \left[0, \frac{1}{\epsilon^L}\right]$ and therefore has bounded variance. So, by the Kolmogorov strong law of large numbers, we have that (3.24) holds. ■

## 3.9 Weighted Per-Decision Importance Sampling (WPDIS) Estimator

Just as the IS estimator was modified to produce the WIS estimator, we can modify the PDIS estimator to get a *weighted per-decision importance sampling* (WPDIS) estimator. Rather than dividing by the number of samples in the batch PDIS estimator, we simply divide by the sum of the importance weights:

$$\text{WPDIS}(\pi_e|H_L,\pi_b) \coloneqq \left(\frac{1}{G_{\text{ub}}-G_{\text{lb}}}\right)\left(\frac{\sum_{i=1}^{n_\mathcal{D}}\sum_{t=1}^{L}\gamma^{t-1}R_t\prod_{j=1}^{t}\frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)}}{\sum_{i=1}^{n_\mathcal{D}}\sum_{t=1}^{L}\gamma^{t-1}\prod_{i=1}^{t}\frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)}}-G_{\text{lb}}\right). \tag{3.25}$$

This estimator was proposed by Precup (2000) for the MDP setting. Although they correctly assert that it may be a biased estimator of $\rho(\pi_e)$, they incorrectly assert that it is a consistent estimator of $\rho(\pi_e)$. To see that it is not consistent, consider what happens if $\pi_e = \pi_i$ for all $i$. This makes all of the action probability ratios one. To further simplify the equation, let $\gamma = 1$, $L > 1$, $G_{\text{lb}} = 0$, and $G_{\text{ub}} = 1$. The estimator then becomes

$$\begin{aligned}\text{WPDIS}(\pi_e|H_L,\pi_e) =& \frac{\frac{\sum_{i=1}^{n_\mathcal{D}}\sum_{t=1}^{L}R_t}{nL}-0}{1-0}\\ =& \frac{\sum_{i=1}^{n_\mathcal{D}}\sum_{t=1}^{L}R_t}{nL}\\ =& \frac{1}{L}\frac{1}{n_\mathcal{D}}\sum_{i=1}^{n_\mathcal{D}}G(H_L^i),\end{aligned}$$

which, if $\rho(\pi_e) \neq 0$, will converge almost surely to $\frac{\rho(\pi_e)}{L} \neq \rho(\pi_e)$. So, not only can the WPDIS estimator be a biased estimator of $\rho(\pi_e)$, but it is also *not* a consistent estimator of $\rho(\pi_e)$, even if using only a single behavior policy. Because the WPDIS estimator is neither unbiased nor consistent, we do not discuss it further or include it in our experiments.

## 3.10 Consistent Weighted Per-Decision Importance Sampling (CWPDIS) Estimator

We propose a new estimator that applies the ideas behind the WIS estimator to the PDIS estimator in a way that allows it to remain a consistent estimator of $\rho(\pi_e)$. Because this estimator is based on PDIS, it only applies to definitions of $G(H_L)$ that are affine functions of the reward, like the normalized discounted return. We call this new estimator the *consistent weighted per-decision importance sampling* (CWPDIS) estimator. If using the normalized discounted return defined in (2.5) for $G(H_L^i)$, then:

$$\text{CWPDIS}(\pi_e|\mathcal{D}) \coloneqq \left(\frac{1}{G_{\text{ub}} - G_{\text{lb}}}\right)\left(\left(\sum_{t=1}^{L}\gamma^{t-1}\underbrace{\frac{\sum_{i=1}^{n_\mathcal{D}} R_t \prod_{j=1}^{t}\frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}{\sum_{i=1}^{n_\mathcal{D}}\prod_{j=1}^{t}\frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}}_{\text{WIS estimate of } \mathbf{E}[R_t|H_L\sim\pi_e]}\right) - G_{\text{lb}}\right). \tag{3.26}$$

For each time, $t$, the PDIS estimator uses ordinary importance sampling to estimate the expected discounted *reward* at time $t$ (see (3.10)). It then sums these $L$ IS estimators (including the discount parameter $\gamma$) to get an estimate of the expected (unnormalized) return. Here we use weighted importance sampling rather than ordinary importance sampling to estimate the expected reward at time $t$. Notice that each weighted importance sampling estimate of $\mathbf{E}[R_t|H_L \sim \pi_e]$ is bounded within $r_{\text{lb}}$ and $r_{\text{ub}}$, and so $\hat{\rho}_{\text{lb}}^{\text{CWPDIS}}(\pi_e, \pi_b) = 0$ and $\hat{\rho}_{\text{ub}}^{\text{CWPDIS}}(\pi_e, \pi_b) = 1$ for all $\pi_e$ and $\pi_b$ if $G(H_L)$ is as defined in (2.5) and $G_{\text{lb}}$ and $G_{\text{ub}}$ are as defined in (2.3) and (2.4).

The CWPDIS estimator can be biased. However, unlike the WPDIS estimator, it is a consistent estimator of $\rho(\pi_e)$:

**Theorem 13.** *If Assumption 1 holds and there is only one behavior policy, i.e., $\pi_i = \pi_b$ for all $i$, then CWPDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$.*

*Proof.* Let

$$X_t \coloneqq \frac{\sum_{i=1}^{n_{\mathcal{D}}} R_t \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}{\sum_{i=1}^{n_{\mathcal{D}}} \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}},$$

so that $\text{CWPDIS}(\pi_e|\mathcal{D}) = \frac{\left(\sum_{t=1}^{L} \gamma^{t-1} X_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}$.

By Properties 1 and 3 we have that if $X_t \xrightarrow{\text{a.s.}} \mathbf{E}[R_t|H_L \sim \pi_e]$ for all $t \in \{1, \ldots, L\}$, then $\text{CWPDIS}(\pi_e|\mathcal{D}) \xrightarrow{\text{a.s.}} \rho(\pi_e)$. To show that $X_t \xrightarrow{\text{a.s.}} \mathbf{E}[R_t|H_L \sim \pi_e]$, we first rewrite $X_t$ my multiplying the numerator and denominator both by $\frac{1}{n_{\mathcal{D}}}$:

$$X_t \coloneqq \frac{\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} R_t \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}{\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_b(A_j^i|O_j^i)}}. \tag{3.27}$$

This proof proceeds by first showing that the numerator of (3.27) converges almost surely to $\mathbf{E}[R_t|H_L \sim \pi_e]$ and then that the denominator converges almost surely to 1. By Property 2, this means that $X_t$ also converges almost surely to $\mathbf{E}[R_t|H_L \sim \pi_e]$, and so we can conclude that $\text{CWPDIS}(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$. Notice that the remainder of the proof holds for any fixed $t \in \{1, \ldots, L\}$.

**Numerator:** Let

$$Y_i \coloneqq R_t \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)}. \tag{3.28}$$

We have that

$$
\begin{aligned}
\mathbf{E}[Y_i|H_L^i \sim \pi_i] =& \mathbf{E}\left[R_t \prod_{j=1}^{t} \frac{\pi_e(A_j|O_j)}{\pi_i(A_j|O_j)} \middle| H_L \sim \pi_i\right] \\
\overset{\textbf{(a)}}{=}& \mathbf{E}\left[R_t \prod_{j=1}^{t} \frac{\pi_e(A_j|O_j)}{\pi_i(A_j|O_j)} \middle| H_t \sim \pi_i\right] \\
=& \sum_{\text{supp}_t\, \pi_i} \Pr(H_t = h_t|\pi_i) r_t \prod_{j=1}^{t} \frac{\pi_e(a_j|o_j)}{\pi_i(a_j|o_j)} \\
\overset{\textbf{(b)}}{=}& \sum_{\text{supp}_t\, \pi_i} \Pr(H_t = h_t|\pi_i) r_t \frac{\Pr(H_t = h_t|\pi_e)}{\Pr(H_t = h_t|\pi_i)} \\
=& \sum_{\text{supp}_t\, \pi_i} \Pr(H_t = h_t|\pi_e) r_t \\
\overset{\textbf{(c)}}{=}& \sum_{\text{supp}_t\, \pi_e} \Pr(H_t = h_t|\pi_e) r_t \\
=& \mathbf{E}\left[R_t|H_t \sim \pi_e\right] \\
\overset{\textbf{(d)}}{=}& \mathbf{E}\left[R_t|H_L \sim \pi_e\right], \qquad (3.29)
\end{aligned}
$$

where **(a)** and **(d)** hold by Lemma 2, **(b)** holds by Lemma 1, and **(c)** holds because Assumption 1 implies Corollary 3. So, by the Khintchine strong law of large numbers, we have that

$$
\underbrace{\frac{1}{n}\sum_{i=1}^{n_{\mathcal{D}}} Y_i}_{\text{numerator of (3.27)}} \xrightarrow{\text{a.s.}} \mathbf{E}\left[R_t|H_L \sim \pi_e\right].
$$

**Denominator:** By Lemma 3 we have that

$$
\mathbf{E}\left[\prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)} \middle| H_L^i \sim \pi_i\right] = 1. \qquad (3.30)
$$

Furthermore, each term, $\prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)}$ is identically distributed for each $i \in \{1, \ldots, n_{\mathcal{D}}\}$. So, by the Khintchine strong law of large numbers, we have that

$$
\frac{1}{n_{\mathcal{D}}}\sum_{i=1}^{n_{\mathcal{D}}} \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)} \xrightarrow{\text{a.s.}} 1.
$$

■

**Theorem 14.** *If Assumption 1 holds and there exists a constant $\epsilon > 0$ such that $\pi_i(a|o) \geq \epsilon$ for all $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and $(a, o)$ where $\pi_e(a|o) \neq 0$, then CWPDIS$(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$ even if there are multiple behavior policies.*

*Proof.* Recall the definition of $X_t$ from (3.27), such that $\text{CWPDIS}(\pi_e|\mathcal{D}) = \frac{\left(\sum_{t=1}^{L} \gamma^{t-1} X_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}}$. By Properties 1 and 3 we have that if $X_t \xrightarrow{\text{a.s.}} \mathbf{E}[R_t|H_L \sim \pi_e]$ for all $t \in \{1, \ldots, L\}$, then $\text{CWPDIS}(\pi_e|\mathcal{D}) \xrightarrow{\text{a.s.}} \rho(\pi_e)$.

This proof proceeds by first showing that the numerator of $X_t$ converges almost surely to $\mathbf{E}[R_t|H_L \sim \pi_e]$ and then that the denominator converges almost surely to 1. By Property 2, this means that $X_t$ also converges almost surely to $\mathbf{E}[R_t|H_L \sim \pi_e]$, and so we can conclude that $\text{CWPDIS}(\pi_e|\mathcal{D})$ is a consistent estimator of $\rho(\pi_e)$. Notice that the remainder of the proof holds for any fixed $t \in \{1, \ldots, L\}$.

**Numerator:** Let $Y_i$ be defined as in (3.28). We have from (3.29) that $\mathbf{E}[Y_i|H_L^i \sim \pi_i] = \mathbf{E}\left[R_t|H_L \sim \pi_e\right]$. Furthermore,

$$Y_i \in \left[\min\left\{0, \frac{r_{\text{lb}}}{\epsilon^t}\right\}, \max\left\{0, \frac{r_{\text{ub}}}{\epsilon^t}\right\}\right],$$

and so each $Y_i$ has bounded variance. So, by the Kolmogorov strong law of large numbers, we have that

$$\underbrace{\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} Y_i}_{\text{numerator of } X_i} \xrightarrow{\text{a.s.}} \mathbf{E}\left[R_t|H_L \sim \pi_e\right].$$

**Denominator:** By Lemma 3 we have that (3.30) holds. Furthermore, each term, $\prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)} \in [0, \epsilon^t]$, and therefore has bounded variance. So, by the Kolmogorov strong law of large numbers, we have that

$$\frac{1}{n_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{D}}} \prod_{j=1}^{t} \frac{\pi_e(A_j^i|O_j^i)}{\pi_i(A_j^i|O_j^i)} \xrightarrow{\text{a.s.}} 1.$$

■

## 3.11 Deterministic Behavior Policies

Notice that our derivation of IS (Equation (3.4)) assumes that the support of the target distribution (evaluation policy) is a subset of the support of the sampling distribution (behavior policy), or at least that if $q(x) = 0$ then $f(x)p(x) = 0$. In our application this is enforced by Assumption 1. However, Assumption 1 may not always hold if a behavior policy is deterministic.

We show that, even if Assumption 1 does *not* hold, the expected value of the IS estimator is less than $\rho(\pi_e)$. That is, on average, it produces an underestimate of $\rho(\pi_e)$. This means that when we use it later for high-confidence off-policy evaluation, the lower bounds that we generate will be lower bounds even if Assumption 1 does not hold.

**Theorem 15.** *Let $p$ and $q$ be the probability mass functions for two distributions over a discrete set of events, $\mathcal{X}$, and $f : \mathcal{X} \to [0, \infty)$. Then the expected importance sampling estimator, $\mathbf{E}\left[\frac{p(X)}{q(X)} f(X) \middle| X \sim q\right]$, is less than $\mathbf{E}[f(X)|X \sim p]$.*

*Proof.*

$$
\begin{aligned}
\mathbf{E}[f(X)|X \sim p] &= \sum_{x \in \operatorname{supp} p} p(x) f(x) \\
&= \sum_{x \in \operatorname{supp} q} p(x) f(x) + \sum_{x \in \operatorname{supp} p \setminus \operatorname{supp} q} p(x) f(x) - \underbrace{\sum_{x \in \operatorname{supp} q \setminus \operatorname{supp} p} p(x) f(x)}_{\overset{(a)}{=} 0} \\
&\overset{(b)}{\geq} \sum_{x \in \operatorname{supp} q} p(x) f(x) \\
&= \sum_{x \in \operatorname{supp} q} \underbrace{\frac{q(x)}{q(x)}}_{=1} p(x) f(x) \\
&= \sum_{x \in \operatorname{supp} q} q(x) \frac{p(x)}{q(x)} f(x) \\
&= \mathbf{E}\left[ \frac{p(X)}{q(X)} f(X) \middle| X \sim q \right],
\end{aligned}
$$

where **(a)** this integral is zero because, from the definition of $\operatorname{supp} p$, we have $p(x) = 0$ for every $x \in \operatorname{supp} q \setminus \operatorname{supp} p$, and **(b)** this inequality holds because $f(x) \geq 0$ for all $x$. ■

It follows from Theorem 15 that the expected value of the IS estimators (batch and single-trajectory) will never be above $\rho(\pi_e)$:

**Corollary 4.** $\mathbf{E}[\hat{\rho}^{IS}(\pi_e|H_L, \pi_b)] \leq \rho(\pi_e)$ *for all* $\pi_e$ *and* $\pi_b$ *where* $H_L \sim \pi_b$.

*Proof.* This follows from Theorem 15 since $\hat{\rho}^{\mathrm{IS}}(\pi_e|H_L, \pi_b)$ is the ordinary IS estimator. ■

**Corollary 5.** $\mathbf{E}[IS(\pi_e|\mathcal{D})] \leq \rho(\pi_e)$ *for all* $\pi_e$ *and* $\pi_b$ *where* $H_L \sim \pi_b$.

*Proof.* This follows from Corollary 4 and the definition of $\mathrm{IS}(\pi_e|\mathcal{D})$. ■

Next we can show that the same holds for the NPDIS estimators:

**Theorem 16.** $\mathbf{E}[\hat{\rho}^{NPDIS}(\pi_e|H_L, \pi_b)] \leq \rho(\pi_e)$ *for all* $\pi_e$ *and* $\pi_b$ *where* $H_L \sim \pi_b$.

*Proof.* Consider the application of ordinary importance sampling to $\gamma^{t-1}(R_t - r_{\text{lb}})$, which is always positive. By Theorem 15 we have that

$$\mathbf{E}\left[\underbrace{\gamma^{t-1}(R_t - r_{\text{lb}})}_{f(X)} \underbrace{\prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}}_{\frac{p(X)}{q(X)}} \middle| \underbrace{H_L \sim \pi_b}_{X \sim q}\right] \leq \underbrace{\mathbf{E}\left[\gamma^{t-1}(R_t - r_{\text{lb}}) \middle| H_L \sim \pi_e\right]}_{\mathbf{E}[f(X)|X \sim p]}. \quad (3.31)$$

So,

$$\begin{aligned}
&\mathbf{E}\left[\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_b) \middle| H_L \sim \pi_b\right] \\
=&\mathbf{E}\left[\frac{\left(\sum_{t=1}^{L} \gamma^{t-1}(R_t - r_{\text{lb}}) \prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)}\right) + r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \middle| H_L \sim \pi_b\right] \\
=&\frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1}(R_t - r_{\text{lb}}) \prod_{i=1}^{t} \frac{\pi_e(A_i|O_i)}{\pi_b(A_i|O_i)} \middle| H_L \sim \pi_b\right]\right) + r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\
\overset{\textbf{(a)}}{\leq}&\frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1}(R_t - r_{\text{lb}}) \middle| H_L \sim \pi_e\right]\right) + r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\
=&\frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1} R_t \middle| H_L \sim \pi_e\right]\right) - \left(\sum_{t=1}^{L} \gamma^{t-1} r_{\text{lb}}\right) + r_{\text{lb}} \frac{\gamma^L - 1}{\gamma - 1} - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\
=&\frac{\left(\sum_{t=1}^{L} \mathbf{E}\left[\gamma^{t-1} R_t \middle| H_L \sim \pi_e\right]\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \\
=&\mathbf{E}\left[\frac{\left(\sum_{t=1}^{L} \gamma^{t-1} R_t\right) - G_{\text{lb}}}{G_{\text{ub}} - G_{\text{lb}}} \middle| H_L \sim \pi_e\right] \\
=&\rho(\pi_e),
\end{aligned}$$

where **(a)** comes from (3.31). ■

**Corollary 6.** $\mathbf{E}[PDIS(\pi_e|\mathcal{D})] \leq \rho(\pi_e)$ *for all* $\pi_e$ *and* $\pi_b$ *where* $H_L \sim \pi_b$.

*Proof.* This follows from Theorem 16 and the definition of NPDIS$(\pi_e|\mathcal{D})$. ■

Next we consider the estimators that are based on weighted importance sampling. First, we define what it means for a sequence of estimators to be consistent underestimators, which is the consistency equivalent to ensuring that the expected value of an estimator is no larger than the target value.

**Definition 4.** *Let $\theta$ be a real number and $(\hat{\theta}_n)_{n=1}^{\infty}$ be an infinite sequence of random variables. We call $\hat{\theta}_n$, a* **consistent underestimator** *of $\theta$ if and only if $\hat{\theta}_n \xrightarrow{a.s.} \theta'$ for some $\theta' \leq \theta$.*

Notice that the weighted importance sampling estimators are *not* consistent underestimators of $\rho(\pi_e)$ if Assumption 1 does not hold. Consider the following counterexample. We use weighted importance sampling to estimate $\mathbf{E}[f(X)|X \sim p]$ given $n$ samples, $\{X_i\}_{i=1}^n$, where each $X_i \sim q$. The WIS estimator in this case is

$$\text{WIS}(p, \{X_i\}_{i=1}^n, q) \coloneqq \frac{\sum_{i=1}^n \frac{p(X_i)}{q(X_i)} f(X_i)}{\sum_{i=1}^n \frac{p(X_i)}{q(X_i)}}.$$

If $X \in \{x_1, x_2\}, p(x_1) = 0.5, p(x_2) = 0.5, q(x_1) = 0, q(x_2) = 1, f(x_1) = 0$, and $f(x_2) = 1$, then $\mathbf{E}[f(X)|X \sim p] = 0.5$. However, $\text{WIS}(p, \{X_i\}_{i=1}^n, q) \xrightarrow{\text{a.s.}} 1$.

In summary, the IS and NPDIS estimators will be useful for generating lower bounds on $\rho(\pi_e)$ when Assumption 1 is not guaranteed since they do not overestimate $\rho(\pi_e)$ in expectation. However, the other estimators should be avoided if Assumption 1 is not satisfied.

## 3.12 Empirical Comparison

The effectiveness of each method depends mainly on the similarity of the policies being compared, how close to deterministic the policies are, how long the trajectories are, and whether or not the bulk of the return is received near the end of the episode. All of these properties can be varied using the gridworld domain. So, we compare IS, PDIS, NPDIS, WIS, and CWPDIS using five different policies for the gridworld. The policies are:

1. $\pi 1$ **(random):** This policy selects each of the four actions with probability 0.25 regardless of the observation. It takes a long time to reach the goal and often

visits the punishing state that gives a reward of $-10$. This is the same policy that was used as the behavior policy in the examples included thus far in this chapter, i.e., the behavior policy from Figure 3.1.

2. $\pi 2$ **(mediocre):** This policy was discovered using a few iterations of a black box optimization routine to maximize $\rho$. It moves quickly to the goal and infrequently visits the state that gives a reward of $-10$. This is the same policy that was used as the evaluation policy in the examples included thus far in this chapter, i.e., the evaluation policy from Figure 3.1.

3. $\pi 3$ **(mediocre+):** This policy is $\pi 2$ but with all action probabilities made slightly more extreme (the most common action is given a slightly higher probability). This results in a slight improvement over $\pi 2$.

4. $\pi 4$ **(near-optimal):** This policy is a hand coded near-optimal policy that moves quickly to the position $(2, 4)$ without hitting $(2, 2)$. It then usually takes the action to move down, and occasionally takes the action to move right. Once it is in $(2, 4)$ it moves almost deterministically to $(4, 4)$.

5. $\pi 5$ **(near-optimal+):** This policy is a slightly more deterministic version of $\pi 4$.

The performance of each policy is presented graphically in Figure 3.5.

Each of the following empirical comparisons is performed the same way. First, a behavior and evaluation policy are selected. We then repeat the following using various numbers of trajectories, $n_{\mathcal{D}}$. We sample $n_{\mathcal{D}}$ trajectories and use each estimator to estimate $\rho(\pi_e)$ using these $n_{\mathcal{D}}$ trajectories. We compute the squared error of each estimator by comparing its value to the ground truth values computed for Figure 3.5. We repeat this process 1,000 times for each $n_{\mathcal{D}}$ and record the *mean squared error* (MSE) of each estimator. We then produce a plot where the horizontal axis is the

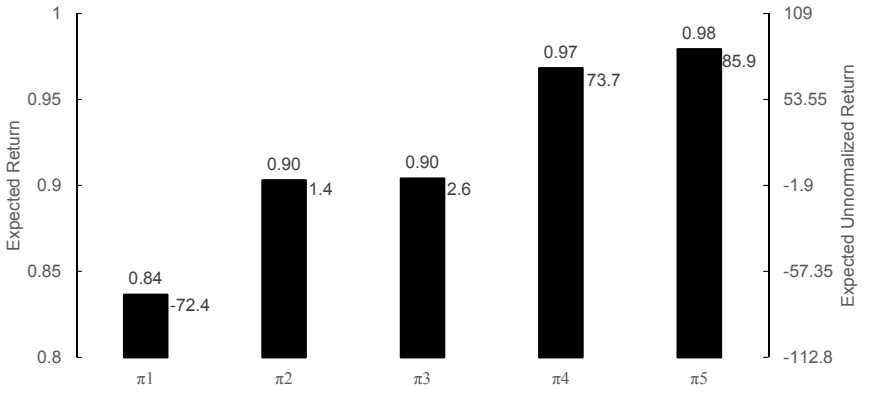


**Figure 3.5.** Expected return (left vertical axis) and expected unnormalized return (right vertical axis) for each of the five policies that we consider. These values were computed for each policy as the average (on-policy) sample return from 100,000 trajectories. We treat these values as ground truth (standard error bars are too small to be visible).

number of trajectories used, $n_{\mathcal{D}}$, and where we the vertical axis is the mean squared error.

We include only the upper standard deviation error bars because the plots use logarithmic scales for the vertical (and also horizontal) axis. The lower error bars would almost always go to below zero, i.e., they would go off the bottom of the plot, which would cause unnecessary clutter.

First we used $\pi 1$ for both the evaluation and behavior policies. In the on-policy setting the importance weights are always one, and so every method degenerates to computing the mean sample return. They all therefore have the same MSE and the same error bars. These results are depicted in Figure 3.6.

Next, consider Figure 3.7, which uses $\pi_b = \pi 1$ and $\pi_e = \pi 2$—data from the random policy to evaluate a significantly better policy. As expected, IS performs the worst. PDIS provides a significant improvement over IS, but it will not be practical for our later use, as discussed previously. NPDIS, the variant of PDIS that will be applicable later, performs much more similarly to IS than PDIS. Although not clear due to the logarithmic scale, NPDIS has 72% the MSE of IS, on average. The closest it comes to

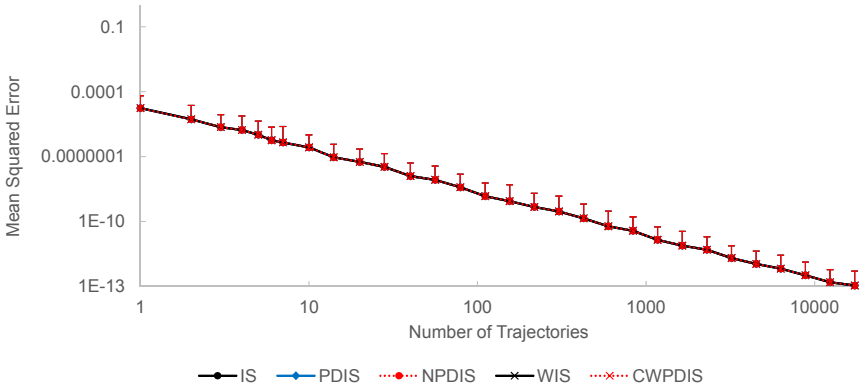


**Figure 3.6.** Behavior policy = $\pi 1$ and evaluation policy = $\pi 1$.

the IS curve is 80% the MSE of IS. WIS performs exceptionally well, with much lower MSE than IS, PDIS, and NPDIS. However, recall that it is not an unbiased estimator of $\rho(\pi_e)$. Still, its lower MSE suggests that it is a better estimator of $\rho(\pi_e)$ in this case. Finally, the consistent and weighted variant of PDIS, CWPDIS, performs almost identically to WIS. These same general trends are apparent in Figure 3.8, which uses $\pi_b = \pi 2$ and $\pi_e = \pi 3$ (data from a reasonable policy to evaluate a similar but slightly better policy).

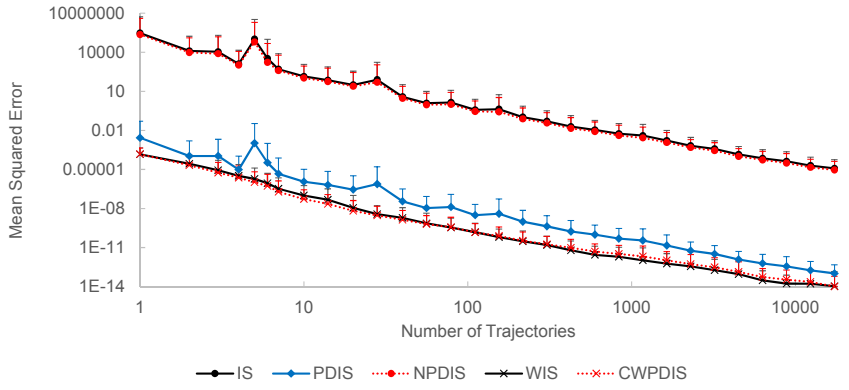


**Figure 3.7.** Behavior policy = $\pi 1$ and evaluation policy = $\pi 2$.

However, Figure 3.9, which uses $\pi_b = \pi 3$ and $\pi_e = \pi 4$ is quite different. To see what is going on, consider the action probabilities in position $(2, 4)$ under the two policies: $\pi 3(\text{down}|(2,4)) = 0.340$, $\pi 3(\text{right}|(2,4)) = 0.463$, $\pi 4(\text{down}|(2,4)) = 0.993$,

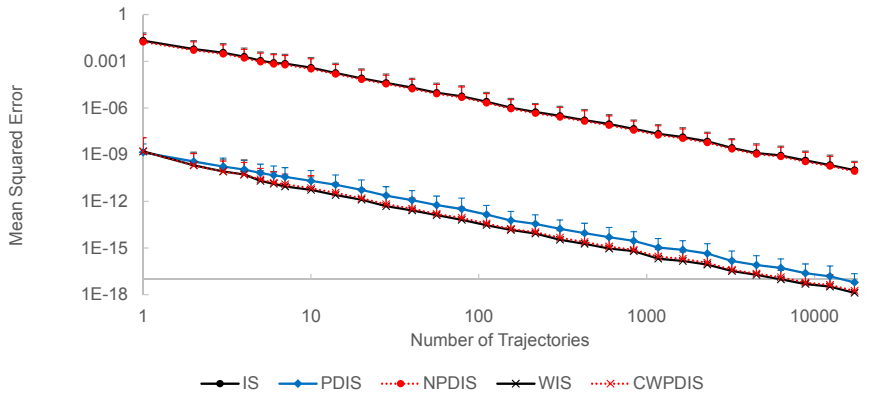


**Figure 3.8.** Behavior policy = $\pi 2$ and evaluation policy = $\pi 3$.

and $\pi 4(\text{right}|(2,4)) = 0.006$. Trajectories that are likely under $\pi 4$ include multiple selections of the down action in position $(2,4)$ to accrue positive rewards. However, it is unlikely that a trajectory generated by $\pi 3$ will select the down action many times in position $(2,4)$. So, trajectories that are likely under $\pi 4$ are unlikely under $\pi 3$.

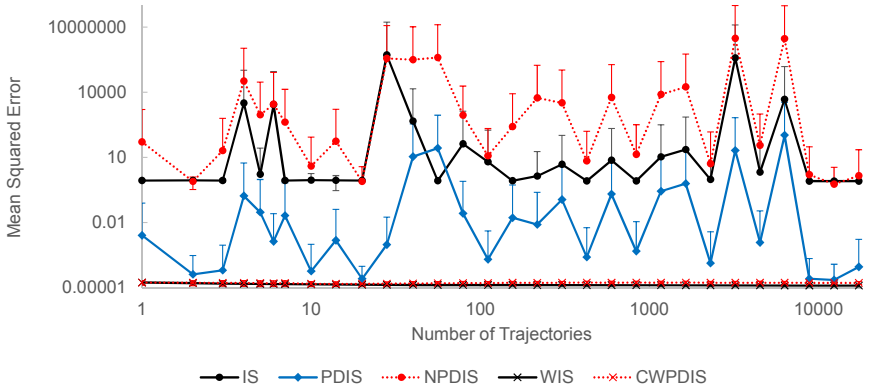


**Figure 3.9.** Behavior policy = $\pi 3$ and evaluation policy = $\pi 4$.

Moreover, consider one particular trajectory, $h_L$, which results from the action sequence down, down, down, right, followed by down until the episode terminates due to time running out. We computed that $\Pr(H_L = h_L|\pi 3) \approx 1.4 \times 10^{-47}$ and $\Pr(H_L = h_L|\pi 4) \approx 0.5$. That is, under $\pi 4$, this trajectory occurs 50% of the time, yet under $\pi 3$ it almost never occurs. The importance weight for this trajectory is

approximately $\frac{0.5}{1.4\times10^{-47}} = 7 \times 10^{46}$. Its return is approximately 0.98, making the importance weighted return for this trajectory $\hat{\rho}^{\text{IS}}(\pi_e|h_L, \pi_b) \approx 6.87 \times 10^{46}$.

Consider the expected value of $\hat{\rho}^{\text{IS}}(\pi 4|H_L, \pi 3)$:

$$\begin{aligned}\mathbf{E}\left[\hat{\rho}^{\text{IS}}(\pi 4|H_L, \pi 3)\middle|H_L \sim \pi 3\right] &= \sum_{h_L \in \mathcal{H}_L} \Pr(H_L = h_L|\pi 3)\hat{\rho}^{\text{IS}}(\pi 3|h_L, \pi 4) \\ &\approx 0.97.\end{aligned}$$

Notice that the trajectory that we have described contributes

$$\Pr(H_L = h_L|\pi 3)\hat{\rho}^{\text{IS}}(\pi 3|h_L, \pi 4) \approx 0.5$$

to the summation—more than half of the expected value comes from the magnitude of the importance weighted return associated with the single exceedingly unlikely trajectory that we described earlier. So, if that trajectory is not sampled (as it never was), then the IS estimator will remain a significant underestimate of $\rho(\pi_e)$.

The impact that this has is clear in Figure 3.10, which shows the actual estimates of $\rho(\pi 4)$ as more and more trajectories are appended to $\mathcal{D}$ (e.g., the estimate using 2 trajectories uses the first trajectory plus one new trajectory). This plot shows only one trial—there is no averaging (the expected value of the unbiased estimators would be $\rho(\pi 4)$). Initially, trajectories that are likely under $\pi 3$ but unlikely under $\pi 4$ are sampled and the IS estimator is near zero. Eventually a trajectory is sampled that is both unlikely under $\pi 3$ and likely under $\pi 4$. This trajectory has a very large importance weight and a large return and results in a large jump up in the IS estimator. More trajectories that are likely under $\pi 3$ are then sampled and the IS estimator slowly decays back down until another unlikely trajectory with a large importance weighted return is sampled and the IS estimator jumps back up again. This process repeats.

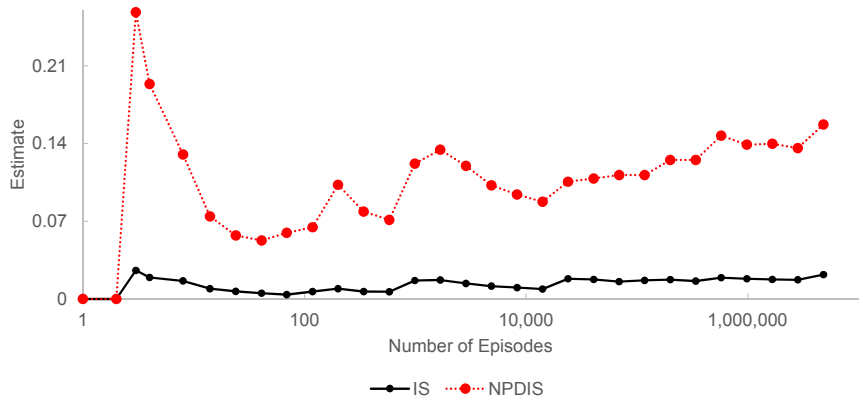


**Figure 3.10.** Actual estimates from the IS and NPDIS estimators using $\pi_b = \pi 3$ and $\pi_e = \pi 4$.

The same general property is true of the NPDIS estimator, as evidenced by Figure 3.10. However, consider what happens if the beginning of a trajectory is likely under $\pi 4$ but the end of the trajectory is very unlikely under $\pi 4$. Overall, the trajectory is unlikely and so the IS estimator will give it a small importance weight and it will result in an underestimate of $\rho(\pi 4)$. However, the NPDIS estimator uses the probability of just the beginning of the trajectory when estimating the expected early rewards in the trajectory, and so it can produce large estimates for individual rewards in the trajectory. This means that the NPDIS estimator can increase its estimate if a trajectory begins in a way that is likely under $\pi 4$, while IS requires the entire trajectory to be likely under $\pi 4$. This is evident in Figure 3.10 since the NPDIS estimator increases its value before the IS estimator does (the curves for the IS and NPDIS estimator were computed using the same sets of trajectories).

Next consider what is happening with the PDIS estimator when using $\pi_b = \pi 3$ and $\pi_e = \pi 4$. Consider a trajectory that is likely under $\pi 3$ but unlikely under $\pi 4$, which results in a small importance weight for most rewards (recall that PDIS uses importance weights for each individual reward rather than one importance weight for the entire return). The individual rewards can be positive or negative, and so the small importance weight can decrease a positive reward to create an underestimate

like in IS, *or* it could increase a negative reward to create an overestimate. As a result, PDIS does not have as strong of a tendency to underestimate $\rho(\pi 4)$ as IS and NPDIS do. In this case (this is not always a property of PDIS), it tends to start closer to $\rho(\pi 3) \approx 0.90$ and slowly works its way up to $\rho(\pi 4) \approx 0.97$, as shown in Figure 3.11, which is identical to Figure 3.10 but for PDIS, WIS, and CWPDIS.

Finally consider what happens with the WIS and CWPDIS estimators when using $\pi_b = \pi 3$ and $\pi_e = \pi 4$. Recall from the discussion of WIS that the weighted methods, when using a single sample, are an unbiased estimator of $\rho(\pi 3)$ (since the importance weights cancel out). So, the WIS and CWPDIS estimators begin as estimates of $\rho(\pi 3)$ and increase up to $\rho(\pi 4)$ as the number of trajectories increases. Notice that even after 4 million trajectories (the span of the plot) none of the estimators (PDIS, WIS, or CWPDIS) has made much progress towards $\rho(\pi 4) \approx 0.97$. This is because the trajectories that result in the largest importance weighted returns are exceptionally unlikely (recall the most important trajectory has probability $1.4 \times 10^{-47}$ of occurring, which means that it is unlikely to occur in only 4 million trajectories).

Next we used $\pi_b = \pi 4$ and $\pi_e = \pi 5$, the results of which are depicted in Figure 3.12. This setting is different from the others because the behavior policy results in long trajectories (usually they take the full 100 time steps before transitioning to the absorbing state). So, whereas before the bulk of the return was obtained at the end of the episode (the reward of +10 for reaching the terminal state), in this case the rewards are fairly evenly spread across the entire trajectory. This means that NPDIS and CWPDIS perform much better than IS and WIS, respectively. That is, the MSE of WIS averages four times the MSE of CWPDIS while the MSE of IS averages 15 times the MSE of NPDIS.

It is important to notice that overall performance is good for all estimators with $\pi_b = \pi 4$ and $\pi_e = \pi 5$. That is, even though the behavior policy is nearly deterministic, the estimators still perform well. This is because, for every $o$ and $a$ where $\pi 4(a|o)$ is

near one, $\pi 5(a|o) > \pi 4(a|o)$. That is, $\pi 5$ is also nearly deterministic in the same way as $\pi 4$. So, the trajectory most needed to evaluate $\pi 5$ is likely under $\pi 4$. Furthermore, the ratio $\pi 5(a|o)/\pi 4(a|o)$ is not too large regardless of whether the most likely action was taken or not.

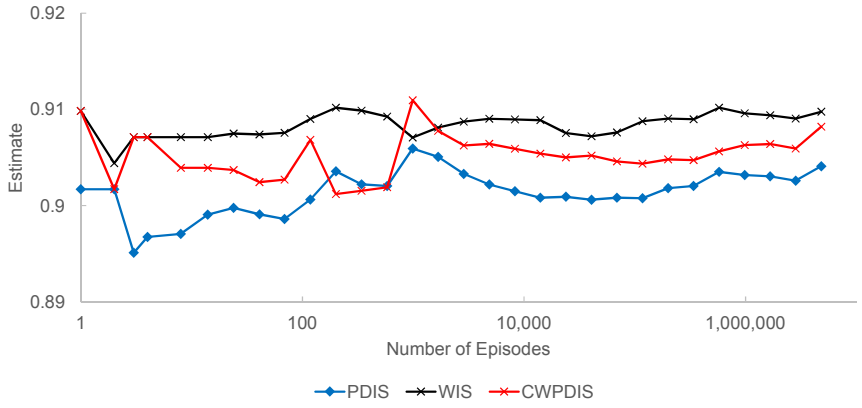


**Figure 3.11.** Actual estimates from the PDIS, WIS, and CWPDIS estimators using $\pi_b = \pi 3$ and $\pi_e = \pi 4$.

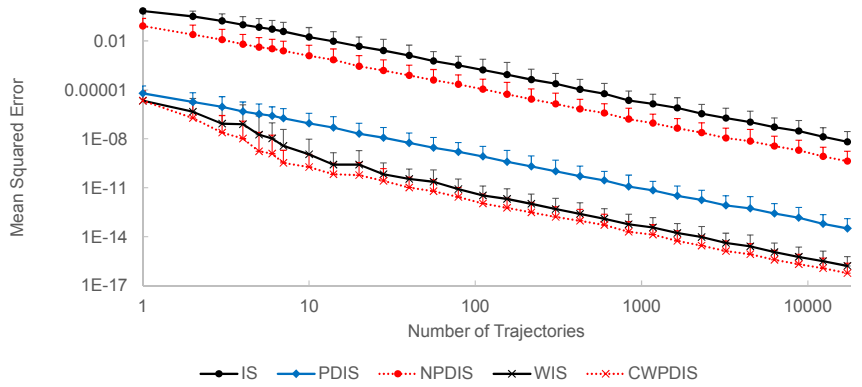


**Figure 3.12.** Behavior policy = $\pi 4$ and evaluation policy = $\pi 5$.

So far we have always used evaluation policies that are better than the behavior policies. We now consider the opposite setting, where the behavior policy is better than the evaluation policy. First, we used $\pi_b = \pi 2$ and $\pi_e = \pi 1$, the result of which is depicted in Figure 3.13. The estimators all perform poorly, although not as poorly as when $\pi_b = \pi 3$ and $\pi_e = \pi 4$. However, the reason for the poor performance

is essentially the same—many of the trajectories required to evaluate $\pi 1$ have low probability under $\pi 2$.

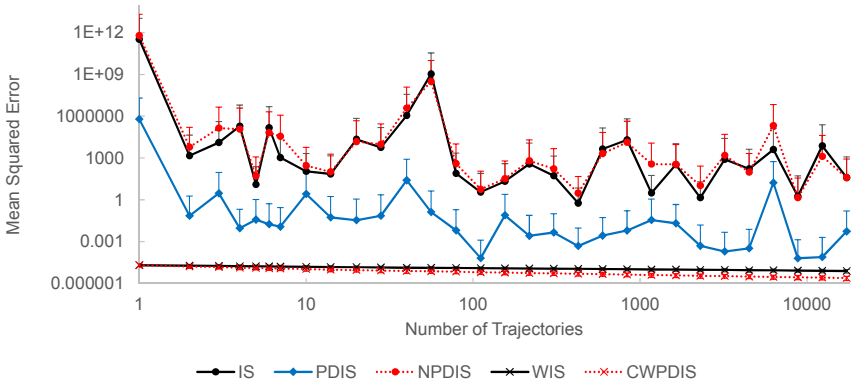


**Figure 3.13.** Behavior policy = $\pi 2$ and evaluation policy = $\pi 1$.

This is not a problem when using $\pi_b = \pi 3$ and $\pi_e = \pi 2$, as shown in Figure 3.14. In this case all estimators perform well, with the same general ordering of their performance.

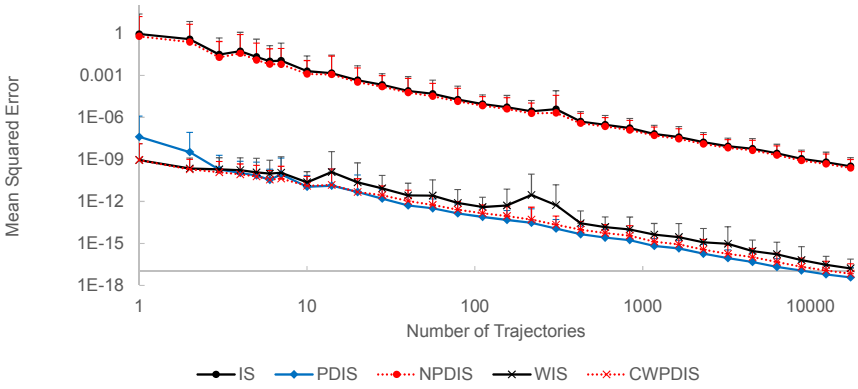


**Figure 3.14.** Behavior policy = $\pi 3$ and evaluation policy = $\pi 2$.

The problems arise again when using $\pi_b = \pi 4$ and $\pi_e = \pi 3$, although not quite as severely. This experiment is depicted in Figure 3.15. The problems are not as severe because $\pi 3$ is not as deterministic as $\pi 4$, and so there isn't one single trajectory that dominates $\mathbf{E}[\hat{\rho}^{\text{IS}}(\pi 3|H_L, \pi 4)|H_L \sim \pi 4]$. Still, when $\pi 4$ deviates only a few times within a trajectory from its usual action choices, the probability of the trajectory can

become very small, which results in a large importance weight. The sensitivity of the unweighted importance sampling methods (IS, PDIS, NPDIS) to these outliers is evident, as is the stability of WIS and CWPDIS.

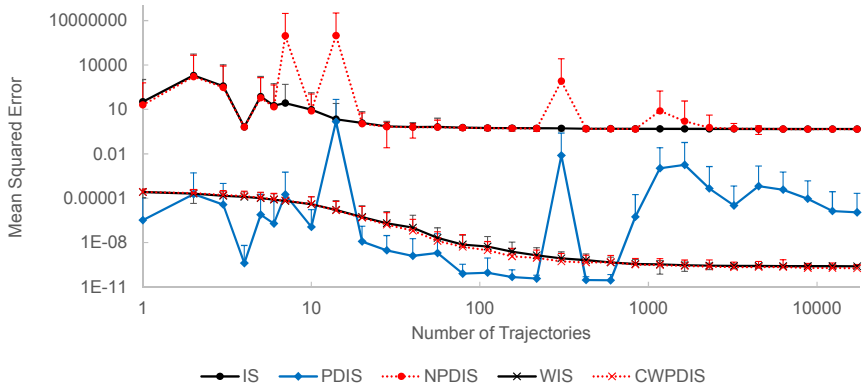


**Figure 3.15.** Behavior policy = $\pi 4$ and evaluation policy = $\pi 3$.

Lastly, we used $\pi_b = \pi 5$ and $\pi_e = \pi 4$, the result of which is depicted in Figure 3.16. This results in the same problem that we saw when using $\pi_b = \pi 4$ and $\pi_e = \pi 3$—when the behavior policy deviates only a few times within a trajectory from its usual action choices, the probability of the trajectory becomes very small, which results in a large importance weight. In this case this is even more extreme and so, for example, CWPDIS finishes with 1/4 the MSE that it had originally (for comparison, in Figure 3.12 it finished with $2.85 \times 10^{-11}$ the MSE it began with).

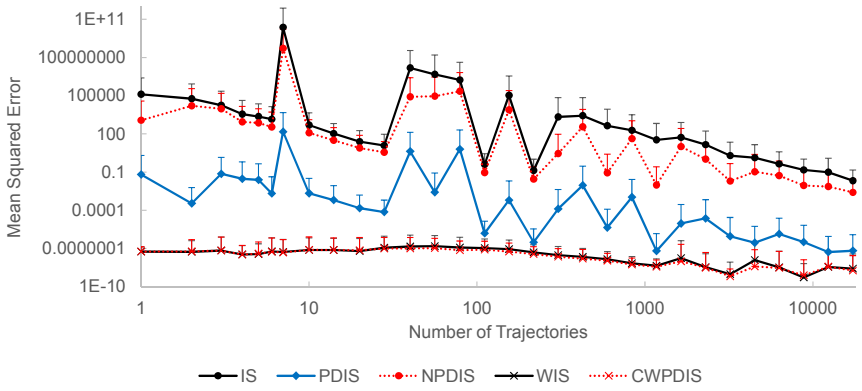


**Figure 3.16.** Behavior policy = $\pi 5$ and evaluation policy = $\pi 4$.

## 3.13 Discussion and Conclusion

We have reviewed the IS, PDIS, WIS, and WPDIS estimators, and derived the NPDIS and CWPDIS estimators. It was known that IS, PDIS, and WIS are consistent estimators if a single behavior policy is used. We showed that IS, PDIS, WIS, NPDIS, and CWPDIS are all consistent estimators if a single behavior policy is used or if multiple behavior policies that meet a mild restriction are used. All of these estimators typically require that the support of the evaluation policy be a subset of the support of the behavior policy (Assumption 1). We showed that, even if this assumption does not hold, the expected values of the IS and PDIS estimators will never be above $\rho(\pi_e)$, which means that they will still be useful for producing lower bounds on $\rho(\pi_e)$ in subsequent chapters.

Our empirical results were as expected. IS performs the worst in almost every case. PDIS performs significantly better—often on par with the weighted methods (WIS, CWPDIS). However, the PDIS estimator is not bounded below by zero, and so it will be of little use to us in the next chapter. To remedy this, we derived the NPDIS estimator, which typically resulted in MSE 1.5 to 15 times lower than the MSE of the IS estimator. However, this improvement is minor in comparison to the reduction in MSE produced by WIS.

We found that WIS was much less sensitive to unlikely trajectories and typically had several orders lower MSE than IS and NPDIS. However, we were able to further improve upon WIS with CWPDIS—a weighted variant of PDIS—particularly when the trajectories are long so that the bulk of the return is not concentrated towards the end of the episode. Because it can be challenging to determine which estimator is best to use for each situation, we provide a decision diagram in Figure 3.17.

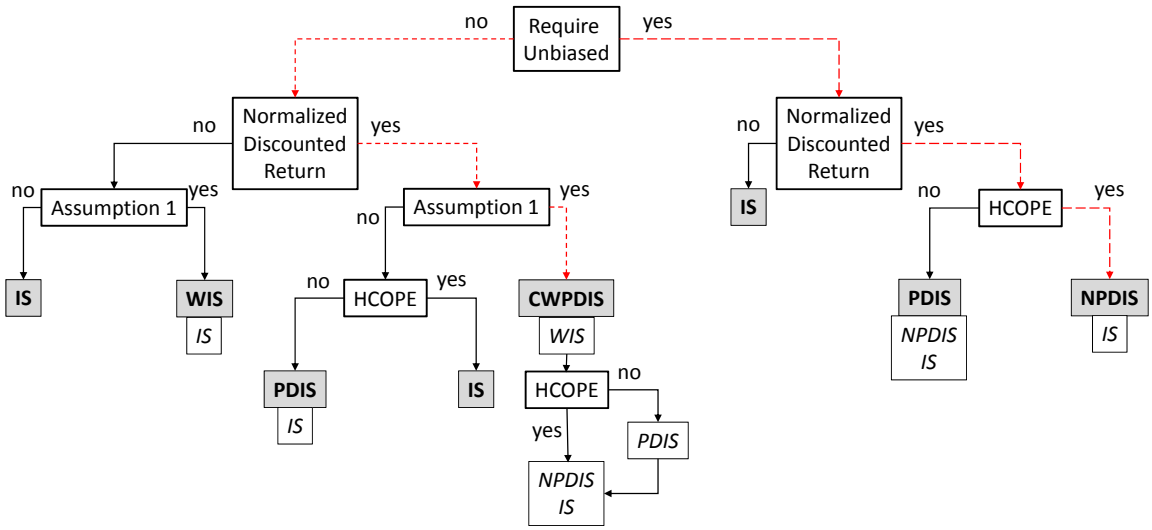


**Figure 3.17.** Decision diagram for deciding which importance sampling variant to use. The recommended method is presented in a gray-filled box in bold. The other applicable methods are listed in order of preference thereunder (in italics). The top node corresponds to whether or not an unbiased estimator is required. The second level nodes, "normalized discounted return," correspond to whether $R$ is defined to be the normalized discounted return (see (2.5)). The decision nodes labeled "Assumption 1" are asking whether Assumption 1 holds. Even if it does not, you may select the answer "yes" if you are not concerned with error introduced by this false assumption. The "HCOPE" decision node denotes whether or not the estimator will be used for HCOPE (the topic of the next chapter). If it will be, then the estimator should be lower bounded by a number close to $\rho(\pi_e)$ (which is not the case with PDIS). The dotted red paths are the two that will be used in the subsequent chapters—they use CWPDIS when unbiased estimates are not required and NPDIS when they are.

# CHAPTER 4

# HIGH CONFIDENCE OFF-POLICY EVALUATION

In many potential applications of RL algorithms, some current policy (often chosen by a domain expert) has already been deployed. An engineer or researcher, who we refer to as the *user*, may attempt to improve upon the current policy using an RL method and historical data collected from the deployment of the current policy. After selecting and applying any of the many applicable RL algorithms to obtain a newly proposed policy, the user is faced with a problem: should the newly proposed policy actually be deployed?

RL algorithms often require careful tuning of hyperparameters like step sizes, which results in the common impression that RL algorithms are not black boxes that "just work," but rather brittle tools that require expert knowledge and luck to apply successfully. Due to this reputation, whether deserved or not, the policies proposed by RL algorithms are often *not* deployed, especially for applications where deployment of a bad policy can be costly or dangerous.

**In this chapter we present a method for evaluating a newly proposed policy using historical data, which can instill the user with confidence that the newly proposed policy will perform well if actually deployed.** Importantly, this evaluation can be *off-policy*—the newly proposed policy can be evaluated without ever actually being deployed and using historical data from any number of previous policies. Also, this chapter is agnostic to what RL algorithm was used to compute the newly proposed policy—the methods presented here can be used in conjunction with any RL algorithm.

Many *off-policy evaluation* methods use historical data to predict the expected performance of any policy. However, they do not provide confidence bounds to quantify how accurate their estimates are. This is troublesome because off-policy evaluation methods are known to produce estimates that have high variance. It is therefore reasonable for the user to be skeptical of performance predictions made by existing off-policy evaluation methods.

To overcome this, our evaluation method, which we call a *high confidence off-policy evaluation* (HCOPE) method, produces a high confidence lower bound on the performance of the newly proposed policy rather than an estimate of its performance. For example, if the performance of the currently deployed policy is known, then our method can produce statements of the form "with confidence 95%, the newly proposed policy will perform at least as well as the current policy."

This chapter is organized as follows. In Section 4.1 we formalize the goal of this chapter before discussing related work in Section 4.2. We then review *concentration inequalities* in Sections 4.3, 4.4, and 4.5. In Section 4.6 we present our general approach. In Section 4.7 we describe a related method in detail. We then derive a concentration inequality in Section 4.8 that makes our approach feasible using less historical data than earlier methods. We provide pseudocode for the resulting methods in Section 4.9 before presenting experiments in Section 4.10 and concluding in Section 4.11.

The primary contribution of this chapter is a concentration inequality that is well suited to HCOPE. Other minor contributions include the bringing together of relevant state-of-the-art methods (both concentration inequalities and variants of importance sampling) that can be leveraged for HCOPE, and the proposal of *risk quantification plots*, which quantify the risk associated with deploying a new policy.

## 4.1 Problem Description

We assume that we are given a data set, $\mathcal{D}$, that consists of $n_{\mathcal{D}}$ trajectories, $\{H_L^i\}_{i=1}^{n_{\mathcal{D}}}$, each labeled by the policy that generated it, $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$, i.e.,

$$\mathcal{D} = \left\{ \left(H_L^i, \pi_i\right) : i \in \{1, \ldots, n_{\mathcal{D}}\}, H_L^i \sim \pi_i \right\}. \tag{4.1}$$

Note that $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$ are *behavior policies*—those that generated the batch of data (trajectories). Finally, we denote by $\pi_e$ the *evaluation policy*—the newly proposed policy that should be evaluated using the data set $\mathcal{D}$. Although some trajectories in $\mathcal{D}$ may have been generated using the evaluation policy, we are particularly interested in the setting where some or all of the behavior policies are different from the evaluation policy.

**Our goal is to construct an algorithm that takes as input the historical data, $\mathcal{D}$, an evaluation policy, $\pi_e$, a confidence level, $\delta$, and produces a $1-\delta$ confidence lower bound on $\rho(\pi_e)$.** We refer to this algorithm as an HCOPE algorithm. Formally, the algorithm is a function $\Lambda$, where the lower bound that it produces given $\pi_e, \mathcal{D}$, and $\delta$ is $\Lambda(\pi_e|\mathcal{D}, \delta)$. A good HCOPE algorithm is one that produces tight lower bounds on $\rho(\pi_e)$, i.e., where $\Lambda(\pi_e|\mathcal{D}, \delta)$ is large.

It is important to understand which of these quantities are fixed and which are random. We assume that $\delta$ and $\pi_e$ are fixed (not random variables). We also assume that the behavior policies, $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$, are fixed. However, the trajectories, $\{H_L^i\}_{i=1}^{n_{\mathcal{D}}}$ are random variables—each $H_L^i$ can be sampled by generating a trajectory using the policy $\pi_i$. So, $\mathcal{D}$ is a random variable and therefore $\Lambda(\pi_e|\mathcal{D}, \delta)$ is as well.

When we say that $\Lambda(\pi_e|\mathcal{D}, \delta)$ should be a $1-\delta$ confidence lower bound on $\rho(\pi_e)$, we mean that the following should hold:

$$\Pr\Big(\rho(\pi_e) \geq \Lambda(\pi_e|\mathcal{D}, \delta)\Big) \geq 1 - \delta. \tag{4.2}$$

The distinction here between probabilities and confidences is crucial in order to properly interpret the output of an HCOPE algorithm. Let $\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}$ be a sample of $\mathcal{D}$ so that $\Lambda(\pi_e|\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}, \delta)$ is a sample of $\Lambda(\pi_e|\mathcal{D}, \delta)$. A common misunderstanding that we have encountered when presenting this work arises because it is *not* correct to say that

$$\Pr\left(\rho(\pi_e) \geq \Lambda(\pi_e|\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}, \delta)\right) \geq 1 - \delta, \tag{4.3}$$

since there is no randomness in $\rho(\pi_e) \geq \Lambda(\pi_e|\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}, \delta)$—it is either true or false. This is a subtle but crucial point. Consider a motivating example where an executive assigns a researcher to compute a 95% confidence lower bound ($\delta = 0.05$) on the performance of a new policy, $\pi_e$, using historical data, $\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}$. The researcher uses an HCOPE method to compute $\Lambda(\pi_e|\{(h_L^i, \pi_i)\}_{i=1}^{n_\mathcal{D}}, 0.05) = 0.8$. It would be incorrect for the researcher to say that the new policy's performance will be at least 0.8 with probability 0.95 (this is English for (4.3), which we have established is incorrect). Instead, the researcher should say that 0.8 is a 95% confidence lower bound on the new policy's performance (this is English for (4.2)).

We present two different approaches to HCOPE—exact and approximate methods. *Exact HCOPE* methods should provide the guarantee specified above without requiring any additional (possibly false) assumptions. The benefit of exact HCOPE methods is that they provide a real guarantee about the performance of a new policy if the environment fits our formalization of POMDPs. The drawback of exact HCOPE methods is that they require more historical data (trajectories) than approximate HCOPE methods to produce tight bounds on the performance of the evaluation policy.

*Approximate HCOPE* methods can use approximations and additional (possibly false) assumptions that can help improve data efficiency. For example, they may assume that data that are approximately normally distributed are actually normally

distributed. The benefit of approximate HCOPE algorithms over exact HCOPE algorithms is that they can provide tight confidence bounds on the performance of the evaluation policy using much less historical data. The drawback of approximate HCOPE methods is that, even if the environment fits the POMDP formalization that we use, the bounds that they produce are only approximate bounds—the error rate will be *approximately* $\delta$, rather than upper bounded by $\delta$.

## 4.2 Related Work

In this section we review related off-policy evaluation methods as well as the two existing approaches to HCOPE of which we are aware.

### 4.2.1 Off-Policy Evaluation

Whereas our HCOPE approach provides a high confidence lower bound on the performance of an evaluation policy using historical data, off-policy evaluation methods attempt to estimate the performance of the evaluation policy but do not provide any confidence bounds on their estimates. Several methods for off-policy evaluation based on importance sampling were reviewed in detail in the previous chapter.

Thomas et al. (2015a) propose an off-policy evaluation method, which they call *complex weighted importance sampling* (CWIS), that generalizes the *ECR estimator* (Tetreault and Littman, 2006). They show that their method, which is founded on fewer typically false assumptions than previous methods, produces an estimator with lower mean squared error than existing methods, including IS and WIS, across several examples (Thomas et al., 2015a). The primary drawbacks of their approach are that it is computationally inefficient and that it relies on the availability of good features for *linear value function approximation* (Sutton and Barto, 1998).

While this chapter is primarily concerned with bounding the performance of a policy, much work in off-policy evaluation has focused on how to estimate the expected

return if the episode where to deterministically start from a known state. The target function of this approach is called the *value function* associated with a policy, $\pi$, and is denoted by $v^\pi$. Formally,

$$v^\pi(s) \coloneqq \mathbf{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \middle| \pi, S_1 = s\right],$$

where $\gamma \in [0, 1]$. The problem of estimating $v^{\pi_e}$ given an off-policy data set, $\mathcal{D}$, is particularly challenging if the state set is large or continuous so that a value, $\hat{v}^{\pi_e}(s)$, cannot be stored for each $s \in \mathcal{S}$. In this setting, a parameterized estimator, $\hat{v}^{\pi_e}_w$, is often used, where $w$ is a vector of weights. It was not until 2009 that a convergent method was developed for estimating $v^{\pi_e}$ with $\hat{v}^{\pi_e}_w$ using off-policy data (Sutton et al., 2009). This convergent method, the *gradient temporal-difference* (GTD) method, was later improved so that its *backwards implementation* (Sutton and Barto, 1998) is exactly equivalent to its forward implementation (van Hasselt et al., 2014), and not just approximately so (Bertsekas and Tsitsiklis, 1996, page 198).

Another promising avenue of off-policy evaluation research involves constructing models of the transition dynamics (and of the reward function if it is not known). The hope of these methods is that an accurate model can be computed using fewer samples than would be required to generate an accurate off-policy estimate of the value function without constructing a model. This model can then be used to estimate the value of any policy (Mannor et al., 2007). This approach can be particularly powerful if some structure of the environment is known *a priori* (Hallak et al., 2015).

### 4.2.2 Other Methods for High-Confidence Off-Policy Evaluation

We are aware of two previous attempts at providing practical confidence bounds on the estimates produced by off-policy evaluation methods (Bottou et al., 2013, Mannor et al., 2007). Although working independently, Bottou et al. took an approach quite similar to ours. The convergence of independent efforts to similar solutions is

encouraging. During the derivation of our approach we discuss where our approach branches from theirs. The drawbacks of their approach relative to ours are that their approach is only approximate (it does not provide a true guarantee, whereas our approach does), it discards data (our approach does not), it is only applicable when there is a single behavior policy (our approach can work with many behavior policies), and it uses a simple heuristic for setting an important hyperparameter that we automatically optimize from the data (a threshold, $c$). In our experiments we compare to their approach and show that, in practice, our approach can produce tighter exact confidence bounds than their approximate confidence bounds.

Mannor et al. (2007) proposed a model-based approach to HCOPE wherein the historical data is used to estimate the transition function and reward function. The performance of any evaluation policy can then be estimated by treating the approximations of the transition and reward functions as though they are exact. Mannor et al. (2007) provide upper bounds on the bias and variance of this off-policy evaluation approach. However, their approach has three drawbacks relative to ours:

1. It is derived for the MDP setting, not the full POMDP setting.

2. It assumes that the state and action sets are finite.

3. The bounds on the bias and variance of their estimator are expressed in terms of the true transition function and reward function. If the empirical estimates are used in place of the true transition and reward functions in the bounds, then their method is only an approximate HCOPE method.

### 4.2.3 Finite-Sample Bounds for Off-Policy Evaluation

Recent work has shown that the GTD algorithm (mentioned in Section 4.2.1) and other related algorithms are true stochastic gradient algorithms for primal-dual saddle-point objective functions (Liu et al., 2015). This new understanding of off-

policy evaluation methods allows for finite-sample bounds on the error of value function approximations. These finite-sample bounds quantify how the off-policy value function estimates produced by GTD differ from the true value function as the number of iterations of GTD increases. This is much like how the Chernoff-Hoeffding inequality (discussed in the following section) quantifies how much the sample mean of some random variables differs from their true mean as the number of random variables increases. For example, Liu et al. (2015, Proposition 5) show that after $n$ iterations of GTD, with probability at least $1 - \delta$ (using their notation, and given a few assumptions):

$$\|V - \bar{v}_n\|_\xi \leq \frac{1 + \gamma\sqrt{\rho_{\max}}}{1 - \gamma}\|V - \Pi V\|_\xi + \sqrt{\frac{2\tau_C \tau \xi_{\max}}{\sigma_{\min}(A^\intercal M^{-1} A)} \operatorname{Err}(\bar{\theta}_n, \bar{y}_n)}. \tag{4.4}$$

The left side of (4.4) is a measure of the error in the value function estimate produced by GTD, and the right side of (4.4) is the finite-sample bound, which depends on many (typically unknown) properties of both the MDP and the choice of function approximator. One approach to HCOPE that could be fruitful, would be to produce high-confidence bounds on all of the unknown terms on the right side of this equation. These bounds, together with (4.4), could be used to produce high-confidence off-policy bounds on the performance of a policy (by running GTD or a similar algorithm and using these bounds to bound the error in the resulting estimate of the value of the initial state).

Instead of this approach, we propose the use of *concentration inequalities* to bound the error of the estimates produced by the unbiased importance sampling estimators described in the previous chapter. The benefit of this approach is that it does not rely on bounding many unknown properties of the problem at hand. This not only makes our approach more simple, but it avoids the over conservativeness associated with combining several confidence bounds using the union bound. Still, the approach

based on finite-sample bounds that we have outlined here could be a promising avenue of future work.

## 4.3 Exact Concentration Inequalities

As discussed in the previous chapter, we can use IS, PDIS, and NPDIS to generate an unbiased estimator of $\rho(\pi)$ from each trajectory in $\mathcal{D}$. That is, for each $i \in \{1, \ldots, n_{\mathcal{D}}\}$ and $\dagger \in \{\text{IS, PDIS, NPDIS}\}$:

$$\mathbf{E}[\hat{\rho}^{\dagger}(\pi_e | H_L^i, \pi_i)] = \rho(\pi_e).$$

The remaining challenge in HCOPE is to use $n_{\mathcal{D}}$ such estimators to produce a high confidence lower bound on $\rho(\pi_e)$. This is exactly what *concentration inequalities* (CIs) do—they use samples of a random variable to provide probabilistic statements about how the probability mass or density of the random variable is concentrated.

Let $\{X_i\}_{i=1}^n$ be $n$ independent random variables. A common form of CIs (a form which all CIs discussed in this dissertation can be written in) is:

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq f(X_1, \ldots, X_n, \delta)\right) \geq 1 - \delta, \tag{4.5}$$

for any $\delta \in [0, 1]$, where $f$ is some function that ensures that the inequality holds (each concentration inequality that we discuss has a different $f$, and we cite a proof that each such $f$ causes (4.5) to hold). Different CIs will have different additional restrictions on $\{X_i\}_{i=1}^n$, such as that they are bounded or identically distributed. CIs often, but not always, use $f$ of the form:

$$f(X_1, \ldots, X_n, \delta) = \frac{1}{n}\sum_{i=1}^{n} X_i - g(X_1, \ldots, X_n, \delta),$$

where $g(X_1, \ldots, X_n, \delta) \geq 0$ and $\lim_{n\to\infty} g(X_1, \ldots, X_n, \delta) = 0$ (not every such $g$ will cause (4.5) to hold, but some do). This means that the concentration inequality states that

$$\Pr\left(\underbrace{\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right]}_{\text{true mean}} \geq \underbrace{\frac{1}{n}\sum_{i=1}^{n} X_i}_{\text{sample mean}} - \underbrace{g(X_1, \ldots, X_n, \delta)}_{\text{positive terms that go to zero as } n \to \infty}\right) \geq 1 - \delta,$$

for some specific $g$. However, not all concentration inequalities are of this form.

For our application, each $X_i$ is an importance weighted return: $X_i \coloneqq \hat{\rho}^{\dagger}(\pi_e|H_L^i, \pi_i)$. We have that

$$\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} \hat{\rho}^{\dagger}(\pi_e|H_L^i, \pi_i)\right] = \rho(\pi_e),$$

if Assumption 1 holds and $\dagger \in \{\text{IS, PDIS, NPDIS}\}$, so the concentration inequality can be written as:

$$\Pr\left(\rho(\pi_e) \geq f(\hat{\rho}^{\dagger}(\pi_e|H_{L1}, \pi_1), \ldots, \hat{\rho}^{\dagger}(\pi_e|H_{Ln}, \pi_n), \delta)\right) \geq 1 - \delta,$$

which means that $f(\hat{\rho}^{\dagger}(\pi_e|{H_L}^1, \pi_1), \ldots, \hat{\rho}^{\dagger}(\pi_e|{H_L}^n, \pi_n), \delta)$ is a $1 - \delta$ confidence lower bound on $\rho(\pi_e)$. Notice that this requires $\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)$ to be an unbiased estimator of $\rho(\pi_e)$ for any $\pi_e, \pi_b$, and $H_L \sim \pi_b$. Since the WIS and CWPDIS estimators are not unbiased estimators, this means that their use results in approximate HCOPE methods. Furthermore, the WIS and CWPDIS estimators have only been defined in their batch form—there is no obvious way to define $\hat{\rho}^{\text{WIS}}(\pi_e|H_L, \pi_b)$ and $\hat{\rho}^{\text{CWPDIS}}(\pi_e|H_L, \pi_b)$. So, the early parts of this chapter will focus on methods that apply for $\dagger \in \{\text{IS, PDIS, NPDIS}\}$ (later we will explicitly discuss using WIS and CWPDIS for approximate HCOPE).

In this application, the $X_i$ (importance weighted returns) are independent because each trajectory is sampled independently. If there is only one behavior policy,

then the $X_i$ are also identically distributed because the trajectories are all produced by the same behavior policy. However, if there are multiple behavior policies, then the importance weighted returns are not necessarily identically distributed (each behavior policy induces a different distribution over trajectories, and thus a different distribution over importance weighted returns).

Many concentration inequalities depend on the range of the random variables, $X_i$. In our case, these bounds are $\hat{\rho}^{\dagger}_{\text{lb}}(\pi_e, \pi_b)$ and $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e, \pi_b)$, which are defined for each value of $\dagger$ in the previous chapter (see Table 3.1). Recall that these bounds often span a massive range.

We review several relevant concentration inequalities below. We emphasize their dependence on the range of the $X_i$, because, for our application, the range of the importance weighted returns tends to be so massive that it dominates the concentration inequalities. We also specify any additional assumptions that are made by the different concentration inequalities (e.g., that the $X_i$ are identically distributed, which would mean that the historical data must come from only one behavior policy).

The *Chernoff-Hoeffding* (CH) inequality is one of the best known. It allows for $X_i$ that are not identically distributed (multiple behavior policies). Formally:

**Theorem 17** (Chernoff-Hoeffding (CH) Inequality)**.** *Let* $\{X_i\}_{i=1}^n$ *be* $n$ *independent random variables such that* $\Pr\left(X_i \in [a_i, b_i]\right) = 1$*, for all* $i \in \{1, \ldots, n\}$*, where all* $a_i \in \mathbb{R}$ *and* $b_i \in \mathbb{R}$*. Then*

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq \frac{1}{n}\sum_{i=1}^{n} X_i - \sqrt{\frac{\ln\left(\frac{1}{\delta}\right)\sum_{i=1}^{n}(b_i - a_i)^2}{2n^2}}\right) \geq 1 - \delta. \tag{4.6}$$

*Proof.* See the work of Massart (2007). ■

To easily see the dependence on the range of the random variables, consider the case where $X_i \in [0, b]$, so $a_i = 0$ and $b_i = b$, for all $i$. Equation (4.6) can then be written as

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq \frac{1}{n}\sum_{i=1}^{n} X_i - b\sqrt{\frac{\ln\left(\frac{1}{\delta}\right)}{2n}}\right) \geq 1-\delta.$$

Since $b$ will be large in our application, the crucial relationship here is how quickly the terms that depend on $b$ go to zero as $n \to \infty$. In the CH inequality, the term is $b\sqrt{\frac{\ln(1/\delta)}{2n}} = \Theta\left(\frac{b}{\sqrt{n}}\right)$, where here $\Theta$ assumes its definition from asymptotic analysis (Cormen et al., 2009).

Notice that the only statistic of the random variables used by the CH inequality is the sample mean—the sample variance and other statistics are not used. Maurer and Pontil (2009, Theorem 11) derived an inequality, which they refer to an an *empirical Bernstein bound*, that uses both the sample mean and the sample variance. The additional information provided by the sample variance allows for a better dependence on the range of the random variables:

**Theorem 18** (Maurer and Pontil's Empirical Bernstein (MPeB) Inequality)**.** *Let* $\{X_i\}_{i=1}^n$ *be* $n$ *independent random variables such that* $\Pr(X_i \in [a,b]) = 1$*, for all* $i \in \{1,\ldots,n\}$*, where* $a \in \mathbb{R}$ *and* $b \in \mathbb{R}$*. Then*

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq \underbrace{\frac{1}{n}\sum_{i=1}^{n} X_i}_{sample\ mean} - \frac{7(b-a)\ln\left(\frac{2}{\delta}\right)}{3(n-1)} - \sqrt{\frac{2\ln\left(\frac{2}{\delta}\right)}{n}\underbrace{\frac{1}{n(n-1)}\sum_{i,j=1}^{n}\frac{(X_i-X_j)^2}{2}}_{sample\ variance}}\right) \geq 1-\delta.$$

*Proof.* See the work of Maurer and Pontil (2009), where (using their notation) we first normalize $\mathbf{X}$ and then apply their Theorem 11 with $1-\mathbf{X}$ instead of $\mathbf{X}$. ■

For easy comparison to the CH inequality, consider the case where all $X_i \in [0,b]$. In this setting

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq \frac{1}{n}\sum_{i=1}^{n} X_i - \frac{7b\ln\left(\frac{2}{\delta}\right)}{3(n-1)} - \frac{1}{n}\sqrt{\frac{\ln\left(\frac{2}{\delta}\right)}{n-1}\sum_{i,j=1}^{n}(X_i-X_j)^2}\right) \geq 1-\delta. \tag{4.7}$$

The term that depends on $b$ in the MPeB inequality is $\frac{7b\ln\left(\frac{2}{\delta}\right)}{3(n-1)} = \Theta\left(\frac{b}{n}\right)$, which goes to zero more quickly than the term in the CH inequality, which is $\Theta\left(\frac{b}{\sqrt{n}}\right)$. For applications like ours where the range is the limiting factor, this means that the MPeB inequality tends to produce tighter confidence bounds.

However, notice that the MPeB inequality requires each random variable to have the same range, whereas the CH inequality allowed for random variables with different ranges. We can trivially select $a = \min_{i\in\{1,\ldots,n\}} a_i$ and $b = \max_{i\in\{1,\ldots,n\}} b_i$ if the $X_i$ actually have different ranges. However, if these bounds are loose for some $X_i$, then the MPeB may be loose. Consider, for example, if one $X_i$ is bounded such that $X_i \in [0, 100]$ while the other $n-1$ random variables are bounded such that $X_i \in [0, 0.1]$. Let $n = 100$ and $\delta = 0.5$ and consider the terms in CH and MPeB that depend on the range (recall that we would like these terms to be as close to zero as possible). In the CH inequality the term is

$$\frac{1}{n}\sqrt{\frac{\ln\left(\frac{1}{\delta}\right)\sum_{i=1}^{n}(b_i - a_i)^2}{2n^2}} \approx 0.01.$$

However, in the MPeB inequality the term is

$$\frac{7b\ln\left(\frac{2}{\delta}\right)}{3(n-1)} \approx 8.7.$$

So, even though MPeB should scale better with the range of the random variables as $n$ grows, this example suggests that the CH inequality may perform better for small $n$ if the bounds on some $X_i$ are loose bounds on other $X_i$. In practice this means that the MPeB inequality may be a loose bound when the historical data comes from multiple behavior policies. Notice that if one tries to remedy this by normalizing each random variable prior to applying MPeB (so that they all do have the same range), then the resulting lower bound is not a lower bound on $\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right]$.

Recall that the MPeB inequality improved upon the CH inequality by taking into account additional statistics (the sample variance). Anderson (1969) proposed a concentration inequality that takes into account even more of the available information: the entire empirical cumulative distribution function. This concentration inequality had one unspecified parameter, an optimal value for which was later derived by Massart (1990) to produce the following concentration inequality:

**Theorem 19** (Anderson and Massart's (AM) Inequality)**.** *Let $\{X_i\}_{i=1}^n$ be $n$ independent and identically distributed random variables such that $X_i \geq a$, for all $i \in \{1, \ldots, n\}$, where $a \in \mathbb{R}$. Then*

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq Z_n - \sum_{i=0}^{n-1}(Z_{i+1} - Z_i)\min\left\{1, \frac{i}{n} + \sqrt{\frac{\ln(2/\delta)}{2n}}\right\}\right) \geq 1 - \delta,$$

*where $Z_0 = a$ and $\{Z_i\}_{i=1}^n$ are $\{X_i\}_{i=1}^n$, sorted such that $Z_1 \leq Z_2 \leq \ldots \leq Z_n$.*

*Proof.* This follows from the works of Anderson (1969) and Massart (1990). ■

Unlike the CH and MPeB inequalities, which hold for independent random variables, the AM inequality only holds for independent and identically distributed random variables, i.e., $\{X_i\}_{i=1}^n$ should also be identically distributed. In the context of our problem, this means that the AM bound can only be used when all of the trajectories in $\mathcal{D}$ were generated by a single behavior policy.

This drawback of the AM inequality comes with a major benefit—the AM inequality has no direct dependence on the range of the random variables. Rather, it depends on the largest *observed* value, $Z_n$. In our application, we upper bound the importance weighted returns by $\hat{\rho}^{\dagger}_{\text{ub}}(\pi_e, \pi_b)$ (see Table 3.1), which assumes that the largest action probability ratio occurs at every time step, which is exceedingly unlikely in most cases. This means that the largest observed sample of $\hat{\rho}^{\dagger}(\pi_e|H_L, \pi_b)$ will often be several orders of magnitude smaller than the upper bound on the ran-

dom variables, which makes the AM inequality significantly tighter than the other concentration inequalities.

However, if the random variables do not have heavy upper tails (their range is not large), then the AM inequality tends to produce loose lower bounds due to its inherent reliance on the Dvoretsky-Kiefer-Wolfowitz inequality (Dvoretzky et al., 1956), which is not ideal for bounding the mean (Diouf and Dufour, 2005). The general looseness of the confidence intervals produced by the AM inequality for distributions without heavy tails suggests that there may still be room for improvement in the confidence intervals that it provides for heavy tailed distributions.

Unlike the CH and MPeB inequalities, which quantify how much the sample mean can usually differ from the true mean, Bubeck et al. (2012) provide a concentration inequality that quantifies how much the sample *median of means* can usually differ from the true mean. This results in a bound that is applicable to unbounded random variables (it was developed for random variables with heavy tailed distributions that are not necessarily sub-Gaussian).

Although Bubeck et al.'s inequality does not depend directly on the range of the random variables, it requires the variances of the random variables to be known. Maurer and Pontil (2009, Theorem 10) provided a method for bounding the true variance of the random variables from samples, which can be used in place of the true variance in the inequality of Bubeck et al. (with one application of the union bound). However the bound on the variance depends on the range of the random variables, so the resulting concentration inequality still depends on the range of the random variables:

**Theorem 20** (Bubeck and Maurer's (BM) inequality)**.** *Let* $\{X_i\}_{i=1}^n$ *be* $n$ *independent and identically distributed random variables such that* $X_i \in [a, b]$*, for all* $i \in \{1, \ldots, n\}$*, where* $a \in \mathbb{R}$ *and* $b \in \mathbb{R}$*. Let* $\delta \in (0, 1]$*,* $k = \lfloor \min\{8 \ln(e^{1/8}/\delta), n/2\} \rfloor$*, and* $N = \lfloor n/k \rfloor$*. Let*

$$\hat{\mu}_1 = \frac{1}{N}\sum_{t=1}^{N} X_t$$
$$\hat{\mu}_2 = \frac{1}{N}\sum_{t=N1}^{2N} X_t$$
$$\dots$$
$$\hat{\mu}_k = \frac{1}{N}\sum_{t=(k-1)N+1}^{kN} X_t.$$

*Let $\hat{\mu}_M$ denote the median of $\hat{\mu}_1, \hat{\mu}_2, \dots, \hat{\mu}_k$ (the median of means). Then*

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] \geq \hat{\mu}_M - (b-a)\sqrt{\frac{192\left[\sqrt{\frac{1}{n(n-1)}\sum_{i,j=1}^{n}\frac{(x_i-x_j)^2}{2}} + \sqrt{\frac{2\ln(2/\delta)}{n-1}}\right]^2 \ln\left(\frac{2e^{1/8}}{\delta}\right)}{n}}\right) \geq 1-\delta. \quad (4.8)$$

*Proof.* This follows from combining the results of Bubeck et al. (2012, Lemma 2) and Maurer and Pontil (2009, Theorem 10) using the union bound (also known as Boole's inequality). ■

The terms in the BM inequality that depend on $b$ are:

$$b\sqrt{\frac{192\left[\sqrt{\frac{1}{n(n-1)}\sum_{i,j=1}^{n}\frac{(x_i-x_j)^2}{2}} + \sqrt{\frac{2\ln(2/\delta)}{n-1}}\right]^2 \ln\left(\frac{2e^{1/8}}{\delta}\right)}{n}} = \Theta\left(\frac{b}{\sqrt{n}}\right),$$

which is the same as the CH inequality. Also notice that the BM inequality requires the random variables to be identically distributed, which for our application means that the historical data from only one behavior policy can be used.

## 4.4 Approximate Concentration Inequalities

The lower bounds produced by the concentration inequalities described thus far are impressive in that they make relatively weak assumptions about the distributions of the random variables (e.g., they do not assume that the random variables are

normally distributed). However, this general applicability means that the bounds that they produce are often not as tight as the bounds that could be produced if additional information about the form of the random variables' distributions was available.

**Here we present *approximate concentration inequalities*, which provide bounds that are usually much tighter than those of the exact concentration inequalities, but which make (typically false) assumptions about the distributions of some statistics.**

First, consider the distribution of the sample mean, $\bar{X}_n \coloneqq \frac{1}{n}\sum_{i=1}^{n} X_i$, which is an unbiased estimator of $\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right]$. The *central limit theorem* (CLT) says that, as $n \to \infty$, $\bar{X}_n$ becomes normally distributed:

**Theorem 21** (Lyapunov Central Limit Theorem)**.** *Let $\{X_i\}_{i=1}^{n}$ be $n$ independent random variables drawn from distributions with means $\mu_i$ and finite variances $\sigma_i^2$, for $i \in \{1, \ldots, n\}$. Then*

$$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{d} \mathcal{N}\left(\frac{1}{n}\sum_{i=1}^{n}\mu_i, \frac{\sum_{i=1}^{n}\sigma_i^2}{n}\right),$$

*where $\mathcal{N}(\mu, \sigma^2)$ denotes a normal distribution with mean $\mu$ and standard deviation $\sigma$, and $\xrightarrow{d}$ denotes convergence in distribution (Sen and Singer, 1993).*

*Proof.* See the work of Sen and Singer (1993, Theorem 3.3.2). ■

In practice, statisticians often suggest that the CLT justifies the assumption that a sample mean like $\bar{X}_n$ is normally distributed if $n \geq 30$. In some applications of HCOPE, there may be thousands or even hundreds of thousands of trajectories (see, for example, the digital marketing example provided later), in which case the assumption that $\bar{X}_n$ is normally distributed is quite reasonable. In particular, the true environment often does not exactly fit our formalization of POMDPs (due to nonsta-

tionarity). In these cases, the false assumption that $\bar{X}_n$ is normally distributed may be minor in comparison to the false assumption that the environment is a POMDP.

If we introduce the (false, yet often reasonable) assumption that $\bar{X}_n$ is actually normally distributed, then we can apply a one-tailed *Student's t-test* (TT):

**Theorem 22** (Student's $t$-test (TT))**.** *Let $\{X_i\}_{i=1}^n$ be $n$ independent random variables such that $\bar{X}_n \coloneqq \frac{1}{n}\sum_{i=1}^n X_i$ is a normally distributed random variable. Then*

$$\Pr\left(\mathbf{E}\left[\frac{1}{n}\sum_{i=1}^n X_i\right] \geq \frac{1}{n}\sum_{i=1}^n X_i - \frac{\sqrt{\frac{1}{n-1}\sum_{i=1}^n \left(X_i - \bar{X}_n\right)^2}}{\sqrt{n}} t_{1-\delta,n-1}\right) \geq 1-\delta,$$

*where $t_{1-\delta,\nu}$ denotes the $100(1-\delta)$ percentile of the Student's $t$ distribution with $\nu$ degrees of freedom (i.e., tinv$(1-\delta,\nu)$ in* MATLAB*).*

*Proof.* See the work of Walpole et al. (2007, Section 10.7). ■

Notice that the TT has no direct dependence on the range of the random variables– in fact it assumes that at least some of the $X_i$ are unbounded in order for $\bar{X}_n$ to be normally distributed. As we will show later, the TT tends to be overly conservative (provide a valid, but loose bound) when the $X_i$ are distributed with a heavy upper tail, as is common in our application. That is, it makes errors (gives a lower bound on the mean of at least $\rho_-$) significantly less than $100\delta\%$ of the time.

If $\bar{X}_n$ were truly normally distributed, then the error rate of the TT would be exactly $100\delta\%$. This suggests an alternative approach—use the available samples to estimate the true distribution of $\bar{X}_n$ and produce a confidence bound specialized to that distribution. This is the high level idea behind bootstrap confidence bounds. These approaches are still only approximate (they do not guarantee at most a $1-\delta$ error rate) since they assume that $\bar{X}_n$ comes from the predicted distribution.

*Bias corrected and accelerated bootstrap* (BCa) (Davison and Hinkley, 1997, Efron and Tibshirani, 1993) is perhaps the most popular such method. It is best expressed

as an algorithm rather than an equation, and is therefore presented as pseudocode in the following subsection (Algorithm 4.6). For our application, we will find that it produces the tightest confidence intervals. Even though the bounds produced by BCa are only approximate, BCa is often used to analyze the results of medical research (Champless et al., 2003, Folsom et al., 2003). This suggests that the lower bounds produced by BCa may be sufficiently reliable even for high risk applications.

Using the bootstrap estimate of the distribution of $\bar{X}_n$, BCa can correct for the heavy tails in our data to produce lower bounds that are not overly conservative like those of the TT. As with the TT, for some distributions the bounds produced by BCa may have error rates larger than $\delta$. It is therefore best to think of the TT and BCa as trying to produce an error rate of approximately $\delta$, as opposed to upper bounding the error rate by $\delta$.

To emphasize the over-conservativeness of TT relative to BCa when used for distributions with heavy tails, we compared the lower bounds produced by both methods when applied to samples from a gamma distribution. The results are provided in Figure 4.1. The different properties of the exact and approximate concentration inequalities are summarized in Table 4.1. The entries for "BEA" and "CUT" correspond to other concentration inequalities that we present later.

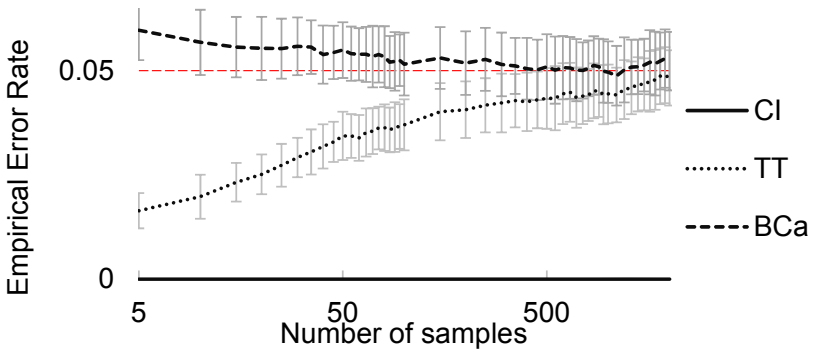


**Figure 4.1.** Empirical error rates when estimating a 95% confidence lower-bound on the mean of a gamma distribution (shape parameter $k = 2$ and scale parameter $\theta = 50$) using TT, BCa, and also the two exact concentration inequalities (AM and CUT) that are applicable to random variables with no upper bound (they both perform identically in this plot, and so are represented by a single line, labeled as "CI"). The gamma distribution used has a heavy upper-tail similar to that of importance weighted returns. The logarithmically scaled horizontal axis is the number of samples used to compute the lower bound (from 5 to 2000) and the vertical axis is the mean empirical error rate over 1,000,000 trials. This error rate is the number of times the lower bound on the mean was greater than the true mean, divided by the number of samples. The error bars show the sample standard deviation. Note that CI is overly conservative, with zero error in all the trials (it is on the $x$-axis). The $t$-test is initially conservative, but approaches the allowed 5% error rate as the number of samples increases. BCa has an error rate above 5%, but remains close to 5% throughout most of the plot.

| Name | Direct Dependence on $b$ | Identically Distributed Only | Exact or Approximate | Reference | Notes |
|---|---|---|---|---|---|
| CH | $\Theta\left(\frac{b}{\sqrt{n}}\right)$ | No | Exact | (Massart, 2007) | None |
| MPeB | $\Theta\left(\frac{b}{n}\right)$ | No | Exact | (Maurer and Pontil, 2009, Theorem 11) | Requires all random variables to have the same range. |
| AM | None | Yes | Exact | (Anderson, 1969, Massart, 1990) | Depends on the largest observed sample. Loose for distributions without heavy tails. |
| BM | $\Theta\left(\frac{b}{\sqrt{n}}\right)$ | Yes | Exact | (Bubeck et al., 2012) | None. |
| TT | None | No | Approximate | (Walpole et al., 2007) | Assumes $X_n$ is normally distributed. Tends to give conservative lower bounds for random variables with heavy upper tails. |
| BCa | None | No | Approximate | (Davison and Hinkley, 1997, Efron and Tibshirani, 1993) | Assumes $X_n$ comes from the bootstrap distribution. |
| BEA | None | Yes | Approximate | (Bottou et al., 2013) | Discards some data. |
| CUT | None | No | Exact | Theorem 23 | None. |

**Table 4.1.** Summary of properties of the exact and approximate concentration inequalities discussed in this dissertation.

## 4.5 Pseudocode for Exact and Approximate Concentration Inequalities

Pseudocode is provided below for each of the concentration inequalities that we have discussed. The CH inequality is presented in Algorithm 4.1, the MPeB inequality is presented in Algorithm 4.2, the AM inequality is presented in Algorithm 4.3, the BM inequality is presented in Algorithm 4.4, the TT is presented in Algorithm 4.5, and BCa is presented in Algorithm 4.6. The pseudocode in Algorithm 4.6 for BCa was derived from that of Carpenter and Bithell (2000), with notation changed as necessary to avoid conflicts with our other notation. It uses $B = 2000$ resamplings as suggested by practitioners (Davison and Hinkley, 1997, Efron and Tibshirani, 1993), $\Phi$ to denote the cumulative distribution function of the normal distribution, $\mathbf{1}_A$ to denote one if $A$ is true and 0 otherwise, and $\#\xi_i < \bar{X}_n$ to denotes the number of $\xi_i$ that are less than $\bar{X}_n$. Some comments are provided in the pseudocode for BCa in order to provide a high level understanding for what the different portions of code are doing. However, for a detailed description of the pseudocode, see the work of Carpenter and Bithell (2000).

**Algorithm 4.1:** CH$(X_1, \ldots, X_n, \delta, a_1, \ldots, a_n, b_1, \ldots, b_n)$: Uses the CH inequality to return a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$.
**Assumes:** The $X_i$ are independent and $X_i \in [a_i, b_i]$ for all $i$.

**return** $\frac{1}{n}\sum_{i=1}^n X_i - \sqrt{\frac{\ln\left(\frac{1}{\delta}\right)\sum_{i=1}^n (b_i-a_i)^2}{2n^2}}$;

**Algorithm 4.2:** MPeB$(X_1, \ldots, X_n, \delta, a, b)$: Uses the MPeB inequality to return a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$.
**Assumes:** The $X_i$ are independent and $X_i \in [a, b]$ for all $i$.

$b \leftarrow \max_{i\in\{1,\ldots,n\}} b_i$;

**return** $\frac{1}{n}\sum_{i=1}^n X_i - \frac{7(b-a)\ln\left(\frac{2}{\delta}\right)}{3(n-1)} - \frac{1}{n}\sqrt{\frac{\ln\left(\frac{2}{\delta}\right)}{n-1}\sum_{i,j=1}^n (X_i - X_j)^2}$;

To simplify later pseudocode, we provide a uniform interface to all of the concentration inequalities in Algorithm 4.7.

**Algorithm 4.3:** AM$(X_1, \ldots, X_n, \delta, a)$: Uses the AM inequality to return a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Assumes:** The $X_i$ are independent and identically distributed and $X_i \geq 0$ for all $i$.

Let $z_1, \ldots, z_n$ be $X_1, \ldots, X_n$, sorted in ascending order;
$z_0 \leftarrow a$;
**return** $z_n - \sum_{i=0}^{n-1}(z_{i+1} - z_i)\min\left\{1, \frac{i}{n} + \sqrt{\frac{\ln(2/\delta)}{2n}}\right\}$;

**Algorithm 4.4:** BM$(X_1, \ldots, X_n, \delta, a, b)$: Uses the BM inequality to return a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Assumes:** The $X_i$ are independent and identically distributed and $X_i \in [a, b]$ for all $i$.

$k \leftarrow \lfloor \min\{8\ln(e^{1/8}/\delta), n/2\} \rfloor$ ; `// Select number of groups`
$N \leftarrow \lfloor n/k \rfloor$ ; `// Number of random variables in each group`
**for** $i = 1$ **to** $k$ **do**
  $\hat{\mu}_i \leftarrow \frac{1}{N}\sum_{t=(i-1)N+1}^{iN} X_t$ ; `// Compute the mean of each group`
$\hat{\mu}_M \leftarrow$MEDIAN$(\hat{\mu}_1, \hat{\mu}_2, \ldots, \hat{\mu}_k)$ ; `// Median of means`
**return** $\hat{\mu}_M - (b-a)\sqrt{\frac{192\left[\sqrt{\frac{1}{n(n-1)}\sum_{i,j=1}^{n}\frac{(x_i-x_j)^2}{2}} + \sqrt{\frac{2\ln(2/\delta)}{n-1}}\right]^2 \ln\left(\frac{2e^{1/8}}{\delta}\right)}{n}}$;

**Algorithm 4.5:** TT$(X_1, \ldots, X_n, \delta)$: Uses the TT to return an approximate $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Assumes:** The $X_i$ are independent random variables with finite variance..

**return** $\frac{1}{n}\sum_{i=1}^{n} X_i - \frac{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \bar{X}_n\right)^2}}{\sqrt{n}} t_{1-\delta,n-1}$;

**Algorithm 4.6:** BCa$(X_1, \ldots, X_n, \delta)$: Uses BCa to return an approximate $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Assumes:** The $X_i$ are independent random variables.

$\bar{X}_n \leftarrow \frac{1}{n}\sum_{i=1}^{n} X_i$; `// Sample mean`
$B \leftarrow 2000$; `// Number of bootstrap resamplings to perform`
`// Generate` $B$ `resamplings of the data and store the sample means`
**for** $i = 1$ **to** $B$ **do**
    Randomly sample $n$ elements of $x \in \mathbf{X}$, with replacement;
    Set $\xi_i$ to be the mean of these $n$ samples;
`// Estimate the` ***bias constant***, $z_0$ `(Efron, 1987)`
Sort the vector $\xi = (\xi_1, \xi_2, \ldots, \xi_B)$ such that $\xi_i \leq \xi_j$ for $1 \leq i < j \leq B$;
$z_0 \leftarrow \Phi^{-1}\left(\frac{\{\#\xi_i < \bar{X}_n\}}{B}\right)$;
`// Estimate the skew,` $a$, `of the distribution of` $\bar{X}_n$
**for** $i = 1$ **to** $n$ **do**
    Set $y_i$ to be the mean of $\mathbf{X}$ excluding the $i^{\text{th}}$ element:
    $y_i \leftarrow \frac{1}{n-1}\sum_{j=1}^{n} \mathbf{1}_{(j \neq i)} X_j$;
$\bar{y} \leftarrow \frac{1}{n}\sum_{i=1}^{n} y_i$;
$a \leftarrow \frac{\sum_{i=1}^{n}(\bar{y}-y_i)^3}{6\left[\sum_{i=1}^{n}(\bar{y}-y_i)^2\right]^{3/2}}$; `// Standard equation for estimating skewness`
`// Get bootstrap confidence bound using a linear interpolation (Carpenter and Bithell, 2000) and the computed bias and skew terms`
$z_L \leftarrow z_0 - \frac{\Phi^{-1}(1-\delta)-z_0}{1+a(\Phi^{-1}(1-\delta)-z_0)}$;
$Q \leftarrow (B+1)\Phi(z_L)$;
$l \leftarrow \min\{\lfloor Q \rfloor, B-1\}$;
**return** $\bar{X}_n + \frac{\Phi^{-1}(\frac{Q}{B+1})-\Phi^{-1}(\frac{l}{B+1})}{\Phi^{-1}(\frac{l+1}{B+1})-\Phi^{-1}(\frac{l}{B+1})}(\xi_Q, \xi_{Q+1})$;

**Algorithm 4.7:** LowerBound$^{\ddagger}(X_1, \ldots, X_n, \delta, a_1, \ldots, a_n, b_1, \ldots, b_n)$: Uses $\ddagger \in \{$CH, MPeB, AM, BM, TT, BCa, CUT$\}$ to return a $1 - \delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Assumes:** The assumptions of the underlying method, $\ddagger$, are satisfied.
**Notice:** Some methods, $\ddagger$, do not rely on all of the inputs, and so some inputs are not necessary for some $\ddagger$.

```
if ‡ = CH then
    return CH(X_1, ..., X_n, δ, a_1, ..., a_n, b_1, ..., b_n);
else if ‡ ∈ {MPeB, BM} then
    return ‡(X_1, ..., X_n, δ, min_{i∈{1,...,n}} a_i, max_{i∈{1,...,n}} a_i);
else if ‡ = AM then
    return AM(X_1, ..., X_n, δ, a_1);
else if ‡ ∈ {TT, BCa, CUT} then
    return ‡(X_1, ..., X_n, δ);
else if ‡ = BEA then
    return BEA(X_1, ..., X_n, δ, min_{i∈{1,...,n}} a_i);
```

## 4.6 Approach

Our general approach to HCOPE combines off-policy evaluation methods like IS, PDIS, NPDIS, WIS, and CWPDIS with exact and approximate concentration inequalities. Our approach to HCOPE when using IS, PDIS, or NPDIS is presented as pseudocode in Algorithm 4.8. This pseudocode does not allow for the weighted importance sampling variants because they are only defined in their batch forms—we have not defined (and it is not clear how to define) $\hat{\rho}^{\text{WIS}}(\pi_e|H_L, \pi_b)$ or $\hat{\rho}^{\text{CWPDIS}}(\pi_e|H_L, \pi_b)$.

**Algorithm 4.8:** $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta)$: Compute a $1-\delta$ confidence lower bound on $\rho(\pi_e)$ using the historical data $\mathcal{D} = \{(H_L^i, \pi_i) : i \in \{1, \ldots, n\}, H_L^i \sim \pi_i\}$. The importance sampling method is specified by $\dagger \in \{\text{IS, PDIS, NPDIS}\}$ and the concentration inequality by $\ddagger \in \{\text{CH, MPeB, AM, BM, TT, BCa, CUT}\}$. Any combination of $\dagger$ and $\ddagger$ is allowed except for $\dagger = \text{PDIS}$ and $\ddagger = \text{CUT}$, as discussed later.

**Assumes:** If $\ddagger \in \{\text{AM, BM}\}$, then there is only one behavior policy, i.e., $\pi_i = \pi_j$ for all $i, j$. Also, if $\dagger = \text{PDIS}$, then Assumption 1 is required.

**for** $i = 1$ **to** $n$ **do**
  $X_i \leftarrow \hat{\rho}^{\dagger}(\pi_e | H_L^i, \pi_i)$ ; `// Compute importance weighted returns`
**for** $i = 1$ **to** $n$ **do**
  $a_i \leftarrow \hat{\rho}^{\dagger}_{\text{lb}}(\pi_e, \pi_i)$; `// Compute lower bound on` $X_i$
  $b_i \leftarrow \hat{\rho}^{\dagger}_{\text{ub}}(\pi_e, \pi_i)$; `// Compute upper bound on` $X_i$
**return** $\text{LowerBound}^{\ddagger}(X_1, \ldots, X_n, \delta, a_1, \ldots, a_n, b_1, \ldots, b_n)$;

The weighted importance sampling estimators, WIS and CWPDIS, are not suitable for use with the exact concentration inequalities because they are only defined as batch methods. It is also unclear how they can be suitably used with TT without using some form of bootstrapping to estimate the variance of the WIS and CWPDIS estimators. However, it is straightforward to use both WIS and CWPDIS with BCa, since BCa can be used to lower bound any statistic of the data (not just the sample mean). In Algorithm 4.9 we present pseudocode for HCOPE using WIS and CWPDIS with BCa.

Notice that $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta)$ is an exact HCOPE method if $\dagger \in \{\text{IS, PDIS, NPDIS}\}$ and $\ddagger \in \{\text{CH, MPeB, AM, BM, CUT}\}$, and an approximate HCOPE method if $\dagger \in \{\text{WIS, CWPDIS}\}$ or $\ddagger \in \{\text{TT,BCa}\}$. Also, notice that it is defined for all combinations of

$$\dagger \in \{\text{IS, PDIS, NPDIS, WIS, CWPDIS}\}$$

$$\ddagger \in \{\text{CH, MPeB, AM, BM, TT, BCa, CUT}\},$$

except for when $\dagger \in \{\text{WIS, CWPDIS}\}$ and $\ddagger \neq \text{BCa}$. Also notice that so far we have defined HCOPE methods for CUT, an exact concentration inequality that we present later, but we have not yet defined HCOPE methods for BEA (the approximate concentration inequality described in the next section).

**Algorithm 4.9:** $\text{HCOPE}^{\dagger}_{\text{BCa}}(\pi_e, \mathcal{D}, \delta)$: Compute a $1-\delta$ confidence lower bound on $\rho(\pi_e)$ using the historical data $\mathcal{D} = \{(H_L^i, \pi_i) : i \in \{1, \ldots, n\}, H_L^i \sim \pi_i\}$. The importance sampling method is specified by $\dagger \in \{\text{WIS, CWPDIS}\}$ and BCa is used.

**Assumes:** Nothing—any number of behavior policies can be used and Assumption 1 is not required (although it will increase the accuracy of the approximate lower bound).

**Notice:** The only differences between this pseudocode and Algorithm 4.6 are the lines that are highlighted in red.

$\bar{X}_n \leftarrow \dagger(\pi_e|\mathcal{D})$; // Sample of statistic to be bounded ($\hat{\theta}$ in most BCa literature)

$B \leftarrow 2000$; // Number of bootstrap resamplings to perform

// Generate $B$ resamplings of the data and store the sample statistics

**for** $i = 1$ **to** $B$ **do**

  Randomly sample $n_{\mathcal{D}}$ trajectories, $H_L^i$, and their behavior policies, $\pi_i$, from $\mathcal{D}$, with replacement. Store these $n_{\mathcal{D}}$ resampled trajectories in $\mathcal{D}'$;

  $\xi_i \leftarrow \dagger(\pi_e|\mathcal{D}')$;

// Estimate the *bias constant*, $z_0$ (Efron, 1987)

Sort the vector $\xi = (\xi_1, \xi_2, \ldots, \xi_B)$ such that $\xi_i \leq \xi_j$ for $1 \leq i < j \leq B$;

$z_0 \leftarrow \Phi^{-1}\left(\frac{\{\#\xi_i < \bar{X}_n\}}{B}\right)$;

// Estimate the skew, $a$, of the distribution of $\bar{X}_n$

**for** $i = 1$ **to** $n$ **do**

  Set $y_i$ to be the mean of $\mathbf{X}$ excluding the $i^{\text{th}}$ element:

  $y_i \leftarrow \frac{1}{n-1}\sum_{j=1}^{n} \mathbf{1}_{(j \neq i)} X_j$;

$\bar{y} \leftarrow \frac{1}{n}\sum_{i=1}^{n} y_i$;

$a \leftarrow \frac{\sum_{i=1}^{n}(\bar{y}-y_i)^3}{6\left[\sum_{i=1}^{n}(\bar{y}-y_i)^2\right]^{3/2}}$; // Standard equation for estimating skewness

// Get bootstrap confidence bound using a linear interpolation (Carpenter and Bithell, 2000) and the computed bias and skew terms

$z_L \leftarrow z_0 - \frac{\Phi^{-1}(1-\delta)-z_0}{1+a(\Phi^{-1}(1-\delta)-z_0)}$;

$Q \leftarrow (B+1)\Phi(z_L)$;

$l \leftarrow \min\{\lfloor Q \rfloor, B-1\}$;

**return** $\bar{X}_n + \frac{\Phi^{-1}(\frac{Q}{B+1})-\Phi^{-1}(\frac{l}{B+1})}{\Phi^{-1}(\frac{l+1}{B+1})-\Phi^{-1}(\frac{l}{B+1})}(\xi_Q, \xi_{Q+1})$;

## 4.7 Using Clipped Importance Weights

Bottou et al. (2013) presented an approach to HCOPE that is similar to ours. Although their derivation predates ours,[1] the two efforts were independent. Bottou et al. (2013) propose using IS with MPeB for HCOPE, with one change—the importance weights are "clipped". That is, let $c$ be the $5^{\text{th}}$ largest importance weight for the trajectories in $\mathcal{D}$. We call $c$ the *threshold*. When computing $\hat{\rho}^{\text{IS}}(\pi_e|H_L^i, \pi_i)$, set the importance weight to zero if it is larger than $c$. This removes the largest importance weights and ensures that all of the random variables, $X_i$, that are provided to MPeB are in the range $[0, c]$. If $c \ll \hat{\rho}^{\text{IS}}_{\text{ub}}(\pi_e, \pi_b)$, then this can make MPeB significantly tighter. However, because the threshold, $c$, depends on the data, the range of the $X_i$ depends on the data, and thus MPeB is not technically applicable with $[0, c]$ as the range for the $X_i$ (MPeB assumes that the range of the random variables is a fixed constant, and not a random variable that depends on the data). Even so, Bottou et al. propose using $[0, c]$ as the range of the $X_i$ when using MPeB, and so their HCOPE method is only approximate (Bottou et al., 2013, footnote 7). We present their method in Algorithm 4.10.

In the next section we present a new exact concentration inequality based on an approach that is similar to that of Bottou et al. (2013). The differences between the method we propose and that of Bottou et al. are enumerated below:

1. Rather than clipping values, we "collapse" them. That is, if a value, $x$, is greater than a threshold, $c$, we set $x \leftarrow c$ rather than $x \leftarrow 0$ (which effectively discards data).

[1] This work was performed from March 2014 to August 2015. Our first publication of this work was in January 2015 (Thomas et al., 2015c), two years after the work of Bottou et al. (2013). However, since we were not aware of their work until mid-2015, this dissertation contains the first comparison of our approach to theirs.

**Algorithm 4.10:** $\text{HCOPE}^{\text{IS}}_{\text{BEA}}(\pi_e, \mathcal{D}, \delta)$: Compute a $1-\delta$ confidence lower bound on $\rho(\pi_e)$ using the historical data $\mathcal{D} = \{(H^i_L, \pi_i) : i \in \{1, \dots, n\}, H^i_L \sim \pi_i\}$. This uses the method of Bottou et al. (2013), which is based on IS with clipped weights and MPeB.
**Assumes:** There is only one behavior policy, i.e., $\pi_i = \pi_j$ for all $i, j$.

**for** $i = 1$ **to** $n$ **do**
  $w_i \leftarrow \prod_{t=1}^{L} \frac{\pi_e(a_t|o_t)}{\pi_b(a_t|o_t)}$ ; `// Importance weight`
$c \leftarrow$ the fifth largest element of $(w_i)_{i=1}^n$; `// Compute the threshold,` $c$
**for** $i = 1$ **to** $n$ **do**
  **if** $w_i \geq c$ **then**
    $w_i \leftarrow 0$ ; `// Clip the importance weight if it is too large`
  $X_i \leftarrow w_i G(H_L)$ ; `// Clipped importance weighted return`
**return** $\text{MPeB}(X_1, \dots, X_n, \delta, 0, c)$; `// Notice that each` $X_i \in [0, c]$

2. We collapse the importance weighted *returns*, as opposed to clipping the importance *weights*.

3. We show that our approach is viable for multiple behavior policies.

4. Whereas their approach is specific to combining MPeB with IS, our approach is a general concentration inequality like CH and MPeB. This means that it may be useful outside of the RL community, and also that it can be used with PDIS and NPDIS.

5. Whereas their approach is only an approximate HCOPE method, ours is an exact concentration inequality (which means that it can produce an exact HCOPE method if combined with an unbiased estimator like IS, PDIS, or NPDIS).

6. We optimize the threshold, $c$, in a more principled way and show that its optimization is crucial to good performance.

## 4.8 A New Concentration Inequality

Here we present a new exact concentration inequality that is particularly well suited to HCOPE with IS or NPDIS (not PDIS, because it can produce negative

estimates). That is, it performs well when the random variables, $\{X_i\}_{i=1}^n$ are non-negative and have large upper bounds that would make other exact concentration inequalities like CH and MPeB perform poorly. It is an extension of MPeB that relies on two key insights: **1)** removing the upper tail of a distribution can only lower its expected value, and **2)** MPeB can be generalized to handle random variables with different ranges if it is used to lower bound a value that is greater than or equal to the expected value of each random variable.

Here we present an intuitive description of how the derivation of our concentration inequality will proceed before presenting a formal proof. Intuitively, we first assume that a threshold, $c_i$, has been provided for each random variable, $X_i$, and that each $X_i \geq 0$. Later we will discuss how to select each $c_i$ automatically from the data. We then collapse the random variables $\{X_i\}_{i=1}^n$ to produce $Y_i \coloneqq \min\{X_i, c_i\}$ for each $i \in \{1, \ldots, n\}$. Notice that

$$\begin{aligned} \mathbf{E}\left[Y_i\right] =& \mathbf{E}[\min\{X_i, c_i\}] \\ \leq& \mathbf{E}[X_i]. \end{aligned} \tag{4.9}$$

So, if some value, $\mu_{\text{lb}}$, is a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n Y_i]$, then it is also a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$. Next, recall that MPeB performs poorly if each random variable has a different range. To overcome this, we normalize each $Y_i$ so that it is in the range $[0, 1]$. Since $Y_i \in [0, c_i]$, we can define the normalized $Y_i$ to be $Z_i \coloneqq \frac{Y_i}{c_i} \in [0, 1]$. We then apply MPeB to $Z_i$ to produce a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n Z_i]$. Next we must extract from this lower bound a lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$. We do this by leveraging the second property listed above to undo the normalization to get a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n Y_i]$, which we know from (4.9) is also a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$.

Our approach for collapsing the tails of the distributions and then bounding the means of the new distributions is similar to bounding the truncated mean and is

a form of Winsorization (Wilcox and Keselman, 2003). We present our concentration inequality in Theorem 23, and we refer to it as the *collapsed upper tail* (CUT) inequality.

**Theorem 23.** *[CUT Inequality] Let $\{X_i\}_{i=1}^n$ be $n$ independent real-valued bounded random variables such that for each $i \in \{1, \ldots, n\}$, we have $\Pr(0 \leq X_i) = 1$, $\mathbf{E}[X_i] \leq \mu$, and the fixed real-valued threshold $c_i > 0$. Let $\delta > 0$ and $Y_i \coloneqq \min\{X_i, c_i\}$. Then with probability at least $1 - \delta$, we have*

$$\mu \geq \underbrace{\left(\sum_{i=1}^n \frac{1}{c_i}\right)^{-1} \sum_{i=1}^n \frac{Y_i}{c_i}}_{\text{empirical mean}} - \underbrace{\left(\sum_{i=1}^n \frac{1}{c_i}\right)^{-1} \frac{7n\ln(2/\delta)}{3(n-1)}}_{\text{term that goes to zero as } 1/n \text{ as } n \to \infty} - \underbrace{\left(\sum_{i=1}^n \frac{1}{c_i}\right)^{-1} \sqrt{\frac{\ln(2/\delta)}{n-1} \sum_{i,j=1}^n \left(\frac{Y_i}{c_i} - \frac{Y_j}{c_j}\right)^2}}_{\text{term that goes to zero as } 1/\sqrt{n} \text{ as } n \to \infty}. \tag{4.10}$$

*Proof.* We define $n$ independent random variables, $\mathbf{Z} = \{Z_i\}_{i=1}^n$, as $Z_i \coloneqq \frac{Y_i}{c_i}$. Thus, we have

$$\bar{Z} \coloneqq \frac{1}{n}\sum_{i=1}^n Z_i = \frac{1}{n}\sum_{i=1}^n \frac{Y_i}{c_i}. \tag{4.11}$$

Since $\mathbf{E}[Y_i] \leq \mathbf{E}[X_i] \leq \mu$, we may write

$$\mathbf{E}[\bar{Z}] = \frac{1}{n}\sum_{i=1}^n \frac{\mathbf{E}[Y_i]}{c_i} \leq \frac{\mu}{n}\sum_{i=1}^n \frac{1}{c_i}. \tag{4.12}$$

Notice that the $Z_i$ random variables, and therefore also the $(1 - Z_i)$ random variables, are $n$ independent random variables with values in $[0, 1]$. So, using Theorem 11 of Maurer and Pontil (2009), with probability at least $1 - \delta$, we have that

$$\mathbf{E}[1 - \bar{Z}] \leq 1 - \bar{Z} + \sqrt{\frac{2V_n(1 - \mathbf{Z})\ln(2/\delta)}{n}} + \frac{7\ln(2/\delta)}{3(n-1)}, \tag{4.13}$$

where the empirical variance, $V_n(1-\mathbf{Z})$, is defined as

$$\begin{aligned} V_n(1-\mathbf{Z}) &\coloneqq \frac{1}{2n(n-1)} \sum_{i,j=1}^{n} \big((1-Z_i)-(1-Z_j)\big)^2 \\ &= \frac{1}{2n(n-1)} \sum_{i,j=1}^{n} \left(\frac{Y_i}{c_i} - \frac{Y_j}{c_j}\right)^2. \end{aligned} \tag{4.14}$$

The claim follows by replacing $\bar{Z}$, $\mathbf{E}[\bar{Z}]$, and $V_n(1-\mathbf{Z})$ in (4.13) with (4.11), (4.12), and (4.14). ■

Notice that if there is some $b \in \mathbb{R}$ such that $\Pr(X_i \leq b) = 1$ and $c_i = b$ for all $i$, then Theorem 23 degenerates to MPeB. Also, despite the nested sum, $\sum_{i,j}$, the right side of (4.10) can be evaluated in linear time (a single pass over the samples), since we may write

$$\begin{aligned} \sum_{i,j=1}^{n} (Z_i - Z_j)^2 &= \sum_{i,j=1}^{n} Z_i^2 - 2Z_iZ_j + Z_j^2 \\ &= 2n \sum_{i=1}^{n} Z_i^2 - 2\left(\sum_{i=1}^{n} Z_i\right)^2, \end{aligned}$$

and so (4.10) may be rewritten as

$$\mu \geq \left(\sum_{i=1}^{n} \frac{1}{c_i}\right)^{-1} \left[\sum_{i=1}^{n} \frac{Y_i}{c_i} - \frac{7n\ln(2/\delta)}{3(n-1)} - \sqrt{\frac{2\ln(2/\delta)}{n-1}\left(n\sum_{i=1}^{n}\left(\frac{Y_i}{c_i}\right)^2 - \left(\sum_{i=1}^{n}\frac{Y_i}{c_i}\right)^2\right)}\right].$$

Also notice that the CUT inequality requires $\mathbf{E}[X_i] \leq \mu$, which is more general than requiring $\mathbf{E}[X_i] = \mu$. This is useful when using IS or NPDIS in a setting where Assumption 1 does not hold, since then we have from Corollary 4 and Theorem 16 that $\mathbf{E}[\hat{\rho}^{\dagger}(\pi_e|H_L,\pi_b)] \leq \rho(\pi_e)$ for $\dagger \in \{\text{IS, NPDIS}\}$. This means that we can get a

high confidence lower bound on $\rho(\pi_e)$ by using the CUT inequality with the IS or NPDIS estimators, even if Assumption 1 does not hold. However, notice from the proofs of Corollary 4 and Theorem 16 that the lower bound will become loose as the number of observation-action pairs where Assumption 1 is violated increases.

In order to use the result of Theorem 23 for HCOPE, we must select the values of the $c_i$, i.e., the thresholds beyond which the distributions of the $X_i$ are collapsed. To simplify this procedure, we select a single $c > 0$ and set $c_i = c$ for all $i$. When $c$ is too large, it loosens the bound just like a large range, $b$, does for MPeB (see (4.7)). On the other hand, when $c$ is too small, it decreases the expected values of the collapsed random variables, $Y_i$, which also loosens the bound. The optimal $c$ must properly balance this trade-off between the range and mean of the $Y_i$.

Figure 4.2 illustrates this trade-off using easily reproduced synthetic data. Notice that the lower bound produced by the CUT inequality is sensitive to the value of $c$. When $c$ is too small, the lower bound approaches zero because every sample, $X_i$, is collapsed to $Y_i = 0$. When $c$ is too large, performance also degrades because of the CUT inequality's direct dependence on $c$. Also notice that the optimal value of $c$ changes with the number of samples. Also, particularly when there are only a few samples, the CUT inequality is sensitive to $c$ being too large. Since we are primarily interested in achieving tight lower bounds with little data, this means that finding a near-optimal value for $c$ will be crucial.

We are therefore faced with the challenge of determining a near-optimal value for $c$. Theorem 23 requires the thresholds, $c_i$, to be fixed—i.e., they should not be computed using realizations of any $X_i$. If $c_i$ depends on any $X_j$ where $i \neq j$, then $Z_i$ and $Z_j$ are not independent and so Theorem 11 of Maurer and Pontil (2009) cannot be applied in our proof. If $c_i$ depends on $X_i$, then 4.12 may not hold because $c_i$ is a random variable and so a covariance term, $\mathrm{Cov}(Y_i, 1/c_i)$, is missing. This means that we must select a value, $c$, *without looking at the data that is used by the CUT*

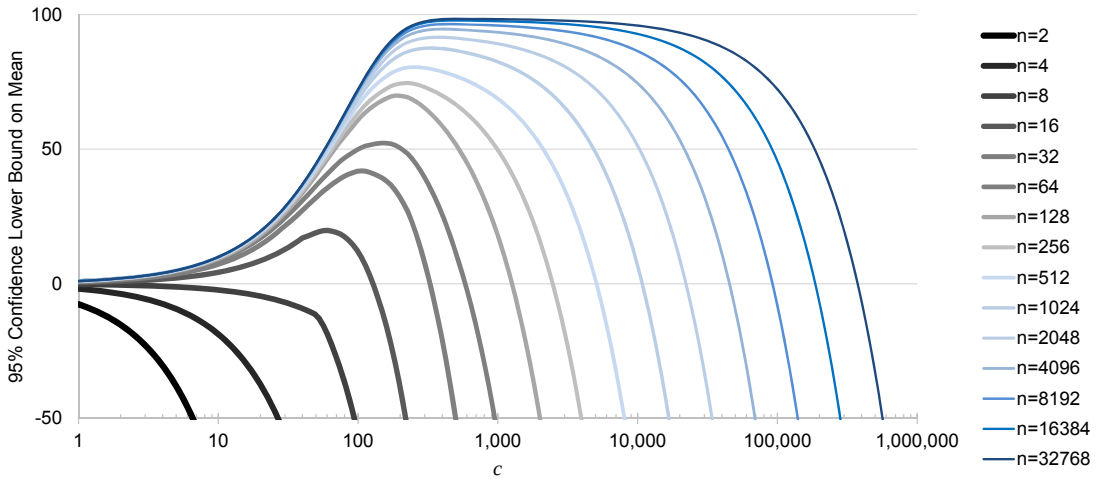


**Figure 4.2.** This figure uses samples from the gamma distribution with shape parameter $k = 2$ and scale parameter $\theta = 50$, which has a true mean of 100. The plot shows the 95% confidence lower bound on the mean produced by the CUT inequality using $n$ samples for various $c$ (specified on the horizontal axis). For readers without color, notice that the thinner curves correspond to larger $n$. Notice the logarithmic scale of the horizontal axis. For any $n$, an optimal value of $c$ is one that causes the curve to take its largest value.

*inequality*. Recall that BEA selected $c$ based on the data—this is why it is only an *approximate* HCOPE method.

To select $c$ in a way that produces an *exact* HCOPE method, we propose partitioning the data set, $\{X_i\}_{i=1}^n$, into two sets, $\mathcal{X}_{\text{pre}}$ and $\mathcal{X}_{\text{post}}$. $\mathcal{X}_{\text{pre}}$ is used to estimate the optimal threshold, $c$, and $\mathcal{X}_{\text{post}}$ is used to compute the lower bound (the RHS of (4.10)). Although in this scheme $c$ is random, it does not depend on any of the data that is used by the CUT inequality, and so the requirements of Theorem 23 are satisfied.

If too much data is allocated to $\mathcal{X}_{\text{pre}}$, then there will not be enough data in $\mathcal{X}_{\text{post}}$ to provide a tight lower bound, regardless of how well optimized $c$ is. However, if too little data is allocated to $\mathcal{X}_{\text{pre}}$, then it may not be possible to predict what value of $c$ will work well when using the CUT inequality with $\mathcal{X}_{\text{post}}$. This is an interesting optimal stopping problem—one sample should be allocated to $\mathcal{X}_{\text{pre}}$ at a time, until

some algorithm determines that no more samples are needed (reducing the number of remaining samples is no longer worth the incremental improvement to the estimate of an optimal threshold, $c$). However, a principled solution to this problem is beyond the scope of this work. Instead we present a scheme that works well for exact HCOPE.

We place 1/5 of the samples in $\mathcal{X}_{\text{pre}}$ and the remainder in $\mathcal{X}_{\text{post}}$.[2] We found that 1/5 of the samples is sufficient to optimize $c$, and if it is more samples than needed, it can only reduce performance by at most $1/5 = 20\%$, i.e., our approach will "waste" at most 20% of the data. Our scheme for optimizing $c$ then proceeds as follows. First we describe how $\mathcal{X}_{\text{pre}}$ can be used to compute an estimate of the output of the CUT inequality if applied to $\mathcal{X}_{\text{post}}$ with any value of $c$. We then search for the value of $c$ that maximizes this estimator.

If $\mathcal{X} = \{X_i\}_{i=1}^n$, let $\widehat{\text{CUT}}(\mathcal{X}, \delta, c, m)$ be an estimate of what the CUT inequality would output if using $m$ samples rather than $n$ samples. We will apply this estimator with $\mathcal{X} = \mathcal{X}_{\text{pre}}$ and $m = |\mathcal{X}_{\text{post}}|$. To derive the estimator that we use, notice that the CUT inequality is based on the number of samples, the sample mean of the collapsed random variables, and the sample variance of the collapsed random variables. So, we use the sample mean and sample variance from the collapsed $\mathcal{X}$, but $m$ as the number of samples:

$$\widehat{\text{CUT}}(\mathcal{X}, \delta, c, m) \coloneqq \underbrace{\frac{1}{n}\sum_{i=1}^{n} \min\{X_i, c\}}_{\text{sample mean of } \mathcal{X} \text{ (after being collapsed)}} - \frac{7c\ln(2/\delta)}{3(m-1)} - \sqrt{\frac{\ln(2/\delta)}{m} \underbrace{\frac{2}{n(n-1)}\left(n\sum_{i=1}^{n}(\min\{X_i, c\})^2 - \left(\sum_{i=1}^{n}\min\{X_i, c\}\right)^2\right)}_{\text{sample variance of } \mathcal{X} \text{ (after being collapsed)}}}, \tag{4.15}$$

[2] In our experiments we randomly select 1/5 of the data for $\mathcal{X}_{\text{pre}}$. An alternate implementation might use stratified sampling to, for example, ensure that identically distributed random variables are split so that approximately 1/5 are in $\mathcal{X}_{\text{pre}}$.

where $n = |\mathcal{X}|$. Notice that $\widehat{\text{CUT}}(\mathcal{X}_{\text{pre}}, \delta, c, m)$, is the lower bound produced by the CUT inequality (rather than a prediction) if $m = |\mathcal{X}_{\text{pre}}|$.

Although intuitively appealing, even if the $X_i$ are all identically distributed, $\widehat{\text{CUT}}(\mathcal{X}, \delta, c, m)$ is not necessarily an unbiased estimator of the expected output of the CUT inequality if using $m$ samples, i.e., it is not necessarily the case that

$$\mathbf{E}\left[\widehat{\text{CUT}}(\mathcal{X}, \delta, c, m)\right] = \mathbf{E}\left[\widehat{\text{CUT}}(\mathcal{X}', \delta, c, m)\right],$$

if $\mathcal{X} = \{X_i\}_{i=1}^n$, $\mathcal{X}' = \{X_i'\}_{i=1}^m$, and all $X_i$ and $X_i'$ are independent and identically distributed random variables. Although the sample mean in (4.15) when computing $\widehat{\text{CUT}}(\mathcal{X}, \delta, c, m)$ is an unbiased estimator of the expected sample mean when computing $\widehat{\text{CUT}}(\mathcal{X}', \delta, c, m)$, the sample *standard deviation* is *not*. This is because, although the sample variance (using Bessel's correction) is an unbiased estimator of the true variance, the sample standard deviation is not an unbiased estimator of standard deviation.

We can now specify how $c^*$, an estimate of the optimal value for $c$, should be chosen given $\mathcal{X}_{\text{pre}}$ and $\mathcal{X}_{\text{post}}$:

$$c^* \in \arg\max_c \widehat{\text{CUT}}(\mathcal{X}_{\text{pre}}, \delta, c, |\mathcal{X}_{\text{post}}|).$$

Notice that, although $c^*$ depends on $|\mathcal{X}_{\text{post}}|$, it does not depend on the values of any $X_i \in \mathcal{X}_{\text{post}}$, and so we can then compute the $1 - \delta$ confidence lower bound on $\mu$ specified by Theorem 23, i.e., $\widehat{\text{CUT}}(\mathcal{X}_{\text{post}}, \delta, c^*, |\mathcal{X}_{\text{post}}|)$.

## 4.9 Pseudocode

Pseudocode for the CUT inequality is provided in Algorithm 4.11. Notice that the CUT inequality requires the random variables to be nonnegative, and so it will

not be applicable with PDIS for exact HCOPE. This is why Algorithm 4.8 disallows the combination of † = PDIS and ‡ = CUT.

**Algorithm 4.11:** $\mathrm{CUT}(X_1, \ldots, X_n, \delta)$: Uses the CUT inequality to return a $1-\delta$ confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^{n} X_i]$.
**Constants:** This algorithm has a real-valued hyperparameter, $c_{\min} \geq 0$, which is the smallest allowed threshold. It should be chosen based on the application. For HCOPE we use $c_{\min} = 1$.
**Assumes:** The $X_i$ are independent random variables such that $\Pr(X_i \geq 0) = 1$ for all $i \in \{1, \ldots, n\}$.

Randomly select 1/5 of the $X_i$ and place them in a set $\mathcal{X}_{\text{pre}}$ and the remainder in $\mathcal{X}_{\text{post}}$;
`// Optimize threshold using` $\mathcal{X}_{\texttt{pre}}$
$c^\star \in \arg\max_{c \in [1,\infty]} \widehat{\mathrm{CUT}}(\mathcal{X}_{\text{pre}}, \delta, c, |\mathcal{X}_{\text{post}}|)$; `//` $\widehat{\texttt{CUT}}$ `is defined in` (4.15)
$c^* = \max\{c_{\min}, c^*\}$; `// Do not let` $c^*$ `become too small`
`// Compute lower bound using optimized threshold,` $c^*$ `and` $\mathcal{X}_{\texttt{post}}$
**return** $\widehat{\mathrm{CUT}}(\mathcal{X}_{\text{post}}, \delta, c^\star, |\mathcal{X}_{\text{post}}|)$;

### 4.9.1 Other Uses of the CUT Inequality

In some cases, we might be provided with a lower bound, for which we would like to determine the confidence that the lower bound will hold. Given a confidence level, $\delta$, we can use (4.10) to compute a $1-\delta$ confidence lower bound on $\mu$ (it is the right side of (4.10)). If we are provided with a lower bound, $\mu_{\text{lb}}$, we can solve (4.10) for $1-\delta$ to get the confidence with which the lower bound holds:

**Corollary 7.** *Let $\{X_i\}_{i=1}^n$ be $n$ independent real-valued bounded random variables such that for each $i \in \{1, \ldots, n\}$, we have $\Pr(0 \leq X_i) = 1$, $\mathbb{E}[X_i] \leq \mu$, and the fixed real-valued threshold $c_i > 0$. Let $Y_i \coloneqq \min\{X_i, c_i\}$, $\mu_{lb}$ be any real number and*

$$k_1 = \frac{7n}{3(n-1)},$$

$$k_2 = \sqrt{\frac{2}{(n-1)}\left(n\sum_{i=1}^{n}\left(\frac{Y_i}{c_i}\right)^2 - \left(\sum_{i=1}^{n}\frac{Y_i}{c_i}\right)^2\right)},$$

$$k_3 = \mu_{lb}\sum_{i=1}^{n}\frac{1}{c_i} - \sum_{i=1}^{n}\frac{Y_i}{c_i},$$

$$k_4 = \frac{-k_2 + \sqrt{k_2^2 - 4k_1k_3}}{2k_1}.$$

*Then $\mu_{lb}$ is a lower bound on $\mu$ with confidence*

$$1-\delta = \begin{cases} 1 - \min\{1, 2\exp\left(-k_4^2\right)\} & \text{if } k_4 \text{ is real and positive,} \\ 0 & \text{otherwise.} \end{cases}$$

*Proof.* This follows from solving (4.10) for $\delta$. ■

### 4.9.2 High Confidence Upper Bounds

Theorem 23 can also be used to generate a high-confidence upper bound on independent random variables $\{X_i\}_{i=1}^n$ if $X_i \leq b_i$. This can be accomplished by first negating the $X_i$ and then translating them so that they are always positive. That is, let $X_i' = b_i - X_i$, for all $i$. We can then apply Theorem 23 to $\{X_i'\}_{i=1}^n$ to produce a high confidence lower bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i']$, from which an upper bound on $\mathbf{E}[\frac{1}{n}\sum_{i=1}^n X_i]$ can be extracted. However, notice that this scheme will only work well if the $X_i'$ have heavy upper tails. For example, for HCOPE this is not the case if bounding the mean of the IS estimator for each trajectory. In this setting it is more appropriate to negate and translate the *returns*, rather than the *importance weighted returns*. Importance sampling can then be used with the negated and translated returns.

| Label in legend | † | ‡ | Exact | Comments |
|---|---|---|---|---|
| CH | NPDIS | CH | Yes | |
| MPeB | NPDIS | MPeB | Yes | |
| AM | NPDIS | AM | Yes | |
| BM | NPDIS | BM | Yes | |
| TT | NPDIS | TT | No | |
| BCa | NPDIS | BCa | No | |
| BEA | IS | BEA | No | |
| CUT | NPDIS | CUT | Yes | Recommended (exact) |
| CUT+IS | IS | CUT | Yes | For fair comparison to BEA |
| BCa2 | CWPDIS | BCa | Yes | Recommended (approximate) |

**Table 4.2.** The combinations of † and ‡ that we include in our plots. The first column corresponds to the label in the legends of future plots. The second and third columns are the importance sampling and concentration inequality methods that are used. The four column specified whether or not the method is an exact or approximate HCOPE method.

## 4.10 Experiments

In this section we provide an empirical comparison of the various exact and approximate concentration inequalities. First we must decide which variants of importance sampling to use. We use NPDIS for almost all experiments. We select NPDIS over IS because we saw in the experiments of the previous chapter that NPDIS performs better. We select NPDIS over PDIS because PDIS performs poorly for exact HCOPE since $\hat{\rho}_{\text{lb}}^{\text{PDIS}}(\pi_e, \pi_b) \ll 0$ in most cases, and we do not have machinery to effectively handle the heavy *lower tail* of the distribution of $\hat{\rho}^{\text{NPDIS}}(\pi_e|H_L, \pi_b)$. We select NPDIS over WIS and CWPDIS because the former produces an *exact* HCOPE method. For approximate HCOPE, we also consider using CWPDIS with BCa.

Since $\text{HCOPE}_{\text{BEA}}^{\dagger}$ is only defined for † = IS, we use IS with BEA. However, this results in an unfair comparison to CUT—is the better performance of $\text{HCOPE}_{\text{CUT}}^{\text{NPDIS}}$ due to its use of CUT or NPDIS? To answer this, we also show results for $\text{HCOPE}_{\text{CUT}}^{\text{IS}}$. The combinations of † (importance sampling variant) and ‡ (concentration inequality) that we use are presented in Table 4.2.

We present results using three domains: the gridworld described in Section 2.5, the canonical mountain car domain (Sutton and Barto, 1998), and a digital marketing example using real data. Unless otherwise specified, the plots that we show depict the 95% confidence lower bound on $\rho(\pi_e)$ produced by the various HCOPE approaches when using different numbers of trajectories, $n_{\mathcal{D}}$ (specified on the horizontal axis). The plots use a logarithmic horizontal axis. The curves are all averaged over 100 trials and standard error error bars are included. Since a higher lower bound is better, the higher the curves the better. For clarity, we sometimes show multiple versions of the same plot with different scales on the vertical axis. The exact HCOPE methods use dashed or dotted lines while the approximate HCOPE methods use solid lines. The legend entry for "True" denotes the true value of $\rho(\pi_e)$, which is being lower bounded.

The most important trend to notice is that BCa2 always gives the tightest lower bounds, while CUT gives the tightest lower bounds of the exact HCOPE methods.

### 4.10.1 Gridworld

The gridworld POMDP is described in Section 2.5, and here we use the same five policies that were used in the previous chapter and which are described in Section 3.12. The results for various behavior and evaluation policies are presented in Figures 4.3, 4.4, 4.5, 4.6, and 4.7. Notice the following general trends:

1. **Approximate HCOPE:** BCa2 gives far tighter bounds than the other methods, which showcases the power of BCa when combined with CWPDIS. Also, BCa outperforms TT in every case. BEA performs the worst of the approximate HCOPE methods, and is also worse than CUT and CUT+IS (which are exact HCOPE methods).

2. **Exact HCOPE:** CH, MPeB, and BM perform poorly—there were no experiments where they returned lower bounds above zero, which is a trivial lower bound. This is to be expected because they all have strong dependencies on the

range of the random variables. Also notice that although CH sometimes begins with better lower bounds than MPeB, MPeB eventually catches up because of its better dependence on the range of the random variables.

AM performs much better than CH, MPeB, and BM, as is expected because it has no direct dependence on the range of the random variables. However our concentration inequality, CUT, performs the best in almost every case—only losing to AM in some cases when using a small number of trajectories (where the lower bounds are so loose that they are likely of little practical use).

Even though BEA is only an approximate HCOPE method, it performs worse than CUT+IS in every experiment (recall that we compare to CUT+IS rather than CUT to avoid conflating the benefits of CUT with the benefits of NPDIS). This suggests that our automatic optimization of the threshold works better than their *ad hoc* scheme for selecting the threshold (although it may also be due to our collapsing of the random variables as opposed to their clipping of the importance weights).

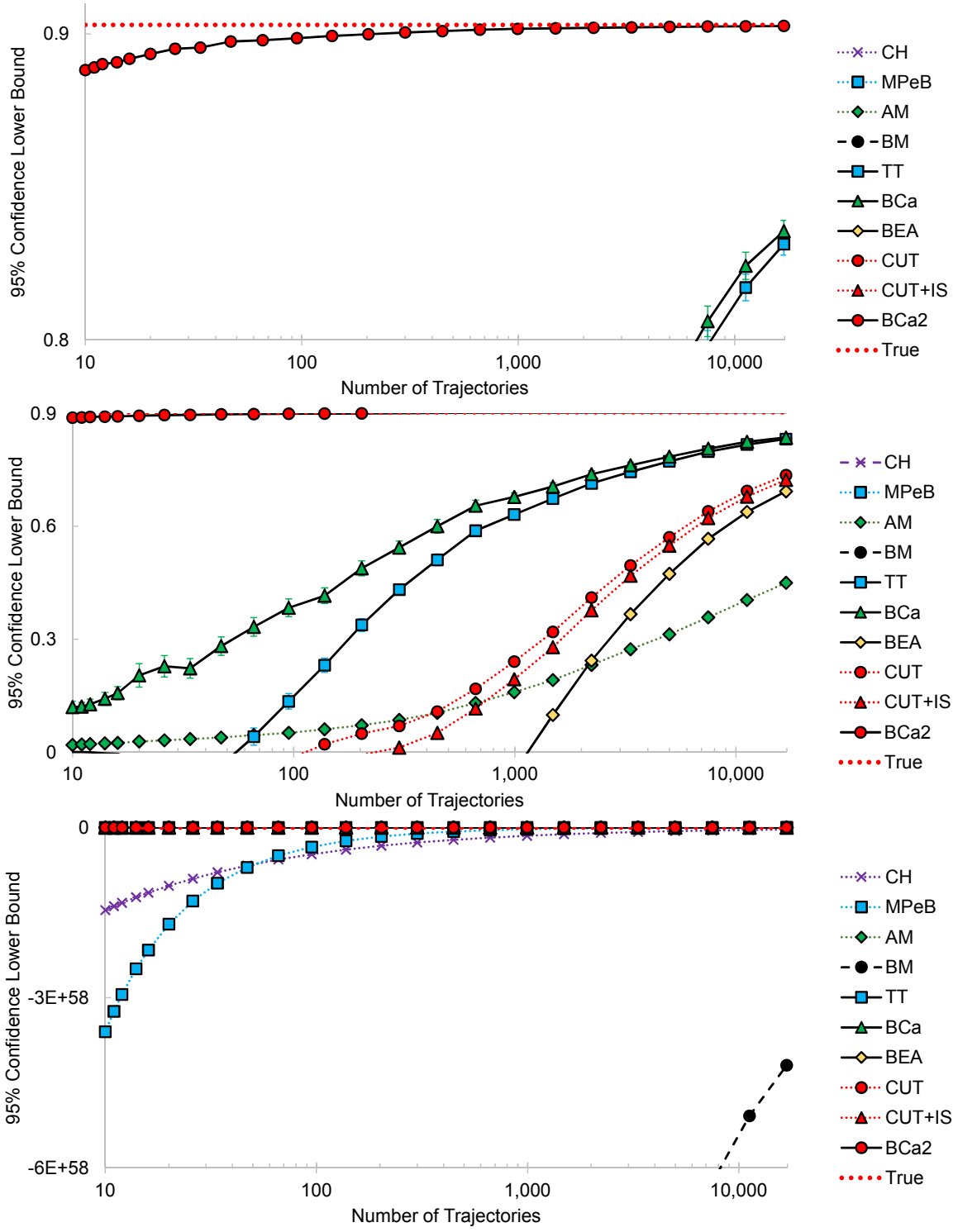


**Figure 4.3.** Lower bounds on $\rho(\pi 2)$ using trajectories from $\pi 3$.

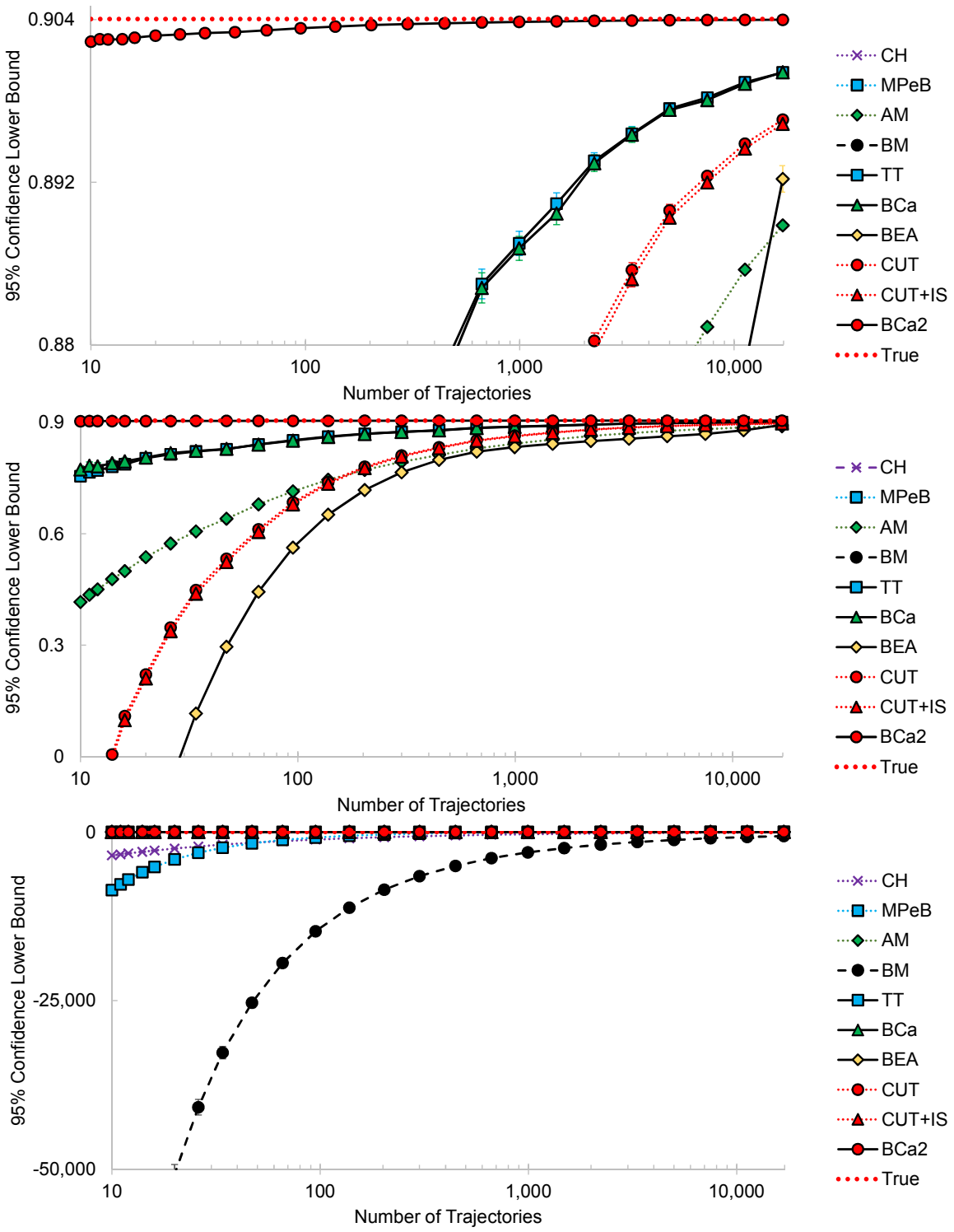


**Figure 4.4.** Lower bounds on $\rho(\pi 3)$ using trajectories from $\pi 2$.

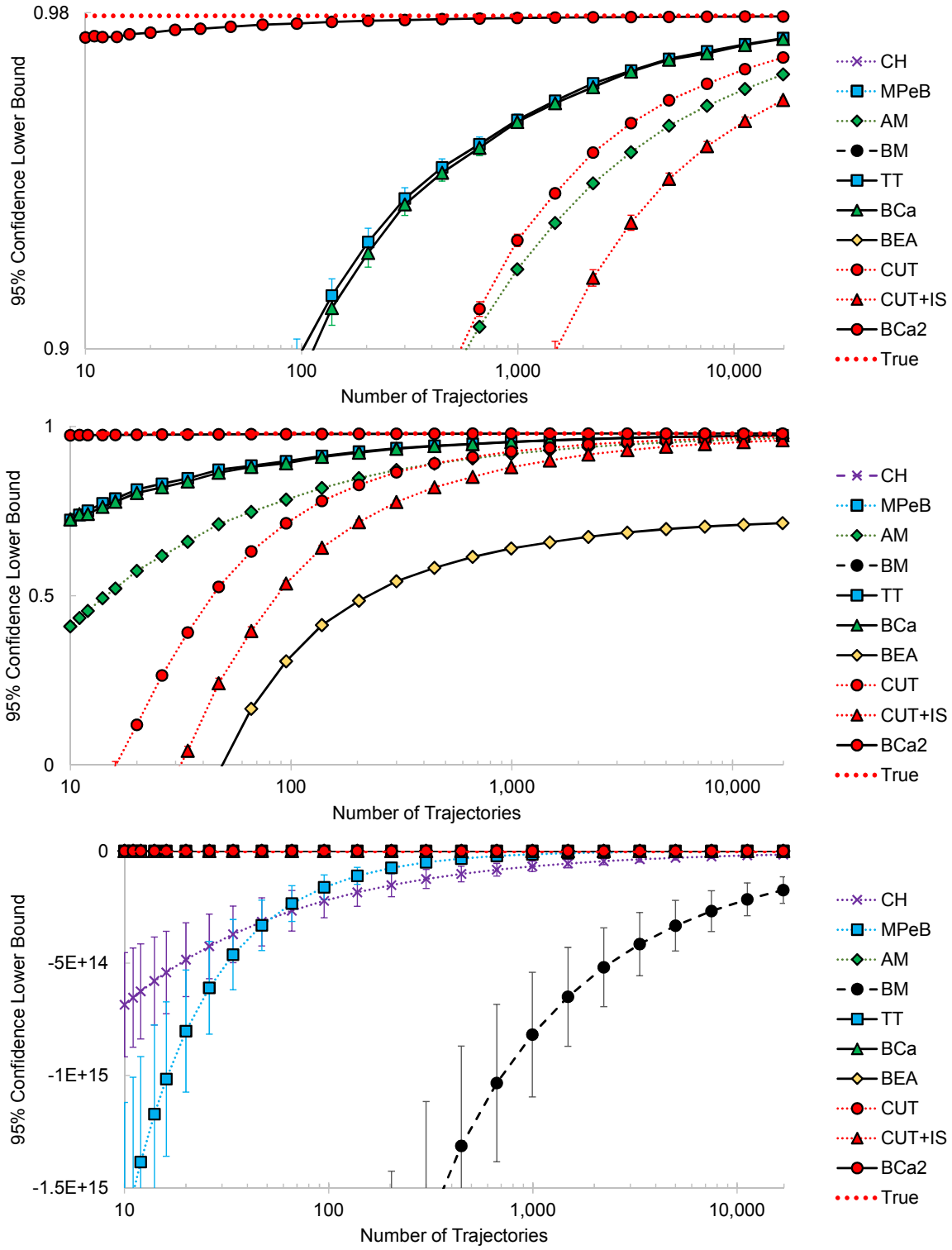


**Figure 4.5.** Lower bounds on $\rho(\pi 5)$ using trajectories from $\pi 4$.

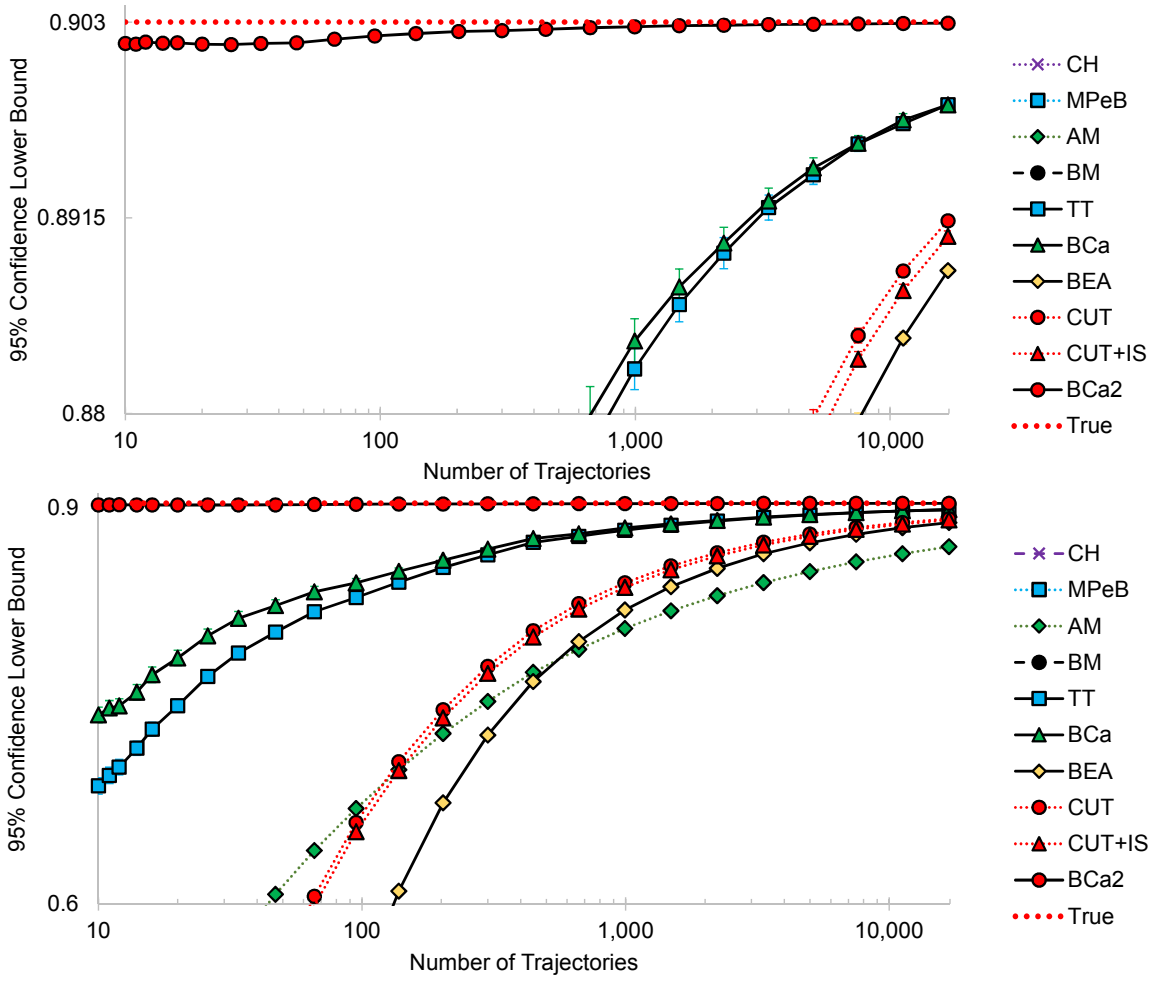


**Figure 4.6.** Lower bounds on $\rho(\pi 2)$ using trajectories from $\pi 3$.

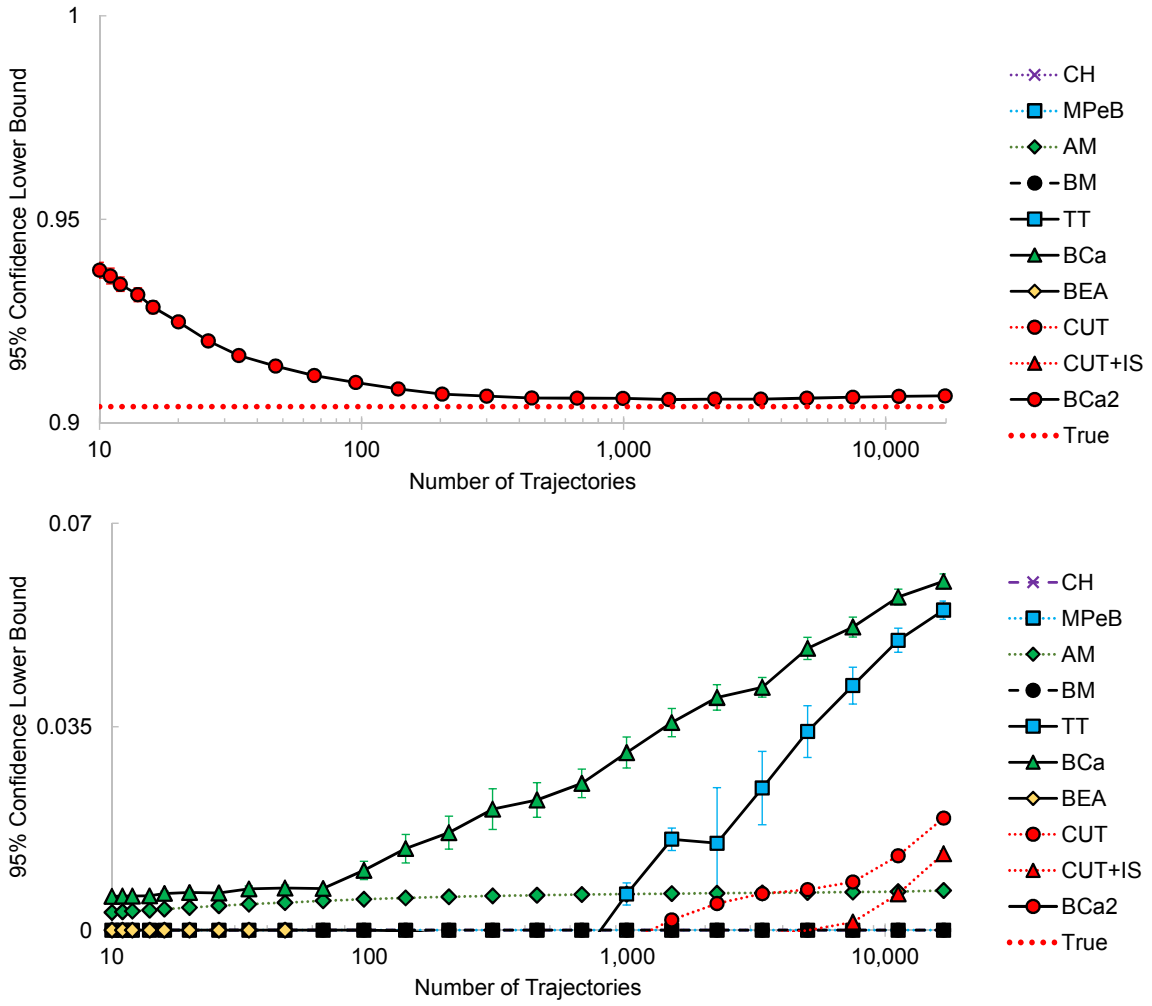


**Figure 4.7.** Lower bounds on $\rho(\pi 3)$ using trajectories from $\pi 4$.

Consider Figure 4.7 in more detail. We saw in Figure 3.15 that off-policy evaluation is challenging in this setting with $\pi_b = \pi 4$ and $\pi_e = \pi 3$. This means that HCOPE is also challenging. Notice the large discrepancy between the performance of BCa2 and the other methods. This discrepancy is present partly because CWPDIS produces much better estimates of $\rho(\pi 3)$ than NPDIS (again, see 3.15). This is also the only plot in this chapter where a mean lower bound produced by BCa2 is larger than $\rho(\pi_e)$.

To understand why BCa2 produces lower bounds that are too large, first recall from Figure 4.1 that BCa can produce expected error rates larger than desired when the random variables have a heavy upper tail. However, the too-large lower bounds produced by BCa2 in Figure 4.7 are primarily due to a different cause: recall from the previous chapter that the weighted importance sampling variants, WIS and CWPDIS, begin (when using a single trajectory) as unbiased estimators of $\rho(\pi_b)$, not $\rho(\pi_e)$. As more trajectories are provided, they change from unbiased estimators of $\rho(\pi_b)$ towards being unbiased estimators of $\rho(\pi_e)$. So, when using only a few trajectories, the CWPDIS estimator used by BCa2 is a better estimator of $\rho(\pi 4) \approx 0.97$, and so BCa2 produces high confidence lower bounds on $\rho(\pi 4)$, which are not valid lower bounds on $\rho(\pi 3)$. As the number of trajectories increases, CWPDIS shifts towards correctly estimating $\rho(\pi 3)$, and so BCa2 produces lower bounds that are more appropriate.

Although the behavior of BCa2 in this case (Figure 4.7) is concerning, it is not damning. Notice that if $\rho(\pi_e) \geq \rho(\pi_b)$, then there is no concern since a lower bound on $\rho(\pi_b)$ is also a lower bound on $\rho(\pi_e)$. In the other case, where $\rho(\pi_e) < \rho(\pi_b)$, BCa2 produces high confidence lower bounds that are too large, *however*, they are reasonable lower bounds of $\rho(\pi_b)$. Later we will use HCOPE methods to search for policies that are predicted to outperform the current behavior policy with high confidence. That is, we will search for a policy $\pi$ such that we can guarantee that $\rho(\pi) > \rho(\pi_b)$ (where an approximation to $\rho(\pi_b)$ may be used). If $\rho(\pi) \geq \rho(\pi_b)$, then

BCa2 (and in general using CWPDIS as a foundation for HCOPE) does not tend to produce lower bounds that are too large on average. However, if $\rho(\pi) < \rho(\pi_b)$, then even though BCa2 may tend to produce lower bounds that are too large, they will also tend to be viable lower bounds on $\rho(\pi_b)$, and therefore will not result in the erroneous conclusion that $\rho(\pi) > \rho(\pi_b)$ with high confidence.

To emphasize this point, we plot the empirical error rates of the different methods in Figures 4.8, 4.9, 4.10, 4.11, 4.12. We compute this error rate in two ways. First we compute the number of times that each method produced a lower bound larger than $\rho(\pi_e)$ and divide it by the total number of trials (100). We refer to this statistic as the *empirical error rate*. Next we computed the number of times that each method produced a lower bound larger than $\max\{\rho(\pi_e), \rho(\pi_b)\}$, which is a better indicator of how often each method will produce an error in the high confidence off-policy *improvement* work in the next chapter. We refer to this statistic as the *empirical severe error rate*. We only plot the empirical severe error rate for the cases where $\rho(\pi_b) > \rho(\pi_e)$, since it is the same as the empirical error rate otherwise.

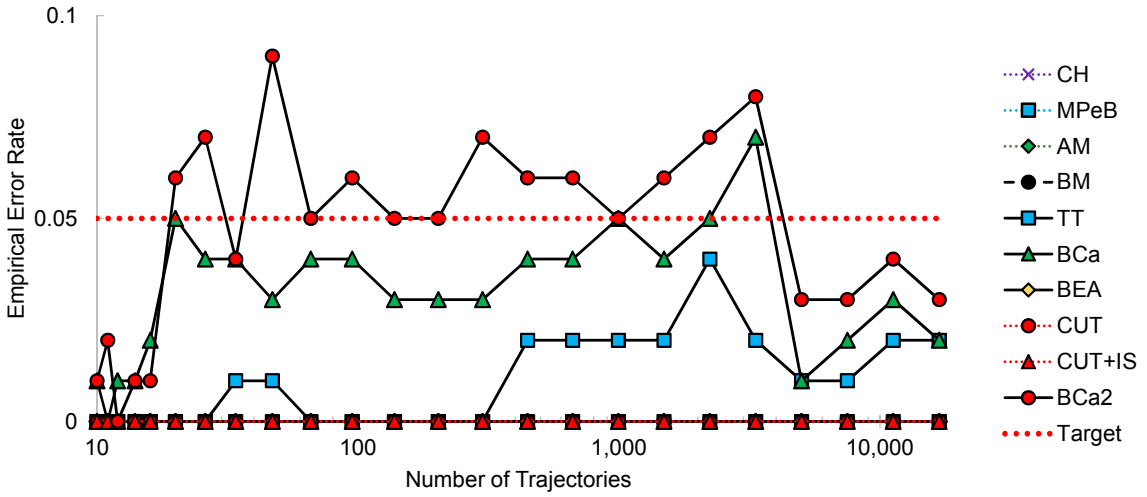


**Figure 4.8.** Empirical error rate when using $\pi_b = \pi 1$ and $\pi_e = \pi 2$.

Notice that in each plot showing empirical error rates, BCa, and TT have error rates of *approximately* 0.05, not error rates *upper bounded* by 0.05 (recall that we

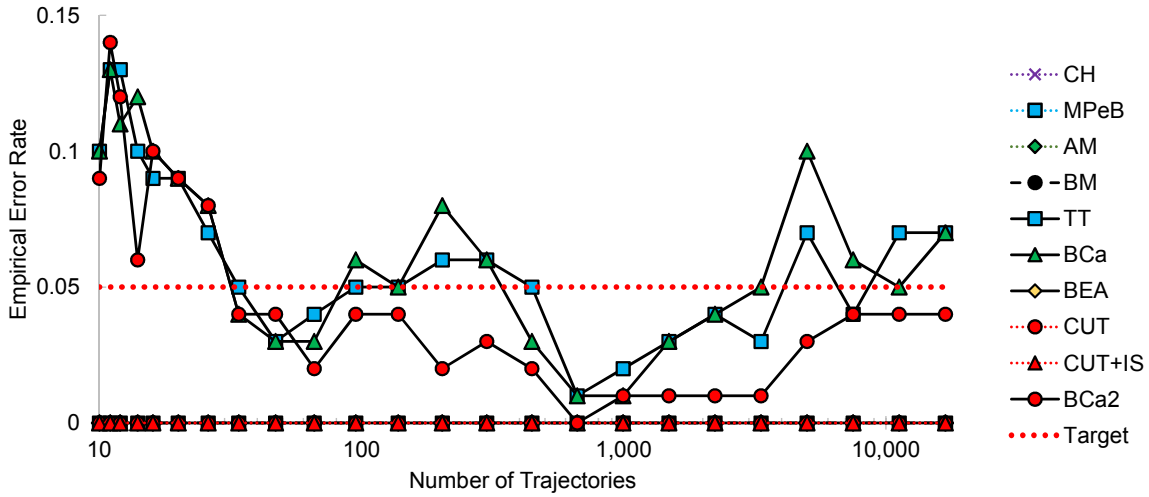


**Figure 4.9.** Empirical error rate when using $\pi_b = \pi 2$ and $\pi_e = \pi 3$.

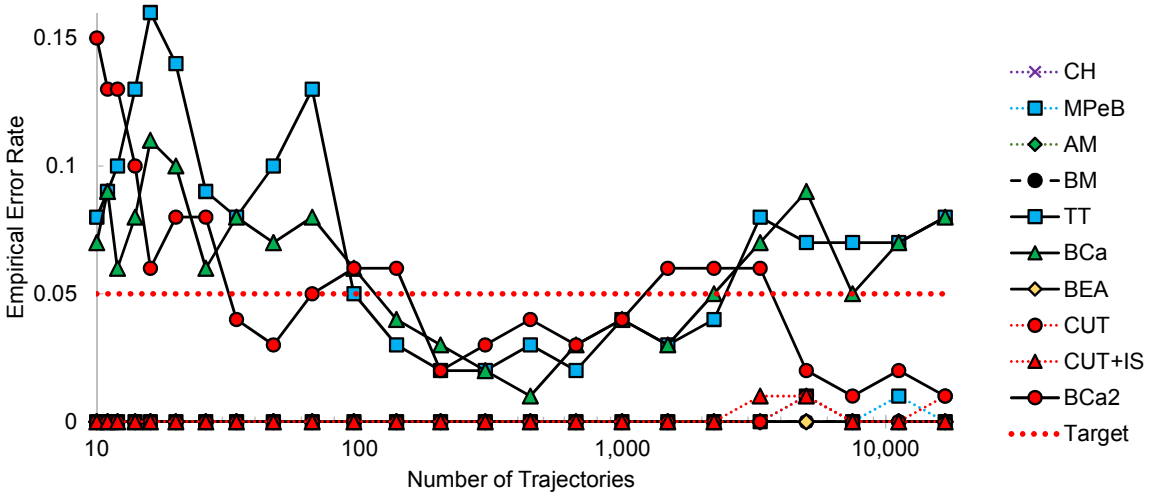


**Figure 4.10.** Empirical error rate when using $\pi_b = \pi 4$ and $\pi_e = \pi 5$.

used $\delta = 0.05$). For approximate HCOPE, we can therefore think of $\delta = 0.05$ as the *target* error rate, which is marked in the plots. BCa2 has a similar error rate in Figures 4.8, 4.9, and 4.10, where $\rho(\pi_e) \geq \rho(\pi_b)$. However, in Figures 4.11 and 4.12 the empirical error rate of BCa2 is significantly too large when there are few trajectories (with more trajectories its error rates are similar to those of BCa and TT). However, in these figures, the empirical *severe* error rate of BCa2 remains reasonable, which

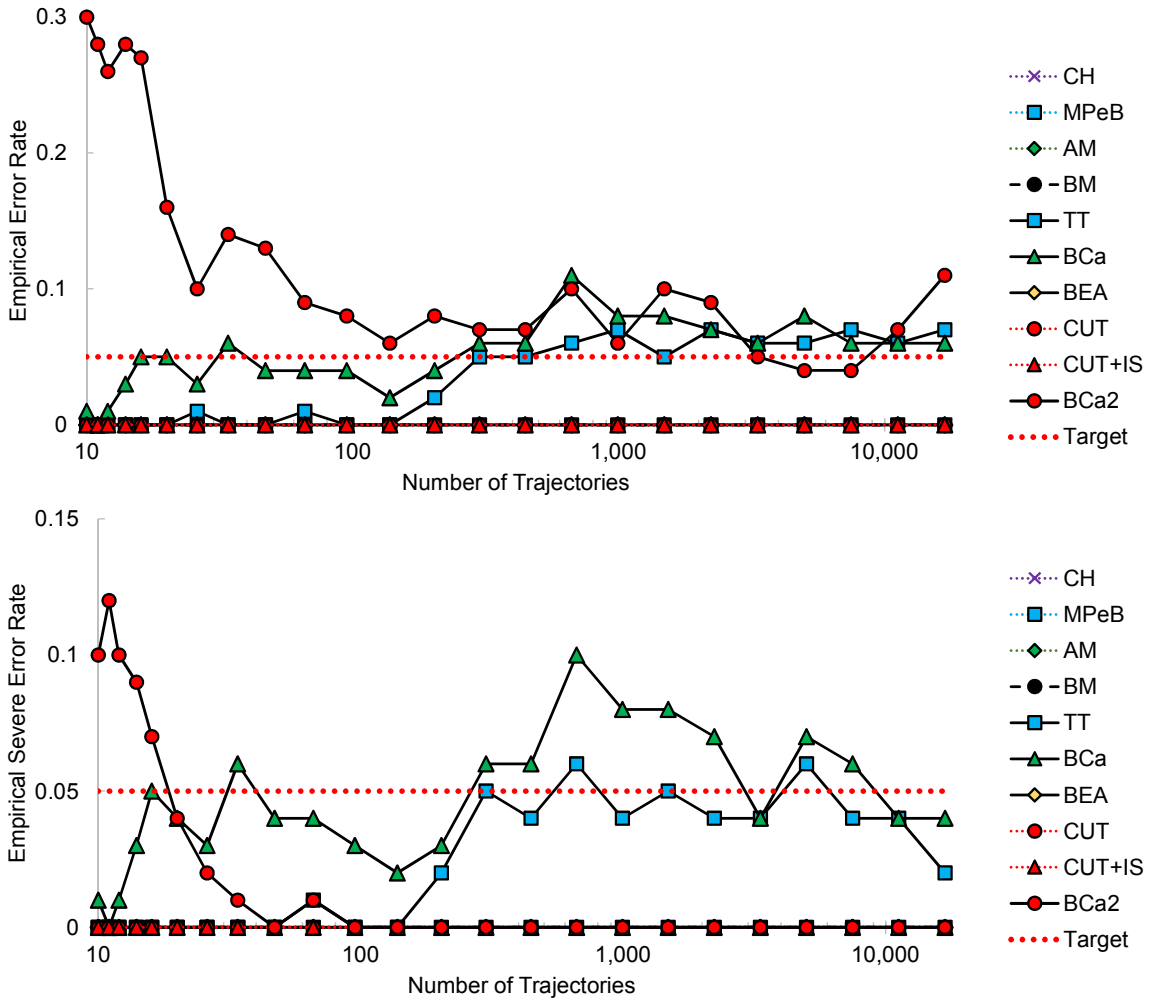


**Figure 4.11.** Empirical error rate and empirical severe error rate when using $\pi_b = \pi 3$ and $\pi_e = \pi 2$.

means that it will still produce reasonable high confidence lower bounds for our later applications.

Also notice that these plots support the theory—the exact HCOPE methods produce lower bounds that have error rates below 0.05 at all times. It is also evident that the exact HCOPE methods are overly conservative since their error rates are near zero in every plot. This highlights the difference between the exact and approximate HCOPE methods. The exact methods provide an actual high confidence lower bound on $\rho(\pi_e)$, and are typically overly conservative. The approximate HCOPE methods

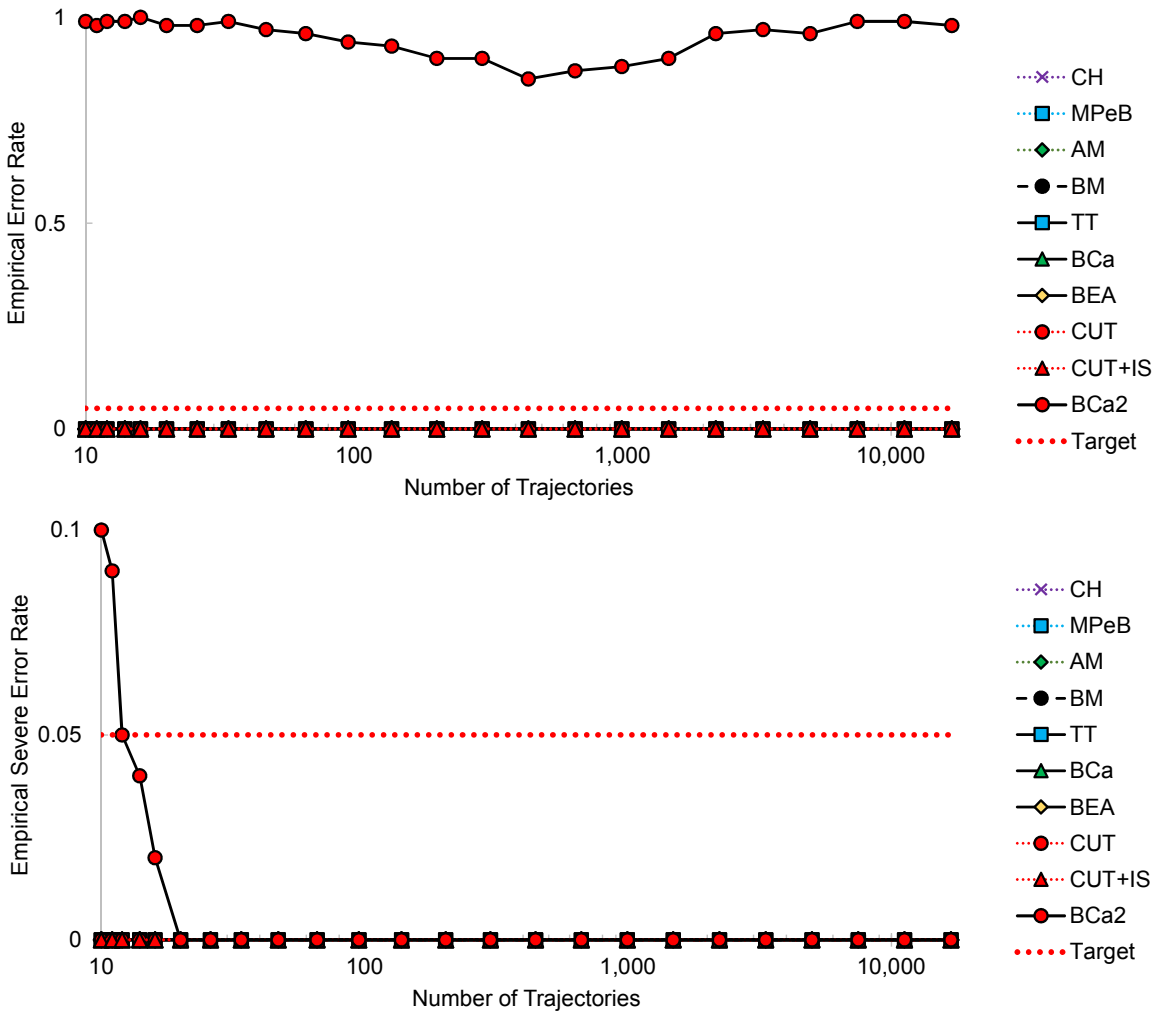


**Figure 4.12.** Empirical error rate and empirical severe error rate when using $\pi_b = \pi 4$ and $\pi_e = \pi 3$.

produce error rates much closer to the allowed level, however, they sometimes produce error rates that are too large.

### 4.10.2 Mountain Car

Next we validated our HCOPE methods on the canonical *mountain car* domain (Sutton and Barto, 1998), modified so that each action chosen by the agent is held fixed for 20 time steps. This modification shortens the trajectories to avoid numerical instability issues (particularly numerical overflows within some of the concentration inequalities that depend on the range of the random variables). In the mountain car domain, the agent is controlling a car via three actions: {forward, reverse, none}, which specify the direction that the car accelerates (with none denoting no acceleration caused by the agent). The car begins at the bottom of a valley, and the goal is to reach the top of the hill in front of the car. However, the car does not have enough power to drive straight up the hill. So, the agent must learn to reverse up the hill behind it before accelerating forwards.

The domain is depicted in Figure 4.13. The agent observes its current horizontal position and velocity. The exact dynamics of the domain are specified by Sutton and Barto (1998), with the one exception that we make each action last for 20 time steps (one time step in our domain is 20 time steps in their domain). Each episode terminates when the agent reaches the goal, and a reward of $-1$ is provided at each time step to motivate the agent to reach the goal as quickly as possible. We force every episode to terminate within $L = 100$ time steps (2,000 time steps in the original mountain car domain).

An optimal policy for this domain requires 5 time steps to reach the goal, which corresponds to 100 time steps in the original domain. In our experiments, the behavior policy, $\pi_b$, is already a decent policy that requires an average of 28 time steps to reach the goal. The evaluation policy, $\pi_e$, is a significantly better policy (found using Sarsa($\lambda$) (Sutton and Barto, 1998)), that requires an average of 7 time steps to reach the goal. Figure 4.14 depicts the results of using different numbers of trajectories with different exact and approximate HCOPE methods to lower bound $\rho(\pi_e)$ with

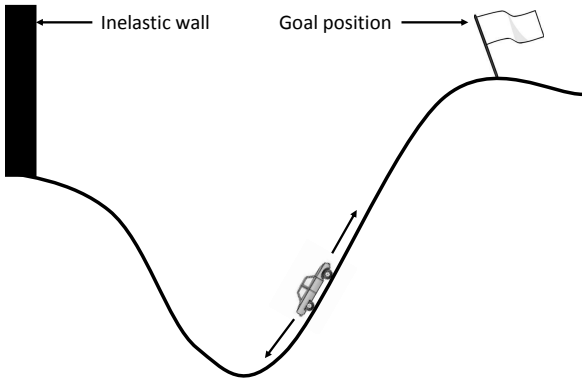


**Figure 4.13.** Graphical depiction of the mountain car domain.

95% confidence. Figure 4.15 presents the empirical error rates, which are similar to those of the gridworld experiments.

We observe the same trends as before—BCa2 gives the tightest bounds, followed by BCa and TT, which are also approximate HCOPE methods. The best exact HCOPE method is CUT, followed closely by CUT+IS. The next best is BEA (an approximate method), and then AM (an exact method). CH, MPeB and BM are far behind the others. Although CH produces a tighter lower bound than MPeB when there are few trajectories, MPeB performs better when more trajectories are available because of its better dependence on the range of the random variables.

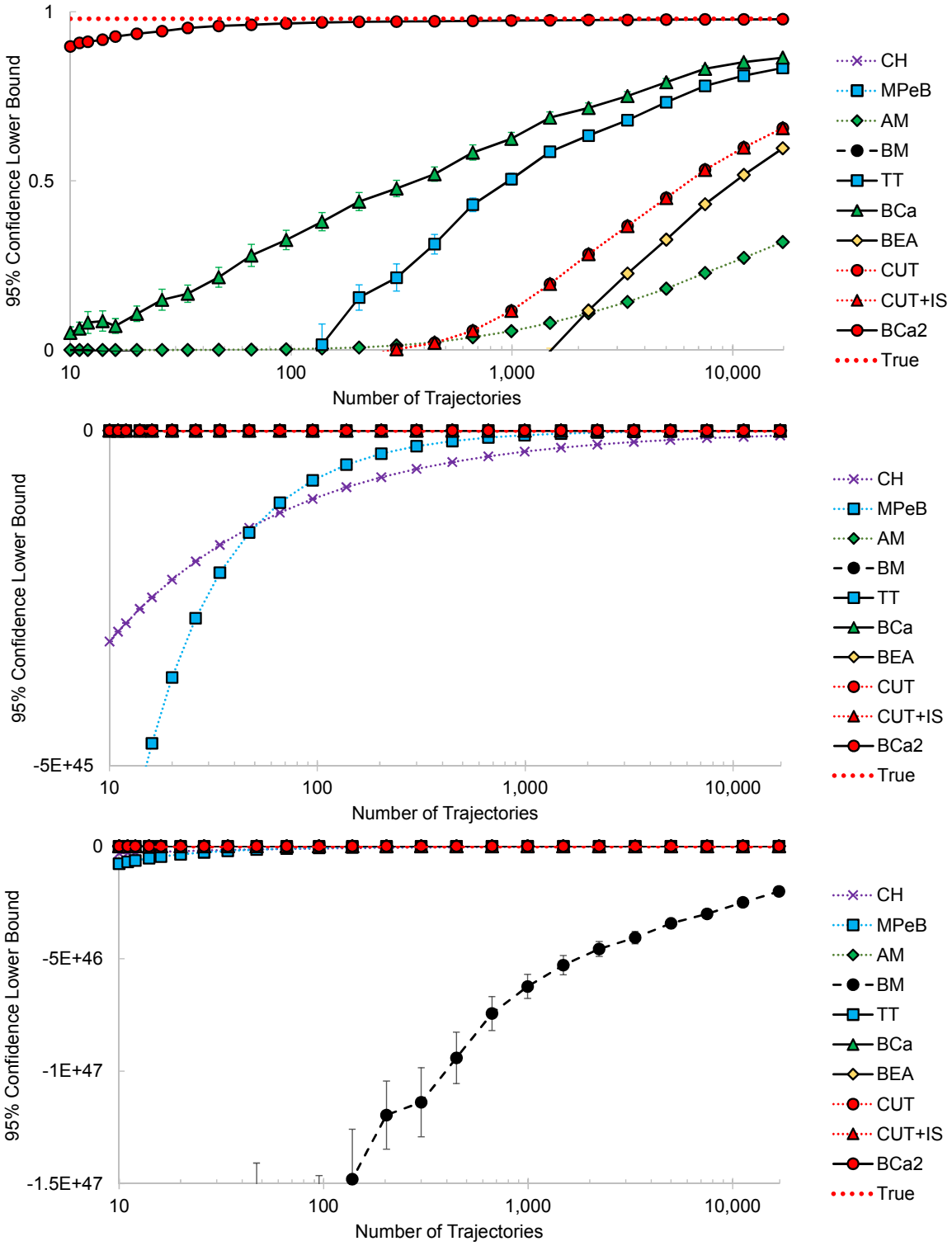


**Figure 4.14.** Exact and approximate HCOPE results on the mountain car domain.

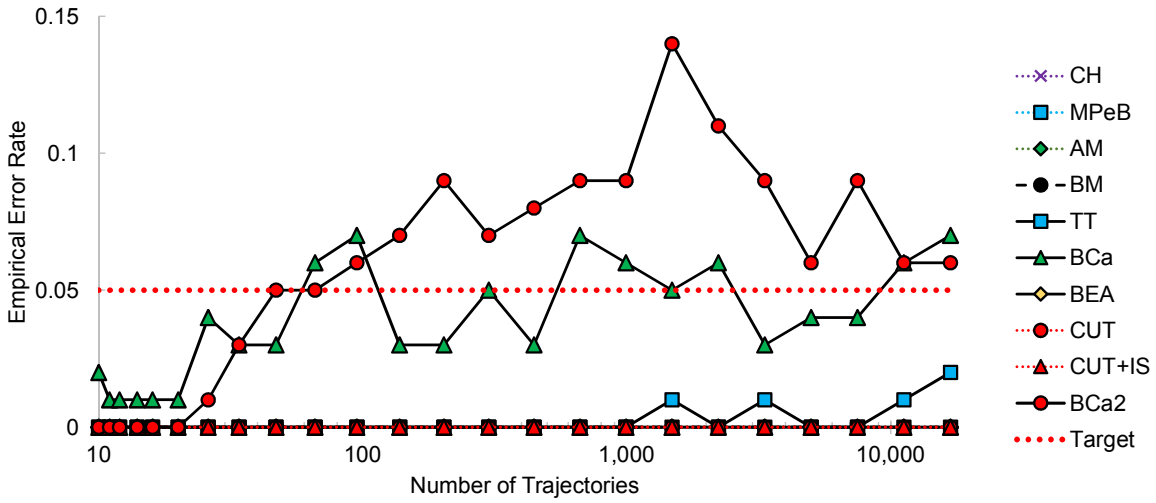


**Figure 4.15.** Empirical error rates on the mountain car domain.

### 4.10.3 Digital Marketing Domain

Our final empirical study uses a digital marketing domain that is based on real-world data from a Fortune 20 company. Adobe Marketing Cloud is a powerful set of tools that allows companies to fully leverage digital marketing using both automated and manual solutions. It has been deployed widely across the internet, with approximately seven out of every ten dollars transacted on the web passing through one of Adobe's products. Adobe Target, one of the six core components of Adobe Marketing Cloud, allows for automated user-specific targeting of advertisements and campaigns. When a user requests a webpage that contains an advertisement, the decision of which advertisement to show is computed based on a vector containing all of the known features of the user. A trajectory consists of the history of interactions with a single user.[3]

This problem tends to be treated as a bandit problem, where an agent treats each advertisement as a possible action and attempts to maximize the probability

[3]For simplicity, we do not consider issues that arise due to the parallel and continuing nature of the problem—we ignore that trajectories occur in parallel (Silver et al., 2013) and that there is no way to determine whether a trajectory has ended (users may return after an arbitrarily long hiatus).

that the user clicks on the advertisement. Although this greedy approach has been successful, it does not necessarily also maximize the total number of clicks from each user over his or her lifetime. It has been shown that more far-sighted reinforcement learning approaches to this problem can improve significantly upon bandit solutions (Theocharous and Hallak, 2013).

In order to avoid the large costs associated with deployment of a bad policy, in this application it is imperative that new policies proposed by RL algorithms are thoroughly evaluated prior to deployment. Because off-policy evaluation methods are known to have high variance, estimates of performance without associated confidences are not sufficient. However, our HCOPE methods *can* provide sufficient evidence supporting the deployment of a new policy to warrant its execution.

We used real data, captured with permission from the website of a Fortune 20 company that receives hundreds of thousands of visitors per day, and which uses Adobe Target, to train a simulator using a proprietary in-house system identification tool at Adobe Research. The simulator produces a vector of 10 real-valued features that provide a compressed representation of all of the available information about a user.[4] The advertisements are clustered into two high-level classes that the agent must select between. After the agent selects between these two classes of advertisements, the user either clicks (reward of +1) or does not click (reward of 0) and the feature vector describing the user is updated. We selected $L = 10$ (recall that $L$ is the maximum trajectory length) and $\gamma = 1$. We selected $L = 10$ because few visitors returned to the website more than 10 times. Notice that using $L = 10$ is a significant improvement over the myopic bandit approach, which effectively uses $L = 1$.

[4]The digital marketing data used in this dissertation comes from a different company than the data used in our original HCOPE publication (Thomas et al., 2015c). Most notably, the feature vectors describing users have been refined down to 10 features from 31.

The (unnormalized) expected return of a policy for a digital marketing application is known as its *lifetime value* (LTV):

$$\text{LTV}(\pi) \coloneqq \mathbf{E}\left[\sum_{t=1}^{L} R_t \middle| \pi\right].$$

To make our results more easily interpretable, we report the expected *click-through rate* (CTR):

$$\text{CTR}(\pi) \coloneqq \mathbf{E}\left[\frac{\text{clicks}}{\text{visits}} 100\%\right].$$

So, a CTR of 0.004 means that 4 out of every thousand user visits results in a click on an advertisement. LTV is more appropriate than CTR for evaluating policies for digital marketing because it can account for how different policies can result in different numbers of user visits (Theocharous et al., 2015a,b). However, since we use a fixed horizon of $L = 10$, the two are equal: $\text{CTR}(\pi) = \text{LTV}(\pi)/L$.

The problem of improving the current policy for advertisement selection is a particularly challenging problem because the reward signal is sparse—users usually do not click on the advertisements. If each of the two clusters of advertisements are shown with probability 0.5, then the average probability that a user will click is 0.0027, i.e., the CTR of this policy is 0.0027. This means that most trajectories provide no feedback. Also, whether a user clicks or not is close to random, so returns have high variance.

We selected a reasonable initial policy, $\pi_b$, which has a CTR of 0.004322. We computed a better policy,[5] $\pi_e$, with CTR $\approx 0.00496$, which is a $\approx 14.8\%$ improvement. Figure 4.16 shows the 95% confidence lower bound on the CTR of $\pi_e$ produced using the various HCOPE methods and different numbers of trajectories, and Figure 4.17

[5]The method used to compute $\pi_e$ is of little importance—it could have been produced by any RL algorithm. In this case we used Sarsa($\lambda$) with a manually tuned step size and eligibility trace decay parameter.

shows the corresponding empirical error rates. Yet again we see the same trends—BCa2 performs the best, followed by BCa and TT. Of the exact HCOPE methods, CUT performs the best, followed closely by CUT+IS.

### 4.10.4 Risk Quantification Plot

In this section we introduce *risk quantification plots* (RQPs), which quantify the risk associated with deploying the evaluation policy, $\pi_e$. Reporting the 95% confidence lower bound on $\rho(\pi_e)$ (or the lower bound on CTR($\pi_e$) for the digital marketing application) provides some insight into how well $\pi_e$ will likely perform. However, in the cases where the high confidence lower bound is larger than $\rho(\pi_e)$, it says nothing about how bad the performance of $\pi_e$ might be. To fully quantify the risk associated with deploying $\pi_e$ requires a high confidence lower bound on $\rho(\pi_e)$ to be computed for every possible value of $\delta$ (or a dense sampling of $\delta \in [0, 1]$). A RQP is a plot that shows the observed performance of the current policy ($\pi_b$ in this case) as well as the lower bound on $\rho(\pi_e)$ for a dense sampling of values of $\delta$. A RQP may also include markings that denote off-policy (not high confidence) predictions of the evaluation policy's performance.

To emphasize that our approach can handle multiple behavior policies, we use the same $\pi_b$ as was used in Figure 4.16 to generate 90% of the historical data (recall that the CTR of $\pi_b$ is $\approx 0.004322$). We then use the policy, $\pi_b'$, that always selects between the two advertisement clusters with equal probability to generate the other 10% of the data (recall that the CTR of $\pi_b'$ is $\approx 0.0027$). This usage of the random policy ($\pi_b'$) for a small subset of the users is common in industry to allow for better statistical analysis of historical data (Theocharous et al., 2015a,b). We used 100,000 trajectories to generate the RQP because 100,000 users is a realistic amount of historical data for this particular client. We used the same evaluation policy, $\pi_e$, as was used in Figure 4.16, which has a CTR of $\approx 0.00447$. We used $\text{HCOPE}_{\text{CUT}}^{\text{NPDIS}}$ to produce the

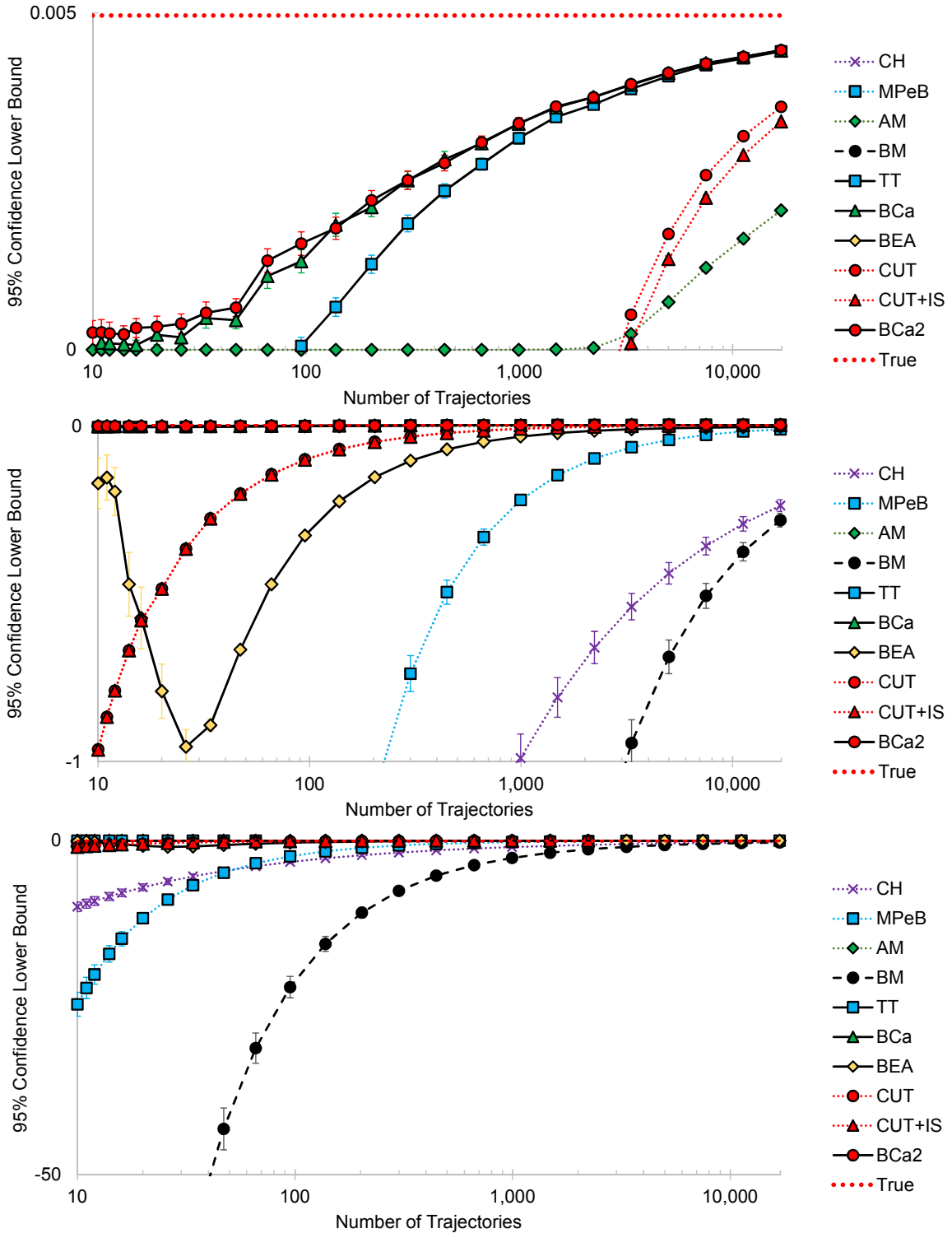


**Figure 4.16.** Exact and approximate HCOPE results on the digital marketing domain.

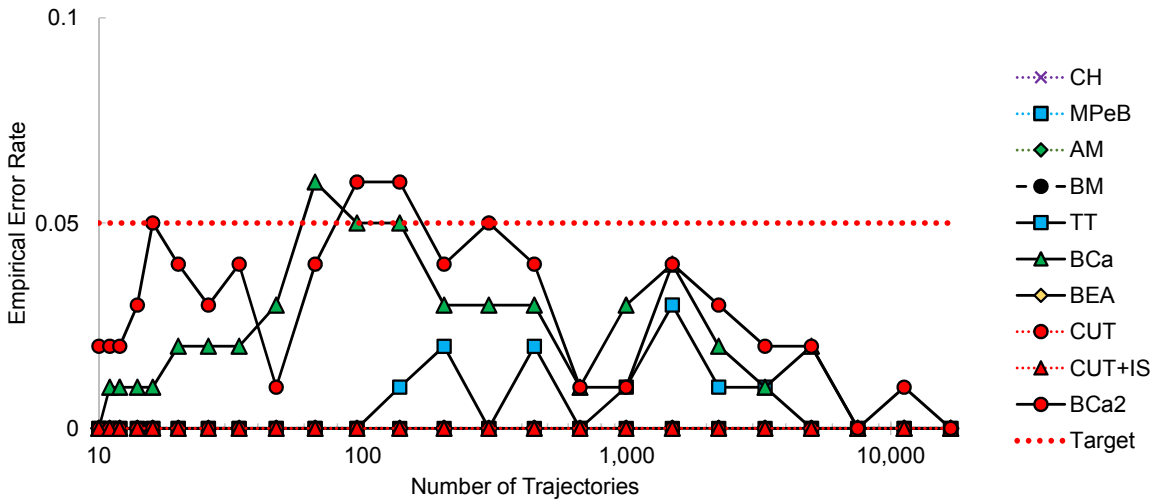


**Figure 4.17.** Empirical error rates on the digital marketing domain.

high confidence lower bounds, since it was the best performing of the exact HCOPE methods.

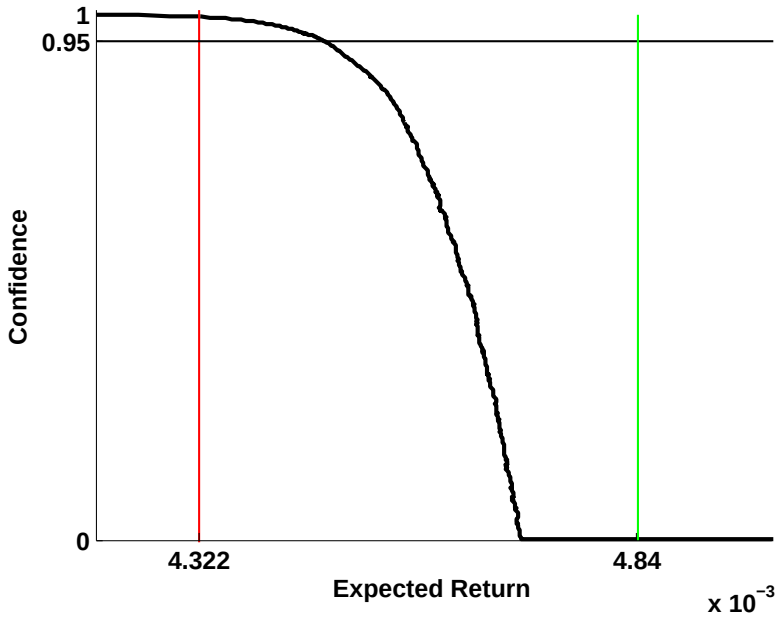


**Figure 4.18.** Risk quantification plot for the digital marketing domain, generated using $\text{HCOPE}_{\text{CUT}}^{\text{NPDIS}}$ with 100,000 trajectories. The 95% confidence lower bound on $\rho(\pi_e)$ is 0.00447. The vertical red line at 0.00432 denotes the observed CTR of $\pi_b$, and the vertical green line at 0.00484 denotes the prediction from CWPDIS of the CTR of $\pi_e$.

This RQP is exceptionally compelling evidence supporting the deployment of the policy $\pi_e$ in place of $\pi_b$. It shows that $\pi_e$ will outperform $\pi_b$ with very high confidence (greater than 95%), and also that it will be a significant improvement upon $\pi_b$ with high confidence (the 95% confidence lower bound on the CTR is $\approx 0.00447$). Also, CWPDIS predicts that $\rho(\pi_e) \approx 0.00484$, which is an 8% improvement. Also, because we are using a simulator, we can easily deploy the evaluation policy for a large number of trajectories to estimate its true performance. Figure 4.19 shows the RQP augmented to include the "true" CTR of $\pi_e$ (computed from 2 million on-policy trajectories). Notice that, because the behavior policy has lower CTR than the evaluation policy, CWPDIS underestimates the CTR of the evaluation policy.

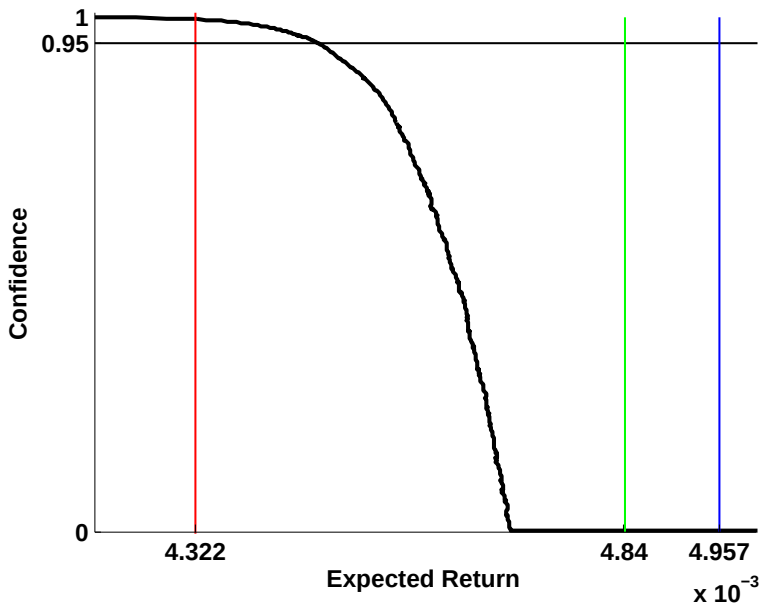


**Figure 4.19.** Figure 4.18, modified to also show the "true" CTR of $\pi_e$ (the vertical blue at 0.004957).

Notice that the RQP is not restricted to the off-policy setting. If a new policy is deployed to a small portion of the users (or for a brief duration), the machinery that we have proposed can be used to determine whether or not the observed performance

of the policy warrants its continued use. Furthermore, any historical data from past policies can be merged with on-policy data to produce a new data set, $\mathcal{D}$, that can also be used. Lastly, notice that the RQP does *not* ensure that every bound given holds simultaneously with the specified confidence. This is due to the problem of *multiple comparisons*, which is discussed in more detail in the next chapter.

At the time of writing this dissertation, researchers at Adobe Research continue to use RQPs to evaluate new policies, and are working to transfer the technology to the product team for Adobe Test and Target.

## 4.11 Discussion and Conclusions

We have presented tools for *high confidence off-policy evaluation* (HCOPE). These tools leverage historical data to provide practical guarantees about the performance of any new policy. Most importantly, they can be used to instill someone with confidence that the policy proposed by any RL algorithm will actually perform well, without requiring it to be deployed. We also introduced *risk quantification plots*, which can use a combination of HCOPE and off-policy evaluation methods like CWPDIS to provide compelling evidence supporting the use of a new policy. The primary drawback of the HCOPE methods that we propose is that they are based on the (often false) assumption that the environment is a POMDP. Deriving confidence bounds on $\rho(\pi_e)$ using weaker stationarity assumptions would be an interesting direction of future research.

Although the bringing together of existing tools to craft HCOPE algorithms is one of the major contributions of this chapter, we also derived a new concentration inequality, which we call the *collapsed upper tail* (CUT) inequality, and showed that it outperforms all of the existing concentration inequalities that we compared it with (with the one exception that the concentration inequality of Anderson (1969) and

Massart (1990) (AM) performs better that the CUT inequality when little data is available and the resulting bounds are both loose).[6]

Two of the new variants of importance sampling, NPDIS and CWPDIS, which we derived in the previous chapter, serve as the foundation of the HCOPE methods that we presented in this chapter. Not only do these methods perform better than IS and WIS (which were used in our original publication of this work (Thomas et al., 2015c)), but we showed in Section 3.11 that an HCOPE method that uses NPDIS does not require the behavior policies to be stochastic everywhere (Assumption 1).

There are two other clear avenues of future work. First, our scheme for automatically selecting the threshold parameter, $c$, in the CUT inequality is *ad hoc*. This could be improved, especially by a method for adaptively determining how many samples should be used to select $c$. Second, in our experiments the AM inequality was the best method after the methods based on CUT. If the AM inequality could be modified to use a statistic other than the Dvoretsky-Kiefer-Wolfowitz inequality (Dvoretzky et al., 1956), then it may outperform the CUT inequality.

We present in Figure 4.20 a decision diagram to assist with selecting which variant of HCOPE to use for a specific application.

[6]This is not a concern for us, so we will use the CUT inequality in the remainder of this dissertation. If this is a concern for some application, then the CUT and AM inequalities can be combined using the union bound to produce a $1 - \delta$ confidence bound: $\max\{\text{CUT}(X_1, \ldots, X_n, \delta/2), \text{AM}(X_1, \ldots, X_n, \delta/2)\}$ that has performance similar to the AM inequality when there are few samples and similar to the CUT inequality when there are more samples.

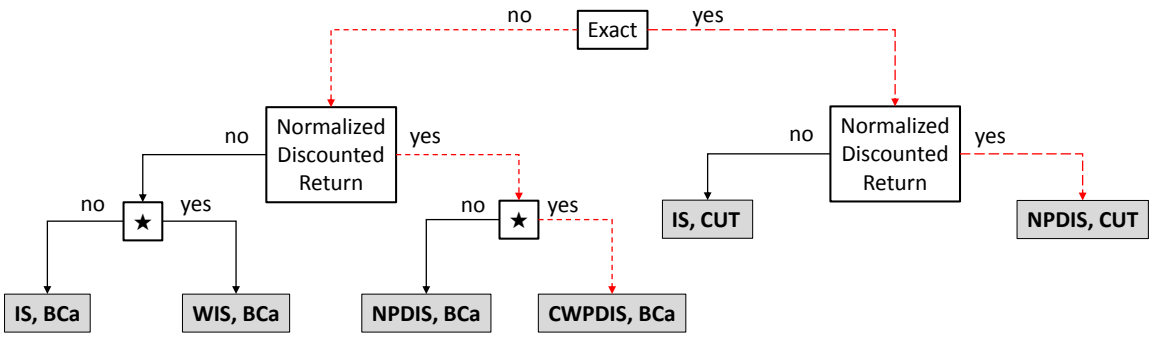


**Figure 4.20.** Decision diagram for deciding which variant of HCOPE to use. The recommended method is presented in a gray-filled box in bold. The top node corresponds to whether or not an exact HCOPE method is required (yes) or whether an approximate HCOPE method is acceptable (no). The second level nodes, "normalized discounted return," correspond to whether $R$ is defined to be the normalized discounted return (see (2.5)). The decision nodes labeled $\star$ denote the question: "Is it acceptable if the approximate HCOPE method returns lower bounds on $\max\{\rho(\pi_e), \max\{\rho(\pi_i)\}_{i=1}^{n_{\mathcal{D}}}\}$?" That is, if the performance of the evaluation policy is worse than that of the best behavior policy, is it acceptable if the lower bounds produced by the approximate HCOPE method are lower bounds on the performance of the best behavior policy? If BCa is too computationally expensive (even using smaller values of $B$ in the BCa pseudocode), then BCa can be replaced with TT in this diagram. The dotted red paths are the two that will be used in the next chapter—we will use CWPDIS with BCa for approximate HCOPE and NPDIS with the CUT inequality for exact HCOPE.

# CHAPTER 5

# SAFE POLICY IMPROVEMENT

In the previous chapter we showed how historical data can be used to lower bound the performance of a policy using *high confidence off-policy evaluation* (HCOPE) methods. We argued that these lower bounds can be used to provide the user of an RL algorithm with confidence that a new policy will perform well. However, the previous chapter was agnostic to *how* the new policy was proposed. In this chapter we investigate *how* the new policy should be chosen.

One approach is to use an existing batch off-policy reinforcement learning algorithm like fitted $Q$-iteration (Ernst et al., 2005) or least squares policy iteration (Lagoudakis and Parr, 2001). However, the policies that these methods produce, and in general the policies that are predicted to have high expected return based on the historical data, are *not* necessarily the policies that will have the highest lower bounds on their performance. If a proposed policy will only be deployed if its performance can be lower bounded by a sufficiently large value, then it (the proposed policy) should be generated based both on its predicted performance *and* a prediction of whether the lower bound on its performance will be sufficiently large to warrant deployment.

In this chapter we present batch and incremental policy improvement algorithms that balance this trade-off between predicted performance and predicted lower bound when searching for "safe" policies—policies that are guaranteed to improve upon a user-specified baseline with a user-specified confidence level. If the user specifies a baseline or confidence that is too high for the algorithm to produce a safe policy given the available data, our algorithms return No Solution Found. We call our batch

policy improvement algorithm *safe policy improvement* (SPI), and our incremental policy improvement algorithm DAEDALUS.

## 5.1 Problem Description

We assume that we are given a data set, $\mathcal{D}$, that consists of $n_{\mathcal{D}}$ trajectories, $\{H_L^i\}_{i=1}^{n_{\mathcal{D}}}$, each labeled by the policy that generated it, $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$, i.e.,

$$\mathcal{D} = \left\{ \left(H_L^i, \pi_i\right) : i \in \{1, \ldots, n_{\mathcal{D}}\}, H_L^i \sim \pi_i \right\}. \tag{5.1}$$

Note that $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$ are *behavior policies*—those that generated the batch of data (trajectories). We are interested in algorithms, which we denote by $\Psi$, that take as input a performance level, $\rho_-$, a confidence level $\delta$, and the historical data, $\mathcal{D}$, and output either NO SOLUTION FOUND, or a policy, $\pi \in \Pi$. That is,

$$\Psi(\rho_-, \delta, \mathcal{D}) \in \{\text{NO SOLUTION FOUND}\} \cup \Pi.$$

It is important to understand which of these quantities are fixed and which are random. We assume that $\delta$ and $\rho_-$ are fixed (not random variables). We also assume that the behavior policies, $\{\pi_i\}_{i=1}^{n_{\mathcal{D}}}$, are fixed. However, the trajectories, $\{H_L^i\}_{i=1}^{n_{\mathcal{D}}}$ are random variables—each $H_L^i$ can be sampled by generating a trajectory using the policy $\pi_i$. So, $\mathcal{D}$ is a random variable and therefore $\Psi(\rho_-, \delta, \mathcal{D})$ is as well.

We call any policy, $\pi$, where $\rho(\pi) < \rho_-$ a *bad policy* and any policy, $\pi$, where $\rho(\pi) \geq \rho_-$ a *good policy*. Intuitively, we call an RL algorithm *safe* if it is unlikely that it will propose a bad policy. Formally, we call $\Psi$ *safe* if the following holds:

$$\Pr\Big(\Psi(\rho_-, \delta, \mathcal{D}) \in \{\pi \in \Pi : \rho(\pi) < \rho_-\}\Big) < \delta. \tag{5.2}$$

There is some subtlety to this definition that makes it a weaker safety guarantee than it might at first appear to be. This subtlety is different from the common miscon-

ceptions regarding probability and confidences that were discussed in the previous chapter. If $\Psi$ is safe and it returns a policy, $\pi$, then it is *not* correct to say that $\rho(\pi) \geq \rho_-$ with confidence $1 - \delta$.

This is best understood through an example where an algorithm is safe because it usually returns No Solution Found. Consider a simple safe algorithm, $\Psi'$, which returns No Solution Found with probability $1 - \delta$. Recall that (5.2) requires the probability that a bad policy is returned to be at most $\delta$. The probability that $\Psi'$ returns a policy *at all*, regardless of how good it is, is at most $\delta$. So, $\Psi'$ satisfies (5.2), regardless of how $\rho_-$ is selected.

Let $\rho_-$ be large enough that $\rho_- > \rho(\pi^*)$, where $\pi^*$ is an optimal policy.[1] If $\Psi'(\rho_-, \delta, \mathcal{D}) \neq$ No Solution Found, then $\Psi'(\rho_-, \delta, \mathcal{D}) \in \{\pi \in \Pi : \rho(\pi) < \rho_-\}$—every policy that the safe algorithm $\Psi'$ returns will have performance below the specified baseline because the baseline, $\rho_-$, is above the performance of every policy. So, it is incorrect to say that if $\Psi$ is safe and it returns a policy, $\pi$, then with confidence $1 - \delta$ we have that $\rho(\pi) \geq \rho_-$ ($\Psi'$ is a counter-example—it is safe, but if it returns a policy, $\pi$, then $\rho(\pi) < \rho_-$ always).

It is correct to say that if $\Psi$ is safe then with confidence $1 - \delta$ it will return either No Solution Found or a policy $\pi$ where $\rho(\pi) \geq \rho_-$. The key difference between this statement and the incorrect one is that this statement allows $\Psi$ to decrease the probability of a bad policy being proposed by rarely suggesting changes to the policy. That is, safe algorithms do *not* guarantee that they will improve or even change the current policy. In Section 5.2 we discuss policy improvement algorithms that have the stronger guarantee that with high probability they will *will* return a policy, and it *will* be an improvement.

---

[1] An *optimal* policy is a policy, $\pi^*$, such that there does not exist a policy $\pi'$ where $\rho(\pi') > \rho(\pi^*)$.

Because safe algorithms do not guarantee that they will change the policy, construction of a safe algorithm is not challenging (or particularly interesting), since the algorithm, $\Psi$, is trivially safe if it always returns No Solution Found. We therefore desire a safe algorithm that returns No Solution Found infrequently, and a good policy ($\pi \in \{\pi \in \Pi : \rho(\pi) \geq \rho_-\}$) frequently. **In this chapter we strive to create policy improvement algorithms that are safe and which frequently return good policies.** Our empirical results suggest that, when provided with enough data, our safe algorithms frequently return policies that are improvements over a reasonable baseline, $\rho_-$. For example, in Figure 5.3 in Section 5.6, our safe policy improvement algorithms returned a policy during all trials using 5,000 trajectories of historical data.

If the exact HCOPE component of a safe RL algorithm is replaced with an approximate HCOPE method, then the confidence level, $\delta$, is only *approximately* guaranteed, and so we refer to the algorithm as *semi-safe*. We refer to a policy, $\pi$, (as opposed to an algorithm) as *safe* if we can ensure, from historical data, that $\rho(\pi) \geq \rho_-$ with confidence $1 - \delta$. Notice that "a policy is safe" is a statement about our belief concerning that policy given the historical data, and not a statement about the policy itself (for comparison, saying that a policy is good or bad is a statement about the policy itself).

We will present a safe batch RL algorithm that uses historical data to make a single change to the policy. We then show how it can be applied repeatedly to make multiple incremental improvements to the policy. If there are many policies that might be deemed safe, then safe RL algorithms should return one that is expected to perform best, i.e.,

$$\pi^* \in \arg\max_{\text{safe } \pi} \hat{\rho}(\pi|\mathcal{D}),$$

where $\hat{\rho}(\pi|\mathcal{D}) \in \mathbb{R}$ is a prediction of $\rho(\pi)$ computed from $\mathcal{D}$. In our experiments we will use CWPDIS (see (3.26)) for $\hat{\rho}$.

## 5.2 Related Work

The only existing safe RL algorithms that we are aware of are *conservative policy iteration* (CPI) (Kakade, 2003, Kakade and Langford, 2002a) and its derivatives (Pirotta et al., 2013). CPI is an *approximate policy iteration* algorithm (Bertsekas, 2011) that guarantees policy improvement with high confidence. The CPI algorithm has three steps:

1. Execute the current policy to generate $m$ time steps of data. Use these data to compute a new policy, $\pi'$.

2. Execute the current policy to generate $n$ trajectories. Use this data to perform a statistical analysis of $\pi'$.

3. Based on the statistical analysis, either stop or make $\pi'$ the current policy and return to step 1.

CPI guarantees with high confidence that the sequence of policies that it produces will have strictly increasing performance until it stops. How close CPI gets to an optimal policy before stopping depends on a parameter, $\epsilon$, which scales how much data is needed at each step, i.e., $m$ and $n$ depend on $\epsilon$.

The paper that introduced CPI (Kakade and Langford, 2002a) does not address how large $m$ must be (see the first sentence of their section 7.1). However, $m$ is defined in Sham Kakade's dissertation (Kakade, 2003, Lemma 7.3.4) as (using his notation):

$$m \coloneqq O\left(\frac{A^2H^2}{\epsilon^2}\left(\log\left(|\Pi|\right) + \log\left(\frac{1}{\delta}\right)\right)\right),$$

if the set of policies being considered is finite and

$$m \coloneqq O\left(\frac{H^2}{\epsilon^2}\left(\log\left(\mathrm{VC}(\Pi)\right) + \log\left(\frac{1}{\delta}\right)\right)\right),$$

otherwise. The definition of $n$ is specified in the appendix of the work of Kakade and Langford (2002a) as:

$$n \coloneqq O\left(\frac{R^2}{\epsilon^2}\log\left(\frac{R^2}{\delta\epsilon^2}\right)\right).$$

The meanings of the various symbols here are not important. Rather, we present these definitions to show that $m$ and $n$ are defined in big-O notation. Determining the exact values for $m$ and $n$ requires delving into the proof of Lemma 7.3.4 of Sham Kakade's dissertation and the appendix of the work of Kakade and Langford (2002a), respectively.

The fact that $m$ and $n$ are defined using big-O notation suggests that it was not intended for CPI to be implemented using the values of $m$ and $n$ that actually guarantee policy improvement with high confidence. Instead, $m$ and $n$ show how the amount of data required by CPI scales with different properties of the MDP (e.g., the size of the action set, which they denote by $A$). So, although CPI can be viewed as a safe RL algorithm, it was not intended to be implemented as such, and we have found that doing so requires impractically large amounts of historical data (large $m$ and $n$). For example, CPI requires more trajectories of historical data to make a *single* improvement to the policy for the gridworld than our methods require to converge to a near-optimal policy (it would be a horizontal line at the performance of the initial policy for the entire span of our plots, e.g., Figure 5.5). Furthermore, CPI requires that data can be collected from a *$\mu$-restart distribution* (Kakade and Langford, 2002b)—a requirement that we do not have.

So far we have compared CPI unfavorably to our safe RL methods. We would like to emphasize that CPI was not intended to compete with our safe RL methods, and so

comparing the two is unfair to CPI (our methods require much less historical data). CPI has several properties that our safe RL methods do not. Most importantly, CPI guarantees policy improvement with high confidence if the current policy is sufficiently far from optimal—with high probability the policy will be updated, and it will be an improvement. By contrast, our methods only do the latter—they do not guarantee that the policy will be changed at all, regardless of how far from optimal it is. Furthermore, whereas our safe algorithms observe the historical data before deciding whether or not to return a policy, CPI specifies *a priori* exactly how much historical data is needed to guarantee policy improvement with high confidence, and so it can guarantee convergence (with high probability) to a near-optimal policy within a fixed and finite number of episodes. This desirable property, which our methods lack, allowed CPI to become the foundation of PAC RL research.

## 5.3 Predicting Lower Bounds

At a high level, our methods will partition the historical data into a small training set and a larger testing set. The small training set is used to select a single *candidate policy*, $\pi_c$. The larger testing set is used to compute a high confidence lower bound on $\rho(\pi_c)$. If the high confidence lower bound is above $\rho_-$, then $\pi_c$ is returned, otherwise No Solution Found is returned. The candidate policy, $\pi_c$, is selected such that:

1. We predict from the historical data in the training set that $\rho(\pi_c)$ will be large.
2. We predict from the historical data in the training set that $\pi_c$ will have a high confidence lower bound (computed from the testing set) of at least $\rho_-$.

In order to search for a candidate policy with the second property, we require a method for using a small amount of historical data to predict what the lower bound on the performance of a policy will be when it is computed later using the remaining historical data. This section describes two such methods.

In this section we redefine $\text{HCOPE}^{\dagger}_{\ddagger}$ to take four inputs rather than three—recall that before we wrote $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta)$. Let the additional input, $m$, be a positive integer and $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta, m)$ be a prediction of what $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta)$ would be if $\mathcal{D}$ contained $m$ trajectories. In this chapter we will only use $\ddagger \in \{\text{CUT}, \text{BCa}\}$. If $\mathcal{D}$ contains exactly $m$ trajectories, then let $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta, m) = \text{HCOPE}^{\dagger}_{\ddagger}(\pi_e, \mathcal{D}, \delta)$.

Pseudocode for $\text{HCOPE}^{\dagger}_{CUT}(\pi_e, \mathcal{D}, \delta, m)$ is presented in Algorithm 5.1. This algorithm can be viewed as the combination of Algorithms 4.8 and 4.11, but using $\widehat{\text{CUT}}$ (defined in (4.15)) rather than CUT. To highlight the occurances of $m$, we color it red in the pseudocode.

**Algorithm 5.1:** $\text{HCOPE}^{\dagger}_{CUT}(\pi_e, \mathcal{D}, \delta, m)$: Predict what the $1-\delta$ confidence lower bound on $\rho(\pi_e)$ using the historical data $\mathcal{D} = \{(H^i_L, \pi_i) : i \in \{1, \ldots, n\}, H^i_L \sim \pi_i\}$ would be if $|\mathcal{D}| = m$. The importance sampling method is specified by $\dagger \in \{\text{IS}, \text{NPDIS}\}$. Assumption 1 is not required.

**for** $i = 1$ **to** $n$ **do**
  $X_i \leftarrow \hat{\rho}^{\dagger}(\pi_e | H^i_L, \pi_i)$ ; `// Compute importance weighted returns`
**if** $n < 200$ **then**
  Select $i$ from the discrete uniform distribution over $\{1, \ldots, n\}$;
  $c^* = X_i$;
  $\mathcal{X}_{\text{post}} = \{X_j\}_{j=1}^n \setminus \{X_i\}$;
**else**
  Randomly select 1/20 of the $X_i$ and place them in a set $\mathcal{X}_{\text{pre}}$ and the remainder in $\mathcal{X}_{\text{post}}$;
  `// Optimize threshold using` $\mathcal{X}_{\text{pre}}$
  $c^\star \in \arg\max_{c \in [1,\infty]} \widehat{\text{CUT}}(\mathcal{X}_{\text{pre}}, \delta, c, m)$; `//` $\widehat{\text{CUT}}$ `is defined in` (4.15)
$c^* = \max\{c_{\min}, c^*\}$; `// Do not let` $c^*$ `become too small`
`// Compute lower bound using optimized threshold,` $c^*$ `and` $\mathcal{X}_{\text{post}}$
**return** $\widehat{\text{CUT}}(\mathcal{X}_{\text{post}}, \delta, c^\star, m)$;

Updating HCOPE when using BCa is also straightforward—we resample $m$ trajectories rather than $|\mathcal{D}|$ trajectories. The resulting pseudocode is provided in Algorithm 5.2.

**Algorithm 5.2:** $\text{HCOPE}^{\dagger}_{\text{BCa}}(\pi_e, \mathcal{D}, \delta, m)$: Predict what the $1-\delta$ confidence lower bound on $\rho(\pi_e)$ using the historical data $\mathcal{D} = \{(H_L^i, \pi_i) : i \in \{1, \ldots, n\}, H_L^i \sim \pi_i\}$ would be if $|\mathcal{D}| = m$. The importance sampling method is specified by $\dagger \in \{\text{WIS, CWPDIS}\}$ and BCa is used.

**Assumes:** Nothing—any number of behavior policies can be used and Assumption 1 is not required (although it will increase the accuracy of the approximate lower bound).

$\bar{X}_n \leftarrow \dagger(\pi_e|\mathcal{D})$; `// Sample of statistic to be bounded (` $\hat{\theta}$ `in most BCa literature)`

$B \leftarrow 2000$; `// Number of bootstrap resamplings to perform`

`// Generate` $B$ `resamplings of the data and store the sample statistics`

**for** $i = 1$ **to** $B$ **do**

  Randomly sample $m$ trajectories, $H_L^i$, and their behavior policies, $\pi_i$, from $\mathcal{D}$, with replacement. Store these $m$ resampled trajectories in $\mathcal{D}'$;

  $\xi_i \leftarrow \dagger(\pi_e|\mathcal{D}')$;

`// Estimate the` ***bias constant*** $z_0$ `(Efron, 1987)`

Sort the vector $\xi = (\xi_1, \xi_2, \ldots, \xi_B)$ such that $\xi_i \leq \xi_j$ for $1 \leq i < j \leq B$;

$z_0 \leftarrow \Phi^{-1}\left(\frac{\{\#\xi_i < \bar{X}_n\}}{B}\right)$;

`// Estimate the skew,` $a$`, of the distribution of` $\bar{X}_n$

**for** $i = 1$ **to** $n$ **do**

  Set $y_i$ to be the mean of $\mathbf{X}$ excluding the $i^{\text{th}}$ element:

  $y_i \leftarrow \frac{1}{n-1}\sum_{j=1}^{n} \mathbf{1}_{(j \neq i)} X_j$;

$\bar{y} \leftarrow \frac{1}{n}\sum_{i=1}^{n} y_i$;

$a \leftarrow \frac{\sum_{i=1}^{n}(\bar{y}-y_i)^3}{6\left[\sum_{i=1}^{n}(\bar{y}-y_i)^2\right]^{3/2}}$; `// Standard equation for estimating skewness`

`// Get bootstrap confidence bound using a linear interpolation (Carpenter and Bithell, 2000) and the computed bias and skew terms`

$z_L \leftarrow z_0 - \frac{\Phi^{-1}(1-\delta)-z_0}{1+a(\Phi^{-1}(1-\delta)-z_0)}$;

$Q \leftarrow (B+1)\Phi(z_L)$;

$l \leftarrow \min\{\lfloor Q \rfloor, B-1\}$;

**return** $\bar{X}_n + \frac{\Phi^{-1}(\frac{Q}{B+1})-\Phi^{-1}(\frac{l}{B+1})}{\Phi^{-1}(\frac{l+1}{B+1})-\Phi^{-1}(\frac{l}{B+1})}(\xi_Q, \xi_{Q+1})$;

## 5.4 Safe Policy Improvement (SPI)

This section is based on our previous work where SPI was referred to as *high confidence policy improvement* (HCPI) (Thomas et al., 2015c). We adopt the name SPI because it was incorrect to imply that our methods guarantee policy improvement with high confidence (like CPI does). For more details on this, refer to the previous two sections.

In this and subsequent sections we present algorithms that contain lines of the form $\pi_c \leftarrow \arg\max_{\pi \in \Pi} f(\pi)$, for some $f : \Pi \to \mathbb{R}$. These lines call for a large search of policy space for a policy that maximizes some objective function, $f$. Any off-the-shelf black box optimization algorithm can be used for this search. For our experiments in Section 5.6 we used CMA-ES (Hansen, 2006).

Consider a simple approach to SPI:

1. **(Policy Selection):** Select a *candidate policy*, $\pi_c \leftarrow \arg\max_\pi \text{HCOPE}^\dagger_\ddagger(\pi, \mathcal{D}, \delta)$.
2. **(Safety Test):** If $\text{HCOPE}^\dagger_\ddagger(\pi_c, \mathcal{D}, \delta) \geq \rho_-$ then return $\pi_c$, otherwise return No Solution Found.

This approach is *not* safe (and in preliminary experiments we found that it produced error rates that were significantly too large). The problem with this approach stems from computing $\pi_c$ from the same data, $\mathcal{D}$, that is used to test it for safety. This is akin to gathering data for a scientific study, using the data to form a hypothesis, and then using that same data to test the hypothesis—it is biased in favor of accepting the hypothesis even if it is wrong. This problem is related to the *multiple comparisons problem*, which we review below.

### 5.4.1 Testing Safety of Multiple Policies

Although the multiple comparisons problem is well studied (Benjamin and Hochberg, 1995), we briefly review it in our context. In this section we consider our safety test (checking whether $\text{HCOPE}^\dagger_\ddagger(\pi_c, \mathcal{D}, \delta) \geq \rho_-$) from a hypothesis testing perspective

(Wilcox, 2012), where the null hypothesis is that $\rho(\pi) < \rho_-$. Consider the following policy improvement scheme, which uses the historical data, $\mathcal{D}$, to propose a new policy, $\pi$. First, randomly select $\pi$. Then use $\mathcal{D}$ to test whether $\pi$ is safe, i.e., check whether $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_c, \mathcal{D}, \delta) \geq \rho_-$. If it is, then return $\pi$, otherwise return No Solution Found. As desired, this naïve scheme will only make Type 1 errors (declare $\pi$ to be safe when it is not) with probability at most $\delta$.

Now consider what would happen if the policy improvement mechanism randomly selects *two* policies, $\pi_1$ and $\pi_2$. Both $\pi_1$ and $\pi_2$ are tested for safety. If both are deemed unsafe, the policy improvement mechanism returns No Solution Found. If one or both are deemed safe, then it returns one of the policies that was deemed safe. The policy returned by this scheme can result in Type 1 errors with probability larger than $\delta$, i.e., it does not provide the desired safety guarantee. To see why, consider the case where both $\pi_1$ and $\pi_2$ should be declared unsafe. The test $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_i, \mathcal{D}, \delta) \geq \rho_-$ ensures that the probability of incorrectly declaring $\pi_i$ to be safe is at most $\delta$, for $i \in \{1, 2\}$. However, if a Type 1 error is made on either policy, then the proposed policy improvement scheme will make a Type 1 error by returning a policy that should not have been declared safe. So, the probability that the policy improvement scheme makes a Type 1 error can be as large as $2\delta$.

Although approaches exist to account for the multiple comparisons problem (Benjamin and Hochberg, 1995), they all result in the individual hypothesis tests being less data efficient than if only a single hypothesis were to be tested. In the context of scientific experiments, where the multiple comparisons problem most frequently arises, this is unavoidable because there are many hypotheses that must be tested (e.g., which symptoms are reduced by a drug). However, in our setting there is no requirement that we test multiple hypotheses. We will therefore avoid the multiple comparisons problem by only testing a single hypothesis (we will set aside some of the historical data and will use it to test a *single* policy for safety).

### 5.4.2 Algorithm

As suggested by the previous discussion, searching for a safe policy is a particularly challenging problem due to the multiple comparisons problem. To avoid this problem we propose a policy improvement algorithm that partitions the historical data into a small *training set*, $\mathcal{D}_{\text{train}}$, and a larger *testing set*, $\mathcal{D}_{\text{test}}$. The algorithm has the high level structure:

1. Partition the historical data into two sets, $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$.
2. **(Policy Selection):** Select a *candidate policy*, $\pi_c$, using the training set, $\mathcal{D}_{\text{train}}$.
3. **(Safety Test):** If $\text{HCOPE}^{\dagger}_{\ddagger}(\pi_c, \mathcal{D}_{\text{test}}, \delta) \geq \rho_-$ then return $\pi_c$, otherwise return No Solution Found.

Notice that $\mathcal{D}_{\text{test}}$ is only used once at the end during the "Safety Test" step to test whether a single policy, $\pi_c$ is safe. This means that the multiple comparisons problem is not an issue—only a single hypothesis is tested using $\mathcal{D}_{\text{test}}$, and the outcome of that test determines whether a policy is returned. So, this algorithm is safe regardless of how the candidate policy, $\pi_c$, is generated from $\mathcal{D}_{\text{train}}$. The viability of this approach hinges on our ability to effectively use a small amount of historical data to select a candidate policy that will both pass the safety test and have high performance (large $\rho(\pi_c)$). The remainder of this section therefore focuses on how we should use $\mathcal{D}_{\text{train}}$ to select $\pi_c$.

Our safe policy improvement algorithm, $\text{SPI}^{\dagger,\star}_{\ddagger}$, is presented in Algorithm 5.3, where we specify the meaning of $\star$ and formally define GetCandidatePolicy later. To simplify later pseudocode, this method, called SPI, assumes that the trajectories have already been partitioned into $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$. In practice, we place 1/5 of the trajectories in the training set and the remainder in the testing set. Also, notice that if SPI uses $\ddagger$ = CUT and $\dagger$ = NPDIS, then it is a safe RL algorithm, while using $\ddagger$ = BCa and $\dagger$ = CWPDIS results in a semi-safe RL algorithm.

**Algorithm 5.3:** $\text{SPI}^{\dagger,\star}_{\ddagger}(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}}, \delta, \rho_-)$: Use the historical data, partitioned into $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$, to search for a safe policy (with $1-\delta$ confidence lower bound at least $\rho_-$). If none is found, then return No Solution Found. Although other $\dagger$ and $\ddagger$ could be used, we have only provided complete pseudocode for $(\dagger, \ddagger) \in \{(\text{NPDIS,CUT}), (\text{CWPDIS,BCa})\}$. We allow for $\star \in \{\text{None}, k\text{-fold}\}$. Assumption 1 is not required.

$\pi_c \leftarrow \text{GetCandidatePolicy}^{\dagger,\star}_{\ddagger}(\mathcal{D}_{\text{train}}, \delta, \rho_-, |\mathcal{D}_{\text{test}}|)$;
**if** $\mathit{HCOPE}^{\dagger}_{\ddagger}(\pi_c, \mathcal{D}_{test}, \delta) \geq \rho_-$ **then**
  **return** $\pi_c$;
**return** No Solution Found

SPI is presented in a top-down manner and makes use of the method $\text{GetCandidatePolicy}^{\dagger,\star}_{\ddagger}(\mathcal{D}, \delta, \rho_-, m)$, which searches for a candidate policy. The input $m$ specifies the number of trajectories that will be used during the subsequent safety test. We will present two versions of $\text{GetCandidatePolicy}^{\dagger,\star}_{\ddagger}$, which we differentiate between using the $\star$ superscript, which may stand for None or $k$-fold. Since our notation is becoming cluttered, we remind the reader of the meaning of the symbols:

$$\text{SPI}^{\dagger=\text{importance sampling variant},\star=\text{GetCandidatePolicy variant}}_{\ddagger=\text{concentration inequality variant}}$$

Before presenting the two variants of GetCandidatePolicy, we define an objective function, $f^{\dagger}_{\ddagger}$, as:

$$f^{\dagger}_{\ddagger}(\pi, \mathcal{D}, \delta, \rho_-, m) \coloneqq \begin{cases} \hat{\rho}(\pi|\mathcal{D}) & \text{if } \text{HCOPE}^{\dagger}_{\ddagger}(\pi, \mathcal{D}, \delta, m) \geq \rho_-, \\ \text{HCOPE}^{\dagger}_{\ddagger}(\pi, \mathcal{D}, \delta, m) & \text{otherwise.} \end{cases}$$

Intuitively, $f^{\dagger}_{\ddagger}$ uses $\text{HCOPE}^{\dagger}_{\ddagger}(\pi, \mathcal{D}, \delta, m)$ to predict what the lower bound on $\rho(\pi)$ would be if it were to be computed using $m$ trajectories of historical data. If the predicted lower bound is below $\rho_-$, then it returns the predicted lower bound. However, if the predicted lower bound is at least $\rho_-$, then it returns the predicted performance

of $\pi$, $\hat{\rho}(\pi|\mathcal{D})$. This will serve as an objective function for $\textsc{GetCandidatePolicy}_{\ddagger}^{\dagger,\star}$ (where $\mathcal{D}$ is $\mathcal{D}_{\text{train}}$ and $m$ is $|\mathcal{D}_{\text{test}}|$).

Consider $\textsc{GetCandidatePolicy}_{\ddagger}^{\dagger,\text{None}}$, which is presented in Algorithm 5.4. This method uses all of the available training data to search for the policy that is predicted to perform the best, subject to it also being predicted to pass the safety test. That is, if no policy is found that is predicted to pass the safety test, it returns the policy, $\pi$, that it predicts will have the highest lower bound on $\rho(\pi)$. If policies are found that are predicted to pass the safety test, it returns one that is predicted to perform the best (according to $\hat{\rho}$).

**Algorithm 5.4:** $\textsc{GetCandidatePolicy}_{\ddagger}^{\dagger,\text{None}}(\mathcal{D}_{\text{train}}, \delta, \rho_-, m)$: Use the historical data, partitioned into $\mathcal{D}_{\text{train}}$ to search for the candidate policy that is predicted to be safe and perform the best (or to be closest to safe if none are predicted to be safe). Although other $\dagger$ and $\ddagger$ could be used, we have only provided complete pseudocode for $(\dagger, \ddagger) \in \{(\text{NPDIS},\text{CUT}), (\text{CWPDIS},\text{BCa})\}$. Assumption 1 is not required.

```
return arg max_π f_‡^†(π, D_train, δ, ρ_-, m);
```

The benefits of this approach are its simplicity and that it works well when there is an abundance of data. However, when there are few trajectories in $\mathcal{D}$ (e.g., due to a cold start), this approach has a tendency to overfit—it finds a policy that it predicts will perform exceptionally well and which will easily pass the safety test, but actually fails the subsequent safety test in SPI. We call this method $\star = \text{None}$ because it does not use any methods to avoid overfitting.

In supervised learning research it is common to introduce a regularization term, $\lambda\|w\|$, into the objective function in order to prevent overfitting. Here $w$ is the model's weight vector and $\|\cdot\|$ is some measure of the complexity of the model (often the $L_1$ or squared $L_2$-norm), and $\lambda$ is a parameter that is tuned using a model selection method like cross-validation. This term penalizes solutions that are too complex, since they are likely to be overfitting the training data.

Here we can use the same intuition, where we control for the complexity of the solution policy using a regularization parameter, $\lambda$, that is optimized using $k$-fold cross validation (a popular model selection technique). Just as the squared $L^2$-norm relates the complexity of a weight vector to its squared distance from the zero vector, we define the complexity of a policy to be some notion of its distance from some initial policy, $\pi_0$. If there is a single behavior policy or if learning began with a specific policy, then that policy can be used as $\pi_0$. Otherwise, $\pi_0$ can be chosen to be the policy that always selects each action with equal probability. In order to allow for an intuitive meaning of $\lambda$, rather than adding a regularization term to our objective function, $f^{\dagger}_{\ddagger}(\cdot, \mathcal{D}_{\text{train}}, \delta, \rho_{-}, |\mathcal{D}_{\text{test}}|)$, we directly constrain the set of policies that we search over to have limited complexity.[2]

We define a *mixed policy*, $\mu_{\lambda,\pi_0,\pi}$, to be a mixture of $\pi_0$ and $\pi$ with mixing parameter $\lambda \in [0, 1]$. As $\lambda$ increases, the mixed policy becomes more like $\pi$, and as $\lambda$ decreases it becomes more like $\pi_0$. Formally:

$$\mu_{\lambda,\pi_0,\pi}(a|o) \coloneqq \lambda\pi(a|o) + (1 - \lambda)\pi_0(a|o).$$

We can avoid overfitting the training data by only searching the space of *mixed policies*, $\mu_{\lambda,\pi_0,\pi}$, where $\lambda$ is the fixed regularization parameter, $\pi_0$ is the fixed initial policy, and where we search the space of all possible $\pi$. Consider, for example what happens to the probability of action $a$ given the observation $o$ when $\lambda = 0.5$. If $\pi_0(a|o) = 0.4$, then for any $\pi$, we have that $\mu_{\lambda,\pi_0,\pi}(a|o) \in [0.2, 0.7]$. That is, the mixed policy can only change the probability of an action $100\lambda\% = 50\%$ of the way towards 0 or 1, i.e., $100\lambda\%$ towards deterministic action selection. So, using mixed

[2]This is *not* non-standard. In regression, depending on the regularization scheme, there is an equivalence between regularization using weight decay and a constraint on model complexity (Abu-Mostafa et al., 2012, Section 4.2.1)

policies results in our searches of policy space being constrained to some *feasible set* centered around the initial policy, and where $\lambda$ scales the size of this feasible set.

While small values of $\lambda$ can effectively eliminate overfitting by precluding the mixed policy from moving far away from the initial policy, they also limit the quality of the best mixed policy in the feasible set. It is therefore important that $\lambda$ is chosen to balance the tradeoff between overfitting and limiting the quality of solutions that remain in the feasible set. Just as in supervised learning, we use a model selection algorithm, $k$-fold cross-validation, to automatically select $\lambda$.

This approach is provided in Algorithm 5.5, where $\textsc{GetCandidatePolicy}^{\dagger,k\text{-fold}}_{\ddagger}(\mathcal{D}_{\text{train}}, \delta, \rho_-, m)$ uses $k$-fold cross validation to predict the value of $f^{\dagger}_{\ddagger}(\pi, \mathcal{D}_{\text{test}}, \delta, \rho_-, |\mathcal{D}_{\text{test}}|)$ if $\pi$ were to be optimized using $\mathcal{D}_{\text{train}}$ and regularization parameter $\lambda$. $\textsc{CrossValidate}^{\dagger}_{\ddagger}$ is described in Algorithm 5.6. In our implementations we use $k = \min\{20, \frac{1}{2}|\mathcal{D}|\}$ folds for cross validation.

**Algorithm 5.5:** $\textsc{GetCandidatePolicy}^{\dagger,k\text{-fold}}_{\ddagger}(\mathcal{D}_{\text{train}}, \delta, \rho_-, m)$: Use the historical data, $\mathcal{D}_{\text{train}}$, to search for the candidate policy that is predicted to be safe and perform the best (or to be closest to safe if none are predicted to be safe). Although other † and ‡ could be used, we have only provided complete pseudocode for $(\dagger, \ddagger) \in \{(\text{NPDIS,CUT}), (\text{CWPDIS,BCa})\}$. Assumption 1 is not required.

$\lambda^* \leftarrow \arg\max_{\lambda \in [0,1]} \textsc{CrossValidate}^{\dagger}_{\ddagger}(\lambda, \mathcal{D}_{\text{train}}, \delta, \rho_-, m)$;
$\pi^* \leftarrow \arg\max_{\pi} f^{\dagger}_{\ddagger}(\mu_{\lambda^*,\pi_0,\pi}, \mathcal{D}_{\text{train}}, \delta, \rho_-, m)$;
**return** $\mu_{\lambda^*,\pi_0,\pi^*}$;

To see that SPI is a safe algorithm (or semi-safe, depending on the choice of † and ‡), notice that it will only return a policy, $\pi$, if $\text{HCOPE}^{\dagger}_{\ddagger}(\pi, \mathcal{D}_{\text{test}}, \delta) \geq \rho_-$. Recall from the chapter on HCOPE (Chapter 4) that $\Pr\left(\rho(\pi_e) \geq \text{HCOPE}^{\dagger}_{\ddagger}(\pi, \mathcal{D}_{\text{test}}, \delta)\right) \geq 1 - \delta$. So, the probability that SPI returns a policy, $\pi$, where $\rho(\pi) < \rho_-$ is at most $\delta$.

**Algorithm 5.6:** CrossValidate$^{\dagger}_{\ddagger}(\lambda, \mathcal{D}_{\text{train}}, \delta, \rho_-, m)$: Predict the value of $f^{\dagger}_{\ddagger}(\pi, \mathcal{D}_{\text{test}}, \delta, \rho_-, |\mathcal{D}_{\text{test}}|)$ if $\pi$ were to be optimized using $\mathcal{D}_{\text{train}}$ and regularization parameter $\lambda$. Although other $\dagger$ and $\ddagger$ could be used, we have only provided complete pseudocode for $(\dagger, \ddagger) \in \{(\text{NPDIS},\text{CUT}), (\text{CWPDIS},\text{BCa})\}$. Assumption 1 is not required.

Partition $\mathcal{D}_{\text{train}}$ into $k$ subsets, $\mathcal{D}_1, \ldots, \mathcal{D}_k$, of approximately the same size;
result $\leftarrow 0$;
**for** $i = 1$ **to** $k$ **do**
    $\widehat{\mathcal{D}} \leftarrow \bigcup_{j \neq i} \mathcal{D}_j$;
    $\pi^\star \leftarrow \arg\max_\pi f^{\dagger}_{\ddagger}(\mu_{\lambda,\pi_0,\pi}, \widehat{\mathcal{D}}, \delta, \rho_-, m)$;
    result $\leftarrow$ result $+ f^{\dagger}_{\ddagger}(\mu_{\lambda,\pi_0,\pi^\star}, \mathcal{D}_i, \delta, \rho_-, m)$;
**return** result$/k$;

## 5.5 Multiple Policy Improvements: Daedalus

The SPI algorithm is a batch method that can be applied to any historical data, $\mathcal{D}$. However, it can also be used in an incremental manner by executing new safe policies whenever they are found. The user might choose to change $\rho_-$ at each iteration, e.g., to reflect an estimate of the performance of the best policy found so far or the performance of the most recently proposed policy. However, for simplicity in our pseudocode and experiments, we assume that the user fixes $\rho_-$ as an estimate of the performance of the initial policy. This scheme for selecting $\rho_-$ is appropriate when trying to convince a user to deploy an RL algorithm to tune a currently fixed initial policy, since it guarantees with high confidence that it will not decrease performance.

Our algorithm maintains a list, $\mathcal{C}$, of the policies that it has deemed safe. When generating new trajectories, it always uses the policy in $\mathcal{C}$ that is expected to perform best. $\mathcal{C}$ is initialized to include a single initial policy, $\pi_0$, which is the same as the baseline policy used by GetCandidatePolicy$^{\dagger,k\text{-fold}}_{\ddagger}$. This online safe learning algorithm is presented in Algorithm 5.7,[3] which takes as input an additional constant,

[3] If trajectories from one or more behavior policies are available *a priori*, then $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}}$, and $\mathcal{C}$ can be initialized accordingly.

$\beta$, that denotes the number of trajectories to be generated by each policy. We assume that $\beta$ is specified by the application. We name this algorithm DAEDALUS$^{\dagger,\star}_{\ddagger}$ after the mythological character who promoted safety when he encouraged Icarus to use caution.

**Algorithm 5.7:** DAEDALUS$^{\dagger,\star}_{\ddagger}(\pi_0, \delta, \rho_-, \beta)$: Make repeated safe policy improvements. The initial policy is $\pi_0$, $\delta$ and $\rho_-$ are the confidence level and lower bounds for ensuring safety, and $\beta$ is a positive integer that specifies how many trajectories to generate between policy updates. Although other $\dagger$ and $\ddagger$ could be used, we have only provided complete pseudocode for $(\dagger, \ddagger) \in \{(\text{NPDIS,CUT}), (\text{CWPDIS,BCa})\}$. We allow $\star \in \{\text{None}, k\text{-fold}\}$. Assumption 1 is not required.

$\mathcal{C} \leftarrow \{\pi_0\}$;
$\mathcal{D}_{\text{train}} \leftarrow \emptyset$;
$\mathcal{D}_{\text{test}} \leftarrow \emptyset$;
**while true do**
  $\widehat{\mathcal{D}} \leftarrow \mathcal{D}_{\text{train}}$;
  $\pi^* \leftarrow \arg\max_{\pi \in \mathcal{C}} \hat{\rho}(\pi|\widehat{\mathcal{D}})$;
  Generate $\beta$ trajectories using $\pi^*$ and append $\lceil 1/5 \rceil$ to $\mathcal{D}_{\text{train}}$ and the rest to $\mathcal{D}_{\text{test}}$;
  $\pi_c \leftarrow \text{SPI}^{\dagger,\star}_{\ddagger}(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}}, \delta, \rho_-)$;
  $\widehat{\mathcal{D}} \leftarrow \mathcal{D}_{\text{train}}$;
  **if** $\pi_c \neq$ NO SOLUTION FOUND **and** $\hat{\rho}(\pi_c|\widehat{\mathcal{D}}) > \max_{\pi \in \mathcal{C}} \hat{\rho}(\pi|\widehat{\mathcal{D}})$ **then**
    $\mathcal{C} \leftarrow \mathcal{C} \cup \pi_c$;
    $\mathcal{D}_{\text{test}} \leftarrow \emptyset$;

The benefits of $\ddagger = k$-fold are biggest when only a few trajectories are available, since then GETCANDIDATEPOLICY$^{\dagger}_{\text{None}}$ is prone to overfitting. When there is a lot of data, overfitting is not a big problem, and so the additional computational complexity of $k$-fold cross-validation is not justified. In our implementations of DAEDALUS$^{\dagger}_{k\text{-fold}}$, we therefore only use $\ddagger = k$-fold until the first policy is successfully added to $\mathcal{C}$, and $\ddagger$ = None thereafter. This provides the early benefits of $k$-fold cross-validation without incurring its full computational complexity.

The DAEDALUS$^{\dagger,\star}_{\ddagger}$ algorithm ensures safety with each newly proposed policy. That is, during each iteration of the while-loop, the probability that a new policy, $\pi$, where

$\rho(\pi) < \rho_-$, is added to $\mathcal{C}$ is at most $\delta$. The multiple comparison problem is not relevant here because this guarantee is per-iteration. However, if we consider the safety guarantee over multiple iterations of the while-loop, it applies and means that the probability that at least one policy, $\pi$, where $\rho(\pi) < \rho_-$, is added to $\mathcal{C}$ over $k$ iterations is at most $\min\{1, k\delta\}$ ($k\delta$ can be larger than one, for example, if there are $k = 100$ calls to $\text{SPI}^{\dagger,\star}_{\ddagger}$ and $\delta = 0.05$). However, the per-iteration guarantee can be applied to the last of several iterations of the while-loop. That is, if $k$ iterations of the while-loop are executed, the probability that, during the last iteration, a new policy, $\pi$, where $\rho(\pi) < \rho_-$, is added to $\mathcal{C}$ is at most $\delta$.

We define $\text{DAEDALUS2}^{\dagger,\star}_{\ddagger}$ to be $\text{DAEDALUS}^{\dagger,\star}_{\ddagger}$ but with line 12 ($\mathcal{D}_{\text{test}} \leftarrow \emptyset$) removed. The multiple hypothesis testing problem does *not* affect $\text{DAEDALUS2}^{\dagger,\star}_{\ddagger}$ more than $\text{DAEDALUS}^{\dagger,\star}_{\ddagger}$, since the safety guarantee is per-iteration. However, a more subtle problem is introduced: the importance weighted returns from the trajectories in the testing set, $\hat{\rho}(\pi_c | \tau_i^{\mathcal{D}_{\text{test}}}, \pi_i^{\mathcal{D}_{\text{test}}})$, are not necessarily unbiased estimates of the true performance, $\rho(\pi_c)$.

This happens because the policy, $\pi_c$, is computed in part from the trajectories in $\mathcal{D}_{\text{test}}$ that are used to test it for safety. In $\text{DAEDALUS}^{\dagger,\star}_{\ddagger}$ $\mathcal{D}_{\text{test}}$ is cleared so that trajectories that influenced the choice of the current candidate policy will not be used during its safety test. This dependence is depicted in Figure 5.1. Importantly, notice that the ability of $\mathcal{D}_{\text{train}}$ to impact the candidate policy is limited. $\mathcal{D}_{\text{train}}$ impacts whether the safety test is passed or not—a Boolean value. This Boolean value impacts which policy will be used to generate the next batch of historical data, which in turn is used to select the next candidate policy. So, the candidate policy may have a dependence on some trajectories in $\mathcal{D}_{\text{train}}$, however this dependence is likely small.

We also modify $\text{DAEDALUS2}^{\dagger,\star}_{\ddagger}$ by changing lines 5 and 9 to $\widehat{\mathcal{D}} \leftarrow \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{test}}$, which introduces an additional minor dependence of $\pi_c$ on the trajectories in $\mathcal{D}_{\text{test}}$.

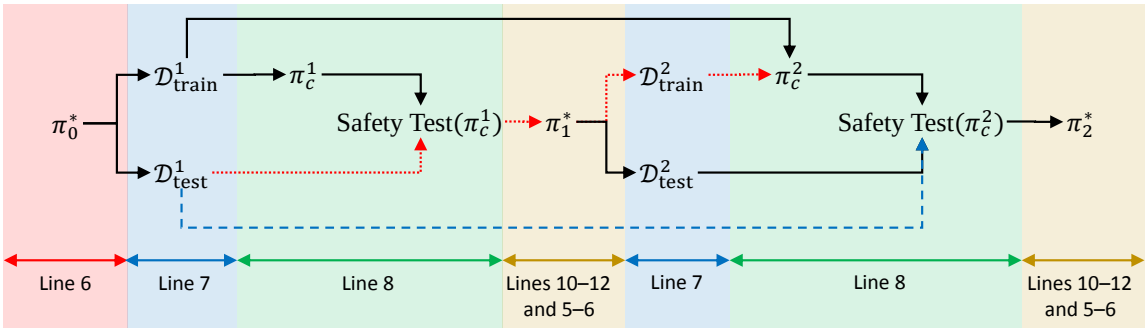


**Figure 5.1.** This diagram depicts influences as DAEDALUS2$^{\dagger,\star}_{\ddagger}$ runs. The line numbers that each part of the diagram corresponds to are provided at the bottom of the figure. First the initial policy, $\pi_0^*$, is used to generate sets of trajectories, $\mathcal{D}^1_{\text{train}}$ and $\mathcal{D}^1_{\text{test}}$, where superscripts denote the iteration. Next $\mathcal{D}^1_{\text{train}}$ is used to select the candidate policy, $\pi_c^1$. Next, $\pi_c^1$ is tested for safety using the trajectories in $\mathcal{D}^1_{\text{test}}$ (this safety test occurs within line 8 of Algorithm 5.7, on line 2 of SPI$^{\dagger,\star}_{\ddagger}$). The result of the safety test influences which policy, $\pi_1^*$, will be executed next—it will either be $\pi_0^*$ or $\pi_c^1$, depending on the outcome of the safety test within SPI$^{\dagger,\star}_{\ddagger}$. The policy $\pi_1^*$ is then used to produce $\mathcal{D}^2_{\text{train}}$ and $\mathcal{D}^2_{\text{test}}$ as before. Next, both $\mathcal{D}^1_{\text{train}}$ and $\mathcal{D}^2_{\text{train}}$ are used to select the next candidate policy, $\pi_c^2$. This policy is then tested for safety using the trajectories in $\mathcal{D}^1_{\text{test}}$ and $\mathcal{D}^2_{\text{test}}$. The result of this test influences which policy, $\pi_2^*$, will be executed next, and the process continues. Notice that $\mathcal{D}^1_{\text{test}}$ is used when testing $\pi_c^2$ for safety (as indicated by the dashed blue line) even though it also influences $\pi_c^2$ (as indicated by the dotted red path). This is akin to performing an experiment, using the collected data ($\mathcal{D}^1_{\text{test}}$) to select a hypothesis ($\pi_c^2$ is safe), and then using that same data to test the hypothesis. DAEDALUS$^{\dagger,\star}_{\ddagger}$ does not have this problem because the dashed blue line is not present.

Although our theoretical analysis applies to DAEDALUS$^{\dagger,\star}_{\ddagger}$, we propose the use of DAEDALUS2$^{\dagger,\star}_{\ddagger}$ because the ability of the trajectories, $\mathcal{D}^i_{\text{test}}$, to bias the choice of which policy to test for safety in the future ($\pi_c^j$, where $j > i$) towards a policy that $\mathcal{D}^i_{\text{test}}$ will deem safe, is small. However, the benefits of DAEDALUS2$^{\dagger,\star}_{\ddagger}$ over DAEDALUS$^{\dagger,\star}_{\ddagger}$ are significant—the set of trajectories used in the safety tests increases in size with each iteration, as opposed to always being of size $\beta$. So, in practice, we expect the over-conservativeness of our concentration inequality to far outweigh the small bias that is introduced by DAEDALUS2$^{\dagger,\star}_{\ddagger}$.

Furthermore, notice that DAEDALUS2$^{\text{CUT},\star}_{\ddagger}$ is safe, and not just semi-safe, if we consider its execution only up until the point where the policy is first changed, since

then the trajectories are always generated by $\pi_0$, which is not influenced by any of the testing data.

## 5.6 Experiments

In this section we provide an empirical comparison of the variants of SPI and Daedalus2. We do not compare to any other algorithms because, to the best of our knowledge, there are no other practical[4] safe reinforcement learning algorithms.[5] The previous chapters suggest that we should use $\dagger$ = NPDIS and $\ddagger$ = CUT for exact HCOPE (safe algorithms) and $\dagger$ = CWPDIS and $\ddagger$ = BCa for approximate HCOPE (semi-safe algorithms). We therefore focus on these two settings and compare the use of $\star = k$-fold and $\star$ = None.

For all of the domains, we select $\rho_-$ to be the performance of the initial policy,[6] $\delta = 0.05$, and $\beta$ to be approximately the amount of data that we might expect to see between policy improvement steps (e.g., for the digital marketing domain we use $\beta = 25{,}000$, which corresponds to a week of historical data from a small company). The plots in this section are all averaged over 100 trials, with standard error bars shown.

The gridworld that we used to validate HCOPE is particularly challenging when it comes to policy improvement—even well-tuned ordinary RL algorithms require an unexpectedly large number of trajectories to converge to a near-optimal policy. We therefore use a simplified variant for testing SPI and Daedalus. In this variant

---

[4]As discussed before, CPI can be a safe reinforcement algorithm if enough trajectories are used. However, we argued that it was not intended to be implemented in this way and that doing so would require an impractical amount of data (the number of trajectories that CPI requires to make a single change to the policy is larger than the entire span of our plots).

[5]As in the rest of this dissertation, we refer specifically to our definition of a *safe reinforcement learning algorithm* from Section 5.1.

[6]For experiments where we set $\rho_-$ to be larger, requiring significant improvements to the policy before it is changed, see our previous work (Thomas et al., 2015d).

the rewards are all changed to $-1$. We begin with a hand-tuned policy that has an expected normalized return of 0.22, and which almost always finds the goal within 10 time steps. We therefore truncate episodes to ensure that they terminate within $L = 10$ time steps.

Figure 5.2 shows the results of running SPI with various settings. As anticipated, BCa outperforms CUT (the approximate method outperforms the exact method, which must provide a stronger safety guarantee). Using $n = 50$ trajectories, only ($\ddagger = \text{BCa}, \star = k$-fold) produced any changes to the policy, however all methods were able to improve the policy with $n = 700$ trajectories. When using $\ddagger = \text{CUT}$, using $\star = k$-fold results in a large improvement in performance over $\star =$None. However, when using $\ddagger = \text{BCa}$, $\star = k$-fold results in a small decrease in performance when using $n = 700$ trajectories, which suggests that in this case overfitting of the training set was not an issue. When overfitting the training data is not an issue, $\star = k$-fold cross-validation sometimes erroneously over-restricts the set of policies that are considered, and therefore results in slightly worse performance than $\star =$None.

We see this same pattern for mountain car (Figure 5.3) and the digital maketing domain (Figure 5.4). Impressively, the semi-safe SPI methods are able to improve the policies for mountain car and digital marketing using as few as $n = 50$ and $n = 1{,}000$ trajectories, respectively. For mountain car the safe methods require just hundreds of trajectories of historical data to improve upon the initial policy. For the digital marketing domain the safe and semi-safe policy improvement algorithm can reliably produce policy improvements with realistic amounts of data—historical data from a few thousand users. For both the mountain car and digital marketing domains we again see the trend that using $\star = k$-fold is primarily beneficial when using $\ddagger =$CUT.

Figures 5.5, 5.6, and 5.7 depict the results on these three domains when using $\text{Daedalus2}^{\dagger,\star}_{\ddagger}$ for multiple policy improvements. There are a few interesting trends to notice. First, notice that given the same number of trajectories, $\text{Daedalus2}^{\dagger,\star}_{\ddagger}$

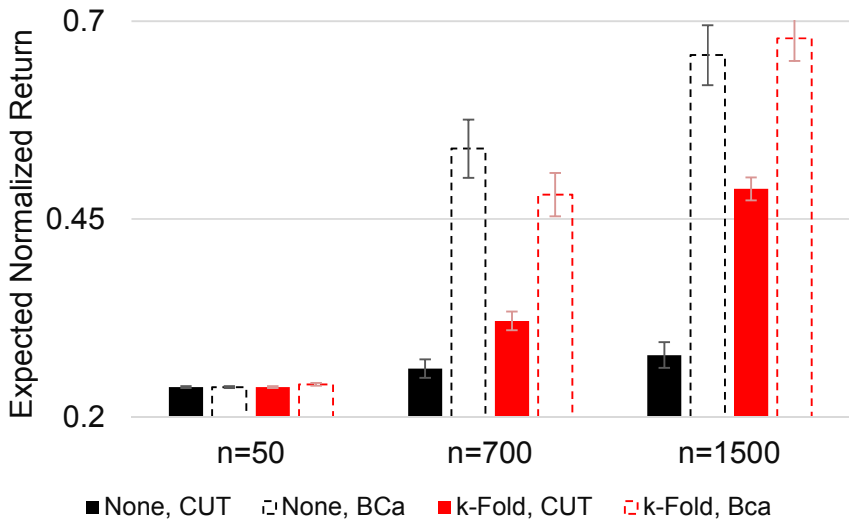


**Figure 5.2.** Performance of $\text{SPI}^{\dagger,\star}_{\ddagger}$ on the simplified gridworld domain, where $\dagger =$ NPDIS and $\ddagger =$ CUT or $\dagger =$ CWPDIS and $\ddagger =$ BCa, and $\star \in \{\text{None}, k\text{-fold}\}$.

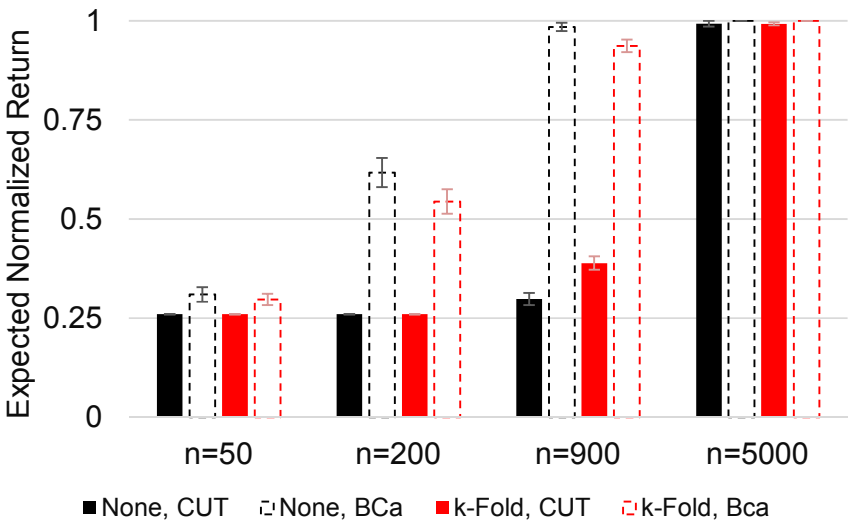


**Figure 5.3.** Performance of $\text{SPI}^{\dagger,\star}_{\ddagger}$ on the mountain car domain, where $\dagger =$ NPDIS and $\ddagger =$ CUT or $\dagger =$ CWPDIS and $\ddagger =$ BCa, and $\star \in \{\text{None}, k\text{-fold}\}$.

tends to outperform SPI (compare, for example, $n = 700$ in Figures 5.2 and 5.5). This is because as $\text{DAEDALUS2}^{\dagger,\star}_{\ddagger}$ improves the policy, it samples trajectories from increasingly good regions of policy space. These trajectories are more informative about even better policies than trajectories that are generated by the initial policy. Next, notice that $k$-fold cross-validation tends to be beneficial when using the exact

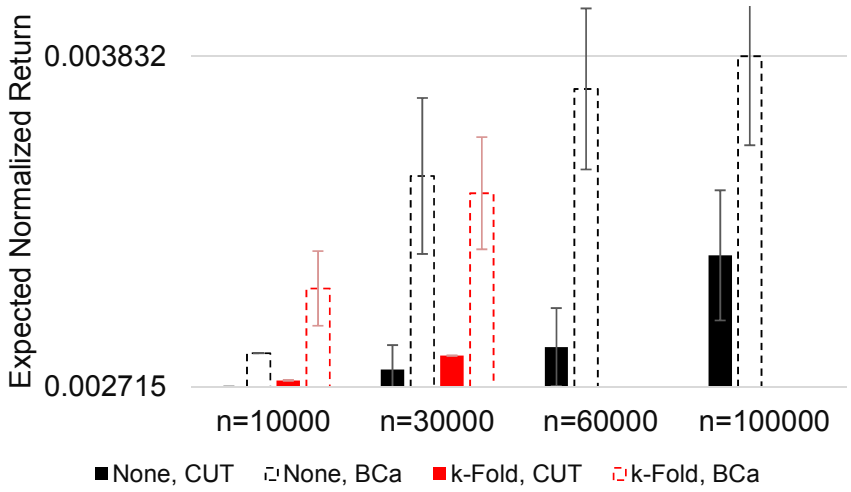


**Figure 5.4.** Performance of $\text{SPI}_{\ddagger}^{\dagger,\star}$ on the digital marketing domain, where $\dagger =$ NPDIS and $\ddagger =$ CUT or $\dagger =$ CWPDIS and $\ddagger =$ BCa, and $\star \in \{\text{None}, k\text{-fold}\}$. Due to runtime limitations, this plot was only averaged over 10 trials and the variants that use $k$-fold cross-validation were not run for $n > 30{,}000$.

methods (CUT), or the approximate methods (BCa) with little data. However, when there is sufficient data (particularly when using BCa), $k$-fold cross-validation often hurts performance. This is likely because there is so much data that overfitting the training set is not an issue.

Impressively, the safe and semi-safe variants of $\text{DAEDALUS2}_{\ddagger}^{\dagger,\star}$ are able to find near optimal policies for the gridworld and mountain car domains using just hundreds of trajectories. They are also able to find significantly improved policies for the digital marketing domain using a realistic amount of historical data.

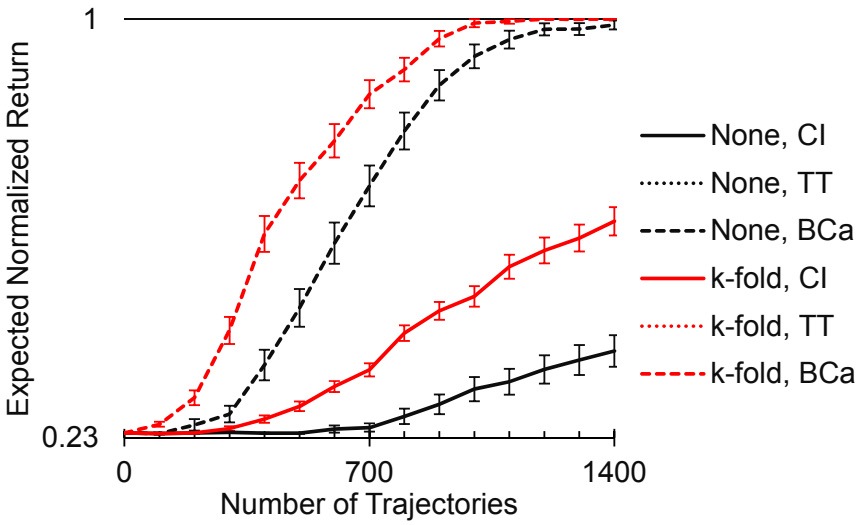


**Figure 5.5.** Performance of DAEDALUS2$^{\dagger,\star}_{\ddagger}$ on the simplified gridworld domain, where $\dagger$ = NPDIS and $\ddagger$ = CUT or $\dagger$ = CWPDIS and $\ddagger$ = BCa, and $\star \in \{\text{None}, k\text{-fold}\}$.

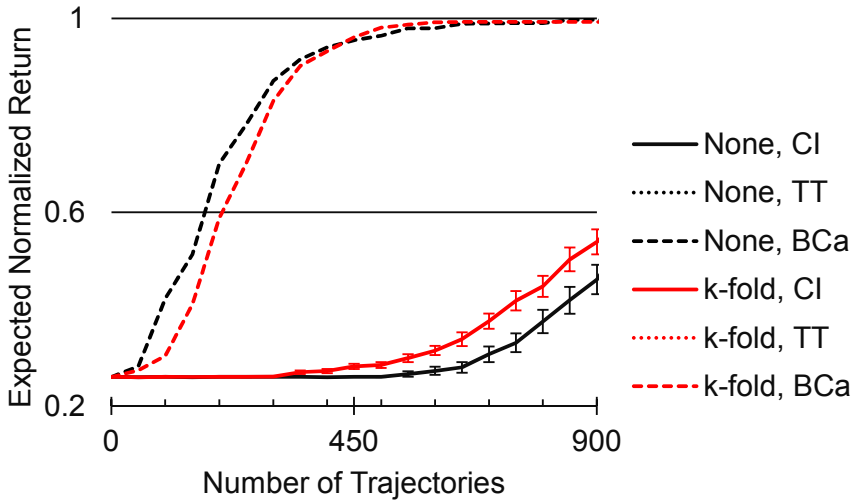


**Figure 5.6.** Performance of DAEDALUS2$^{\dagger,\star}_{\ddagger}$ on the mountain car domain, where $\dagger$ = NPDIS and $\ddagger$ = CUT or $\dagger$ = CWPDIS and $\ddagger$ = BCa, and $\star \in \{\text{None}, k\text{-fold}\}$.

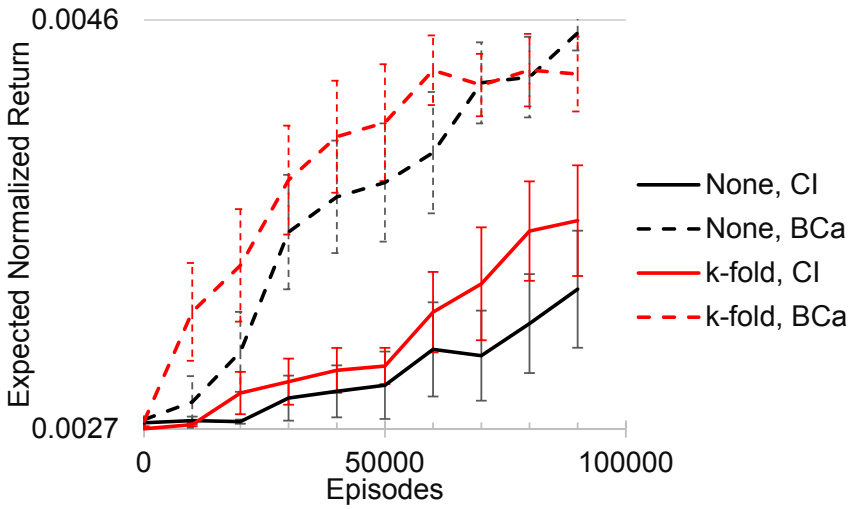


**Figure 5.7.** Performance of DAEDALUS2$^{\dagger,\star}_{\ddagger}$ on the digital marketing domain, where $\dagger$ = NPDIS and $\ddagger$ = CUT or $\dagger$ = CWPDIS and $\ddagger$ = BCa, and $\star \in \{\text{None}, k\text{-fold}\}$. Due to runtime limitations, this plot was only averaged over 10 trials.

## 5.7 Discussion and Conclusions

In this chapter we formalized the notion of a *safe* reinforcement learning algorithm. Recall that our definition of safety is one of many reasonable definitions, as described in Section 2.6. After formalizing the notion of safety in this context, we presented the first practical safe batch and incremental reinforcement learning algorithms. These algorithms allow the user to specify a baseline performance level, $\rho_-$, and a confidence level, $\delta$, to quantify how much risk is acceptable for the application at hand. They then use historical data to search for policies with large expected return, while ensuring that the probability that they return a policy with performance less than $\rho_-$ is at most $\delta$.

The viability of our batch algorithm, *safe policy improvement* (SPI), and our iterative algorithms, Daedalus and Daedalus2, hinge on the foundation that we laid in the previous chapters, which allows us to perform high confidence off-policy evaluation (HCOPE) using remarkably little historical data. However, the creation of policy improvement algorithms given our HCOPE algorithms was not a trivial task—we had to be careful to avoid the problem of multiple comparisons, to avoid using the same data to both select a policy and to test it for safety, and to avoid overfitting our training data.

Although the resulting algorithms require more data than traditional (unsafe) reinforcement learning algorithms *with optimized hyperparameters*, they surprised us with how *little* data was necessary to find safe policy improvements. Unlike PAC RL methods and CPI, which require hundreds of thousands to millions of trajectories to provide statistical guarantees about learning, our methods can function using only tens to hundreds of trajectories, and can converge to near-optimal policies within a few thousand episodes.

**Most importantly, we have shown that safe reinforcement learning is tractable.** We expect that the algorithms that we have proposed leave significant

room for further improvement, as there were too many design choices for us to optimize them all. It is our hope that this preliminary work will convince researchers that high confidence policy improvement is a goal that is within reach, and will result in new algorithms that outperform the methods that we have presented.

# CHAPTER 6

# CONCLUSION AND FUTURE WORK

**In this dissertation we have made several contributions, one of which is particularly significant.**

We began with a formal analysis of importance sampling variants for reinforcement learning in Chapter 3. These importance sampling estimators can be used to estimate the performance of one policy, called the *evaluation policy*, using data collected from the application of a different policy, called the *behavior policy*. We reiterated several known results, including that some of the estimators are unbiased and that if there is only a single behavior policy, then some are strongly consistent. We showed that one estimator from the literature is not a consistent estimator as was previously believed, and we corrected it to produce a consistent estimator. This new estimator, which we call CWPDIS, outperforms all of the other estimators in our experiments. We also showed that most of the estimators are strongly consistent estimators of the evaluation policy's performance even if there are multiple behavior policies (but if an additional restriction is satisfied).

In Chapter 4 we leveraged the importance sampling variants discussed in Chapter 3 to show how historical data from several behavior policies can be used to produce tight confidence bounds on the performance of a new evaluation policy. We presented both exact and approximate methods—the former produce actual statistical confidence bounds, while the latter produce good approximations thereof. Intuitively, while the exact methods have an error rate of *at most* $\delta$ (this is guaranteed by the theoretical derivation, subject to the lone assumption that the environment is a POMDP), the approximate methods have error rates of *approximately* $\delta$. The key to producing

a viable exact HCOPE algorithm was our derivation of a concentration inequality, which we call the *collapsed upper tail* (CUT) inequality, that is particularly well suited to HCOPE. In our experiments we showed how our HCOPE algorithm could be used to provide the user of any reinforcement learning algorithm with confidence that a policy that has never before been used is actually safe to use.

Lastly, in Chapter 5 we showed how our HCOPE algorithms, which built on our importance sampling variants, can be leveraged to perform "safe" policy improvement.[1] The resulting algorithms, *safe policy improvement* (SPI) and DAEDALUS, take as input historical data from any number of previous policies, a confidence level, $\delta$, and a performance lower bound, $\rho_-$ (e.g., the performance of the currently deployed policy). They then search for a policy that is expected to perform best, subject to the requirement that safety is always insured—the probability that a policy is returned with performance below $\rho_-$ is at most $\delta$.

The most significant contribution of this dissertation is not one of the proofs or the new algorithms, but rather the overarching message that **HCOPE and SPI are tractable problems.** When this project began, we had little hope that the problem of HCOPE (particularly *exact* HCOPE) would be tractable, and our first experiments supported this—using ordinary importance sampling with the Chernoff-Hoeffding concentration inequality required millions of trajectories to get a decent high confidence lower-bound on the performance of a policy for a two-state two-action *Markov decision process* (MDP). For these initial experiments (which are not described in this dissertation) we were forced to use this trivial domain because the experiments that are in this dissertation were beyond hope.

This dissertation describes the many incremental improvements that, together, brought HCOPE and SPI from an interesting but unrealistic and impractical idea to

[1] Recall that our definition of safety is one of many reasonable definitions, as described in Section 2.6

a practical set of algorithms. Our empirical studies with these algorithms show the surprising and now indisputable result that HCOPE and SPI are tractable problems. That is, the amount of historical data needed to produce tight confidence bounds on the performance of a policy or to search for a policy with a large lower bound on its performance is *not* as prohibitive as we expected it to be. However, we would like to make it clear that, although HCOPE and SPI are tractable for some problems (more than we anticipated), there are of course many problems that are not *yet* within reach—particularly problems where the behavior and evaluation policies are very different and when trajectories are particularly long.

It is our hope that the algorithms that we have presented will *not* stand the test of time. During their construction we faced many engineering decisions, some of which are described again in the following section. It is likely that at some point we made a decision that degraded the performance of our algorithms. However, they have served their purpose—they have shown that the HCOPE and SPI problems can be solved using a realistic amount of data. We hope that this will inspire researchers to propose their own methods, which improve upon our own, and that the development of increasingly data-efficient safe reinforcement learning algorithms will catalyze the widespread adoption of reinforcement learning algorithms for suitable real-world problems.

## 6.1 Future Work

As suggested by the previous section, we have constructed a sequence of methods that build upon each other, with importance sampling methods at the foundation, HCOPE methods in the middle, and SPI methods at the top. Although we are aware of no mistakes that endanger our safety claims, we suspect that our derivations have been rife with suboptimal engineering decisions that result in decreased data efficiency. For example:

1. Is there a variant of importance sampling that yields even better performance than NPDIS and CWPDIS, particularly if some domain-specific knowledge is available?

2. The AM inequality has no inherent dependence on the upper bound of a random variable (when computing a lower bound on its mean). Can it be improved beyond the CUT inequality by basing it on a statistic other than the Kolmogorov-Smirnov statistic (which is implicit in its use of the Dvoretsky-Kiefer-Wolfowitz inequality)?

3. The CUT inequality is based on the MPeB inequality. Is there a different inequality (that does have a dependence on the upper bound of the random variables) that would produce better results when using our thresholding approach?

4. How best could the threshold, $c$, be computed for the CUT inequality? We use a heuristic method to decide how much data to allocate to optimizing this hyperparameter, and our optimization is based on a biased estimator of how good a given value of $c$ is. Could any of these components be improved?

5. Our approach to HCOPE is based directly on the returns. Could better performance be attained for some applications by instead building a model of the transition dynamics (and reward function, if it is not known), and bounding how much the expected returns computed from this model can differ from the true POMDP?

6. We used a heuristic to decide how much data to use to search for the candidate policy, and how much to use to test it for safety. Can this decision be better optimized?

7. Do other methods for mitigating the problem of multiple comparisons yield better performance (e.g., testing $k$ candidate policies for safety, each with a confidence level of $\delta/k$)?

8. We used a heuristic to determine which candidate policy to return. Can this heuristic be improved? Particularly, some preliminary tests suggests that the candidate policy is often still predicted to be safe, but then fails the subsequent safety test.

9. We used one form of regularization to ensure that we do not over-fit the training data. Do other methods perform better?

10. We used CMA-ES to perform global searches over policy space. These searches have high computational complexity, and may not be feasible for some problems. Could alternate approaches provide equal performance with much lower computational complexity? For example, one might use an off-policy gradient method to estimate the gradient of the objective function with respect to the parameters of a parameterized policy. The search of policy space could then be limited to the parameterized policies that lie along this gradient.

11. Does our use of importance sampling ignore some of the known structure of the problem? For example, might an approach like the one outlined in Section 4.2.3 produce tighter confidence bounds than our HCOPE methods?

Perhaps most importantly, our methods only apply to POMDPs—they do not apply to nonstationary problems.[2] Can HCOPE and SPI methods be developed that are robust to environments where the state-transition probabilities slowly drift between episodes? If so, how should Daedalus be modified? Specifically, if it remains

[2]POMDPs can model nonstationarity only *within* an episode, not across episodes.

as is, it may converge to a nearly-deterministic policy. If the state-transition probabilities change, then sampling historical data from this nearly-deterministic (and now sub-optimal) policy may not be very informative. The introduction of mechanisms that ensure that the policy remains stochastic may be beneficial for nonstationary problems.

# BIBLIOGRAPHY


Y. S. Abu-Mostafa, M. Magdon-Ismail, and H. T. Lin. *Learning from Daga: A Short Course*. AMLBook, 2012.

A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin. Reachability-based safe learning with gaussian processes. In *IEEE Conference on Decision and Control*, pages 1424–1431, 2014.

T. W. Anderson. Confidence limits for the value of an arbitrary bounded random variable with a continuous distribution function. *Bulletin of The International and Statistical Institute*, 43:249–251, 1969.

K. J. Åström and T. Hägglund. *PID Controllers: Theory, Design, and Tuning*. ISA: The Instrumentation, Systems, and Automation Society, 1995.

P. Auer, T. Jaksch, and R. Ortner. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 99:1563–1600, 2010.

Y. Benjamin and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, 57 (1):289–300, 1995.

D. P. Bertsekas. *Dynamic programming and stochastic control*. Academic Press, 1976.

D. P. Bertsekas. Approximate policy iteration: A survey and some new methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011.

D. P. Bertsekas and S. E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

N. Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 2014.

L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

E. Brunskill and L. Li. PAC-inspired option discovery in lifelong reinforcement learning. 2014.

S. Bubeck, N. Cesa-Bianchi, and G. Lugosi. Bandits with heavy tail. *Arxiv*, arXiv:1209.1727, 2012.

J. Carpenter and J. Bithell. Bootstrap confidence intervals: when, which, what? a practical guide for medical statisticians. *Statistics in Medicine*, 19:1141–1164, 2000.

L. E. Champless, A. R. Folsom, A. R. Sharrett, P. Sorlie, D. Couper, M. Szklo, and F. J. Nieto. Coronary heard disease risk prediction in the atherosclerosis risk in communities (ARIC) study. *Journal of Clinical Epidemiology*, 56(9):880–890, 2003.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, third edition, 2009.

A. C. Davison and D. V. Hinkley. *Bootstrap Methods and their Application*. Cambridge University Press, Cambridge, 1997.

M. A. Diouf and J. M. Dufour. Improved nonparametric inference for the mean of a bounded random variable with applicaiton to poverty measures. 2005. URL `http://web.hec.ca/scse/articles/Diouf.pdf`.

A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of a sample distribution function and of the classical multinomial estimator. *Annals of Mathematical Statistics*, 27:642–669, 1956.

B. Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, London, 1993.

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

A. R. Folsom, L. E. Chambless, B. B. Duncan, A. C. Gilbert, and J. S. Pankow. Prediction of coronary heart disease in middle-aged adults with diabetes. *Diabetes Care*, 26(10):2777–2784, 2003.

Z. Guo and E. Brunskill. Concurrent PAC RL. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence*, 2015.

S. Haddadin, M. Suppa, S. Fuchs, T. Bodenmüller, A. Albu-Schäffer, and G. Hirzinger. Towards the robotic co-worker. *Robotics Research*, 70:261–282, 2011.

A. Hallak, F. Schnitzler, T. Mann, and S. Mannor. Off-policy model-based learning under unknown factored dynamics. *Arxiv*, arXiv:1502.03255v1, 2015.

N. Hansen. The CMA evolution strategy: a comparing review. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006.

R. M. Houtman, C. A. Montgomery, A. R. Gagnon, D. E. Calkin, T. G. Dietterich, S. McGregor, and M. Crowley. Allowing a wildfire to burn: Estimating the effect on future fire suppression costs. *International Journal of Wildland Fire*, 22:871–882, 2013.

Future Life Institute. Research priorities for robust and beneficial artificial intelligence: an open letter, January 2015. URL `http://futureoflife.org/misc/open_letter`.

P. A. Ioannou. *Robust Adaptive Control*. Dover Publications, 2012.

J. Jiang. *Large Sample Techniques for Statistics*. Springer-Verlag New York, 2010.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

S. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274, 2002a.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274, 2002b.

H. K. Khalil. *Nonlinear Systems*. Prentice Hall, third edition, 2001.

J. Kim, H. Kim, K. Lakshmanan, and R. Rajkumar. Parallel scheduling for cyber-physical systems: Analysis and case study on a self-driving car. In *International Conference on Cyber-Physical Systems*, pages 31–40, April 2013.

R. M. Kretchmar, P. M. Young, C. Anderson, D. Hittle, M. Anderson, C. Delnero, and J. Tu. Robust reinforcement learning control with static and dynamic stability. *International Journal of Robust and Nonlinear Control*, 11:1469–1500, 2001.

S. Kuindersma, R. Grupen, and A. G. Barto. Variable risk control via stochastic optimization. 32(7):806–825, 2013.

M. Lagoudakis and R. Parr. Model-free least-squares policy iteration. In *Neural Information Processing Systems: Natural and Synthetic*, pages 1547–1554, 2001.

T. Lattimore and M. Hutter. PAC bounds for discounted MDPs. *Arxiv*, arXiv:1202.3890, 2012.

B. Liu, J. Liu, M. Ghavamzadeh, S. Mahadevan, and M. Petrik. Finite-sample analysis of proximal gradient TD algorithms. In *Proceedings of the 31th Conference on Uncertainty in Artificial Intelligence*, 2015.

L. Lubin and M. Athans. Linear quadratic regulator control. In W. S. Levine, editor, *The Control Handbook*, chapter 39. CRC press, 1996.

S. Mannor, D. Simester, P. Sun, and J. N. Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, February 2007.

P. Massart. The tight constraint in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability*, 18(3):1269–1283, 1990.

P. Massart. *Concentration Inequalities and Model Selection*. Springer, 2007.

A. Maurer and M. Pontil. Empirical Bernstein bounds and sample variance penalization. In *Proceedings of the Twenty-Second Annual Conference on Learning Theory*, pages 115–124, 2009.

B. Moore, P. Panousis, V. Kulkarni, L. Pyeatt, and A. Doufas. Reinforcement learning for closed-loop propofol anesthesia: A human volunteer study. In *Innovative Applications of Artificial Intelligence*, pages 1807–1813, 2010.

S. Niekum, A. G. Barto, and L. Spector. Genetic programming for reward function search. In *IEEE Transactions on Autonomous Mental Development*, volume 2, pages 83–90, 2010.

A. O'Dwyer. *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press, 2003.

T. J. Perkins and A. G. Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3:803–832, 2003.

M. Pirotta, M. Restelli, A. Pecorino, and D. Calandriello. Safe policy iteration. In *Proceedings of the 30$^{th}$ International Conference on Machine Learning*, 2013.

M. Powell and J. Swann. Weighted importance sampling— a Monte-Carlo technique for reducing variance. *Inst. Maths. Applics.*, 2:228–236, 1966.

D. Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2000.

R. Y. Rubinstein and D. P. Kroese. *Simulation and the Monte Carlo Method*. Wiley-Interscience, second edition, 2007.

P. K. Sen and J. M. Singer. *Large Sample Methods in Statistics An Introduction With Applications*. Chapman & Hall, 1993.

D. Silver, L. Newnham, D. Barker, S. Weller, and J. McFall. Concurrent reinforcement learning from customer interactions. In *Proceedings of the 30th International Conference on Machine Learning*, pages 924–932, 2013.

S. Singh, R. Lewis, and A. Barto. Where do rewards come from? In *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, Austin, TX, 2009.

J. Sorg, S. Singh, and R. Lewis. Internal rewards mitigate agent boundedness. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

M. T. Söylemez, N. Munro, and H. Baki. Fast calculation of stabilizing PID controllers. *Automatica*, 39(1):121–126, 2003.

R. F. Stengel. *Stochastic Optimal Control*. Wiley-Interscience, 1986.

A. Strehl and M. Littman. A theoretical analysis of model-based interval estimation. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 856–863, 2005.

A. Strehl and M. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

A. Strehl, L. Li, and M. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. Pac model-free reinforcement learning. In *International Conference on Machine Learning*, pages 881–888, 2006.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

R. S. Sutton, C. Szepesvári, and H. R. Maei. A convergend $o(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In *Advances in Neural Information Processing Systems 21*, 2009.

I. Szita and C. Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the International Conference on Machine Learning*, pages 1031–1038, 2010.

J. R. Tetreault and D. J. Littman. Comparing the utility of state features in spoken dialogue using reinforcement learning. In *Proceedings of the Human Language Technology/North American Association for Computational Linguistics*, 2006.

D. Thapa, I. Jung, and G. Wang. Agent based decision support system using reinforcement learning under emergency circumstances. *Advances in Natural Computation*, 3610:888–892, 2005.

G. Theocharous and A. Hallak. Lifetime value marketing using reinforcement learning. In *The 1st Multidisciplinary Conference on Reinforcement Learning and Decision Making*, 2013.

G. Theocharous, P. S. Thomas, and M. Ghavamzadeh. Personalized ad recommendation systems for life-time value optimization with guarantees. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2015a.

G. Theocharous, P. S. Thomas, and M. Ghavamzadeh. Ad recommendation systems for life-time value optimization. In *TargetAd 2015: Ad Targeting at Scale, at the World Wide Web Conference*, 2015b.

P. S. Thomas. A reinforcement learning controller for functional electrical stimulation of a human arm. Master's thesis, Department of Electrical Engineering and Computer Science, Case Western Reserve University, August 2009.

P. S. Thomas, W. Dabney, S. Mahadevan, and S. Giguere. Projected natural actor-critic. In *Advances in Neural Information Processing Systems 26*, 2013.

P. S. Thomas, S. Niekum, G. Theocharous, and G. D. Konidaris. Policy evaluation using the $\Omega$-return. In *In submission*, 2015a.

P. S. Thomas, Y. Pantazis, G. Arampatzis, and I. Gemp. On the benefits of using measure theoretic probability for reinforcement learning research. In *In submission*, 2015b.

P. S. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence off-policy evaluation. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence*, 2015c.

P. S. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence policy improvement. In *International Conference on Machine Learning*, 2015d.

H. van Hasselt, A. R. Mahmood, and R. S. Sutton. Off-policy TD($\lambda$) with true online equivalence. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, 2014.

R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye. *Probability & Statistics for Engineers & Scientists*. Prentice Hall, eighth edition, 2007.

N. Wiener. *God and Golem, Inc. A comment on Certain Points where Cybernetics Impinges on Religion*. The M.I.T. Press, Cambridge, Massachusetts, 1964.

R. R. Wilcox. *Introduction to Robust Estimation and Hypothesis Testing*. Third edition, 2012.

R. R. Wilcox and H. J. Keselman. Modern robust data analysis methods: Measures of central tendency. *Psychological Methods*, 8(3):254–274, 2003.

K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice Hall, 1995.