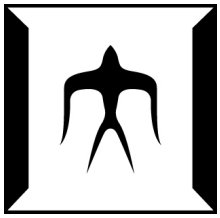
論文 / 著書情報
Article / Book Information

題目(和文)	強化学習の統計的理論とそのロボット制御への応用
Title(English)	Statistical Theory of Reinforcement Learning with Applications to Robot Control
著者(和文)	TINGTING ZHAO
Author(English)	TINGTING ZHAO
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第9557号, 授与年月日:2014年3月26日, 学位の種別:課程博士, 審査員:杉山　将,佐藤　泰介,篠田　浩一,藤井　敦,瀬々　潤
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第9557号, Conferred date:2014/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Statistical Theory of Reinforcement Learning with Applications to Robot Control

Tingting Zhao

March 2014



Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology

Thesis Committee:
Masashi Sugiyama, Chair
Taisuke Sato
Koichi Shinoda
Atsushi Fujii
Jun Sese

Submitted in partial fulfillment of
the requirements for the degree of

Doctor of Engineering

Keywords: reinforcement learning, policy gradient, variance reduction, optimal baseline, importance sampling, robotics

To my family

Abstract

Reinforcement learning offers a framework to robotics such that a robot can autonomously discover the optimal action through the interaction with the underlying environment. Among reinforcement learning methods, the policy gradient approach is a class of flexible and powerful model free methods, particularly for problems with continuous actions such as robot control. A common challenge in this scenario is how to reduce the variance of policy gradient estimates for reliable policy updates. The goal of this thesis is to mitigate the large variance problem of policy gradient estimates.

A classical policy gradient method, REINFORCE, tends to suffer from the instability of gradient estimates and thus leads to a slow convergence. To cope with this problem, a method called *policy gradients with parameter-based exploration* (PGPE) was proposed. The experimental success of PGPE was demonstrated; however, theoretical properties were not clear. In this thesis, we first prove that the variance of gradient estimates in PGPE is smaller than that of REINFORCE under mild assumptions. We then derive the *optimal baseline* for PGPE, which contributes to further reducing the variance. We also theoretically show that PGPE with the optimal baseline is more preferable than REINFORCE with the optimal baseline in terms of the variance of gradient estimates. Finally, we demonstrate the usefulness of the proposed improvement of PGPE through experiments.

However, the standard PGPE still requires a relatively large number of samples to obtain accurate gradient estimates, which can be a critical bottleneck in real-world applications that require large costs and time in data collection. In order to solve this problem, we combine the following three ideas and give a highly data-effective policy gradient method: (a) PGPE, which is a recently proposed policy search method with the low variance of gradient estimates, (b) an *importance*

sampling technique, which allows us to effectively reuse previously gathered data, and (c) an *optimal baseline* technique, which minimizes the variance of gradient estimates while the unbiasedness of the gradient estimates is maintained. For the proposed method, we give theoretical analyses of the variance of gradient estimates and show its usefulness through extensive experiments. Moreover, we also investigate the benefit of the proposed method in complex high-dimensional humanoid robotic experiments, and the results show that the proposed method is promising.

Given the solid theoretical analyses and the encouraging experimental results, we conclude that the proposed methods compare favorably with the corresponding state-of-the-art methods. Therefore, they can be applied to real-world robot control tasks and worth a further study in the future.

Acknowledgments

I am extremely grateful to my supervisor Professor Masashi Sugiyama for his support, patient guidance, enthusiastic encouragement and constructive suggestions during my Ph.D. studies. I would also like to express my great appreciation to my mentor Dr. Hirotaka Hachiya, who introduced me important ideas of reinforcement learning field, gave me valuable advice and productive discussions about my research. The most important is that Dr. Hirotaka Hachiya taught me how to do research and encouraged me a lot during the toughest time of my studies.

For investing the usefulness of my proposed method in humanoid robot tasks, a lot of thanks to Dr. Jun Morimoto of ATR, kyoto, who provided me the simulator of the humanoid robot CBi and introduced me valuable knowledge of robotics. Also, I wish to express my gratitude to my college Voot Tangkaratt, who did a great job in the humanoid robot experiments.

My research project in Japan were financially supported by the Japanese Government MEXT scholarship since October 2010. Without the MEXT scholarship, I could not afford my life in Japan, not to mention my research. Thus, I would like to express my sincere appreciation to MEXT for offering me the scholarship. A lot of thanks to Ning Xie, Gang Niu, Song Liu, Yao Ma, Hao Zhang, Christo and all members in my lab for your helps and the happy days with you together.

The last but definitely not the least, I would like to express my special thanks to my family. I am grateful to my husband for his understanding and support. I know it deeply that without his kind support and encouragement, my ph.d. work would not go well. I would like to show my respect to my mother in law and farther in law. I feel really regret that I could not stay with my mother and take care of her when she was sick during my studies. I would like to say sorry to my parents that I could not honor them for these 3 years. Meanwhile, I want to show

my great thanks to my dear elder brother, who takes very good care of my parents and companies with them all the time so that I can focus on my research in Japan.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Reinforcement learning enables a robot to autonomously discover an optimal behavior in an unknown environment. To this purpose, this thesis contributes to developing statistical approaches to reinforcement learning and providing theoretical supports for the proposed methods.

1.1 Reinforcement Learning in Machine Learning

The idea that we learn by interacting with the environment is probably the first occur to us when we think about the nature of *learning* (Schacter et al., 2011). Learning is the activity of inferring certain unknown facts based on some known facts and some knowledge of the environment. If the subject of learning is a person, it is called *human learning*. Besides human beings, animals can also learn, it is called *animal learning*. Similarly, besides these creatures, programs in computer can also learn, which is referred to as *machine learning*.

Machine learning is a natural outgrowth of the intersection of *computer science* and *statistics* (Mitchell, 2006). However, they have different goals: Computer science emphasizes how to manually program computers; Machine learning emphasizes how to get computers to program themselves and it focuses on predicting in the future; Statistics emphasizes what conclusion can be inferred from data and it focuses on understanding the past. According to the definition of Mitchell (2006), machine learning seeks to answer this question: How can we build computer systems that automatically improve with experience, and what

are the fundamental laws that govern all learning processes. To be more precise, machine learning is programming computers to optimize a performance criterion using example data or past experience.

There are three major learning types in machine learning (Murphy, 2012):

- *Supervised learning*: The goal is to learn a mapping from the input to an output given training data. The training data used in supervised learning is labeled data, e.g.,

  $$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\},$$

  where $\{x_i\}_{i=1}^n$ are the input data, $\{y_i\}_{i=1}^n$ are the labels given by the supervisor, and n is the number of training samples. The form of the output can be anything in principle, but most methods assume that y_i is categorical variable from some finite set $y_i \in \{1, 2, \cdots, C\}$, or a real valued scalar. When y_i is categorical number, the problem is known as *classification*, and when y_i is real value, the problem is called *regression* (Bishop, 2006). Supervised learning plays an important role in applications as diverse as *face detection* and *spam filtering*.

- *Unsupervised learning*: The aim is to find the hidden structure in the data. The training data is given as the un-labeled data, e.g.,

  $$\{x_1, x_2, \ldots, x_n\},$$

  there is no supervisor and we only have input data in unsupervised learning. This is sometimes called *knowledge discovery*. Unsupervised learning is closely related to the problem of density estimation, that is, we want to build models of the form $p(x)$ (Murphy, 2012). Important examples of unsupervised learning are *clustering* and *dimensionality reduction* (Bishop, 2006).

- *Reinforcement learning*: It is concerned with how an *agent* ought to take actions in an unknown *environment* so as to maximize the cumulative *rewards* (Sutton and Barto, 1998). The agent is not told which actions to take, but instead must discover which actions give the most reward. The reward defines

what are the good and bad actions for the agent. Reinforcement learning has been applied successfully to various problems, including *robot control*, *elevator scheduling*, *telecommunications*, and *economics* (Kaelbling et al., 1996).

Reinforcement learning may be understood by contrasting the problem with other areas of study in machine learning. Roughly speaking, reinforcement learning is in the middle of supervised learning and unsupervised learning. In supervised learning, the correct answers are provided by the supervisor in the training samples; In reinforcement learning, the learner has no explicit teacher as in supervised learning, but it does have a reward signal, which directly connects to its environment; In unsupervised learning, the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. The reward distinguishes reinforcement learning from supervised learning and unsupervised learning.

Furthermore, reinforcement learning is essentially different from supervised learning. The problems solved by supervised learning pose no interactive component. The supervised learning methods depend on training and testing samples as independent and identical distributed random variables. Also, the methods are built to operate in the world where every decision has no effect on the future examples. Further, within supervised learning scenarios, the correct answer is given to the learner during the training phase, so there is no ambiguity about action choices. On the other hand, the agent in reinforcement learning, is not told which actions to take, but instead the agents will discover actions which can yield the most reward by themselves. The action affects not only the immediate reward but also the next situation due to the state transition, through that, all subsequent rewards, hence, it affects the future, and the interactions between agent and environment are not independent and identical distributed.

Machine learning and *artificial intelligence* also have been closely related since long before (Mitchell, 2006). In particular, there is greater contact between artificial intelligence and reinforcement learning (Sutton and Barto, 1998). In artificial intelligence, the key issues of the agents are *perceiving*, *searching*, *planning*, *learning*, *acting*, and *communicating* (Poole and Mackworth, 2010). Machine learning includes lots of advanced data analysis, thus it is much more

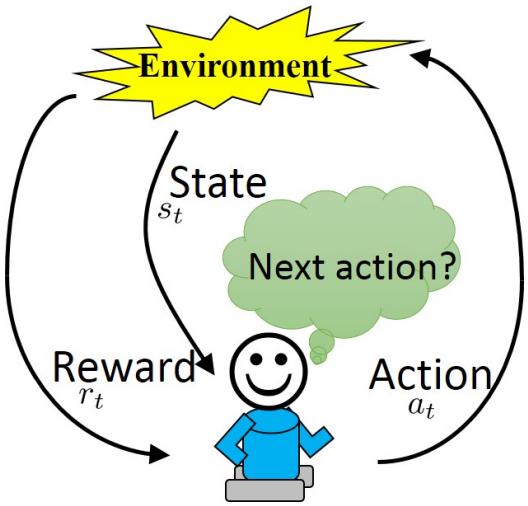


Figure 1.1: Illustration of reinforcement learning framework.

general than the specific learning in artificial intelligence. Nowadays, machine learning is regarded as an independent research field instead of a branch of artificial intelligence, and learning in artificial intelligence mainly means reinforcement learning. On the other hand, reinforcement learning is very closely related to the field of *optimal control* (Kirk, 2012). Both reinforcement learning and optimal control address the problem of finding an optimal policy that optimizes an objective function such as cumulative rewards. However, optimal control assumes perfect knowledge of the environment in the form of a model (Bertsekas, 2000). Reinforcement learning extends ideas from optimal control and stochastic approximation to address the broader and more ambitious goal, which is also referred to as *adaptive optimal control* (Sutton et al., 1992).

A typical setting where reinforcement learning operates is shown in Figure 1.1, the agent receives the state of the environment and a reward associated with the state transition. Then the agent calculates an action which is sent back to the environment. In response, the environment makes a transition to a new state and the cycle is repeated. The problem is to learn a way of behaving so as to maximize the total reward.

Basically, the agent and the environment consist the reinforcement learning system. More specifically, there are four main elements in a reinforcement learning system: a *policy*, a *reward function*, a *return* (or a *value function*), and op-

tionally, a *model* of the environment (Kaelbling et al., 1996). A policy defines the agent's way of behaving at a given time, it is the core of reinforcement learning agent. A reward function defines the goal of a problem, which maps each sensed state of the environment to a single number. The reward function shows what is good in an immediate sense, whereas a return or a value function specifies what is good in a long run. The value function represents expected future rewards as a function of a state or a state-action pair. The return is the expected cumulative rewards along a trajectory. The agent's objective is to find a policy which can maximize the return or the value function. The final element of some reinforcement learning systems is a model of the environment, which describes what the next state of the agent will be given the current state and action, it is used to mimic the behavior of the environment. The model of the environment is optional, which differs *model-based* RL methods and *model-free* RL methods (Buşoniu et al., 2010). The model-based methods explicitly model the environment first, then learn the policy based on the model of the environment. On the other hand, the model-free reinforcement learning approaches learn the policy without explicitly modeling the environment, but learn the policy directly based on the samples obtained from interactions with the environment.

In this thesis, we will focus on the model-free reinforcement learning algorithms.

1.2 Reinforcement Learning in Robotics

From simple house-cleaning robots to humanoid robots, and a variety of robots used in our daily life are increasing drastically. There are many kinds of robots, such as *service robots*, *mobile robots*, *collaborate robots*, and *military robots* (Thrun et al., 2005). Robots can do a great favor to human beings, commercial and industrial robots are now used in widespread performing jobs more cheaply or with greater accuracy and reliability than humans, some robots are also employed for jobs which are too dirty, dangerous or dull to be suitable for humans. It has been reported that 40% of all the robots in the world are in Japan, making Japan the country with the highest number of robots.

In order to perform their jobs, robots need to be controlled so as to take appro-

priate actions in the interacted environment. For example, the robots may figure out how to achieve a task without hitting obstacles, falling over, etc. To date, the controllers for these robots used in our daily life are usually manually designed by a human engineer. Designing robots require extensive experience and high degree of expertise. Moreover, the designed robots are based on the assumption that the robots' behaviors and the environment are correctly modeled (Kober et al., 2013). When robots have to adapt to new environment or when the environment is not modeled sufficiently accurately, the designed robots might be limited. Therefore, it is highly necessary to develop autonomous robots. In this thesis, we try to develop algorithms for the *task-level autonomy* robots, which means a human designer specifies only the task and the robot manages itself to complete it (Deisenroth et al., 2013).

Reinforcement learning offers to robotics a framework, it enables a robot to autonomously discover an optimal action through interactions with environment. As we mentioned in Section 1.1, both reinforcement learning and optimal control address the problem of finding an optimal policy that optimizes an objective function such as cumulative rewards. However, optimal control assumes perfect knowledge of the environment in the form of a model. Thus, optimal control is limited in the development of autonomous robots. Using the power and flexibility of reinforcement learning, the filed of robot control can be further automated.

Let us using humanoid robot *CB-i* to take an example to show how reinforcement learning works in robotics (Cheng et al., 2007). Consider, we try to train CB-i to reach a target object by using its right hand. In this case, the robot must sense the states of the environment to some extent, which is specified by the position of the target object and the internal dynamics of the joint position and velocity. The actions to the robot is the torque sent to motors or the desired accelerations. The function π that generates the actions based on the current position and velocity of the joints is called policy. Then the reinforcement learning problem is to find a policy that maximizes the long term sum of rewards. The reward in this case could be based on the distance between the right hand and the object, the closer the hand to the object, the higher reward the robot can get.

Reinforcement learning is generally a hard problem and many of its challenges are particularly apparent in the robotics. Two main challenges of applying rein-

forcement learning to robotics (Kober et al., 2013):

- High dimensional continuous state and action spaces: Due to the large number of degrees of freedoms in robots, robotic systems have to deal with high dimensional states and actions. As the number of dimensions grows, exponentially more data and computations are needed, which is known as *curse of dimensionality*. In addition, the states and actions of most robots are inherently continuous.

- The high costs of robot interactions with its environment: As robotics deal with complex physical systems, the cost of collecting samples is often prohibitively expensive and too time consuming. For example, a robot learns how to hit a ball in tennis, we need to ask the robot to hit the ball hundreds of times for learning a reliable policy. Hitting the ball hundreds of times may cost several weeks. Moreover, real robot learning tasks require some form of human supervision, there must be an robot engineer spending a lot of time and effort to take care of the robot through maintenance and repairing. Repairing a robot is a factor that can not be negligible, because it is associated with cost, physical labor and long waiting period. For such reasons, real robots samples are expensive in terms of time, labor and potentially, money. Therefore, sample efficient algorithms are essential for robotics.

Considering the above challenges, not every reinforcement learning method is equally suitable for the robotics domain. The reinforcement learning methods developed so far can be categorized into two types: *Policy iteration* where policies are learned based on value function (Sutton and Barto, 1998; Lagoudakis and Parr, 2003) and *policy search* where policy parameters are learned directly to maximize expected future rewards (Williams, 1992; Dayan and Hinton, 1997; Sutton et al., 1999; Kakade, 2002; Sehnke et al., 2010).

Since policy iteration algorithms require filling the complete state action space with data to get the value functions, they struggle with the challenges encountered in robot RL. In order to deal with the continuous state space, value function approximation technique is necessary. In the policy iteration with value function approximation framework, approximation of the value function for the current policy and improvement of the policy based on the learned value function are iteratively

performed until an optimal policy is found. Thus, accurately approximating the value function is a challenge in the value function based approach. So far, various machine learning techniques have been employed for better value function approximation, such as least-squares approximation (Lagoudakis and Parr, 2003), manifold learning (Sugiyama et al., 2008), efficient sample reuse (Hachiya et al., 2009), active learning (Akiyama et al., 2010), and robust learning (Sugiyama et al., 2010). However, because policies are learned indirectly via value functions in policy iteration, improving the quality of value function approximation does not necessarily yield a better policy. Furthermore, because a small change in value functions can cause a big change in policy functions, it is not safe to use the value function based approach for robots. Another weakness of the value function approach is that it is difficult to handle continuous actions because a maximizer of the value function with respect to an action needs to be found for policy improvement. Therefore, policy iteration algorithms in the robotics context is not directly applicable.

On the other hand, policy search algorithms focus on finding optimal policy parameters for a given policy parametrization. It is well suited for robotics as it can cope with high-dimensional continuous state and action spaces. Furthermore, policy search methods allow integrating pre-structured policy for an assigned task straightforwardly (Schaal et al.). In addition, imitation learning from an expert's demonstrations can be used to obtain a good initial policy parameter, which can make the learning process much more efficient (Peters and Schaal, 2006). All these properties simplify the robot learning problem and permit the successful applications to robotics (Bagnell and Schneider, 2001; Kober and Peters, 2011; Ng et al., 2004). Therefore, policy search is often the choice in robotics since it is better at coping with the inherent challenges of robot learning.

In fact, it has been thus far demonstrated that robot learning systems often employ policy search methods rather than value-function based methods (Kober and Peters, 2011; Ng and Jordan, 2000; Peters and Schaal, 2006). Furthermore, the *model-free* approach seems to be more preferred because accurately learning the transition dynamics of complex robots is challenging in robotics(Deisenroth et al., 2013). However, the *model-based* approach is advantageous in that no interaction with the real robot is required once the transition model has been learned and

the learned transition model can be utilized for further simulation (Abbeel et al., 2007; Deisenroth and Rasmussen, 2011). Choice of either going with model-free or model-based methods is not only an ongoing and debatable research theme in machine learning, which is beyond the scope of our study. In this thesis, we will focus on the model free policy search algorithms.

1.3 Contributions

This thesis contributes to developing statistical reinforcement learning algorithms, which enable the robot to autonomously discover the optimal behavior in the unknown environment. In this section, an overview of the contributions in this thesis is explained.

Policy gradient is a useful model-free reinforcement learning approach, but it tends to suffer from instability of gradient estimates. A common challenge in this scenario is how to reduce the variance of policy gradient estimates for reliable policy updates. In this thesis, we analyze and improve the stability of policy gradient methods.

We first prove that the variance of gradient estimates in the policy gradients with parameter-based exploration (PGPE) method (Sehnke et al., 2010) is smaller than that of the classical policy gradient (REINFORCE) method (Williams, 1992) under a mild assumption. We then propose the *optimal baseline* for PGPE, which contributes to further reducing the variance. We also theoretically show that PGPE with the optimal baseline is more preferable than REINFORCE with the optimal baseline in terms of the variance of gradient estimates.

When applying reinforcement learning algorithms to the real world problems, reducing the number of training samples is desirable because the sampling cost is often much higher than the computational cost. Thus, we propose a new policy gradient method equipped with efficient sample reuse, which systematically combines a reliable policy gradient method, PGPE, with *importance sampling* and the optimal constant baseline. We theoretically show that the introduction of the optimal constant baseline can mitigate the large-variance problem of importance weighting under some conditions.

We provide solid theoretical analysis for all the proposed methods. Extensive

experimental results demonstrate that the proposed methods compare favorably with the corresponding state-of-the-art methods.

1.4 Organization

This thesis consists of five chapters (see Figure 1.2). In this section, we give the organization of this thesis.

In Chapter 2, we give the mathematical formulation of reinforcement learning problems and review some existing state-of-the-art algorithms. The reinforcement learning problems are formalized in section 2.1. Then, two fundamental and major paradigms in reinforcement learning are reviewed: we review policy iteration methods in Section 2.2, where we give the definition of the value function, the framework of the policy iteration methods, and a state-of-the-art policy iteration algorithm, i.e., least squares policy iteration; policy search methods are reviewed in Section 2.3, where we review the traditional policy gradients (REINFORCE) method, a natural policy gradients method, policy gradients with parameter based exploration (PGPE) method, and EM-based policy search method.

In Chapter 3, we analyze and improve the stability of the policy gradients method. Section 3.1 describes the motivation and the background knowledge. In Section 3.2, the theoretical properties of REINFORCE and PGPE are investigated. More specifically, we theoretically show that, under a mild condition, PGPE provides more stable gradient estimates than the classical REINFORCE method. In Section 3.3, we further improve the performance of PGPE by deriving the optimal baseline, and give the theoretical analysis of PGPE with the optimal baseline in terms of the variance of gradient estimates. Subsequently, we demonstrate the usefulness of the improved PGPE algorithm through experiments in Section 3.4. Finally, proofs of all the theoretical results given in this chapter are provided in Section 3.5.

In Chapter 4, we propose a new policy gradient method equipped with efficient sample reuse. Section 4.1 gives the motivation and the background knowledge. In Section 4.2, we systematically combine a reliable policy gradient method, PGPE, with importance sampling and the optimal constant baseline, which gives an efficient and practical algorithm. In Section 4.2.2, we theoretically show that the

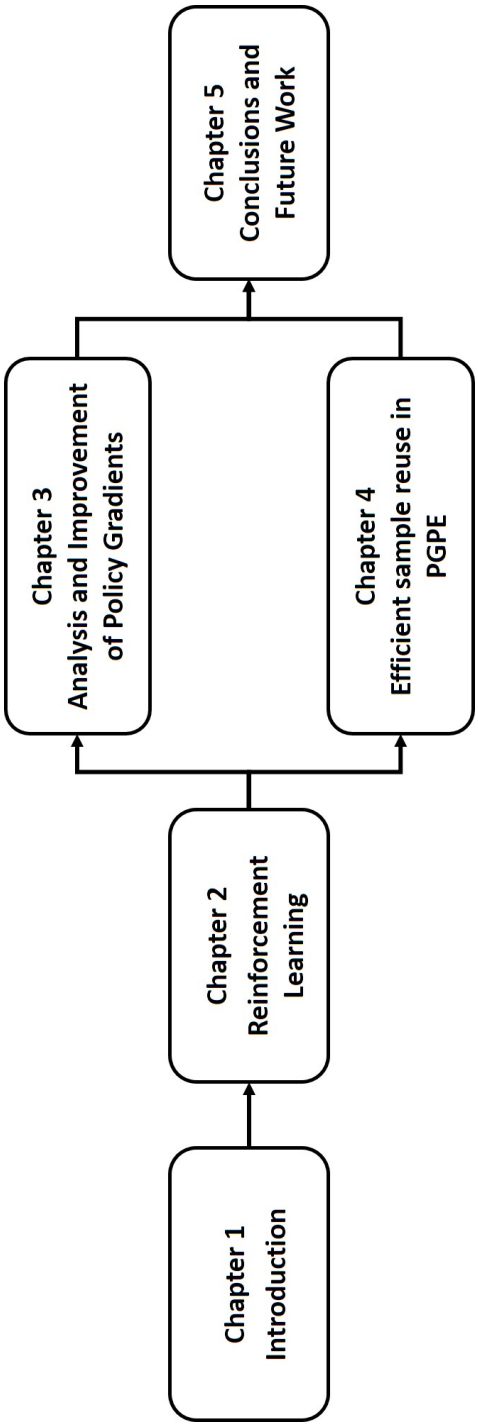


Figure 1.2: Structure of this thesis.

introduction of the optimal constant baseline can mitigate the large-variance problem of importance weighting under some conditions. Subsequently, in Section 4.3, we demonstrate the usefulness of the proposed method through experiments with an artificial domain. We also investigate the effectiveness of the proposed method on high-dimensional problems through robotic experiments in Section 4.3.3. Finally, we give the proofs of all the theoretical results given in this chapter in Section 4.4.

In the end, conclusions and future prospects are discussed in Chapter 5.

Chapter 2

Related Work

This chapter introduces the problem formulation of reinforcement learning: Markov decision processes, then reviews existing algorithms of reinforcement learning.

2.1 Markov Decision Processes

Reinforcement learning problems can be formalized as a *Markov decision process* (MDP) (Sigaud and Garcia, 2013), which is specified by

$$(\mathcal{S}, \mathcal{A}, P_T, P_I, r, \gamma),$$

where

- $\mathcal{S}$ is a set of states,
- $\mathcal{A}$ is a set of actions,
- $P_T(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$ is the transition probability density from current state $\boldsymbol{s}_t$ to next state $\boldsymbol{s}_t$ when action a_t is taken,
- $P_I(\boldsymbol{s})$ is the probability of initial states $\boldsymbol{s}_1$,
- $r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$ is an immediate reward for transition from current state $\boldsymbol{s}_t$ to next state $\boldsymbol{s}_{t+1}$ by taking action a_t,
- $0 < \gamma < 1$ is the discount factor for future rewards.

The states $\mathcal{S}$ and actions $\mathcal{A}$ can be either discrete or continuous. In many robot control problems, those tend to be continuous or very large discrete space, and also high dimensional. In this thesis, we will focus on the high dimensional continuous state and action problems. Let us assume that the dimensionality of state and action space are known. Furthermore, we also assume that the state space $\mathcal{S}$ is fully observed. When this assumption is not true, the problem is called a *partially observable Markov decision process* (POMDP), which is beyond the scope of this thesis.

Policy is the core of reinforcement learning agent, which defines the learning agent's way of behaving at a given time t. Roughly speaking, a policy is a mapping from the perceived states to actions to be taken when in those states, it can either be deterministic or stochastic. Deterministic policy always uses the exact same action for a given state:

$$a_t = \pi(\boldsymbol{s}_t).$$

Stochastic policy maps states to distributions over the action space, i.e.,

$$a_t \sim \pi(a_t|\boldsymbol{s}_t),$$

which represents the conditional probability density of taking action a_t in state $\boldsymbol{s}_t$. The stochastic policy incorporates exploratory actions, and exploration is usually required for getting a better policy in the learning process. For this reason, we employ stochastic policy in this thesis.

The dynamics of an MDP proceeds as follows: Initially, the agent starts from a randomly selected state s_1 following the initial state probability density $p(\boldsymbol{s}_1)$ and chooses an action a_1 based on the policy π. Then the agent makes a transition following the dynamics of the environment $p(\boldsymbol{s}_2|\boldsymbol{s}_1, a_1)$. The transition is repeated T times to get a *trajectory*, which is denoted as $h = [\boldsymbol{s}_1, a_1, \ldots, \boldsymbol{s}_T, a_T]$. Here, T is the time horizon, which can either be finite or infinite. The finite horizon implies that the agent has to control the system within T steps, the corresponding task is called episodic task. On the other side, the infinite horizon MDPs correspond to the continuous tasks. In this thesis, we consider the former case that is finite horizon MDPs.

The *return* (i.e., the discounted sum of future rewards) along h is given by

$$R(h) := \sum_{t=1}^{T} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}),$$

which is the criterion for evaluating a policy. The discount factor γ is introduced to reduce the reward received in the future, which favors rewards received early in the process. Hence, the discounter factor may help us to obtain the policy which takes shorter steps to achieve a certain task because the shorter steps are, the less rewards are discounted.

The expected return for policy π is defined by

$$J_\pi := \int p(h) R(h) \mathrm{d}h,$$

where

$$p(h) = p(\boldsymbol{s}_1) \prod_{t=1}^{T} p(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, a_t) \pi(a_t | \boldsymbol{s}_t)$$

is the probability density of occurring the trajectory h.

The goal of reinforcement learning is to find the optimal policy π^* that maximizes the expected return J_π:

$$\pi^* := \arg\max_\pi J_\pi.$$

2.2 Policy Iteration

In this section, we consider a major category of reinforcement learning algorithms, namely *policy iteration* methods. Policy iteration algorithms evaluate policy by constructing their value functions, and use these value functions to find improved policy. We first introduce the definitions of value functions $V_\pi(\boldsymbol{s})$ and $Q_\pi(\boldsymbol{s}, a)$, then the framework of policy iteration algorithms. At last, we review a state of the art policy iteration algorithm, called *least squares policy iteration* (LSPI).

2.2.1 Value Function

A convenient way to characterize the policy is by using their value functions (Sutton and Barto, 1998). There are two types of value functions: *state value function*

$V_\pi(\boldsymbol{s})$, which estimates how good is the agent to be in a given state under the policy π; *state action value function* $Q_\pi(\boldsymbol{s}, a)$, which evaluates how good it is to perform a given action a in a given state $\boldsymbol{s}$ under the policy π.

The state value function for policy π is defined as

$$V_\pi(\boldsymbol{s}) := E_{\pi,P_T}\left[\sum_{t=1}^{\infty}\gamma^{t-1}r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})|\boldsymbol{s}_1 = \boldsymbol{s}\right],$$

where E_{π,P_T} denotes the expected value following the policy $\pi(a_t|\boldsymbol{s}_t)$ and the transition model $P_T(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$ when starting from the state $\boldsymbol{s}_1 = \boldsymbol{s}$. Similarly, the state action value function $Q_\pi(\boldsymbol{s}, a)$ for policy π is defined as the conditional expected cumulative rewards of taking action a in state $\boldsymbol{s}$ under the policy π:

$$Q_\pi(\boldsymbol{s}, a) := E_{\pi,P_T}\left[\sum_{t=1}^{\infty}\gamma^{t-1}r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})|\boldsymbol{s}_1 = \boldsymbol{s}, a_1 = a\right],$$

where E_{π,P_T} denotes the conditional expectation over the $\{\boldsymbol{s}_t, a_t\}_{t=1}^{\infty}$ following the policy $\pi(a_t|\boldsymbol{s}_t)$ and transition $P_T(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$ starting from $\boldsymbol{s}_1 = \boldsymbol{s}$ and $a_1 = a$.

For any policy π and any state $\boldsymbol{s}$, we can get the following relationship between the value of $\boldsymbol{s}$ and its successor $\boldsymbol{s}'$ (Szepesvari, 2010):

$$V_\pi(\boldsymbol{s}) = E_{\pi,P_T}\left[r(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma V_\pi(\boldsymbol{s}')\right],$$

which is the *Bellman equation* for $V_\pi(\boldsymbol{s})$. The same for the state action value function, we also have the Bellman equation for $Q_\pi(\boldsymbol{s}, a)$:

$$Q_\pi(\boldsymbol{s}, a) = E_{\pi,P_T}\left[r(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma Q_\pi(\boldsymbol{s}', a')\right],$$

where $(\boldsymbol{s}', a')$ is the next state action pair.

One reason for computing the value function for a policy is to help find better policies. Now, we define the optimal policy in terms of the optimal value function. The optimal policy is denoted as π^*, which is given by the optimal state value function defined as

$$V_\pi^*(\boldsymbol{s}) = \max_\pi V_\pi(\boldsymbol{s}).$$

Similarly, the optimal policy also shares the optimal state action value function, which is defined as

$$Q_\pi^*(\boldsymbol{s}, a) = \max_\pi Q_\pi(\boldsymbol{s}, a).$$

Intuitively, the Bellman equation for the optimal state value function $V_\pi^*(\boldsymbol{s})$ shows the fact that the value of the state $\boldsymbol{s}$ under the optimal policy must equal the state action value for the best action from that state $\boldsymbol{s}$:

$$\begin{aligned} & V^*(\boldsymbol{s}) \\ =& \max_a Q^*(\boldsymbol{s}, a) \\ =& \max_a E_{\pi, P_T}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) | \boldsymbol{s}_1 = \boldsymbol{s}, a_1 = a\right] \\ =& \max_a E_{\pi, P_T}\left[r(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma V^*(\boldsymbol{s}') | \boldsymbol{s}_1 = \boldsymbol{s}, a_1 = a\right], \end{aligned}$$

which gives the *Bellman optimal equation* for $V^*(\boldsymbol{s})$. The Bellman optimal equation for the state action value function $Q^*(\boldsymbol{s}, a)$ is

$$Q^*(\boldsymbol{s}, a) = E_{\pi, P_T}\left[r(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma \max_{a'} Q^*(\boldsymbol{s}', a')\right].$$

In this thesis, we focus on the state action value function.

2.2.2 Framework of Policy Iteration

Policy iteration is a major category of reinforcement learning algorithms, which uses value function to obtain the optimal policy (Sutton and Barto, 1998). There are two steps in policy iteration algorithms: *policy evaluation* and *policy improvement*. More specifically, it starts with an arbitrary policy, we denote it as π_1. At each iteration l, the value function $Q_{\pi_l}(\boldsymbol{s}, a)$ of the current policy π_l is determined by solving the Bellman equation, this step is called policy evaluation. When policy evaluation is done, a new policy π_{l+1} is obtained greedily based on the value function $Q_{\pi_l}(\boldsymbol{s}, a)$:

$$\pi_{l+1}(a|\boldsymbol{s}) = \arg\max_a Q_{\pi_l}(\boldsymbol{s}, a),$$

this step is called greedy policy improvement. Policy improvement is also known as the *actor* and policy evaluation is known as the *critic*, because the actor is responsible for the way the agent acts and the critics is responsible for criticizing the way the agent acts. Policy iteration algorithms repeat these two procedures until policy converge:

$$||\pi_{l+1}(a|\boldsymbol{s}) - \pi_l(a|\boldsymbol{s})|| \leq \kappa, \quad \forall s \in \mathcal{S}, \forall a \in \mathcal{A},$$

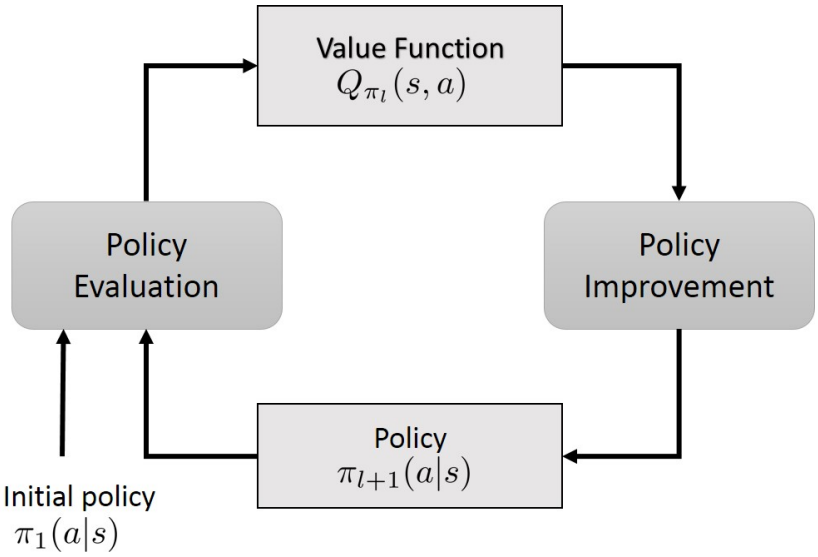


Figure 2.1: Framework of policy iteration method.

where $\kappa > 0$ is usually set small enough and $||\cdot||$ denotes L_2 norm. Figure 2.1 shows the framework of policy iteration algorithm: Starting from initial policy $\pi_1(a|\boldsymbol{s})$, policy evaluation and policy improvement steps are repeated until convergence.

The greedy policy improvement guarantees the improved policy, which is known as the *policy improvement theorem*:

$$Q_{\pi_l}(\boldsymbol{s}, a) \leq Q_{\pi_{l+1}}(\boldsymbol{s}, a),$$

which means that the policy π_{l+1} must be as good as, or better than policy π_l. The equality only holds when $Q_{\pi_{l+1}}(\boldsymbol{s}, a)$ must be $Q^*(\boldsymbol{s}, a)$, and both $\pi_{l+1}(a|\boldsymbol{s})$ and $\pi_l(a|\boldsymbol{s})$ are optimal policies. Thus, policy improvement must give us a strictly better policy except when the policy is already optimal.

In policy iteration, policy improvement can be performed by solving the optimization problem of $Q_{\pi_l}(\boldsymbol{s}, a)$, while the crucial part is policy evaluation. Computing the value function by using Bellman equation is computationally expensive especially when the state action spaces are large, which is known as *curse of dimensionality*. It is computationally intractable when the state and action spaces are continuous. To overcome this problem, value function approximation techniques are proposed.

In next subsection, we will review a state-of-the-art policy iteration algorithm, which employs the value function approximation technique.

2.2.3 Least Squares Policy Iteration

Least squares policy iteration (LSPI) is a policy iteration algorithm, which learns the state action value function with linear architecture for incremental policy improvement within policy iteration framework (Lagoudakis and Parr, 2003). In this subsection, let us review LSPI algorithm.

Let $\hat{Q}_\pi(\boldsymbol{s}, a|\boldsymbol{w})$ be an approximation to $Q_\pi(\boldsymbol{s}, a)$ represented by a parametric approximation with parameters $\boldsymbol{w}$. The basic idea of value function approximation is that parameters $\boldsymbol{w}$ can be adjusted appropriately so that the approximated values are accurate enough to the original values. Typically, function approximation is done using a training set of examples $\{(\boldsymbol{s}, a), Q_\pi(\boldsymbol{s}, a)\}$ that provide the value $Q_\pi(\boldsymbol{s}, a)$ of the target function at a certain point $(\boldsymbol{s}, a)$, which is known as supervised learning. However, in the context of reinforcement learning, the value of the target function is not known in advance, and must be calculated from collected trajectory samples.

A common class of approximation for value function is linear model, which approximates the value function by a linear parametric combination of the basis function :

$$\hat{Q}_\pi(\boldsymbol{s}, a|\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{s}, a),$$

where $\boldsymbol{\phi}(\boldsymbol{s}, a)$ is the k dimensional basis function vector:

$$\boldsymbol{\phi}(\boldsymbol{s}, a) = (\phi_1(\boldsymbol{s}, a), \phi_2(\boldsymbol{s}, a), \cdots, \phi_k(\boldsymbol{s}, a))^T.$$

As the value of the target function is not known, we have to estimate it from the collected samples. Here, a sample is denoted as $(\boldsymbol{s}, a, r, \boldsymbol{s}')$, which is drawn from the process along with the policy π and transition model P_T. A finite set of samples is given as

$$D = \{(\boldsymbol{s}_i, a_i, r_i, \boldsymbol{s}'_i)\}_{i=1}^N.$$

Let Q_{π_l} be the value functions given as a column vector of N data samples under the policy π_l at the l-th iteration, i.e.,

$$Q_{\pi_l} = (Q_{\pi_l}(\boldsymbol{s}_1, a_1), Q_{\pi_l}(\boldsymbol{s}_2, a_2), \ldots, Q_{\pi_l}(\boldsymbol{s}_N, a_N))^T.$$

Also, let $\hat{Q}_{\pi_l}$ be the vector of approximated values as computed by a linear approximation architecture with parameter $\boldsymbol{w}_l$ and basis function $\boldsymbol{\phi}$, i.e.,

$$\hat{Q}_{\pi_l} = \left(\hat{Q}_{\pi_l}(\boldsymbol{s}_1, a_1), \hat{Q}_{\pi_l}(\boldsymbol{s}_2, a_2), \ldots, \hat{Q}_{\pi_l}(\boldsymbol{s}_N, a_N)\right)^T.$$

Now, $\hat{Q}_{\pi_l}$ can be expressed compactly as

$$\hat{Q}_{\pi_l} = \Phi \boldsymbol{w}_l,$$

where $\boldsymbol{w}_l$ is a column vector of length k and Φ is a $N \times k$ matrix of the form

$$\Phi = \begin{pmatrix} \boldsymbol{\phi}(\boldsymbol{s}_1, a_1)^T \\ \boldsymbol{\phi}(\boldsymbol{s}_2, a_2)^T \\ \ldots \\ \boldsymbol{\phi}(\boldsymbol{s}_N, a_N)^T \end{pmatrix}.$$

Each row of Φ contains the value of all basis functions for a certain sample pair $(\boldsymbol{s}, a)$ and each column of Φ contains the value of a certain basis function for all sample pairs.

Bellman Residual Minimizing Approximation Let us recall the Bellman equation for the state action value function:

$$Q_\pi(\boldsymbol{s}, a) = R(\boldsymbol{s}, a) + \gamma E_{\pi, P_T}\left[Q_\pi(\boldsymbol{s}', a')\right], \tag{2.1}$$

where $R(\boldsymbol{s}, a) = E_{p(\boldsymbol{s}'|\boldsymbol{s},a)}[r(\boldsymbol{s}, a, \boldsymbol{s}')]$. In matrix format, the Bellman equation becomes

$$Q_{\pi_l} = \boldsymbol{R} + \gamma E_{\pi_l, P_T}\left[Q'_{\pi_l}\right],$$

where Q_{π_l} and $\boldsymbol{R}$ are vectors of size N.

An obvious way to find a good approximation is to make the approximate value function as close to the Bellman equation as possible. Substituting $\hat{Q}_{\pi_l}$ to Q_{π_l} gives

$$\Phi \boldsymbol{w}_l = \boldsymbol{R} + \gamma E_{\pi, P_T}[\Phi' \boldsymbol{w}_l].$$

Minimizing the L_2 norm of the Bellman residual, i.e., the difference between the left hand side and the right hand side of the above equation, yields a solution

$$\boldsymbol{w}_l = \left((\Phi - \gamma E_{\pi, P_T}[\Phi'])^T (\Phi - \gamma E_{\pi, P_T}[\Phi'])\right)^{-1} (\Phi - \gamma E_{\pi, P_T}[\Phi'])^T \boldsymbol{R},$$

which is known as the *Bellman residual minimizing approximation* to the true value function. Note that the solution is unique since the columns of the basis functions Φ are linearly independent by definition.

Least Squares Fixed Point Approximation Based on the Bellman equation for $Q_\pi(\boldsymbol{s}, a)$, let us define the *Bellman operator* underlying policy π:

$$(T^\pi Q)(\boldsymbol{s}, a) = R(\boldsymbol{s}, a) + \gamma E_{\pi, P_T}\left[Q(\boldsymbol{s}', a')\right].$$

The state action value function $Q_\pi(\boldsymbol{s}, a)$ is the fixed point of the Bellman operator T^π, T^π is an affine linear operator, and it is maximum norm contraction when $0 < \gamma < 1$. With the help of the Bellman operator, the Bellman equation for $Q_\pi(\boldsymbol{s}, a)$ can be written in the compact form

$$T^\pi Q_\pi = Q_\pi.$$

Note that Q_π is the unique solution to the fixed point equation $T^\pi Q_\pi = Q_\pi$.

Another way to find a good approximation is to force the approximate value function to be a fixed point under the Bellman operator:

$$T^{\pi_l} \hat{Q}_{\pi_l} \approx \hat{Q}_{\pi_l}.$$

For that to be possible, the fixed point has to lie in the space of approximate value functions which is the space spanned by the basis functions. Even though $\hat{Q}_{\pi_l}$ lies in that space, in general, $T^{\pi_l} \hat{Q}_{\pi_l}$ may be out of that space and must be projected. As the orthogonal projection minimizes the L_2 norm, we seek an approximate value function that is invariant under one application of the Bellman operator T^{π_l} followed by orthogonal projection:

$$\begin{aligned}\hat{Q}_{\pi_l} =& \Phi^T(\Phi^T\Phi)^{-1}\Phi^T(T^{\pi_l}\hat{Q}_{\pi_l}) \\ =& \Phi^T(\Phi^T\Phi)^{-1}\Phi^T(\boldsymbol{R} + \gamma E_{\pi_l, P_T}[\hat{Q}'_{\pi_l}]).\end{aligned}$$

Note that the orthogonal projection to the column space Φ is well defined because the columns of Φ are linearly independent by definition. Manipulating the above equation, we get the solution:

$$\boldsymbol{w}_l = \left(\Phi^T(\Phi - \gamma E_{\pi, P_T}[\Phi'])\right)^{-1} \Phi^T \boldsymbol{R},$$

which is called the *least squares fixed point approximation* to the true value function.

Comparison of Approximation Methods It is easy to see that the Bellman residual minimizing approximation minimizes the L_2 distance between $\hat{Q}_{\pi_l}$ and $T^{\pi_l}\hat{Q}_{\pi_l}$, whereas the least squares fixed point approximation minimizes the orthogonal projection of that distance, i.e., the distance between $\hat{Q}_{\pi_l}$ and the orthogonal projection of $T^{\pi_l}\hat{Q}_{\pi_l}$. Basically, the Bellman residual minimizing approximation finds the point on the basis functions Φ space where the Bellman operator makes the least changes toward Q_{π_l} only in terms of L_2 distance without considering the change of the direction. However, the least squares fixed point approximation ignores the magnitude of the Bellman operator changes and focuses on the direction of the change.

Clearly, the solutions by these two methods are different since their objective functions are different, except in the case that the true value function Q_{π_l} lies in the basis functions Φ space; in that case, both methods are in fact solving the Bellman equation and their solutions are same. If Q_{π_l} does not lie in the Φ space, there is no clear evidence that which method will find a better solution. However, according to the observation in Lagoudakis and Parr (2003), from a practical point of view, the least squares fixed point approximation experimentally delivers policies that are superior to the ones found using the Bellman residual minimizing approximation. Thus, for the rest of reviewing LSPI, we will focus on the least squares fixed point approximation.

Given the samples D, the solution of least squares fixed point approximation $\hat{Q}_{\pi_l}$ to the true state action value function Q_{π_l} the can be estimated by

$$\hat{\boldsymbol{w}}_l = \boldsymbol{A}^{-1}b,$$

where $\boldsymbol{A}$ can be calculated as

$$\boldsymbol{A} = \frac{1}{N}\sum_{i=1}^{N}\left[\phi(\boldsymbol{s}_i, a_i)\left(\phi(\boldsymbol{s}_i, a_i) - \gamma\phi(\boldsymbol{s}'_i, a'_i)\right)^T\right],$$

and

$$b = \frac{1}{N}\sum_{i=1}^{N}\left[\phi(\boldsymbol{s}_i, a_i)r_i\right].$$

Till now, we get the solution of the approximated state action value function, which is called *least squares temporal difference learning for the state action value function* (LSTDQ).

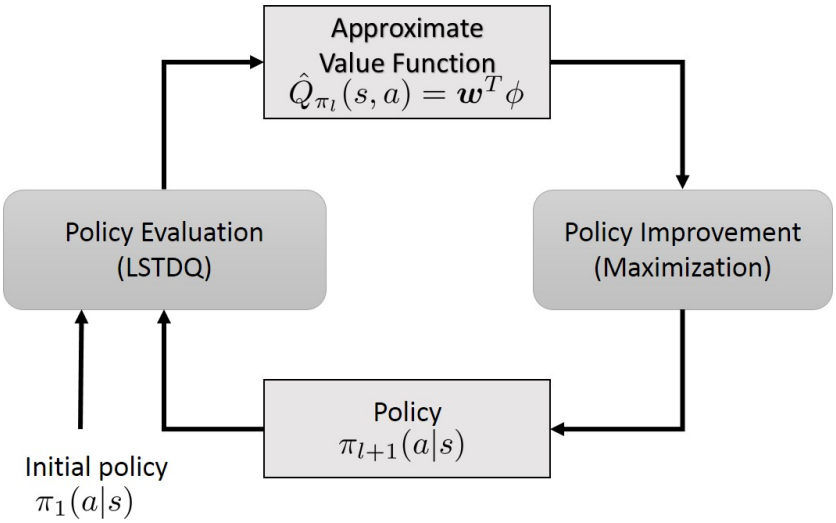


Figure 2.2: Framework of least-squares policy iteration algorithm.

At this point, all ingredients are in place to state the policy evaluation and policy improvement steps of LSPI algorithm. Figure 2.2 shows a block diagram of LSPI that demonstrates how the algorithm fits within the policy iteration framework. The state action value function can be approximated by LSTDQ. The greedy policy π over this approximated value function at any given state $\boldsymbol{s}$ can be obtained through maximization of the approximated values over all actions in $\mathcal{A}$:

$$\pi_{l+1}(\boldsymbol{s}) = \arg\max_a \hat{Q}_{\pi_l}(\boldsymbol{s}, a) = \arg\max_a \boldsymbol{w}_l^T \phi(\boldsymbol{s}, a).$$

So far, the policy is updated using greedy strategy, which is deterministic. However, in practice, stochastic policy is often more useful, because it is required to explore new state action pairs to find better policy in large state action spaces. Thus, the randomness of a resulting policy is taken into account in stochastic probability improvement. Here, we introduce a stochastic policy improvement technique:

$$\pi_{l+1}(a|s) = \frac{\exp(\hat{Q}_{\pi_l}(\boldsymbol{s}, a)/\tau)}{\int \exp(\hat{Q}_{\pi_l}(\boldsymbol{s}, a)/\tau) da},$$

where τ is a positive parameter which determines the randomness of the new policy $\pi_{l+1}(a|s)$. This is called *Gibbs* policy update.

Since policies are learned indirectly via value functions in policy iteration,

improving the quality of value function approximation does not necessarily yield a better policy. Furthermore, because a small change in value functions can cause a big change in policy, it is not safe to use the value function based approach for controlling expensive dynamic systems such as a humanoid robot. Another weakness of the value function approach is that it is difficult to handle continuous actions because a maximizer of the value function with respect to an action needs to be found for policy improvement. One way to address these problems is policy search, which will be reviewed in next subsection.

2.3 Policy Search

Having introduced *policy iteration* in Section 2.2, we now review *policy search*, another major class of reinforcement learning algorithms. These algorithms use optimization techniques to directly search for an optimal policy. Policy search methods use parameterized policies, which are defined as

$$\pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta}),$$

where $\boldsymbol{\theta}$ is the policy parameters. The goal of policy search methods is to find the optimal policy parameters $\boldsymbol{\theta}^*$ that maximizes the expected return $J(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}).$$

The expected return is regarded as the function of policy parameter $\boldsymbol{\theta}$:

$$J(\boldsymbol{\theta}) := \int p(h|\boldsymbol{\theta})R(h)\mathrm{d}h,$$

where the probability density of occurring the trajectory h depends on the policy parameters $\boldsymbol{\theta}$ as:

$$p(h|\boldsymbol{\theta}) = p(\boldsymbol{s}_1)\prod_{t=1}^{T} p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)\pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta}).$$

There are several policy search methods such as the traditional policy gradients method (*REINFORCE*) (Williams, 1992), the natural policy gradients method, policy gradients with parameter-based exploration method, and policy search by expectation-maximization. In this section, we will review these methods.

2.3.1 REINFORCE

REINFORCE is the classical policy search algorithm (Williams, 1992), which directly learns the policy parameters $\boldsymbol{\theta}$ via *gradient ascent*:

$$\boldsymbol{\theta} \longleftarrow \boldsymbol{\theta} + \varepsilon \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where ε is a small positive constant. The gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ is given by

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \int \nabla_{\boldsymbol{\theta}} p(h|\boldsymbol{\theta}) R(h) \mathrm{d}h \\
&= \int p(h|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(h|\boldsymbol{\theta}) R(h) \mathrm{d}h \\
&= \int p(h|\boldsymbol{\theta}) \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta}) R(h) \mathrm{d}h,
\end{aligned}$$

where we used the so-called 'log trick':

$$\nabla_{\boldsymbol{\theta}} p(h|\boldsymbol{\theta}) = p(h|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(h|\boldsymbol{\theta}).$$

Since $p(h|\boldsymbol{\theta})$ is unknown, the expectation is approximated by the empirical average:

$$\nabla_{\boldsymbol{\theta}} \widehat{J}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_t^n|\boldsymbol{s}_t^n, \boldsymbol{\theta}) R(h^n), \tag{2.2}$$

where $h^n := [\boldsymbol{s}_1^n, a_1^n, \ldots, \boldsymbol{s}_T^n, a_T^n]$ is a roll-out sample.

Let us employ the Gaussian policy model with parameter $\boldsymbol{\theta} = (\boldsymbol{\mu}, \sigma)$, where $\boldsymbol{\mu}$ is the mean vector and σ is the standard deviation:

$$\pi(a|\boldsymbol{s}; \boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s}))^2}{2\sigma^2}\right),$$

where $\boldsymbol{\phi}(\boldsymbol{s})$ is an ℓ-dimensional basis function vector.

Then the policy gradients are explicitly given as

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}} \log \pi(a|\boldsymbol{s}, \boldsymbol{\theta}) &= \frac{a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s})}{\sigma^2} \boldsymbol{\phi}(\boldsymbol{s}), \\
\nabla_{\sigma} \log \pi(a|\boldsymbol{s}, \boldsymbol{\theta}) &= \frac{(a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s}))^2 - \sigma^2}{\sigma^3}.
\end{aligned}$$

A drawback of REINFORCE is that the variance of the above policy gradients is large (Peters and Schaal, 2006; Sehnke et al., 2010), which leads to slow convergence.

2.3.2 Natural Policy Gradients

REINFORCE typically uses an Euclidean metric to determine the direction of the parameter update, which basically implying that all parameter dimensions have similarly strong effects on the resulting policy. One of the main reasons for using policy gradients method is that we intend to do just a small change of parameter to the policy while improving the policy. However, small changes in parameter $\boldsymbol{\theta}$ might result in large changes of the policy. Moreover, the gradient estimation is based on the distribution of $\pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta})$ due to sampling, hence, the gradient estimation in the next iteration can also change dramatically. In order to achieve a stable policy update process, it is required that the distribution $\pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta})$ does not change too much in one step. This is the motivation of natural policy gradient method (Kakade, 2002).

To measure the distance between the current policy and the updated policy based upon the distribution of the trajectories, the approximated Kullback-Leibler (KL) divergence is used in the natural gradients method. KL divergence is a similarity measure of two distributions. It has shown that the Fisher Information Matrix $\mathbf{F}_{\boldsymbol{\theta}}$ can be used to approximate the KL divergence between the trajectory distribution based on the current policy $p(h|\boldsymbol{\theta})$ and the trajectory distribution based on the updated policy $p(h|\boldsymbol{\theta}+\triangle\boldsymbol{\theta})$ for sufficiently small parameter changes $\triangle\boldsymbol{\theta}$:

$$\mathrm{KL}(p(h|\boldsymbol{\theta})||p(h|\boldsymbol{\theta}+\triangle\boldsymbol{\theta})) \approx \triangle\boldsymbol{\theta}^T\mathbf{F}_{\boldsymbol{\theta}}\triangle\boldsymbol{\theta},$$

where

$$\mathbf{F}_{\boldsymbol{\theta}} = \int p(h|\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\log p(h|\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\log p(h|\boldsymbol{\theta})^T dh.$$

The natural gradient update $\triangle\boldsymbol{\theta}^{\mathrm{NG}}$ (Amari, 1998) is defined as the update $\triangle\boldsymbol{\theta}$ that is most similar to the traditional gradient update $\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta})$ while the change in the trajectory distribution is bound to a very small number ϵ (i.e., close to zero):

$$\mathrm{KL}(p(h|\boldsymbol{\theta})||p(h|\boldsymbol{\theta}+\triangle\boldsymbol{\theta})) \leq \epsilon.$$

It can be interpreted as follows: determine the maximal improvement of the policy parameter $\triangle\boldsymbol{\theta}$ for a small change of the trajectory distribution $\triangle\boldsymbol{\theta}^T\mathbf{F}_{\boldsymbol{\theta}}\triangle\boldsymbol{\theta}$. Thus, we formulate the natural policy gradients as the following optimization problem:

$$\triangle\boldsymbol{\theta}^{\mathrm{NG}} = \arg\max_{\triangle\boldsymbol{\theta}} \triangle\boldsymbol{\theta}^T\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta}) \quad s.t. \quad \triangle\boldsymbol{\theta}^T\mathbf{F}_{\boldsymbol{\theta}}\triangle\boldsymbol{\theta} \leq \epsilon.$$

The solution to this program is given by

$$\triangle \boldsymbol{\theta}^{\mathrm{NG}} = \mathbf{F}_{\boldsymbol{\theta}}^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ can be estimated by Eq. (2.2), and $\mathbf{F}_{\boldsymbol{\theta}}$ can be approximated as

$$\hat{\mathbf{F}_{\boldsymbol{\theta}}} = \frac{1}{N} \sum_{n=1}^{N} \nabla_{\boldsymbol{\theta}} \log p(h_n|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(h_n|\boldsymbol{\theta})^T,$$

where $h^n := [\boldsymbol{s}_1^n, a_1^n, \ldots, \boldsymbol{s}_T^n, a_T^n]$ is a roll-out sample.

Since the Fisher information matrix is always positive definite, the natural gradients always rotate the traditional gradients less than 90 degree. Therefore, the convergence guarantees from the traditional gradients method.

In contrast to the traditional gradients method, natural policy gradients method avoids premature convergence on plateaus and overaggressive steps on steep ridges. Thus, in practice, a learning process based on natural policy gradients often converges significantly faster for most practical cases (Amari, 1998). The difference between natural policy gradients and traditional policy gradients method is shown in Figure 2.3. In this example, we employ the Gaussian policy model, μ is the mean parameter and σ is the deviation parameter. The task is with linear transition model and quadratic reward function. The contour means the expected return, and the red arrows indicate the resulting gradient. Through the parameter update trajectories shown in Figure 2.3, we can see that the traditional gradients method reduces the variance parameter σ quickly, therefore, will stop exploring. On the other hand, the natural policy gradients method only gradually reduces the variance parameter, in the end, finds the optimal solution faster. From another point of view, the traditional gradients method is inefficient particularly in the large plateaus of the expected return landscape, where the gradients are small and often do not point toward to the optimal solution. However, natural gradients method does not have such problem.

A major issue in natural policy gradients method is that the matrix inversion in the gradient estimations may be numerically brittle and may scale worse (Deisenroth et al., 2013).

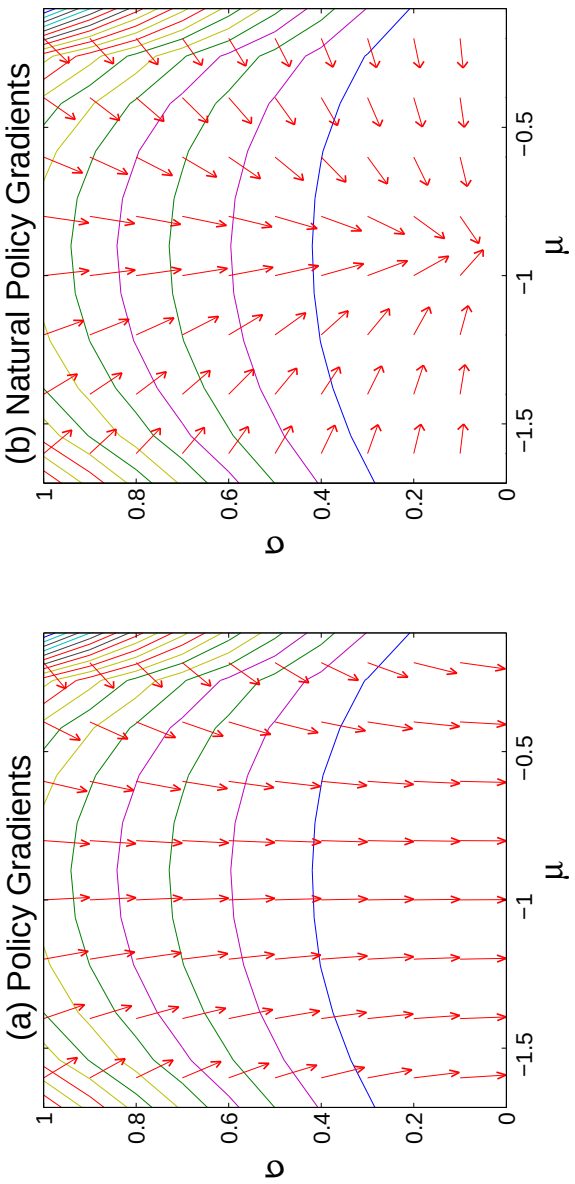


Figure 2.3: Comparison of the natural gradients method and the traditional gradients method on policy parameters update trajectories.

2.3.3 Policy Gradients with Parameter-based Exploration

One of the reasons for large variance of policy gradients in the REINFORCE algorithm is that the empirical average is taken at each time step, which is caused by stochasticity of policies.

In order to mitigate this problem, another method called *policy gradients with parameter-based exploration* (PGPE) was proposed recently (Sehnke et al., 2010).

In PGPE, a linear *deterministic* policy is adopted:

$$\pi(a|\boldsymbol{s},\boldsymbol{\theta}) = \delta(a = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{s})), \tag{2.3}$$

where $\delta(\cdot)$ is the *Dirac delta function*, $\boldsymbol{\phi}(\boldsymbol{s})$ is an ℓ-dimensional basis function vector, and $^\top$ denotes the transpose. The stochasticity is introduced by considering $p(\boldsymbol{\theta}|\boldsymbol{\rho})$, a prior distribution over policy parameter $\boldsymbol{\theta}$ with hyper-parameter $\boldsymbol{\rho}$. Since entire history h is solely determined by a single sample of parameter $\boldsymbol{\theta}$ in this formulation, it is expected that the variance of gradient estimates can be reduced.

The expected return for hyper-parameter $\boldsymbol{\rho}$ is expressed as

$$J(\boldsymbol{\rho}) = \iint p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho})R(h)\mathrm{d}h\mathrm{d}\boldsymbol{\theta}.$$

Differentiating this with respect to $\boldsymbol{\rho}$, we have

$$\begin{aligned}\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) &= \iint p(h|\boldsymbol{\theta})\nabla_{\boldsymbol{\rho}} p(\boldsymbol{\theta}|\boldsymbol{\rho})R(h)\mathrm{d}h\mathrm{d}\boldsymbol{\theta} \\ &= \iint p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho})\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})R(h)\mathrm{d}h\mathrm{d}\boldsymbol{\theta},\end{aligned}$$

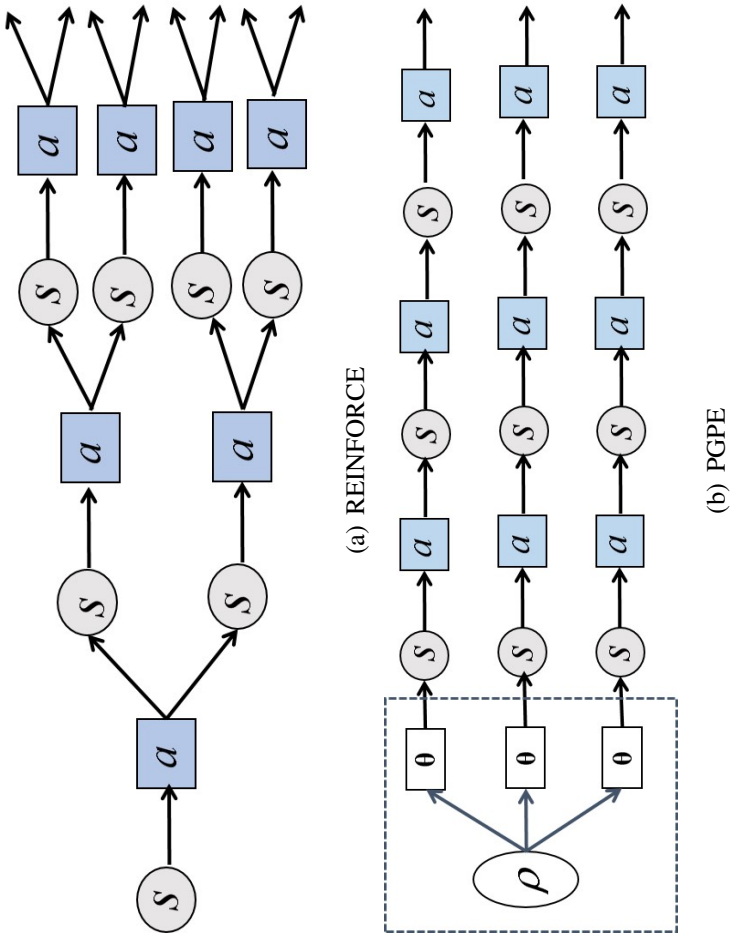
where the log trick for $\nabla_{\boldsymbol{\rho}} p(\boldsymbol{\theta}|\boldsymbol{\rho})$ is used. We then approximate the expectation over h and $\boldsymbol{\theta}$ by the empirical average:

$$\nabla_{\boldsymbol{\rho}} \widehat{J}(\boldsymbol{\rho}) = \frac{1}{N}\sum_{n=1}^{N} \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}^n|\boldsymbol{\rho})R(h^n), \tag{2.4}$$

where each trajectory sample h^n is drawn from $p(h|\boldsymbol{\theta}^n)$ and parameter $\boldsymbol{\theta}^n$ is drawn from $p(\boldsymbol{\theta}^n|\boldsymbol{\rho})$.

Let us employ the Gaussian prior distribution with hyper-parameter $\boldsymbol{\rho} = (\boldsymbol{\eta}, \boldsymbol{\tau})$ to draw parameter vector $\boldsymbol{\theta}$, where $\boldsymbol{\eta}$ is the mean vector and $\boldsymbol{\tau}$ is the vector consisting of the standard deviation in each element:

$$p(\theta_i|\boldsymbol{\rho}_i) = \frac{1}{\tau_i\sqrt{2\pi}} \exp\left(-\frac{(\theta_i - \eta_i)^2}{2\tau_i^2}\right).$$



(a) REINFORCE (b) PGPE

Figure 2.4: Illustrations of exploration strategies in REINFORCE and PGPE. Note that the transition model in this graph is deterministic for illustrative purpose.

Then the derivative of $\log p(\boldsymbol{\theta}|\boldsymbol{\rho})$ with respect to η_i and τ_i are given as follows:

$$\nabla_{\eta_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) = \frac{\theta_i - \eta_i}{\tau_i^2},$$
$$\nabla_{\tau_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) = \frac{(\theta_i - \eta_i)^2 - \tau_i^2}{\tau_i^3}.$$

To aid in understanding PGPE, we give the illustrative graphs of REINFORCE and PGPE in Figure 2.4. Exploration in REINFORCE is by adding perturbations directly to the actions at each time step, which is caused by the stochastic policy. In this way, the randomness increases with the number of time steps, thus, leads to a large variance of the gradient estimates. On the other hand, exploration in PGPE is by drawing the policy parameter $\boldsymbol{\theta}$ from a prior distribution $p(\boldsymbol{\theta}|\boldsymbol{\rho})$ in the beginning, and thereafter the trajectory is deterministic. Fixing the exploration in the beginning can reduce the variance of the gradient estimates, hence, produce more reliable policy updates.

2.3.4 Expectation-Maximization based Policy Search

Gradient based policy update methods require us to specify the learning rate, setting the learning rate can be problematic and often results in an unstable learning process or slow convergence (Kober and Peters, 2011). This problem can be solved by using *Expectation Maximization* (EM) algorithm to update the policy parameters (Dayan and Hinton, 1997). The EM algorithm is an iterative procedure for estimating the maximum likelihood solution of latent variable models, where the parameter updates of every iteration can be obtained in closed form.

EM-based policy search method with Gaussian policy models is called reward weighted regression (RWR) (Peters and Schaal, 2007), where the basic idea is to iteratively update the policy parameters $\boldsymbol{\theta}$ by maximizing a lower bound of expected return. Let us review this algorithm in this subsection.

Let $\boldsymbol{\theta}_l$ be the current policy parameters, where l indicates the iteration number. We first show the lower bound of the logarithmic expected return $\log J(\boldsymbol{\theta})$:

$$\log J(\boldsymbol{\theta}) \geq \int \frac{R(h)p(h|\boldsymbol{\theta}_l)}{J(\boldsymbol{\theta}_l)} \log \frac{p(h|\boldsymbol{\theta})}{p(h|\boldsymbol{\theta}_l)} dh + \log J(\boldsymbol{\theta}_l) := Q_l(\boldsymbol{\theta}).$$

The EM algorithm iteratively updates the parameters $\boldsymbol{\theta}$ by maximizing the lower bound $Q_l(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}_{l+1} := \arg\max_{\boldsymbol{\theta}} Q_l(\boldsymbol{\theta}).$$

Since $\log J(\boldsymbol{\theta}_l) = Q_l(\boldsymbol{\theta}_l)$, the lower bound $Q_l(\boldsymbol{\theta})$ is tight at $\boldsymbol{\theta}_l$. The updated parameter is achieved by maximizing the lower bound, which guarantees that the expected return is monotone increased along the parameter updated direction:

$$J(\boldsymbol{\theta}_{l+1}) \geq J(\boldsymbol{\theta}_l).$$

Therefore, the EM algorithm is guaranteed to converge to a local maximum of expected return $\log J(\boldsymbol{\theta})$.

Let us employ the Gaussian policy model with parameter $\boldsymbol{\theta} = (\boldsymbol{\mu}, \sigma)$, where $\boldsymbol{\mu}$ is the mean vector and σ is the standard deviation:

$$\pi(a|\boldsymbol{s};\boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s}))^2}{2\sigma^2}\right),$$

where $\boldsymbol{\phi}(\boldsymbol{s})$ is an ℓ-dimensional basis function vector.

A useful property of Gaussian policy model is that the log-derivative with respect the policy parameter can be analytically calculated as

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}} \log \pi(a|\boldsymbol{s},\boldsymbol{\theta}) &= \frac{a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s})}{\sigma^2} \boldsymbol{\phi}(\boldsymbol{s}), \\
\nabla_{\sigma} \log \pi(a|\boldsymbol{s},\boldsymbol{\theta}) &= \frac{(a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s}))^2 - \sigma^2}{\sigma^3}.
\end{aligned}$$

By differentiating $Q_l(\boldsymbol{\theta})$ with respect to the policy parameter $\boldsymbol{\theta}$ and setting it to 0:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} Q_l(\boldsymbol{\theta}) =& \nabla_{\boldsymbol{\theta}} \int \frac{R(h)p(h|\boldsymbol{\theta}_l)}{J(\boldsymbol{\theta}_l)} \log \frac{p(h|\boldsymbol{\theta})}{p(h|\boldsymbol{\theta}_l)} dh + \log J(\boldsymbol{\theta}_l) \\
=& \int \frac{R(h)p(h|\boldsymbol{\theta}_l)}{J(\boldsymbol{\theta}_l)} \nabla_{\boldsymbol{\theta}} \log p(h|\boldsymbol{\theta}) dh \\
=& \int \frac{R(h)p(h|\boldsymbol{\theta}_l)}{J(\boldsymbol{\theta}_l)} \nabla_{\boldsymbol{\theta}} \sum_{t=1}^{T} \log \pi(a_t|\boldsymbol{s}_t,\boldsymbol{\theta}) dh \\
=& 0.
\end{aligned}$$

The updated parameter $\boldsymbol{\theta}_{l+1} = (\boldsymbol{\mu}_{l+1}, \sigma_{l+1})^T$ can be analytically obtained as

$$\boldsymbol{\mu}_{l+1} = \left(\int R(h) p(h|\boldsymbol{\theta}_l) dh \sum_{t=1}^{T} \boldsymbol{\phi}(\boldsymbol{s}_t) \boldsymbol{\phi}(\boldsymbol{s}_t)^T \right)^{-1} \left(\int R(h) p(h|\boldsymbol{\theta}_l) dh \sum_{t=1}^{T} a_t \boldsymbol{\phi}(\boldsymbol{s}_t) \right),$$
$$\sigma_{l+1}^2 = \left(\int R(h) p(h|\boldsymbol{\theta}_l) dh \right)^{-1} \left(\int R(h) p(h|\boldsymbol{\theta}_l) dh \sum_{t=1}^{T} (a_t - \boldsymbol{\mu}_{l+1}^T \boldsymbol{\phi}(\boldsymbol{s}_t))^2 \right).$$

Since $p(h|\boldsymbol{\theta})$ is unknown, the updated parameters $\boldsymbol{\theta}_{l+1} = (\boldsymbol{\mu}_{l+1}, \sigma_{l+1})^T$ are approximated by the empirical average:

$$\hat{\boldsymbol{\mu}}_{l+1} = \left(\frac{1}{N} \sum_{n=1}^{N} R(h_n) \sum_{t=1}^{T} \boldsymbol{\phi}(\boldsymbol{s}_t^n) \boldsymbol{\phi}(\boldsymbol{s}_t^n)^T \right)^{-1} \left(\frac{1}{N} \sum_{n=1}^{N} R(h_n) \sum_{t=1}^{T} a_t^n \boldsymbol{\phi}(\boldsymbol{s}_t^n) \right),$$
$$\hat{\sigma}_{l+1}^2 = \left(\frac{1}{N} \sum_{n=1}^{N} R(h_n) \right)^{-1} \left(\frac{1}{N} \sum_{n=1}^{N} R(h_n) \sum_{t=1}^{T} (a_t^n - \boldsymbol{\mu}_{l+1}^T \boldsymbol{\phi}(\boldsymbol{s}_t^n))^2 \right).$$

where $h^n := [\boldsymbol{s}_1^n, a_1^n, \ldots, \boldsymbol{s}_T^n, a_T^n]$ is a roll-out sample.

Chapter 3

Analysis and Improvement of Policy Gradient Estimation

Policy gradient is a useful model-free reinforcement learning approach, but it tends to suffer from instability of gradient estimates. In this chapter, we analyze and improve the stability of policy gradient methods. We first prove that the variance of gradient estimates in the *PGPE* (policy gradients with parameter-based exploration) method is smaller than that of the classical REINFORCE method under a mild assumption. We then derive the optimal baseline for PGPE, which contributes to further reducing the variance. We also theoretically show that PGPE with the optimal baseline is more preferable than REINFORCE with the optimal baseline in terms of the variance of gradient estimates. Finally, we demonstrate the usefulness of the improved PGPE method through experiments.

3.1 Introduction

The goal of *reinforcement learning* (RL) is to find an optimal decision-making policy that maximizes the *return* (i.e., the sum of discounted rewards) through interaction with an unknown environment (Sutton and Barto, 1998). *Model-free* RL is a flexible framework in which decision-making policies are directly learned without going through explicit modeling of the environment. *Policy iteration* and

policy search are two popular formulations of model-free RL[1].

In the policy iteration approach (Kaelbling et al., 1996), the *value function* is first estimated and then policies are determined based on the learned value function. Policy iteration was demonstrated to work well in many real-world applications, especially in problems with discrete states and actions (Tesauro, 1994; Williams and Young, 2007; Abe et al., 2010). Although policy iteration can naturally deal with continuous states by function approximation (Lagoudakis and Parr, 2003), continuous actions are hard to handle due to the difficulty of finding maximizers of value functions with respect to actions. Moreover, since policies are indirectly determined via value function approximation, misspecification of value function models can lead to inappropriate policies even in very simple problems (Weaver and Baxter, 1999; Baxter et al., 2001). Another limitation of policy iteration especially in physical control tasks is that control policies can vary drastically in each iteration. This causes severe instability in the physical system and thus is not favorable in practice.

Policy search is another approach to model-free RL that can overcome the limitations of policy iteration (Williams, 1992; Dayan and Hinton, 1997; Kakade, 2002). In the policy search approach, control policies are directly learned so that the return is maximized, for example, via a gradient method (called the *REINFORCE* method) (Williams, 1992), an EM algorithm (Dayan and Hinton, 1997), and a natural gradient method (Kakade, 2002). Among them, the gradient-based method is particularly useful in physical control tasks since policies are changed gradually. This ensures the stability of the physical system.

However, since the REINFORCE method tends to have a large variance in the estimation of the gradient directions, its naive implementation converges slowly (Marbach and Tsitsiklis, 2004; Peters and Schaal, 2006; Sehnke et al., 2010). Subtraction of the *optimal baseline* (Weaver and Tao, 2001; Greensmith et al., 2004) can ease this problem to some extent, but the variance of gradient estimates is still large. Furthermore, the performance heavily depends on the choice of an initial policy, and appropriate initialization is not straightforward in practice.

To cope with this problem, a novel policy gradient method called *policy gra-*

[1]Policy iteration is originally a model-based RL approach, but it can be driven in a model-free mode by implicitly approximating an environment model with samples.

dients with parameter-based exploration (PGPE) was proposed recently (Sehnke et al., 2010). In PGPE, an initial policy is drawn from a prior probability distribution, and then actions are chosen deterministically. This construction contributes to mitigating the problem of initial policy choice and stabilizing gradient estimates (Rückstiess et al., 2010). Moreover, by subtracting a moving-average baseline, the variance of gradient estimates can be further reduced. Through robot-control experiments, PGPE was demonstrated to achieve more stable performance than existing policy-gradient methods.

The goal of this chapter is to theoretically support the usefulness of PGPE, and to further improve its performance. More specifically, we first give bounds of the gradient estimates of the REINFORCE and PGPE methods. Our theoretical analysis shows that gradient estimates for PGPE have smaller variance than those for REINFORCE under a mild condition. We then show that the moving-average baseline for PGPE adopted in the original paper (Sehnke et al., 2010) has excess variance; we give the optimal baseline for PGPE that minimizes the variance, following the line of Weaver and Tao (2001); Greensmith et al. (2004). We further theoretically show that PGPE with the optimal baseline is more preferable than REINFORCE with the optimal baseline in terms of the variance of gradient estimates. Finally, the usefulness of the improved PGPE method is demonstrated through experiments.

3.2 Variance of Gradient Estimates

In this section, we theoretically investigate the variance of gradient estimates in REINFORCE and PGPE.

For multi-dimensional state space, we consider the *trace* of the covariance matrix of gradient vectors. That is, for a random vector $\boldsymbol{A} = (A_1, \ldots, A_\ell)^\top$, we define

$$\begin{aligned}\mathbf{Var}(\boldsymbol{A}) &= \mathrm{tr}\left(\mathbb{E}\big[(\boldsymbol{A} - \mathbb{E}[\boldsymbol{A}])(\boldsymbol{A} - \mathbb{E}[\boldsymbol{A}])^\top\big]\right) \\ &= \sum_{m=1}^{\ell} \mathbb{E}\Big[(A_m - \mathbb{E}[A_m])^2\Big], \end{aligned} \tag{3.1}$$

where $\mathbb{E}$ denotes the expectation. Let

$$B = \sum_{i=1}^{\ell} \tau_i^{-2},$$

where ℓ is the dimensionality of state $\boldsymbol{s}$.

Below, we consider a subset of the following assumptions:

Assumption (A): $r(\boldsymbol{s}, a, \boldsymbol{s}') \in [-\beta, \beta]$ for $\beta > 0$.

Assumption (B): $r(\boldsymbol{s}, a, \boldsymbol{s}') \in [\alpha, \beta]$ for $0 < \alpha < \beta$.

Assumption (C): For $\delta > 0$, there exist two series $\{c_t\}_{t=1}^T$ and $\{d_t\}_{t=1}^T$ such that

$$\|\boldsymbol{s}_t\|_2 \geq c_t \quad \text{and} \quad \|\boldsymbol{s}_t\|_2 \leq d_t$$

hold with probability at least $(1-\delta)^{1/2N}$ respectively over the choice of sample paths, where $\|\cdot\|_2$ denotes the ℓ_2-norm.

Note that Assumption (B) is stronger than Assumption (A). Let

$$\mathcal{L}(T) = C_T \alpha^2 - D_T \beta^2 / (2\pi),$$

where

$$C_T = \sum_{t=1}^{T} c_t^2 \quad \text{and} \quad D_T = \sum_{t=1}^{T} d_t^2.$$

First, we analyze the variance of gradient estimates in PGPE:

Theorem 3.1. *Under Assumption (A), we have the following upper bounds:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})\right] \leq \frac{\beta^2 (1-\gamma^T)^2 B}{N(1-\gamma)^2},$$
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{J}(\boldsymbol{\rho})\right] \leq \frac{2\beta^2 (1-\gamma^T)^2 B}{N(1-\gamma)^2}.$$

This theorem means that the upper bound of the variance of $\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})$ is proportional to β^2 (the upper bound of squared rewards), B (the trace of the inverse Gaussian covariance), and $(1-\gamma^T)^2/(1-\gamma)^2$, and is inverse-proportional to sample size N. The upper bound of the variance of $\nabla_{\boldsymbol{\tau}} \widehat{J}(\boldsymbol{\rho})$ is twice larger than that of $\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})$. When T goes to infinity, $(1-\gamma^T)^2$ will converge to 1.

Next, we analyze the variance of gradient estimates in REINFORCE:

Theorem 3.2. *Under Assumptions (B) and (C), we have the following lower bound with probability at least* $1-\delta$*:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})\right] \geq \frac{(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}\mathcal{L}(T).$$

Under Assumptions (A) and (C), we have the following upper bound with probability at least $(1-\delta)^{1/2}$*:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})\right] \leq \frac{D_T\beta^2(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}.$$

Under Assumption (A), we have

$$\mathbf{Var}\left[\nabla_{\sigma}\widehat{J}(\boldsymbol{\theta})\right] \leq \frac{2T\beta^2(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}.$$

The upper bounds for REINFORCE are similar to those for PGPE, but they are monotone increasing with respect to trajectory length T. The lower bound for the variance of $\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})$ will be non-trivial if it is positive, i.e., $\mathcal{L}(T) > 0$. This can be fulfilled, e.g., if α and β satisfy

$$2\pi C_T\alpha^2 > D_T\beta^2.$$

Deriving a lower bound of the variance of $\nabla_{\sigma}\widehat{J}(\boldsymbol{\theta})$ is left open as future work.

Finally, we compare the variance of gradient estimates in REINFORCE and PGPE:

Theorem 3.3. *In addition to Assumptions (B) and (C), we assume* $\mathcal{L}(T)$ *is positive and monotone increasing with respect to* T*. If there exists* T_0 *such that* $\mathcal{L}(T_0) \geq \beta^2 B\sigma^2$*, then we have*

$$\mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})] > \mathbf{Var}[\nabla_{\boldsymbol{\eta}}\widehat{J}(\boldsymbol{\rho})]$$

for all $T > T_0$*, with probability at least* $1-\delta$*.*

The above theorem means that PGPE is more favorable than REINFORCE in terms of the variance of gradient estimates of the mean, if trajectory length T is large. This theoretical result would partially support the experimental success of the PGPE method (Sehnke et al., 2010).

3.3 Variance Reduction by Subtracting Baseline

In this section, we give a method to reduce the variance of gradient estimates in PGPE and analyze its theoretical properties.

3.3.1 Basic Idea of Introducing Baseline

It is known that the variance of gradient estimates can be reduced by subtracting a *baseline* b: for REINFORCE and PGPE, modified gradient estimates are given by

$$\nabla_{\boldsymbol{\theta}} \widehat{J}^b(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} (R(h^n) - b) \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_t^n | \boldsymbol{s}_t^n, \boldsymbol{\theta}),$$

$$\nabla_{\boldsymbol{\rho}} \widehat{J}^b(\boldsymbol{\rho}) = \frac{1}{N} \sum_{n=1}^{N} (R(h^n) - b) \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}^n | \boldsymbol{\rho}).$$

The *adaptive reinforcement baseline* (Williams, 1992) was derived as the exponential moving average of the past experience:

$$b(n) = \xi R(h^{n-1}) + (1 - \xi) b(n-1),$$

where $0 < \xi \leq 1$. Based on this, an empirical gradient estimate with the moving-average baseline was proposed for REINFORCE (Williams, 1992) and PGPE (Sehnke et al., 2010).

The above moving-average baseline contributes to reducing the variance of gradient estimates. However, it was shown (Greensmith et al., 2004; Weaver and Tao, 2001) that the moving-average baseline is not optimal; the optimal baseline is, by definition, given as the minimizer of the variance of gradient estimates with respect to a baseline. Following this formulation, the optimal baseline for REINFORCE is given as follows (Peters and Schaal, 2006):

$$\begin{aligned} b^*_{\mathrm{REINFORCE}} &:= \arg\min_b \mathbf{Var}[\nabla_{\boldsymbol{\theta}} \widehat{J}^b(\boldsymbol{\theta})] \\ &= \frac{\mathbb{E}[R(h) \| \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_t | \boldsymbol{s}_t, \boldsymbol{\theta}) \|^2]}{\mathbb{E}[\| \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_t | \boldsymbol{s}_t, \boldsymbol{\theta}) \|^2]}. \end{aligned}$$

However, only the moving-average baseline was introduced to PGPE so far (Sehnke et al., 2010), which is suboptimal. Below, we derive the optimal baseline for PGPE, and study its theoretical properties.

3.3.2 Optimal Baseline for PGPE

Let b^*_{PGPE} be the optimal baseline for PGPE that minimizes the variance:

$$b^*_{\mathrm{PGPE}} := \arg\min_b \mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{J}^b(\boldsymbol{\rho})].$$

Then the following theorem gives the optimal baseline for PGPE:

Theorem 3.4. *The optimal baseline for PGPE is given by*

$$b^*_{\mathrm{PGPE}} = \frac{\mathbb{E}[R(h)\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]}{\mathbb{E}[\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]},$$

and the excess variance for a baseline b is given by

$$\mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{J}^b(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] = \frac{(b - b^*_{\mathrm{PGPE}})^2}{N} \mathbb{E}[\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2].$$

The above theorem gives an analytic-form expression of the optimal baseline for PGPE. When expected return $R(h)$ and the squared norm of characteristic eligibility $\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2$ are independent of each other, the optimal baseline is reduced to average expected return $\mathbb{E}[R(h)]$. However, the optimal baseline is generally different from the average expected return. The above theorem also shows that the excess variance is proportional to the squared difference of baselines $(b - b^*_{\mathrm{PGPE}})^2$ and the expected squared norm of characteristic eligibility $\mathbb{E}[\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]$, and is inverse-proportional to sample size N.

Next, we analyze the contribution of the optimal baseline to the variance with respect to mean parameter $\boldsymbol{\eta}$ in PGPE:

Theorem 3.5. *If $r(\boldsymbol{s}, a, \boldsymbol{s}') \geq \alpha > 0$, we have the following lower bound:*

$$\mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \geq \frac{\alpha^2(1-\gamma^T)^2 B}{N(1-\gamma)^2}.$$

Under Assumption (A), we have the following upper bound:

$$\mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \leq \frac{\beta^2(1-\gamma^T)^2 B}{N(1-\gamma)^2}.$$

This theorem shows that the lower and upper bounds of the excess variance are proportional to α^2 and β^2 (the bounds of squared immediate rewards), B (the trace of the inverse Gaussian covariance), and $(1-\gamma^T)^2/(1-\gamma)^2$, and are inverse-proportional to sample size N. When T goes to infinity, $(1-\gamma^T)^2$ will converge to 1.

3.3.3 Comparison with REINFORCE

Next, we analyze the contribution of the optimal baseline for REINFORCE, and compare it with that for PGPE. It was shown (Greensmith et al., 2004; Weaver and Tao, 2001) that the excess variance for a baseline b in REINFORCE is given by

$$\begin{aligned}&\mathbf{Var}[\nabla_{\boldsymbol{\theta}}\widehat{J}^{b}(\boldsymbol{\theta})] - \mathbf{Var}[\nabla_{\boldsymbol{\theta}}\widehat{J}^{b^*_{\mathrm{REINFORCE}}}(\boldsymbol{\theta})]\\ &= \frac{(b - b^*_{\mathrm{REINFORCE}})^2}{N}\mathbb{E}\left[\left\|\sum_{t=1}^{T}\nabla_{\boldsymbol{\theta}}\log\pi(a_t|\boldsymbol{s}_t,\boldsymbol{\theta})\right\|^2\right].\end{aligned}$$

Based on this, we have the following theorem:

Theorem 3.6. *Under Assumptions (B) and (C), we have the following bounds with probability at least* $1-\delta$*:*

$$\frac{C_T\alpha^2(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2} \leq \mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})] - \mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}^{b^*_{\mathrm{REINFORCE}}}(\boldsymbol{\theta})] \leq \frac{\beta^2(1-\gamma^T)^2D_T}{N\sigma^2(1-\gamma)^2}.$$

The above theorem shows that the lower and upper bounds of the excess variance are monotone increasing with respect to trajectory length T.

In the aspect of the amount of reduction in the variance of gradient estimates, Theorem 3.5 and Theorem 3.6 show that the optimal baseline for REINFORCE contributes more than that for PGPE.

Finally, based on Theorem 3.1 and Theorem 3.5 and based on Theorem 3.2 and Theorem 3.6, we have the following theorem:

Theorem 3.7. *Under Assumptions (B) and (C), we have*

$$\begin{aligned}\mathbf{Var}[\nabla_{\boldsymbol{\eta}}\widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] &\leq \frac{(1-\gamma^T)^2}{N(1-\gamma)^2}(\beta^2-\alpha^2)B,\\ \mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}^{b^*_{\mathrm{REINFORCE}}}(\boldsymbol{\theta})] &\leq \frac{(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}(\beta^2D_T-\alpha^2C_T),\end{aligned}$$

where the latter inequality holds with probability at least $1-\delta$*.*

This theorem shows that the upper bound of the variance of gradient estimates for REINFORCE with the optimal baseline is still monotone increasing with respect to trajectory length T. On the other hand, since $(1-\gamma^T)^2 \leq 1$, the above

upper bound of the variance of gradient estimates in PGPE with the optimal baseline can be further upper-bounded as

$$\mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \leq \frac{(\beta^2 - \alpha^2)B}{N(1-\gamma)^2},$$

which is independent of T. Thus, when trajectory length T is large, the variance of gradient estimates in REINFORCE with the optimal baseline may be significantly larger than the variance of gradient estimates in PGPE with the optimal baseline.

3.4 Experiments

In this section, we experimentally investigate the usefulness of the proposed method, PGPE with the optimal baseline.

3.4.1 Illustration

Let the state space $\mathcal{S}$ be one-dimensional and continuous, and the initial state is randomly chosen from the standard normal distribution. The action space $\mathcal{A}$ is also set to be one-dimensional and continuous. The transition dynamics of the environment is set at

$$s_{t+1} = s_t + a_t + \varepsilon,$$

where $\varepsilon \sim \mathcal{N}(0, 0.5^2)$ is stochastic noise and $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2. The immediate reward is defined as

$$r = \exp\left(-s^2/2 - a^2/2\right) + 1,$$

which is bounded as $1 < r \leq 2$.

Variance and Bias

First, we illustrate the variance of gradient estimates of the following methods:

- **REINFORCE:** REINFORCE without any baselines.
- **REINFORCE-OB:** REINFORCE with the optimal baseline.

- **PGPE:** PGPE without any baselines.
- **PGPE-MB:** PGPE with the moving-average baseline.
- **PGPE-OB:** PGPE with the optimal baseline.

For fair comparison, all of these methods use the same parameter setup: the mean and standard deviation of the Gaussian distribution is set at $\mu = -1.5$ and $\sigma = 1$, and the length of the trajectory is set at $T = 10$ or 50. The discount factor is set at $\gamma = 0.9$, and the number of episodic samples is set at $N = 100$.

Table 3.1 summarizes the variance of gradient estimates over 100 runs, showing that the variance of REINFORCE is overall larger than PGPE. A notable difference between REINFORCE and PGPE is that the variance of REINFORCE significantly grows as T increases, whereas that of PGPE is not influenced that much by T. This well agrees with our theoretical analysis in Section 3.2. The results also show that the variance of PGPE-OB is much smaller than that of PGPE-MB. REINFORCE-OB contributes highly to reducing the variance especially when T is large, which also well agrees with our theory. However, PGPE-OB still provides much smaller variance than REINFORCE-OB.

We also investigate the bias of gradient estimates of each method. Here, we regard gradients estimated with $N = 1000$ as true gradients, and compute the bias of gradient estimates. The results are also included in Table 3.1, showing that introduction of baselines does not increase the bias; rather, it tends to reduce the bias.

Figure 3.1 shows the variance of gradient estimates with respect to the mean parameter as functions of discount factor γ, in $\log_{10}$-scale. The graphs show that, as discount factor γ gets close to 1, the variance increases. This well agrees with our theoretical analysis in Section 3.2. Among the compared methods, PGPE-OB has the smallest variance overall.

Symmetric Sampling for PGPE

In order to improve the convergence property of the PGPE method, a heuristic of using a pair of symmetric samples called the *symmetric sampling method* was introduced (Sehnke et al., 2010). Here, we numerically investigate its effect on the variance of gradient estimates.

Table 3.1: Variance and bias of estimated gradients for toy data.

Method	$T = 10$				$T = 50$			
	Variance		Bias		Variance		Bias	
	μ, η	σ, τ	μ, η	σ, τ	μ, η	σ, τ	μ, η	σ, τ
REINFORCE	13.2570	26.9173	-0.3102	-1.5098	188.3860	278.3095	-1.8126	-5.1747
REINFORCE-OB	0.0914	0.1203	0.0672	0.1286	0.5454	0.8996	-0.2988	-0.2008
PGPE	0.9707	1.6855	-0.0691	0.1319	1.6572	3.3720	-0.1048	-0.3293
PGPE-MB	0.2127	0.3238	0.0828	-0.1295	0.4123	0.8332	0.0925	-0.2556
PGPE-OB	0.0372	0.0685	-0.0164	0.0512	0.0850	0.1815	0.0480	-0.0779
PGPE-MB-SyS	0.1070	0.8087	0.0850	0.2625	0.2717	1.7883	0.1022	0.1124
PGPE-OB-SyS	0.0908	0.1084	-0.0854	0.0640	0.2865	0.3009	0.0460	0.1602

In the symmetric sampling method, perturbation sample $\boldsymbol{\epsilon}_n$ is drawn from distribution $\mathcal{N}(0, \boldsymbol{\tau}^2)$, and then symmetric parameter samples are created as $\boldsymbol{\theta}_n^+ = \boldsymbol{\eta} + \boldsymbol{\epsilon}_n$ and $\boldsymbol{\theta}_n^- = \boldsymbol{\eta} - \boldsymbol{\epsilon}_n$. Let R_n^+ and R_n^- be returns obtained by $\boldsymbol{\theta}_n^+$ and $\boldsymbol{\theta}_n^-$, respectively. Based on these two returns, gradients with respect to $\boldsymbol{\eta}$ are calculated using the difference between the two returns as

$$\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho}) \approx \frac{1}{N} \sum_{n=1}^{N} \frac{\boldsymbol{\epsilon}_n (R_n^+ - R_n^-)}{2\boldsymbol{\eta}^2}.$$

On the other hand, gradients with respect to $\boldsymbol{\tau}$ can not be directly computed from symmetric parameter samples since $\boldsymbol{\theta}^+$ and $\boldsymbol{\theta}^-$ are equally probable under given $\boldsymbol{\tau}$. To cope with this problem, the difference of the mean of the two returns and the baseline is used as

$$\nabla_{\boldsymbol{\tau}} \widehat{J}(\boldsymbol{\rho}) \approx \frac{1}{N} \sum_{n=1}^{N} \nabla_{\boldsymbol{\tau}} \log p(\boldsymbol{\theta}^n | \boldsymbol{\rho}) \left(\frac{R_n^+ + R_n^-}{2} - b \right).$$

Note that, since the symmetric sampling method produces two parameters $\boldsymbol{\theta}^+$ and $\boldsymbol{\theta}^-$, it requires two trajectory samples in every update.

We numerically compare the variance of gradient estimates of the following methods:

- **PGPE-MB-SyS:** PGPE-MB with symmetric sampling.
- **PGPE-OB-SyS:** PGPE-OB with symmetric sampling.

In the previous experiments, the number of episodic samples for non-symmetric sampling methods was set at $N = 100$. If the number of sampled parameters is the same, the symmetric sampling methods will require twice as many trajectory samples (i.e., $N = 200$) as non-symmetric sampling counterparts since the symmetric sampling methods produce two parameters θ^+ and θ^-. For fair comparison, we only use the half number of sampled parameters for the symmetric sampling methods, which requires $N = 100$ trajectory samples.

The bottom half of Table 3.1 shows the numerical results. In terms of the variance of gradient estimates with respect to mean parameter η, PGPE-MB-SyS has smaller variance than PGPE-MB. Thus, symmetric sampling contributes to reducing the variance for the PGPE-MB method, which agrees with the experimental

results reported in Sehnke et al. (2010). However, PGPE-OB (without symmetric sampling) has smaller variance than PGPE-OB-SyS, indicating that symmetric sampling increases the variance for the PGPE-OB method. As for the variance of gradient estimates with respect to deviation parameter τ, symmetric sampling tends to increase the variance both for the PGPE-MB and PGPE-OB methods.

Variance and Policy Parameter Change through Entire Policy-Update Process

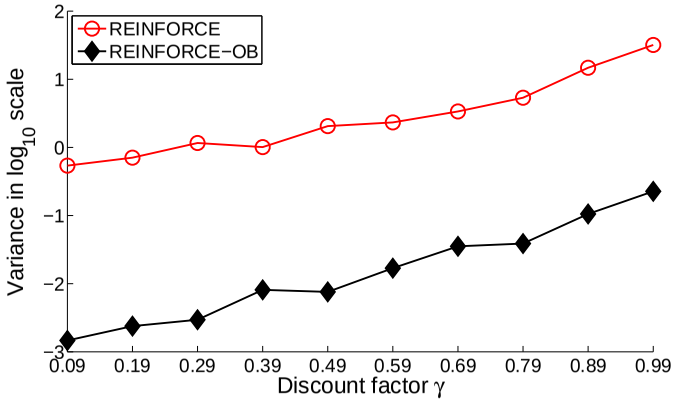
Next, we investigate the variance of gradient estimates when policy parameters are updated over iterations[2].

In this experiment, we set $T = 20$, and the variance is computed from 50 runs. We set $N = 10$ for all the methods, and policies are updated over 50 iterations. In order to evaluate the variance in a stable manner, we repeat the above experiments 20 times with random choice of initial mean parameter μ from $[-3.0, -0.1]$, and investigate the average variance of gradient estimates with respect to mean parameter μ over 20 trials.

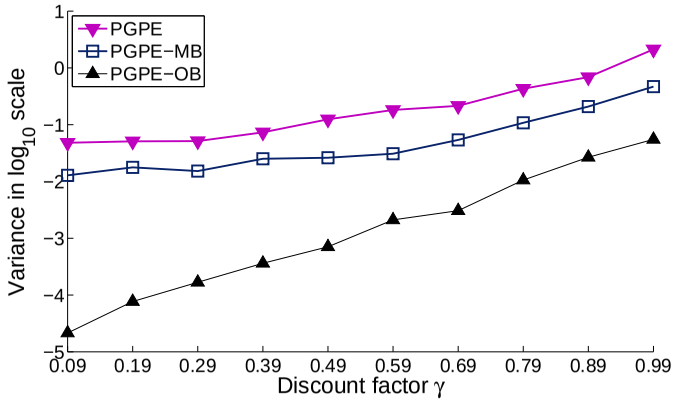
The results are summarized in Figure 3.2. Figure 3.2(a) compares the variance of REINFORCE with/without baselines, whereas Figure 3.2(b) compares the variance of PGPE with/without baselines. These plots show that introduction of baselines contributes highly to the reduction of the variance over iterations. Figure 3.2(c) compares the variance of PGPE-MB and PGPE-MB-SyS, showing that symmetric sampling contributes highly to stabilization. Figure 3.2(d) compares the variance of PGPE-OB and PGPE-OB-SyS, showing that the variance of PGPE-OB (without symmetric sampling) is smaller than that of PGPE-OB-SyS. Overall, in terms of the variance of gradient estimates, PGPE-OB compares favorably with other methods.

Next, we investigate how policy parameters change over 50 iterations. We set $N = 10$ and $T = 10$, and set the initial mean parameter at $\eta = -1.6$, -0.8, or -0.1, and initial deviation parameter at $\tau = 1$. Figure 3.3 depicts the contour of the expected return and illustrates changes of policy parameters over iterations for PGPE-MB and PGPE-OB. In the graphs, the maximum of the return surface is

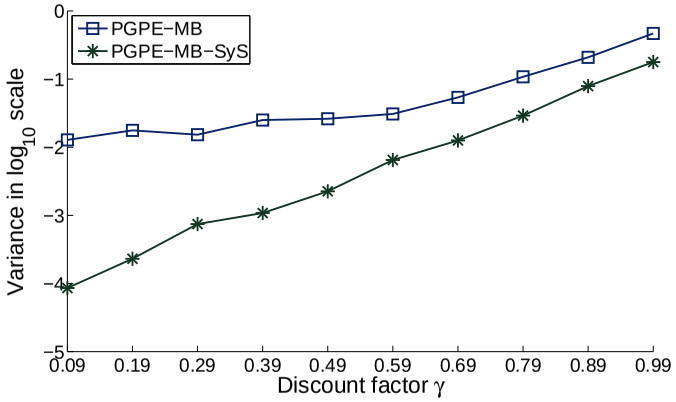
[2] If the deviation parameter σ takes a negative value during the policy-update process, we set it at 0.05.



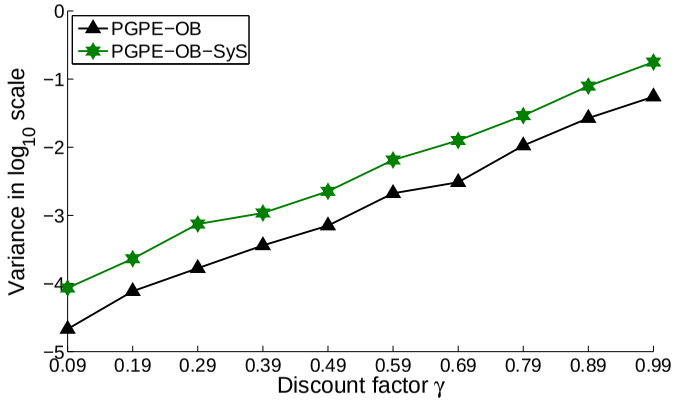
(a) REINFORCE and REINFORCE-OB



(b) PGPE, PGPE-MB, and PGPE-OB

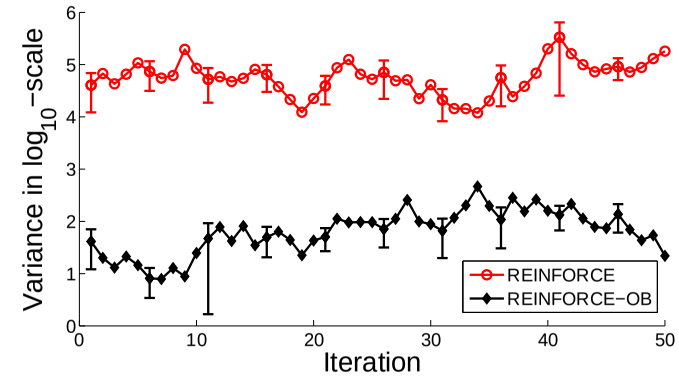


(c) PGPE-MB and PGPE-MB-SyS

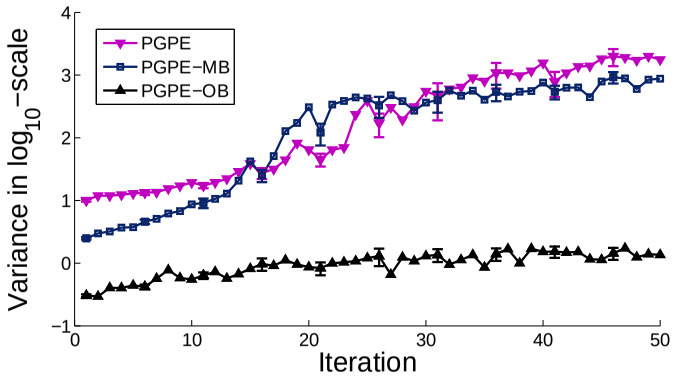


(d) PGPE-OB and PGPE-OB-SyS

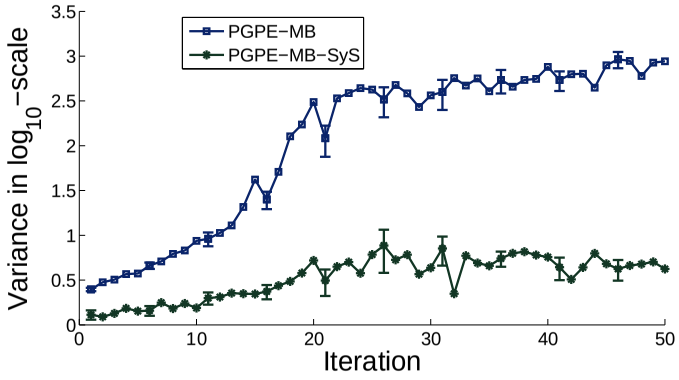
Figure 3.1: Variance of gradient estimates with respect to the mean parameter as functions of discount factor γ for toy data.



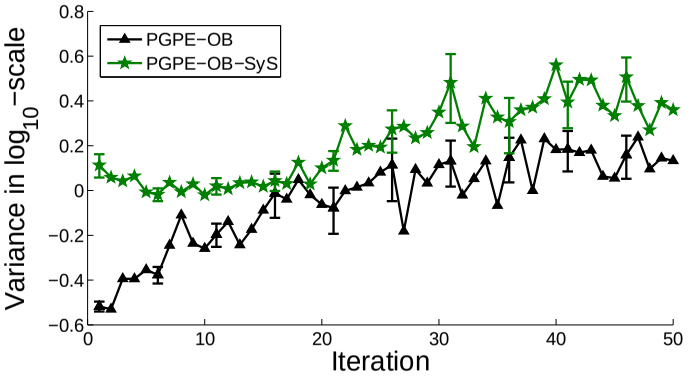
(a) REINFORCE and REINFORCE-OB



(b) PGPE, PGPE-MB, and PGPE-OB

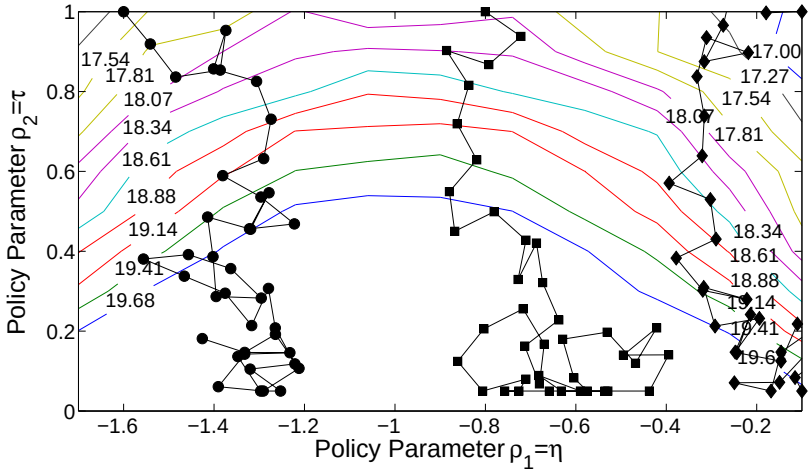


(c) PGPE-MB and PGPE-MB-SyS

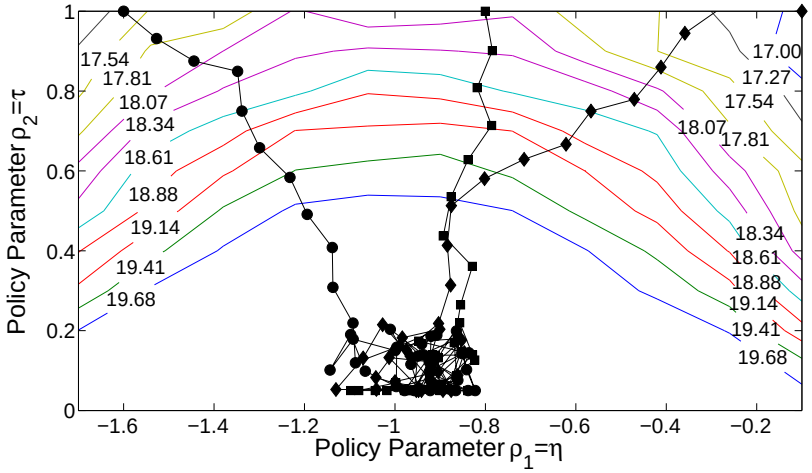


(d) PGPE-OB and PGPE-OB-SyS

Figure 3.2: Variance of gradient estimates with respect to the mean parameter through policy-update iterations for toy data.



(a) PGPE-MB



(b) PGPE-OB

Figure 3.3: Policy parameter change through policy-update iterations for toy data.

located at the middle bottom. Figure 3.3(a) shows that update directions of PGPE-MB are unstable and the three paths do not converge even after 50 iterations. On the other hand, Figure 3.3(b) shows that PGPE-OB gives much more reliable update directions and the three paths converge to a maximum point rapidly.

Performance of Learned Policies

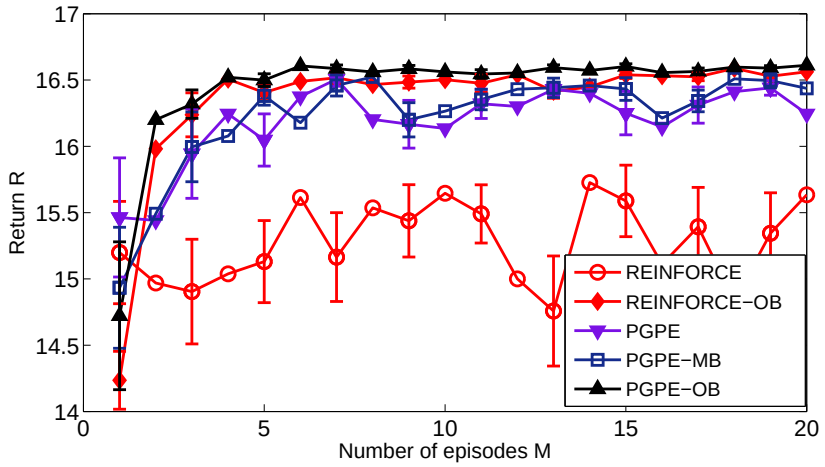
Finally, we evaluate returns obtained by each method. The trajectory length is fixed at $T = 20$, and the maximum number of policy-update iterations is set at 50. We investigate average returns over 20 runs as functions of the number of episodic samples N. We have two experimental results for different initial policies. Figure 3.4(a) shows the results when initial mean parameter μ is chosen randomly from $[-1.6, -0.1]$, which tends to perform well. The graph shows that PGPE-OB performs the best, especially when $N < 5$; then REINFORCE-OB follows with a small margin. PGPE-MB and plain PGPE also work reasonably well, although they are slightly unstable due to larger variance. Plain REINFORCE is highly unstable, which is caused by the huge variance of gradient estimates (see Figure 3.2 again).

Figure 3.4(b) describes the results when initial mean parameter μ is chosen randomly from $[-3.0, -0.1]$, which tends to result in poorer performance. In this setup, difference among the compared methods is more significant than the case with good initial policies. Overall, plain REINFORCE performs very poorly, and even REINFORCE-OB tends to be outperformed by the PGPE methods. This means that REINFORCE is very sensitive to the choice of initial policies. Among the PGPE methods, PGPE-OB works very well and converges quickly.

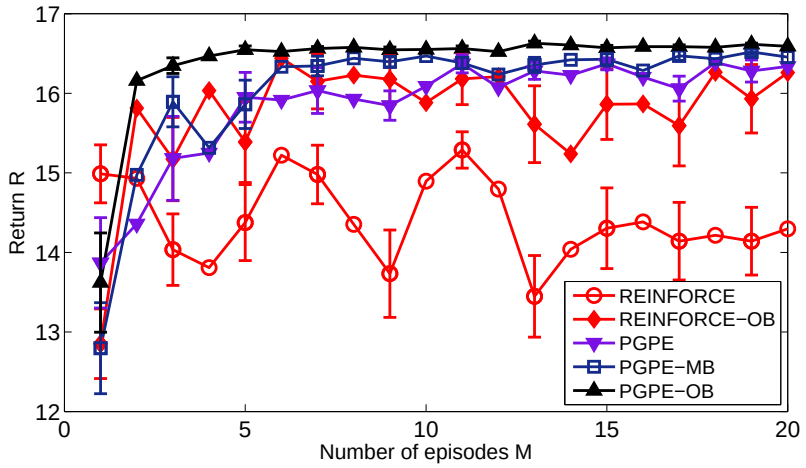
3.4.2 Cart-Pole Balancing

Here, we evaluate the performance of our proposed method in a more complex task of *cart-pole balancing* (Bugeja, 2003). A pole is hanged to the roof of a cart (see Figure 3.5), and the goal is to swing up the pole by moving the cart properly and try to keep the pole at the top.

The state space $\mathcal{S}$ is two-dimensional and continuous, which consists of the angle $\varphi \in [0, 2\pi]$ and angular velocity $\dot{\varphi} \in [-3\pi, 3\pi]$ of the pole. The action space



(a) Good initial policy



(b) Poor initial policy

Figure 3.4: Average returns over 20 runs as functions of the number of episodic samples N for toy data.

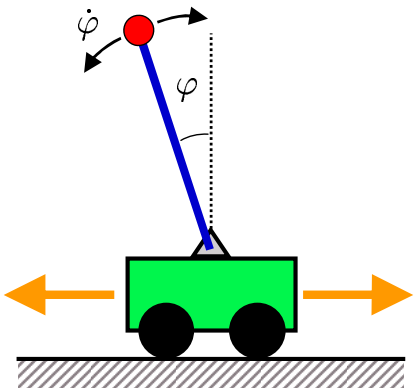


Figure 3.5: Cart-pole balancing.

$\mathcal{A}$ is one-dimensional and continuous, which corresponds to the force applied to the cart (note that we can *not* directly control the pole, but only indirectly through moving the cart). We use the Gaussian policy model for REINFORCE and linear policy model for PGPE, where state $\boldsymbol{s}$ is non-linearly transformed to a feature space via a basis function vector.

We use 20 Gaussian kernels with standard deviation $\sigma = 0.5$ as the basis functions, where the kernel centers are distributed over the following grid points:

$$\{0, \pi/2, \pi, 3\pi/2\} \times \{-3\pi, -3\pi/2, 0, 3\pi/2, 3\pi\}.$$

For the position of pole, we use the polar system where $\varphi = 0$ and $\varphi = 2\pi$ are treated as the same. That is, for the i-th Gaussian center $(c_i, \dot{c}_i)$, the basis function $\phi_i(\boldsymbol{s})$ is given by

$$\phi_i(\boldsymbol{s}) = \exp\left(-\frac{((\cos(\varphi) - \cos(c_i))^2 + (\sin(\varphi) - \sin(c_i))^2)/4 + (\dot{\varphi} - \dot{c}_i)^2/(6\pi)^2}{2\sigma^2}\right).$$

The dynamics of the pole (i.e., the update rule of the angle and the angular velocity) is given by

$$\begin{aligned}
\varphi_{t+1} &= \varphi_t + \dot{\varphi}_{t+1}\Delta t, \\
\dot{\varphi}_{t+1} &= \dot{\varphi}_t + \frac{9.8\sin(\varphi_t) - \alpha w l \dot{\varphi}_t^2 \sin(2\varphi_t)/2 + \alpha\cos(\varphi_t)a_t}{4l/3 - \alpha w l \cos^2(\varphi_t)}\Delta t,
\end{aligned}$$

where $\alpha = 1/W + w$ and a_t is the action taken at time t. We set the problem parameters as: the mass of the cart $W = 8$[kg], the mass of the pole $w = 2$[kg],

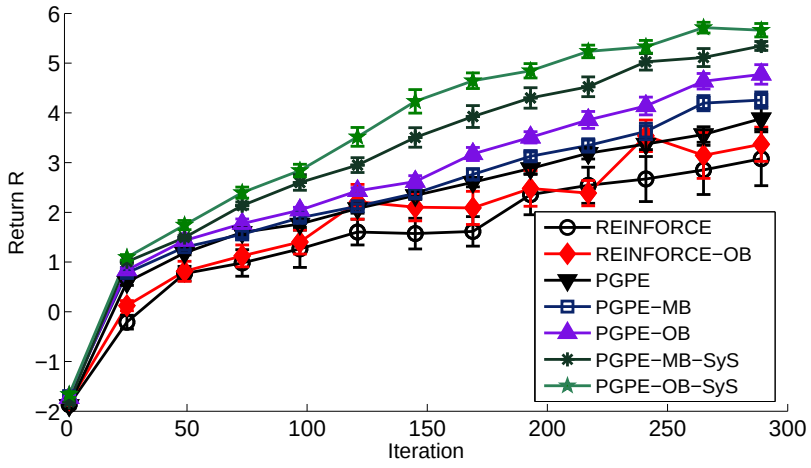


Figure 3.6: Average returns over 10 runs as functions of the number of iterations.

and the length of the pole $l = 0.5$[m]. We set the time step Δt for the position and velocity updates at 0.01[s] and action selection at 0.1[s]. The reward function is defined as

$$r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \cos(\varphi_{t+1}).$$

That is, the higher the pole is, the more rewards we can obtain. The initial policy is chosen randomly, and the initial-state probability density is set to be uniform. The agent collects $N = 100$ episodic samples with trajectory length $T = 40$, and the discount factor is set at $\gamma = 0.9$.

We investigate average returns over 10 trials as the functions of policy-update iterations. The return at each trial is computed over 100 test episodic samples (which are not used for policy learning). The experimental results are plotted in Figure 3.6, showing that the improvement of both plain REINFORCE and REINFORCE-OB tend to be slow, and all PGPE methods outperformed REINFORCE methods overall. Among the PGPE methods, the proposed PGPE-OB converges faster than PGPE-MB and plain PGPE. Moreover, the use of symmetric sampling further improves the performance. Overall, PGPE equipped with both the optimal baseline and symmetric sampling (PGPE-OB-SyS) gives the best performance.

3.5 Proofs of Theoretical Results

In this section, we give proofs of all the theorems in this chapter. First, we give some preliminaries.

If $X \sim \chi^2(k)$, then the non-central moments are given by

$$\mathbb{E}\left[X^n\right] = 2^n \frac{\Gamma(n + k/2)}{\Gamma(k/2)} = k(k+2)\cdots(k+2n-2),$$

where $\Gamma(z)$ is the Gamma function defined as

$$\Gamma(z) := \int_0^{+\infty} t^{z-1} e^{-t} dt.$$

The Gamma function satisfies $\Gamma(z+1) = z\Gamma(z)$, $\Gamma(1/2) = \sqrt{\pi}$, and $\Gamma(1) = 1$.

If $X \sim \mathcal{N}(\mu, \sigma^2)$, central absolute moments (the moments of $|X - \mu|$) are given by

$$\mathbb{E}\left[|x-\mu|^p\right] = \begin{cases} \sigma^p (p-1)!! \sqrt{2/\pi}, & p \text{ is odd,} \\ \sigma^p (p-1)!! & p \text{ is even,} \end{cases}$$

where $n!!$ denotes the double factorial defined by

$$n!! := \begin{cases} n \cdot (n-2) \cdots 5 \cdot 3 \cdot 1 & n \text{ is positive odd,} \\ n \cdot (n-2) \cdots 6 \cdot 4 \cdot 2 & n \text{ is positive even,} \\ 1 & n = 1 \text{ or } 0. \end{cases}$$

3.5.1 Proof of Theorem 3.1

For notational brevity, we denote the i-th component of $\boldsymbol{f}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\eta}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})$ and the i-th component of $\boldsymbol{g}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\tau}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})$ as

$$f_i(\boldsymbol{\theta}) = \nabla_{\eta_i} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}) = \frac{\theta_i - \eta_i}{\tau_i^2},$$
$$g_i(\boldsymbol{\theta}) = \nabla_{\tau_i} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}) = \frac{(\theta_i - \eta_i)^2 - \tau_i^2}{\tau_i^3}.$$

Proof. According to Eq.(4.1), we have

$$\begin{aligned}
\mathbf{Var}[R(h)\boldsymbol{f}(\boldsymbol{\theta})] &\leq \sum_{i=1}^{\ell} \mathbb{E}\left[(Rf_i)^2\right] \\
&= \sum_{i=1}^{\ell} \int p(\theta_i) \left(\sum_{t=1}^{T} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})\right)^2 \left(\frac{\theta_i - \eta_i}{\tau_i^2}\right)^2 d\theta_i \\
&\leq \sum_{i=1}^{\ell} \int p(\theta_i) \left(\sum_{t=1}^{T} \gamma^{t-1} \beta\right)^2 \left(\frac{\theta_i - \eta_i}{\tau_i^2}\right)^2 d\theta_i \\
&= \sum_{i=1}^{\ell} \int p(\theta_i) \left(\frac{\beta(1-\gamma^T)}{1-\gamma}\right)^2 \left(\frac{\theta_i - \eta_i}{\tau_i^2}\right)^2 d\theta_i \\
&= \sum_{i=1}^{\ell} \frac{\beta^2(1-\gamma^T)^2}{\tau_i^2(1-\gamma)^2} \mathbb{E}\left[\left(\frac{\theta_i - \eta_i}{\tau_i}\right)^2\right].
\end{aligned}$$

Let $\psi_i = ((\theta_i - \eta_i)/\tau_i)^2$ for $i = 1, \ldots, \ell$. We could know that $\psi_i \sim \chi^2(1)$ and $\mathbb{E}[\psi_i] = 1$ since $\theta_i \sim \mathcal{N}(\eta_i, \tau_i^2)$, and thus

$$\mathbf{Var}[R(h)\boldsymbol{f}(\boldsymbol{\theta})] \leq \frac{\beta^2(1-\gamma^T)^2 B}{(1-\gamma)^2}.$$

Hence the first part of Theorem 3.1 follows due to

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})\right] = \frac{1}{N} \mathbf{Var}[R(h)\boldsymbol{f}(\boldsymbol{\theta})].$$

Similarly,

$$\begin{aligned}
\mathbf{Var}[R(h)\boldsymbol{g}(\boldsymbol{\theta})] &\leq \sum_{i=1}^{\ell} \mathbb{E}\left[(Rg_i)^2\right] \\
&\leq \sum_{i=1}^{\ell} \frac{\beta^2(1-\gamma^T)^2}{\tau_i^2(1-\gamma)^2} \mathbb{E}\left[\left(\left(\frac{\theta_i - \eta_i}{\tau_i}\right)^2 - 1\right)^2\right].
\end{aligned}$$

Let $\psi_i = ((\theta_i - \eta_i)/\tau_i)^2$ for $i = 1, \ldots, \ell$. Since $\theta_i \sim \mathcal{N}(\eta_i, \tau_i^2)$, we could know that

$$\mathbb{E}\left[(\psi_i - 1)^2\right] = \mathbb{E}\left[\psi_i^2\right] - 2\mathbb{E}[\psi_i] + 1 = 2.$$

Hence

$$\mathbf{Var}[R(h)\boldsymbol{g}(\boldsymbol{\theta})] \leq \frac{2\beta^2(1-\gamma^T)^2 B}{(1-\gamma)^2}.$$

Notice that

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{J}(\boldsymbol{\rho})\right] = \frac{1}{N}\mathbf{Var}[R(h)\boldsymbol{g}(\boldsymbol{\theta})],$$

which completes the proof. □

3.5.2 Proof of Theorem 3.2

To begin with, we note that $\boldsymbol{\mu}$ is a vector and σ is a scalar in REINFORCE. We denote the i-th component of $\boldsymbol{f}(h) = \sum_{t=1}^{T}\nabla_{\boldsymbol{\mu}}\log\pi(a_t \mid \boldsymbol{s}_t, \boldsymbol{\theta})$ and the scalar function $g(h)$ as

$$f_i(h) = \sum_{t=1}^{T}\nabla_{\mu_i}\log\pi(a_t \mid \boldsymbol{s}_t, \boldsymbol{\theta}) = \sum_{t=1}^{T}\frac{a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t}{\sigma^2}s_{t,i},$$
$$g(h) = \sum_{t=1}^{T}\nabla_{\sigma}\log\pi(a_t \mid \boldsymbol{s}_t, \boldsymbol{\theta}) = \sum_{t=1}^{T}\frac{(a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)^2 - \sigma^2}{\sigma^3},$$

where all functions above are parameterized by $\boldsymbol{\theta}$.

Proof. Since

$$\mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})] = \frac{1}{N}\mathbf{Var}[R(h)\boldsymbol{f}(h)],$$
$$\mathbf{Var}[\nabla_{\sigma}\widehat{J}(\boldsymbol{\theta})] = \frac{1}{N}\mathbf{Var}[R(h)g(h)],$$

we can just focus on the bounds of $\mathbf{Var}[R(h)\boldsymbol{f}(h)]$ and $\mathbf{Var}[R(h)g(h)]$.

The upper bound of $\mathbf{Var}[R(h)\boldsymbol{f}(h)]$:

$$\begin{aligned}
\mathbf{Var}[R(h)\boldsymbol{f}(h)] &\le \sum_{i=1}^{\ell}\mathbb{E}\left[(Rf_i)^2\right] \\
&= \mathbb{E}\left[R^2\boldsymbol{f}^\top\boldsymbol{f}\right] \\
&= \int_h p(h)\left(\sum_{t=1}^{T}\gamma^{t-1}r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})\right)^2\left(\sum_{t=1}^{T}\frac{a_t - \boldsymbol{\mu}^\top\boldsymbol{s}_t}{\sigma^2}\boldsymbol{s}_t\right)^\top\left(\sum_{t=1}^{T}\frac{a_t - \boldsymbol{\mu}^\top\boldsymbol{s}_t}{\sigma^2}\boldsymbol{s}_t\right)dh \\
&\le \frac{\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2}\mathbb{E}\left[\left(\sum_{t,t'=1}^{T}\frac{(a_t - \boldsymbol{\mu}^\top\boldsymbol{s}_t)(a_{t'} - \boldsymbol{\mu}^\top\boldsymbol{s}_{t'})}{\sigma^2}\boldsymbol{s}_t^\top\boldsymbol{s}_{t'}\right)\right].
\end{aligned}$$

Let $\xi_t = (a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)/\sigma$ for $t = 1, \ldots, T$. Then, $\xi_1, \ldots, \xi_T$ are independent standard normal variables because of $a_t \sim \mathcal{N}(\boldsymbol{\mu}^\top \boldsymbol{s}_t, \sigma^2)$. Since all $\nabla_{\boldsymbol{\mu}} \log \pi(a_t \mid \boldsymbol{s}_t, \boldsymbol{\theta})$ in $\boldsymbol{f}(h)$ are parameterized by the states $\boldsymbol{s}_t$, and the stochasticity of ξ_t comes only from a_t, it is sufficient to consider fixed states. Given $\{\boldsymbol{s}_t\}_{t=1}^T$, $\xi_1 \boldsymbol{s}_1, \ldots, \xi_T \boldsymbol{s}_T$ are ℓ-dimensional independent normal variables with zero means, that is, $\mathbb{E}[\xi_t \boldsymbol{s}_t] = \boldsymbol{0}$. Hence,

$$
\begin{aligned}
&\mathbb{E}\left[\left(\sum_{t,t'=1}^{T} \frac{(a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)(a_{t'} - \boldsymbol{\mu}^\top \boldsymbol{s}_{t'})}{\sigma^2} \boldsymbol{s}_t^\top \boldsymbol{s}_{t'}\right)\right] \\
&= \mathbb{E}\left[\left(\sum_{t,t'=1}^{T} \xi_t \xi_{t'} \boldsymbol{s}_t^\top \boldsymbol{s}_{t'}\right)\right] \\
&= \sum_{t=1}^{T} \mathbb{E}\left[\xi_t^2 \boldsymbol{s}_t^\top \boldsymbol{s}_t\right] + \sum_{t,t'=1, t \neq t'}^{T} \mathbb{E}[\xi_t \boldsymbol{s}_t]^\top \mathbb{E}[\xi_{t'} \boldsymbol{s}_{t'}] \\
&= \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2 \mathbb{E}\left[\xi_t^2\right].
\end{aligned}
$$

Since $\xi_t \sim \mathcal{N}(0, 1)$, we have $\xi_t^2 \sim \chi^2(1)$ and $\mathbb{E}[\xi_t^2] = 1$. Consequently,

$$
\begin{aligned}
\mathbf{Var}[R(h)\boldsymbol{f}(h)] &\leq \frac{\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2} \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2 \mathbb{E}\left[\xi_t^2\right] \\
&= \frac{\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2} \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2 \\
&\leq \frac{D_T \beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2},
\end{aligned}
$$

with probability at least $(1-\delta)^{1/2N}$.

The upper bound of $\mathbf{Var}[R(h)g(h)]$:

$$\begin{aligned}
&\mathbf{Var}[R(h)g(h)] \\
&\leq \mathbb{E}\left[(Rg)^2\right] \\
&= \int_h p(h) \left(\sum_{t=1}^{T} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})\right)^2 \left(\sum_{t=1}^{T} \frac{(a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)^2 - \sigma^2}{\sigma^3}\right)^2 dh \\
&\leq \frac{\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2} \mathbb{E}\left[\left(\sum_{t=1}^{T} \left(\frac{a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t}{\sigma}\right)^2 - T\right)^2\right].
\end{aligned}$$

Let $\xi_t = (a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)/\sigma$ for $t = 1, \ldots, T$. Then $\xi_1, \ldots, \xi_T$ are independent standard normal variables. Let $\kappa = \sum_{t=1}^{T} \xi_t^2$. Then we have $\kappa \sim \chi^2(T)$ and

$$\mathbb{E}\left[(\kappa - T)^2\right] = \mathbb{E}\left[\kappa^2\right] - 2T\mathbb{E}[\kappa] + T^2 = 2T.$$

Hence

$$\mathbf{Var}[R(h)g(h)] \leq \frac{2T\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2}.$$

The lower bound of $\mathbf{Var}[R(h)f(h)]$: By the same technique used in the corresponding upper bound, we can prove that with probability at least $(1-\delta)^{1/2N}$,

$$\sum_{i=1}^{\ell} \mathbb{E}\left[(Rf_i)^2\right] \geq \frac{C_T \alpha^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2}.$$

On the other hand, based on the existence of $\{d_t\}_{t=1}^T$, there must be $\{d_{t,i}\}_{t=1}^T$ for $i = 1, \ldots, \ell$, such that $d_t^2 = \sum_{i=1}^{\ell} d_{t,i}^2$ and the inequality $|s_{t,i}| \leq d_{t,i}$ holds with probability at least $(1-\delta)^{1/2N\ell}$. Let $\xi_{t,i} = \mathrm{sgn}(s_{t,i})(a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)/\sigma$ for $t = 1, \ldots, T$ and $i = 1, \ldots, \ell$. Then all $\xi_{t,i}$ are independent standard normal variables. Let $\kappa_i = \sum_{t=1}^{T} \xi_{t,i}|s_{t,i}|$ and $\zeta_i = \sum_{t=1}^{T} \xi_{t,i} d_{t,i}$. Then $\kappa_i \sim \mathcal{N}(0, \sum_{t=1}^{T} s_{t,i}^2)$ for fixed $s_{1,i}, \ldots, s_{T,i}$, $\zeta_i \sim \mathcal{N}(0, \sum_{t=1}^{T} d_{t,i}^2)$, and $\mathbb{E}[|\kappa_i| \mid s_{1,i}, \ldots, s_{T,i}] \leq \mathbb{E}[|\zeta_i|]$ holds with probability at least $(1-\delta)^{1/2N\ell}$ over the choice of $s_{1,i}, \ldots, s_{T,i}$ according to the

underlying $p(h)$. When $\int_h p(h)Rf_i dh > 0$, with probability at least $(1-\delta)^{1/2N\ell}$,

$$\begin{aligned}
\int_h p(h)Rf_i dh &\leq \int_{\{h|f_i(h)>0\}} p(h)Rf_i dh \\
&\leq \frac{\beta(1-\gamma^T)}{1-\gamma} \int_{\{h|f_i(h)>0\}} p(h) f_i dh \\
&= \frac{\beta(1-\gamma^T)}{1-\gamma} \int_{\{h|\sum_{t=1}^T \xi_{t,i}|s_{t,i}|>0\}} p(h) \sum_{t=1}^T \xi_{t,i}|s_{t,i}| dh \\
&= \frac{\beta(1-\gamma^T)}{1-\gamma} \int_0^{+\infty} p(\kappa_i)\kappa_i d\kappa_i \\
&= \frac{\beta(1-\gamma^T)}{1-\gamma} \left(\frac{1}{2}\mathbb{E}[|\kappa_i|]\right) \\
&= \frac{\beta(1-\gamma^T)}{1-\gamma} \left(\frac{1}{2}\mathbb{E}_{s_{1,i},\ldots,s_{T,i}}\left[\mathbb{E}_{\kappa_i}[|\kappa_i| \mid s_{1,i},\ldots,s_{T,i}]\right]\right) \\
&\leq \frac{\beta(1-\gamma^T)}{1-\gamma} \left(\frac{1}{2}\mathbb{E}[|\zeta_i|]\right) \\
&= \frac{\beta(1-\gamma^T)}{1-\gamma} \frac{\sqrt{\sum_{t=1}^T d_{t,i}^2}}{\sqrt{2\pi}}.
\end{aligned}$$

When $\int_h p(h)Rf_i dh < 0$, with probability at least $(1-\delta)^{1/2N\ell}$,

$$\int_h p(h)Rf_i dh \geq -\frac{\beta(1-\gamma^T)}{1-\gamma} \frac{\sqrt{\sum_{t=1}^T d_{t,i}^2}}{\sqrt{2\pi}}.$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{\ell} (\mathbb{E}[Rf_i])^2 &= \sum_{i=1}^{\ell} \left(\int_h p(h)Rf_i dh\right)^2 \\
&\leq \sum_{i=1}^{\ell} \frac{\beta^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2} \frac{\sum_{t=1}^T d_{t,i}^2}{2\pi} \\
&= \frac{\beta^2(1-\gamma^T)^2}{2\pi\sigma^2(1-\gamma)^2} \sum_{t=1}^T \sum_{i=1}^{\ell} d_{t,i}^2 \\
&= \frac{\beta^2(1-\gamma^T)^2}{2\pi\sigma^2(1-\gamma)^2} \sum_{t=1}^T d_t^2 \\
&= \frac{D_T\beta^2(1-\gamma^T)^2}{2\pi\sigma^2(1-\gamma)^2},
\end{aligned}$$

with probability at least $(1-\delta)^{1/2N}$.

Finally, with probability at least $(1-\delta)^{1/N}$, we have

$$\begin{aligned}\mathbf{Var}[R(h)\boldsymbol{f}(h)] &= \sum_{i=1}^{\ell} \mathbb{E}\left[(Rf_i)^2\right] - (\mathbb{E}[Rf_i])^2 \\ &\geq \frac{(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2}\mathcal{L}(T).\end{aligned}$$

□

3.5.3 Proof of Theorem 3.3

Proof. According to Theorem 3.1 and Theorem 3.2, we could know that if there exists T_0 such that

$$\frac{(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}\mathcal{L}(T_0) \geq \frac{\beta^2(1-\gamma^T)^2 B}{N(1-\gamma)^2},$$

we could get

$$\mathcal{L}(T_0) \geq \beta^2 B\sigma^2.$$

Under our assumption that $\mathcal{L}(T) > 0$ and $\mathcal{L}(T)$ is monotonically increasing with respect to T, we will have that whenever

$$\exists T_0, \mathcal{L}(T_0) \geq \beta^2 B\sigma^2,$$

there must be

$$\forall T > T_0, \mathbf{Var}[\nabla_{\boldsymbol{\mu}}\widehat{J}(\boldsymbol{\theta})] > \mathbf{Var}[\nabla_{\boldsymbol{\eta}}\widehat{J}(\boldsymbol{\rho})].$$

□

3.5.4 Proof of Theorem 3.4

We denote $\boldsymbol{f}(\boldsymbol{\theta})$ and its i-th component $\boldsymbol{f}_i(\boldsymbol{\theta})$ as

$$\begin{aligned}\boldsymbol{f}(\boldsymbol{\theta}) &= \left(\nabla_{\boldsymbol{\eta}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})^\top, \nabla_{\boldsymbol{\tau}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})^\top\right)^\top = \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}), \\ \boldsymbol{f}_i(\boldsymbol{\theta}) &= (\nabla_{\eta_i} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}), \nabla_{\tau_i} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}))^\top = \nabla_{\boldsymbol{\rho}_i} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}).\end{aligned}$$

Note that we still have

$$\begin{aligned}\mathbf{Var}\left[\nabla_{\boldsymbol{\rho}}\widehat{J}^b(\boldsymbol{\rho})\right] &= \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{J}^b(\boldsymbol{\rho})\right] + \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{J}^b(\boldsymbol{\rho})\right] \\ &= \frac{1}{N}\mathbf{Var}[(R(h)-b)\nabla_{\boldsymbol{\eta}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})] + \frac{1}{N}\mathbf{Var}[(R(h)-b)\nabla_{\boldsymbol{\tau}} \log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})] \\ &= \frac{1}{N}\mathbf{Var}[(R(h)-b)\boldsymbol{f}(\boldsymbol{\theta})].\end{aligned}$$

Proof. According to Eq.(4.1), we have

$$\begin{aligned}\mathbf{Var}[(R(h)-b)\boldsymbol{f}(\boldsymbol{\theta})] &= \sum_{i=1}^{\ell}\mathbb{E}[(R-b)^2\boldsymbol{f}_i^\top\boldsymbol{f}_i] - (\mathbb{E}[(R-b)\boldsymbol{f}_i])^\top(\mathbb{E}[(R-b)\boldsymbol{f}_i]) \\ &= \sum_{i=1}^{\ell}\mathbb{E}[R^2\boldsymbol{f}_i^\top\boldsymbol{f}_i] - 2\mathbb{E}[Rb\boldsymbol{f}_i^\top\boldsymbol{f}_i] + \mathbb{E}[b^2\boldsymbol{f}_i^\top\boldsymbol{f}_i] \\ &\quad - (\mathbb{E}[R\boldsymbol{f}_i] - \mathbb{E}[b\boldsymbol{f}_i])^\top(\mathbb{E}[R\boldsymbol{f}_i] - \mathbb{E}[b\boldsymbol{f}_i]).\end{aligned}$$

Noticing that

$$\begin{aligned}\mathbb{E}[b\boldsymbol{f}_i] &= \int p(\theta_i \mid \boldsymbol{\rho}_i) b \nabla_{\boldsymbol{\rho}_i} \log p(\theta_i \mid \boldsymbol{\rho}_i) d\theta_i \\ &= \int b \nabla_{\boldsymbol{\rho}_i} p(\theta_i \mid \boldsymbol{\rho}_i) d\theta_i \\ &= b \nabla_{\boldsymbol{\rho}_i} \int p(\theta_i \mid \boldsymbol{\rho}_i) d\theta_i \\ &= b \nabla_{\boldsymbol{\rho}_i} 1 \\ &= b(\nabla_{\eta_i} 1, \nabla_{\tau_i} 1)^\top \\ &= (0,0)^\top,\end{aligned}$$

we have

$$\mathbf{Var}[(R(h)-b)\boldsymbol{f}(\boldsymbol{\theta})] = \mathbb{E}[R^2\boldsymbol{f}^\top\boldsymbol{f}] - 2\mathbb{E}[Rb\boldsymbol{f}^\top\boldsymbol{f}] + \mathbb{E}[b^2\boldsymbol{f}^\top\boldsymbol{f}] - \mathbb{E}[R\boldsymbol{f}]^\top\mathbb{E}[R\boldsymbol{f}].$$

The optimal baseline is obtained by minimizing the variance, so that differentiating it with respect to b and setting the result to zero will give us the optimal baseline for PGPE:

$$b^*_{\mathrm{PGPE}} = \frac{\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}]}{\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]}.$$

Subsequently,

$$
\begin{aligned}
&\mathbf{Var}[(R-b)\boldsymbol{f}] - \mathbf{Var}[(R-b^*_{\mathrm{PGPE}})\boldsymbol{f}] \\
&= -2\mathbb{E}[Rb\boldsymbol{f}^\top\boldsymbol{f}] + \mathbb{E}[b^2\boldsymbol{f}^\top\boldsymbol{f}] + 2\mathbb{E}[Rb^*_{\mathrm{PGPE}}\boldsymbol{f}^\top\boldsymbol{f}] - \mathbb{E}[b^{*2}_{\mathrm{PGPE}}\boldsymbol{f}^\top\boldsymbol{f}] \\
&= -2\mathbb{E}[Rb\boldsymbol{f}^\top\boldsymbol{f}] + \mathbb{E}[b^2\boldsymbol{f}^\top\boldsymbol{f}] + 2\frac{\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}]}{\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]}\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}] - \left(\frac{\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}]}{\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]}\right)^2\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}] \\
&= b^2\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}] - 2b\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}] + \frac{(\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}])^2}{\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]} \\
&= \left(b - \frac{\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}]}{\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]}\right)^2\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}] \\
&= \left(b - b^*_{\mathrm{PGPE}}\right)^2\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}],
\end{aligned}
$$

which leads to

$$
\begin{aligned}
&\mathbf{Var}[\nabla_{\boldsymbol{\rho}}\widehat{J}^{b}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\rho}}\widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \\
&= \frac{1}{N}\mathbf{Var}[(R-b)\boldsymbol{f}] - \frac{1}{N}\mathbf{Var}[(R-b^*_{\mathrm{PGPE}})\boldsymbol{f}] \\
&= \frac{(b-b^*_{\mathrm{PGPE}})^2}{N}\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}].
\end{aligned}
$$

□

3.5.5 Proof of Theorem 3.5

We denote the i-th component of $\boldsymbol{f}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\eta}}\log p(\boldsymbol{\theta} \mid \boldsymbol{\rho})$ as

$$
f_i(\boldsymbol{\theta}) = \nabla_{\eta_i}\log p(\boldsymbol{\theta} \mid \boldsymbol{\rho}) = \frac{\theta_i - \eta_i}{\tau_i^2}.
$$

Proof. By the same technique used in the proof of Theorem 3.4, we know, when the baseline $b = 0$,

$$
\mathbf{Var}[\nabla_{\boldsymbol{\eta}}\widehat{J}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\eta}}\widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] = \frac{\left(\mathbb{E}[R\boldsymbol{f}^\top\boldsymbol{f}]\right)^2}{N\mathbb{E}[\boldsymbol{f}^\top\boldsymbol{f}]}.
$$

On one hand,

$$\begin{aligned}\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}] &= \sum_{i=1}^{\ell} \mathbb{E}[f_i^2] \\ &= \sum_{i=1}^{\ell} \mathbb{E}\left[\left(\frac{\theta_i - \eta_i}{\tau_i^2}\right)^2\right] \\ &= \sum_{i=1}^{\ell} \frac{1}{\tau_i^2} \mathbb{E}\left[\left(\frac{\theta_i - \eta_i}{\tau_i}\right)^2\right].\end{aligned}$$

Let $\psi_i = ((\theta_i - \eta_i)/\tau_i)^2$ for $i = 1, \ldots, \ell$. We could know that $\psi_i \sim \chi^2(1)$ and $\mathbb{E}[\psi_i] = 1$ since $\theta_i \sim \mathcal{N}(\eta_i, \tau_i^2)$, and thus

$$\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}] = \sum_{i=1}^{\ell} \frac{1}{\tau_i^2} = B.$$

On the other hand, when $\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] > 0$, we have

$$\begin{aligned}\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] &= \sum_{i=1}^{\ell} \int p(\theta_i) R \left(\frac{\theta_i - \eta_i}{\tau_i^2}\right)^2 d\theta_i \\ &\leq \sum_{i=1}^{\ell} \frac{\beta(1-\gamma^T)}{\tau_i^2(1-\gamma)} \int p(\theta_i) \left(\frac{\theta_i - \eta_i}{\tau_i}\right)^2 d\theta_i \\ &= \sum_{i=1}^{\ell} \frac{\beta(1-\gamma^T)}{\tau_i^2(1-\gamma)} \mathbb{E}[\psi_i] \\ &= \frac{\beta(1-\gamma^T)B}{(1-\gamma)},\end{aligned}$$

while $\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] < 0$, we have

$$\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] \geq -\frac{\beta(1-\gamma^T)B}{(1-\gamma)}.$$

Hence,

$$\frac{\left(\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}]\right)^2}{\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}]} \leq \frac{\beta^2(1-\gamma^T)^2 B}{(1-\gamma)^2}.$$

Similarly,

$$\frac{\left(\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}]\right)^2}{\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}]} \geq \frac{\alpha^2(1-\gamma^T)^2 B}{(1-\gamma)^2},$$

which completes the proof. □

3.5.6 Proof of Theorem 3.6

We denote $\boldsymbol{f}(h) = \sum_{t=1}^{T} \nabla_{\boldsymbol{\mu}} \log \pi(a_t \mid \boldsymbol{s}_t, \boldsymbol{\theta})$.

Proof. It is easy to prove that, when $b = 0$,

$$\mathbf{Var}[\nabla_{\boldsymbol{\mu}} \widehat{J}(\boldsymbol{\theta})] - \mathbf{Var}[\nabla_{\boldsymbol{\mu}} \widehat{J}^{b^*_{\text{REINFORCE}}}(\boldsymbol{\theta})] = \frac{(\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}])^2}{N\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}]}.$$

From the proof of Theorem 3.2, we could have

$$\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}] = \frac{1}{\sigma^2} \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2.$$

On the other hand,

$$\begin{aligned}
&\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] \\
&= \int_h p(h) \left(\sum_{t=1}^{T} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) \right) \left(\sum_{t=1}^{T} \frac{a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t}{\sigma^2} \boldsymbol{s}_t \right)^\top \left(\sum_{t=1}^{T} \frac{a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t}{\sigma^2} \boldsymbol{s}_t \right) dh \\
&\leq \frac{\beta(1-\gamma^T)}{\sigma^2(1-\gamma)} \mathbb{E}\left[\left(\sum_{t,t'=1}^{T} \frac{(a_t - \boldsymbol{\mu}^\top \boldsymbol{s}_t)(a_{t'} - \boldsymbol{\mu}^\top \boldsymbol{s}_{t'})}{\sigma^2} \boldsymbol{s}_t^\top \boldsymbol{s}_{t'} \right) \right] \\
&= \frac{\beta(1-\gamma^T)}{\sigma^2(1-\gamma)} \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2.
\end{aligned}$$

Similarly,

$$\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}] \geq \frac{\alpha(1-\gamma^T)}{\sigma^2(1-\gamma)} \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2.$$

Therefore,

$$\frac{\alpha^2(1-\gamma^T)^2 \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2}{\sigma^2(1-\gamma)^2} \leq \frac{(\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}])^2}{\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}]} \leq \frac{\beta^2(1-\gamma^T)^2 \sum_{t=1}^{T} \|\boldsymbol{s}_t\|^2}{\sigma^2(1-\gamma)^2},$$

and subsequently, with probability at least $(1-\delta)^{1/N}$, we have

$$\frac{C_T \alpha^2(1-\gamma^T)^2}{\sigma^2(1-\gamma)^2} \leq \frac{(\mathbb{E}[R\boldsymbol{f}^\top \boldsymbol{f}])^2}{\mathbb{E}[\boldsymbol{f}^\top \boldsymbol{f}]} \leq \frac{\beta^2(1-\gamma^T)^2 D_T}{\sigma^2(1-\gamma)^2}.$$

From this, the theorem follows. □

3.5.7 Proof of Theorem 3.7

Proof. According to Theorem 3.5, we know

$$\mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \leq \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})\right] - \frac{\alpha^2(1-\gamma^T)^2 B}{N(1-\gamma)^2}.$$

According to Theorem 3.1, we have

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{J}(\boldsymbol{\rho})\right] \leq \frac{\beta^2(1-\gamma^T)^2 B}{N(1-\gamma)^2}.$$

Hence,

$$\mathbf{Var}[\nabla_{\boldsymbol{\eta}} \widehat{J}^{b^*_{\mathrm{PGPE}}}(\boldsymbol{\rho})] \leq \frac{(1-\gamma^T)^2}{N(1-\gamma)^2}(\beta^2 - \alpha^2)B.$$

According to Theorem 3.6, we know that

$$\mathbf{Var}[\nabla_{\boldsymbol{\mu}} \widehat{J}^{b^*_{\mathrm{REINFORCE}}}(\boldsymbol{\theta})] \leq \mathbf{Var}\left[\nabla_{\boldsymbol{\mu}} \widehat{J}(\boldsymbol{\theta})\right] - \frac{C_T \alpha^2(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}$$

will hold with probability at least $(1-\delta)^{1/2}$. Furthermore, according to Theorem 3.2, we have the following upper bound with probability at least $(1-\delta)^{1/2}$:

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\mu}} \widehat{J}(\boldsymbol{\theta})\right] \leq \frac{D_T \beta^2(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}.$$

Eventually, we arrive at the upper bound for REINFORCE with the optimal baseline:

$$\mathbf{Var}[\nabla_{\boldsymbol{\mu}} \widehat{J}^{b^*_{\mathrm{REINFORCE}}}(\boldsymbol{\theta})] \leq \frac{(1-\gamma^T)^2}{N\sigma^2(1-\gamma)^2}(D_T\beta^2 - C_T\alpha^2),$$

with probability at least $1-\delta$. □

3.6 Summary and Discussions

In this chapter, we analyzed and improved the stability of the policy gradient method called PGPE (policy gradients with parameter-based exploration). We theoretically showed that, under a mild condition, PGPE provides more stable gradient estimates than the classical REINFORCE method. We also derived the optimal baseline for PGPE, and theoretically showed that PGPE with the optimal baseline is more preferable than REINFORCE with the optimal baseline in

terms of the variance of gradient estimates. Finally, we demonstrated the usefulness of PGPE with optimal baseline through experiments. We also experimentally showed that the use of symmetric sampling further improves the performance.

Although we focused on the gradient based approach to optimize the distribution of policy parameters, there are many alternative heuristic approaches such as the *genetic algorithm* (GA), *estimation of distribution algorithm* (EDA), and *hill climbing* (HC). GA is a heuristic approach inspired by mutation, selection, and crossover (Goldberg, 1989). In GA, the population of randomly generated individuals are initially constructed. Then, in each iteration, multiple individuals are selected from the current population based on a fitness function, and new populations are formed by crossover between selected individuals with mutations. GAs could be applied to optimizing policy parameters by regarding individuals and the fitness function as policy parameters and the reward function respectively.

EDA is an outgrowth of GAs (Larrañaga and Lozano, 2002). In EDAs, the probability distribution of populations is estimated from selected individuals and new populations are sampled from the distribution. EDAs would be more stable since the difficulty of designing crossover and mutation is diminished. Similarly to GAs, EDA could be applied to optimizing policy parameters. However, estimating a high-dimensional distribution of policy parameters is highly challenging.

HC is an optimization technique which belongs to the class of local search methods (Koza et al., 2003). HC iteratively finds a better parameter by comparing the value of all neighbors of the current parameter. HC could be in principle applied to optimizing policy parameters, but it would not be computationally efficient to compare the return of all neighbored parameters.

These heuristic optimization techniques would be useful approaches to RL problem. Thus, an important future work along this line is to combine the meta-heuristics with the gradient-based method.

The low variance of PGPE was brought by considering a deterministic policy and introducing the stochasticity by drawing a policy parameter from a prior distribution. This per-trajectory formulation was indeed shown to be useful in reducing the variance of policy gradient estimates. However, PGPE has limitations, too. For example, the use of a finite horizon is essential in PGPE, because the gradient estimates need full trajectories. In particular, it is not straightforward

to handle the infinite-horizon case. Another issue is an extension to a partially-observable case. It is known that for every finite Markov decision problem (MDP) there exists a deterministic policy that is optimal (Ross, 1983). However, in a partially-observable MDP (POMDP), the best stationary stochastic policy can be arbitrarily better than the best stationary deterministic policy (Singh et al., 1994). Thus, the deterministic policy in PGPE can be a limitation when extending it to the POMDP framework. It is trivial to extend the current formulation to consider stochastic policies. However, this may lead to an increase of variance and thus slow down convergence.

In episodic policy gradient methods, the optimal baseline which does not bias policy gradient estimates is given by a single scalar for all trajectories (Peters and Schaal, 2006). However, in the non-episodic policy gradient methods, the optimal baseline can depend on the current state (Greensmith et al., 2004; Morimura et al., 2008; Peters and Schaal, 2008). Thus, if a good parameterization for the baseline is known, e.g., in a generalized linear form $b(s_t) = \boldsymbol{w}^T \boldsymbol{\phi}(s_t)$, this can significantly improve the gradient estimation process. However, the selection of the basis function can be difficult and often impractical in robotics (Peters and Schaal, 2006). On the other hand, it is interesting to see that if the value function is used as the baseline function in non-episodic policy gradient methods, such as in Peters and Schaal (2008); Sutton et al. (1999), the term $Q(s, a) - V(s)$ will lead to the *advantage function* (Baird, 1993), where $Q(s, a)$ is action value function and $V(s)$ is the value function.

Chapter 4

Efficient Sample Reuse in Policy Gradients with Parameter-based Exploration

The policy gradient approach is a flexible and powerful reinforcement learning method particularly for problems with continuous actions such as robot control. A common challenge in this scenario is how to reduce the variance of policy gradient estimates for reliable policy updates. In this chapter, we combine the following three ideas and give a highly effective policy gradient method: (a) the *policy gradients with parameter based exploration*, which is a recently proposed policy search method with low variance of gradient estimates, (b) an *importance sampling technique*, which allows us to reuse previously gathered data in a consistent way, and (c) an *optimal baseline*, which minimizes the variance of gradient estimates with their unbiasedness being maintained. For the proposed method, we give theoretical analysis of the variance of gradient estimates and show its usefulness through extensive experiments.

4.1 Introduction

The objective of *reinforcement learning* (RL) is to let an agent optimize its decision-making policy through interaction with an unknown environment (Sutton and Barto, 1998). Among possible approaches, *policy search* has become a popular

method because of its direct nature for policy learning (Bagnell et al., 2004). Particularly, in high-dimensional problems with continuous states and actions, policy search has been shown to be highly useful in practice (Ng and Jordan, 2000; Peters and Schaal, 2006).

Among policy search methods (Buşoniu et al., 2010), gradient-based methods are popular in physical control tasks because policies are changed gradually (Sutton et al., 1999; Kakade, 2002; Peters and Schaal, 2006) and thus steady performance improvement is ensured until a local optimal policy has been obtained. However, since the gradients estimated with these methods tend to have large variance and thus they may suffer from slow convergence.

Recently, a novel approach to using policy gradients called *policy gradients with parameter based exploration* (PGPE) was proposed (Sehnke et al., 2010). PGPE tends to produce gradient estimates with low variance by removing unnecessary randomness from policies and introducing useful stochasticity by considering a prior distribution for policy parameters. PGPE was shown to be more promising than alternative approaches experimentally (Sehnke et al., 2010; Zhao et al., 2012). However, PGPE still requires a relatively large number of samples to obtain accurate gradient estimates, which can be a critical bottleneck in real-world applications that require large costs and time in data collection.

To overcome this weakness, an *importance sampling* technique (Fishman, 1996) is useful under the *off-policy* RL scenario, where a data-collecting policy and the current target policy are different in general (Sutton and Barto, 1998). An importance sampling technique allows us to reuse previously collected data, which are collected following policies different from the current one in a consistent manner (Sutton and Barto, 1998; Shimodaira, 2000). However, naively using an importance sampling technique significantly increases the variance of gradient estimates, which can cause sudden changes in policy updates (Shelton, 2001; Peshkin and Shelton, 2002; Hachiya et al., 2011; Wawrzynski, 2009). To mitigate this problem, variance reduction techniques such as decomposition (Precup et al., 2000), truncation (Wawrzynski, 2009; Uchibe and Doya, 2004), normalization (Shelton, 2001; Peshkin and Shelton, 2002), and flattening (Hachiya et al., 2011) of importance weights are often used. However, these methods commonly suffer from the bias-variance trade-off, meaning that the variance is reduced at the

expense of increasing the bias.

The purpose of this chapter is to propose a new approach to systematically addressing the large variance problem in policy search. Basically, this work is an extension of our previous research (Zhao et al., 2012) to an *off-policy* scenario using an importance weighting technique. More specifically, we first give an off-policy implementation of PGPE called the *importance-weighted PGPE* (IW-PGPE) method for consistent sample reuse. We then derive the optimal baseline for IW-PGPE to minimize the variance of importance-weighted gradient estimates, following (Greensmith et al., 2004; Weaver and Tao, 2001). We show that the proposed method can achieve significant performance improvement over alternative approaches in experiments with an artificial domain. We also investigate that combining the proposed method with the truncation technique can further improve the performance in high-dimensional problems.

4.2 Off-Policy Extension of PGPE

In real-world applications such as robot control, gathering roll-out data is often costly. Thus, we want to keep the number of samples as small as possible. However, when the number of samples is small, policy gradients estimated by the original PGPE are not reliable enough.

The original PGPE is categorized as an *on-policy* algorithm (Sutton and Barto, 1998), where data drawn from the current target policy is used to estimate policy gradients. On the other hand, *off-policy* algorithms are more flexible in the sense that a data-collecting policy and the current target policy can be different. In this section, we extend PGPE to an *off-policy* scenario using importance-weighting, which allows us to reuse previously collected data in a consistent manner. We also theoretically analyze properties of the extended method.

4.2.1 Importance-Weighted PGPE

Let us consider an off-policy scenario where a data-collecting policy and the current target policy are different in general. In the context of PGPE, we consider two hyper-parameters, $\boldsymbol{\rho}$ for the target policy to learn and $\boldsymbol{\rho}'$ for data collection.

Let us denote data samples collected with hyper-parameter $\boldsymbol{\rho}'$ by D':

$$D' = \{(\boldsymbol{\theta}'_n, h'_n)\}_{n=1}^{N'} \overset{i.i.d}{\sim} p(h, \boldsymbol{\theta}|\boldsymbol{\rho}') = p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho}').$$

If we naively use data D' to estimate policy gradients by Eq.(2.4), we have an inconsistency problem:

$$\frac{1}{N'}\sum_{n=1}^{N'} \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}'_n|\boldsymbol{\rho}) R(h'_n) \overset{N'\to\infty}{\nrightarrow} \nabla_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho}),$$

which we refer to as "*non-importance-weighted PGPE*" (NIW-PGPE).

Importance sampling (Fishman, 1996) is a technique to systematically resolve this distribution mismatch problem. The basic idea of importance sampling is to weight samples drawn from a sampling distribution to match the target distribution, which gives a consistent gradient estimator:

$$\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho}) := \frac{1}{N'}\sum_{n=1}^{N'} w(\boldsymbol{\theta}'_n) \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}'_n|\boldsymbol{\rho}) R(h'_n) \overset{N'\to\infty}{\longrightarrow} \nabla_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho}),$$

where

$$w(\boldsymbol{\theta}) = \frac{p(\boldsymbol{\theta}|\boldsymbol{\rho})}{p(\boldsymbol{\theta}|\boldsymbol{\rho}')}$$

is called the *importance weight*.

An intuition behind importance sampling is that if we know how "important" a sample drawn from the sampling distribution is in the target distribution, we can make adjustment by importance weighting. We call this extended method *importance-weighted PGPE* (IW-PGPE).

Now we analyze the variance of gradient estimates in IW-PGPE. For a multi-dimensional space, we consider the *trace* of the covariance matrix of gradient vectors. That is, for a random vector $\boldsymbol{A} = (A_1, \ldots, A_\ell)^\top$, we define

$$\begin{aligned}\mathbf{Var}(\boldsymbol{A}) &= \mathrm{tr}\left(\mathbb{E}\left[(\boldsymbol{A} - \mathbb{E}[\boldsymbol{A}])(\boldsymbol{A} - \mathbb{E}[\boldsymbol{A}])^\top\right]\right), \\ &= \sum_{m=1}^{\ell} \mathbb{E}\left[(A_m - \mathbb{E}[A_m])^2\right], \end{aligned} \tag{4.1}$$

where $\mathbb{E}$ denotes the expectation.

Let

$$B = \sum_{i=1}^{\ell} \tau_i^{-2},$$

where ℓ is the dimensionality of the basis function vector $\boldsymbol{\phi}(\boldsymbol{s})$. For a $\boldsymbol{\rho} = (\boldsymbol{\eta}, \boldsymbol{\tau})$, we have the following theorem:

Theorem 4.1. *Assume that for all $\boldsymbol{s}$, a, and $\boldsymbol{s}'$, there exists $\beta > 0$ such that $r(\boldsymbol{s}, a, \boldsymbol{s}') \in [-\beta, \beta]$, and, for all $\boldsymbol{\theta}$, there exists $0 < w_{\max} < \infty$ such that $0 < w(\boldsymbol{\theta}) \leq w_{\max}$. Then we have the following upper bounds:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{\beta^2 (1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\max},$$
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{2\beta^2 (1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\max}.$$

Theorem 4.1 shows that the upper bound of the variance of $\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})$ is proportional to β^2 (the upper bound of squared rewards), $w_{\max}$ (the upper bound of the importance weight $w(\boldsymbol{\theta})$), B (the trace of the inverse Gaussian covariance), and $(1-\gamma^T)^2/(1-\gamma)^2$, and is inverse-proportional to sample size N'. It is interesting to see that the upper bound of the variance of $\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})$ is twice larger than that of $\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})$.

It is also interesting to see that the upper bounds are the same as the upper bounds for the plain PGPE (Theorem 1 of Zhao et al. (2012)) except for the factor w_{max}; when $w_{\max} = 1$, the bounds are reduced to those of the plain PGPE method. However, if the sampling distribution is significantly different from the target distribution, $w_{\max}$ can take a large value and thus IW-PGPE tends to produce a gradient estimator with large variance (at least in terms of its upper bound). Therefore, IW-PGPE may not be a reliable approach as it is.

Below, we give a variance reduction technique for IW-PGPE, which leads to a highly effective policy gradient algorithm.

4.2.2 Variance Reduction by Baseline Subtraction for IW-PGPE

To cope with the large variance of gradient estimates in IW-PGPE, several techniques have been developed in the context of sample reuse, for example, by flattening (Hachiya et al., 2011), truncating (Wawrzynski, 2009), and normalizing

(Shelton, 2001) the importance weight. Indeed, from Theorem 4.1, we can see that decreasing $w_{\max}$ by flattening or truncating the importance weight reduces the upper bounds of the variance of gradient estimates. However, all of those techniques are based on the bias-variance trade-off, and thus they lead to biased estimators.

Another, and possibly more promising variance reduction technique is subtraction of a constant *baseline* (Sutton, 1984; Williams, 1988; Greensmith et al., 2004; Weaver and Tao, 2001), which reduces the variance *without* increasing the bias. Here, we derive an optimal baseline for IW-PGPE to minimize the variance, and analyze its theoretical properties.

A policy gradient estimator with a baseline $b \in \mathbb{R}$ is defined as

$$\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b}_{\mathrm{IW}}(\boldsymbol{\rho}) := \frac{1}{N'} \sum_{n=1}^{N'} (R(h'_n) - b) w(\boldsymbol{\theta}'_n) \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}'_n | \boldsymbol{\rho}).$$

It is well known that $\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b}_{\mathrm{IW}}(\boldsymbol{\rho})$ is still a consistent estimator of the true gradient for any constant b (Greensmith et al., 2004). Here, we determine the constant baseline b so that the variance is minimized, following the line of Zhao et al. (2012). Let b^* be the optimal constant baseline for IW-PGPE that minimizes the variance:

$$b^* := \arg\min_{b} \mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b}_{\mathrm{IW}}(\boldsymbol{\rho})].$$

Then the following theorem gives the optimal constant baseline for IW-PGPE:

Theorem 4.2. *The optimal constant baseline for IW-PGPE is given by*

$$b^* = \frac{\mathbb{E}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho}')}[R(h) w^2(\boldsymbol{\theta}) \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]}{\mathbb{E}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho}')}[w^2(\boldsymbol{\theta}) \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]},$$

and the excess variance for a constant baseline b is given by

$$\mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b}_{\mathrm{IW}}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})] = \frac{(b-b^*)^2}{N'} \mathbb{E}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho}')}[w^2(\boldsymbol{\theta}) \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2],$$

where $\mathbb{E}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho}')}[\cdot]$ denotes the expectation of the function of random variables h and $\boldsymbol{\theta}$ with respect to $(h, \boldsymbol{\theta}) \sim p(h, \boldsymbol{\theta}|\boldsymbol{\rho}')$.

The above theorem gives an analytic expression of the optimal constant baseline for IW-PGPE. It also shows that the excess variance is proportional to the

squared difference of baselines $(b - b^*)^2$ and the expectation of the product of squared importance weight $w(\boldsymbol{\theta})$ and the squared norm of characteristic eligibility $\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2$, and is inverse-proportional to sample size N'.

Next, we analyze contributions of the optimal baseline to variance reduction in IW-PGPE:

Theorem 4.3. *Assume that for all $\boldsymbol{s}$, a, and $\boldsymbol{s}'$, there exists $\alpha > 0$ such that $r(\boldsymbol{s}, a, \boldsymbol{s}') \geq \alpha$, and, for all $\boldsymbol{\theta}$, there exists $w_{\min} > 0$ such that $w(\boldsymbol{\theta}) \geq w_{\min}$. Then we have the following lower bounds:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \geq \frac{\alpha^2(1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\min},$$
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \geq \frac{2\alpha^2(1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\min}.$$

Assume that for all $\boldsymbol{s}$, a, and $\boldsymbol{s}'$, there exists $\beta > 0$ such that $r(\boldsymbol{s}, a, \boldsymbol{s}') \in [-\beta, \beta]$, and, for all $\boldsymbol{\theta}$, there exists $0 < w_{\max} < \infty$ such that $0 < w(\boldsymbol{\theta}) \leq w_{\max}$. Then we have the following upper bounds:

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{\beta^2(1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\max},$$
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{2\beta^2(1-\gamma^T)^2 B}{N'(1-\gamma)^2} w_{\max}.$$

This theorem shows that the bounds of the variance reduction in IW-PGPE brought by the optimal constant baseline depend on the bounds of the importance weight. If importance weights are larger, using the optimal baseline can reduce the variance more.

Based on Theorems 4.1 and 4.3, we get the following corollary:

Corollary 4.4. *Assume that for all $\boldsymbol{s}$, a, and $\boldsymbol{s}'$, there exists $0 < \alpha < \beta$ such that $r(\boldsymbol{s}, a, \boldsymbol{s}') \in [\alpha, \beta]$, and, for all $\boldsymbol{\theta}$, there exists $0 < w_{\min} < w_{\max} < \infty$ such that $w_{\min} \leq w(\boldsymbol{\theta}) \leq w_{\max}$. Then we have the following upper bounds:*

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{(1-\gamma^T)^2 B}{N'(1-\gamma)^2}(\beta^2 w_{\max} - \alpha^2 w_{\min}),$$
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \leq \frac{2(1-\gamma^T)^2 B}{N'(1-\gamma)^2}(\beta^2 w_{\max} - \alpha^2 w_{\min}).$$

Comparing Theorem 4.1 and this corollary, we can see that the upper bounds for IW-PGPE with the optimal constant baseline are smaller than those for IW-PGPE with no baseline because $\alpha^2 w_{\min} > 0$. Although they are just upper bounds, they can still intuitively show that subtraction of the optimal constant baseline contributes to mitigating the large variance caused by importance weighting. If $w_{\min}$ is larger, then the upper bounds for IW-PGPE with the optimal constant baseline can be much smaller than those for IW-PGPE with no baseline.

4.3 Experimental Results

In this section, we experimentally investigate the usefulness of the proposed method, importance-weighted PGPE with the optimal constant baseline (which we denote by IW-PGPE$_{\text{OB}}$ hereafter). In the experiments, we estimate the optimal constant baseline using all collected data, as suggested in Greensmith et al. (2004); Peters and Schaal (2006); Weaver and Tao (2001). This approach introduces bias into the method because the same sample-set is used both for estimating the gradient and the baseline. Another possibility is to split the data into two parts: One is used for estimating the optimal constant baseline and the other is used for estimating the gradient. However, we found that this splitting approach does not work well in our preliminary experiments.

4.3.1 Illustrative Example

First, we illustrate the behavior of PGPE methods using a toy dataset.

Setup

The dynamics of the environment is defined as

$$s_{t+1} = s_t + a_t + \varepsilon,$$

where $s_t \in \mathbb{R}$, $a_t \in \mathbb{R}$, and $\varepsilon \sim \mathcal{N}(0, 0.5^2)$ is stochastic noise. The initial state s_1 is randomly chosen from the standard normal distribution. The linear deterministic controller is represented by $a_t = \theta s_t$ for $\theta \in \mathbb{R}$. The immediate

reward function is given by

$$r(s_t, a_t) = \exp\left(-s_t^2/2 - a_t^2/2\right) + 1,$$

which is bounded in $(1, 2]$. In the toy dataset experiments, we always set the discount factor at $\gamma = 0.9$, and we always use the adaptive learning rate $\varepsilon = 0.1/\|\nabla_\rho \hat{\mathcal{J}}(\boldsymbol{\rho})\|$ (Matsubara et al., 2010).

Here, we compare the following PGPE methods:

- **PGPE:** Plain PGPE without data reuse (Sehnke et al., 2010).
- **PGPE$_{\mathbf{OB}}$:** Plain PGPE with the optimal constant baseline without data reuse (Zhao et al., 2012).
- **NIW-PGPE:** Data-reuse PGPE without importance weights.
- **NIW-PGPE$_{\mathbf{OB}}$:** Data-reuse PGPE$_{\mathrm{OB}}$ without importance weights.
- **IW-PGPE:** Importance-weighted PGPE.
- **IW-PGPE$_{\mathbf{OB}}$:** Importance-weighted PGPE with the optimal baseline.

Suppose that a small amount of samples consisting of N trajectories with length T is available at each iteration. More specifically, given the hyper-parameter $\boldsymbol{\rho}_L = (\eta_L, \tau_L)$ at the Lth iteration, we first choose the policy parameter θ_n^L from $p(\theta|\boldsymbol{\rho}_L)$, and then run the agent to generate trajectory h_n^L according to $p(h|\theta_n^L)$. Initially, the agent starts from a randomly selected state s_1 following the initial state probability density $p(s_1)$ and chooses an action based on the policy $\pi(a_t|s_t, \theta_n^L)$. Then the agent makes a transition following the dynamics of the environment $p(s_{t+1}|s_t, a_t)$ and receives a reward $r_t = r(s_t, a_t, s_{t+1})$. The transition is repeated T times to get a trajectory, which is denoted as $h_n^L = \{s_t, a_t, r_t, s_{t+1}\}_{t=1}^T$. We repeat the procedure N times, and, the samples gathered at the Lth iteration is obtained, which is expressed as $D^L = \{(\theta_n^L, h_n^L)\}_{n=1}^N$.

In the data-reuse methods, we estimate gradients at each iteration based on the current data and all previously collected data $D^{1:L} = \{D^l\}_{l=1}^L$, by the estimated gradients to update the policy hyper-parameters (i.e., mean η and standard deviation τ). In the plain PGPE method and the plain PGPE$_{\mathrm{OB}}$ method, we only use the

on-policy data D^L to estimate the gradients at each iteration, by the estimated gradients to update the policy hyper-parameters. If the deviation parameter τ takes a value smaller than 0.05 during the parameter-update process, we set it at 0.05.

Below, we experimentally evaluate the variance, bias, and mean squared error of the estimated gradients, trajectories of learned hyper-parameters, and obtained returns.

Estimated Gradients

We investigate how data reuse influences estimated gradients over iterations. Below, we focus on gradients with respect to the mean parameter η.

We randomly choose initial mean parameter η from the standard normal distribution, and fix the initial deviation parameter at $\tau = 1$. We collect $N = 10$ trajectories with the trajectory length $T = 10$ at each iteration, and update hyper-parameters over 20 iterations. Here, the variance and squared bias of estimated gradients at each iteration (e.g., at the Lth iteration, $L = 1, \ldots, 20$) are investigated for $M = 10000$ trials:

$$\mathrm{Var} := \frac{1}{M}\sum_{m=1}^{M}\left\|\nabla_{\eta_L}\widehat{\mathcal{J}}^m(\boldsymbol{\rho}_L) - \frac{1}{M}\sum_{m'=1}^{M}\nabla_{\eta_L}\widehat{\mathcal{J}}^{m'}(\boldsymbol{\rho}_L)\right\|^2,$$

$$\mathrm{Bias}^2 := \left\|\frac{1}{M}\sum_{m=1}^{M}\nabla_{\eta_L}\widehat{\mathcal{J}}^m(\boldsymbol{\rho}_L) - \nabla_{\eta_L}\mathcal{J}(\boldsymbol{\rho}_L)\right\|^2,$$

where $\nabla_{\eta_L}\widehat{\mathcal{J}}^m(\boldsymbol{\rho}_L)$ is an estimated gradient in the m-th trial. More specifically, we estimate the gradients M times with different random seeds at the Lth iteration as follows: We generate samples $D_m^{1:L} = \{D_m^l\}_{l=1}^L$ following the corresponding distributions $\{D_m^l \overset{i.i.d}{\sim} p(h, \theta|\boldsymbol{\rho}_l)\}_{l=1}^L$ in each trial ($m = 1, \ldots, M$), and we estimate the gradient $\nabla_{\eta_L}\widehat{\mathcal{J}}^m(\boldsymbol{\rho}_L)$ with the generated samples $D_m^{1:L}$. The variance and squared bias at the Lth iteration are calculated based on the estimated gradients from M trials. In this experiment, the true gradient $\nabla_{\eta_L}\mathcal{J}(\boldsymbol{\rho}_L)$ at the Lth iteration is approximated by the plain PGPE method using Eq.(2.4) with $N = 10000$ on-policy samples. Note that the sum of the variance and squared bias agrees with

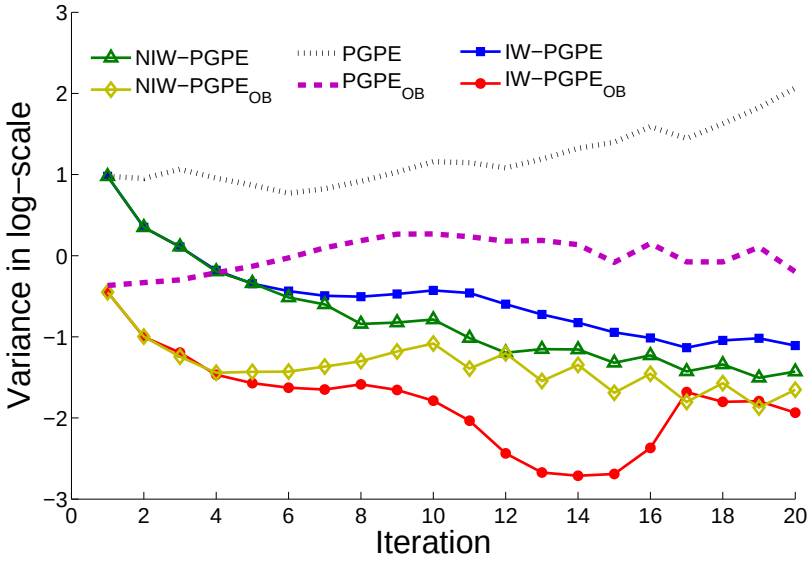
the mean squared error:

$$\mathrm{Var} + \mathrm{Bias}^2 = \frac{1}{M} \sum_{m=1}^{M} \|\nabla_{\eta_L} \widehat{\mathcal{J}}^m(\boldsymbol{\rho}_L) - \nabla_{\eta_L} \mathcal{J}(\boldsymbol{\rho}_L)\|^2. \tag{4.2}$$

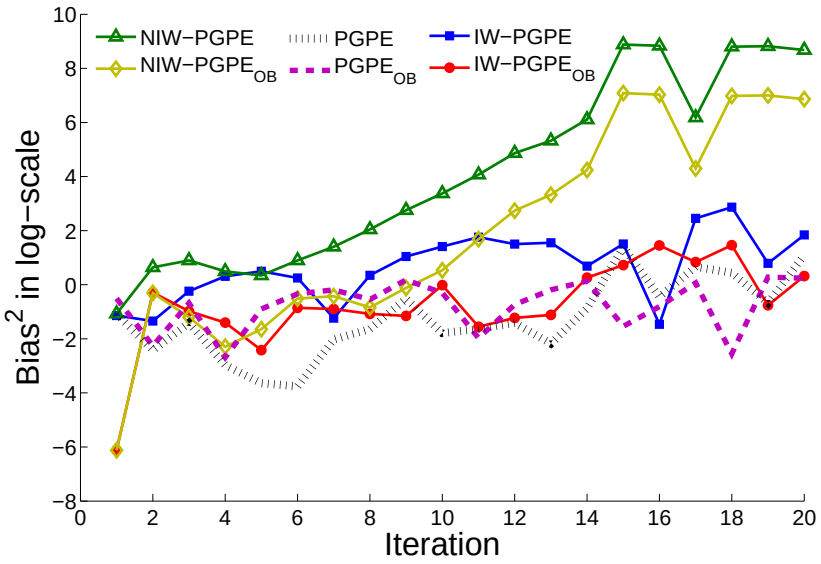
We update the hyper-parameters $\boldsymbol{\rho}_L$ based on the estimated true gradient $\nabla_{\eta_L} \mathcal{J}(\boldsymbol{\rho}_L)$, and obtain $\boldsymbol{\rho}_{L+1}$. Then, we investigate the variance and bias at the next iteration, i.e., the $(L+1)$th iteration, following the above procedures. Figure 4.1 shows the variance and squared bias over 20 iterations.

From Figure 4.1(a), we can see that IW-PGPE$_{\mathrm{OB}}$ provides gradient estimates with the lowest variance among the compared methods. IW-PGPE has a larger variance than NIW-PGPE, which well agrees with our theoretical analysis: According to Theorem 4.1, upper bounds of the variance are proportional to the importance weight, which is always 1 in NIW-PGPE, but is very large in IW-PGPE if the target distribution is significantly different from the sampling distribution. In order to see whether the upper bound of importance weights is really large, we measure the maximum value of importance weights over iterations, which is shown in Figure 4.2. Figure 4.2(a) shows that the maximum value of importance weights tends to be larger over iterations, which further illustrates how importance weights influence the variance of gradient estimates in IW-PGPE.

We can also see that the gap in the variance between IW-PGPE and IW-PGPE$_{\mathrm{OB}}$ tends to be larger over iterations, which is also consistent with our theoretical analysis: According to Theorem 4.3, the larger the importance weight is, the more the optimal constant baseline contributes to reducing the variance. The importance weight may get larger at later iterations, because distributions in the first and the last iterations may be significantly different (Figure 4.2 exactly illustrates this phenomenon). Thus, variance reduction from IW-PGPE to IW-PGPE$_{\mathrm{OB}}$ by the optimal constant baseline tends to be more significant in later iterations. Gradient estimates in both NIW-PGPE$_{\mathrm{OB}}$ and IW-PGPE$_{\mathrm{OB}}$ are with smaller variance than the plain PGPE$_{\mathrm{OB}}$ method, because the more data we use, the smaller variance of gradient estimates we can obtain as expected from the theory. IW-PGPE$_{\mathrm{OB}}$ provides smaller variance than NIW-PGPE$_{\mathrm{OB}}$, which is our expected result: According to Theorem 4.3, if the importance weights are larger, using the optimal constant baseline can reduce variance more, while the importance weights are always 1 in NIW-PGPE$_{\mathrm{OB}}$ (see Figure 4.2(b)). The plain PGPE$_{\mathrm{OB}}$ has smaller



(a) Variance



(b) Bias^2

Figure 4.1: Variance and Bias^2 of gradient estimates with respect to the mean parameter η through parameters update iterations.

variance than the plain PGPE, which well agrees with the results reported in Zhao et al. (2012).

Figure 4.1(b) shows that introduction of the optimal baseline does not increase the bias. NIW-PGPE and NIW-PGPE$_{\mathrm{OB}}$ have very large bias, because naively reusing previous data leads to an inconsistent and biased gradient estimator. The bias of gradient estimates in IW-PGPE is fairly small, because IW-PGPE is not only consistent, but also unbiased. The plain PGPE and plain PGPE$_{\mathrm{OB}}$ are also with small bias, as expected.

Because our proposed IW-PGPE$_{\mathrm{OB}}$ has small bias and the smallest variance among the compared methods, it also gives the smallest mean squared error (see Eq.(4.2)).

Justification of Theorem 4.3

Here, we experimentally justify Theorem 4.3. When we calculate the upper bounds and the lower bounds, w_{max} and w_{min} are the maximum value and minimum value of w in the experiments, since they are theoretically unknown. We investigate the variance reduction from IW-PGPE to IW-PGPE$_{\mathrm{OB}}$, i.e.,

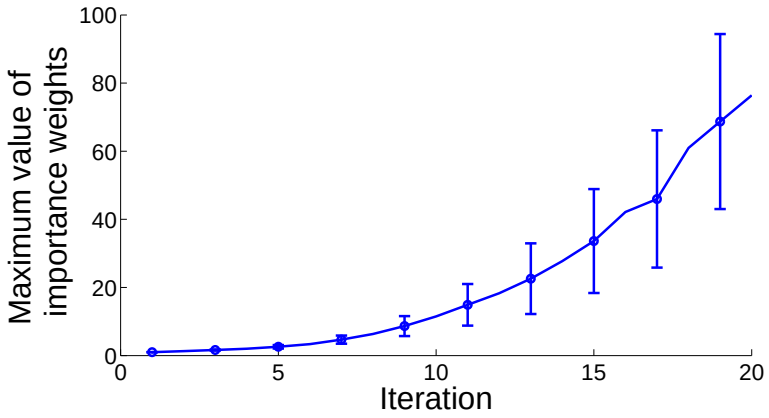
$$\mathbf{Var}\left[\nabla_{\boldsymbol{\rho}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\rho}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right].$$

We first collect $N' = 10$ off-policy trajectories, which are drawn from Gaussian prior with $\rho = (-1.6, 1)$. We then reuse these samples to estimate the gradients with parameters $\rho = (-1.5, 0.5)$, $(-0.8, 0.5)$, and $(-0.1, 0.5)$ in IW-PGPE and IW-PGPE$_{\mathrm{OB}}$, respectively. The variances are calculated from 100 trials.

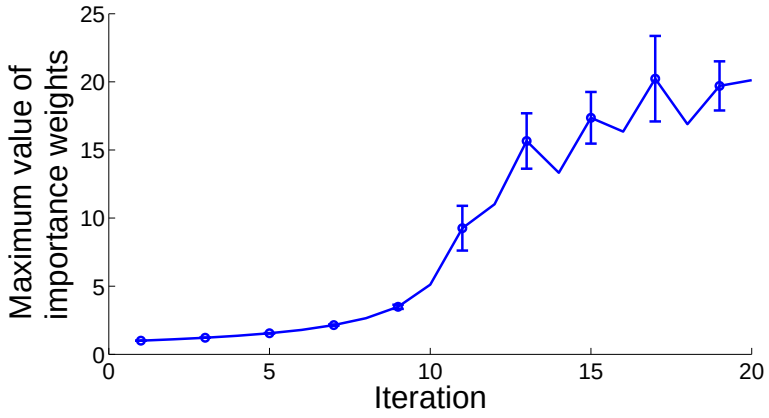
The averaged variance reduction over 100 runs, upper bounds, and lower bounds of variance reduction are summarized in Table 4.1. Through the results, we can see that numerical results of variance reduction are located between the lower bounds and upper bounds. Moreover, we further confirmed that variance reduction from IW-PGPE to IW-PGPE$_{\mathrm{OB}}$ by the optimal constant baseline tends to be more significant when the w_{max} is larger.

Hyper-Parameter Trajectories

Next, we illustrate how learned hyper-parameters change over iterations. Here we compare the behavior of the following three methods: NIW-PGPE, IW-PGPE and



(a) IW-PGPE



(b) IW-PGPE$_{\mathrm{OB}}$

Figure 4.2: Average maximum values of importance weights over 20 runs through parameter update iterations.

Table 4.1: Empirical values, lower bounds, and upper bounds of variance reduction from IW-PGPE to IW-PGPE$_{\text{OB}}$.

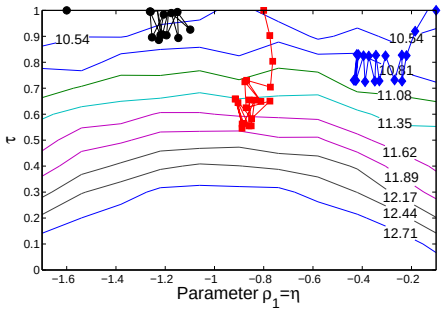
Parameter $\boldsymbol{\rho}$		w		Lower Bounds		Upper Bounds		Variance Reduction	
η	τ	w_{min}	w_{max}	η	τ	η	τ	η	τ
-1.5	0.5	2.29e-5	2.0134	3.89e-4	7.78 e-4	136.6583	273.32	45.3058($\pm$1.5742)	71.4705($\pm$ 2.4567)
-0.8	0.5	9.87e-14	3.0643	1.67e-12	3.34e-12	207.9887	415.98	77.3764($\pm$2.6514)	109.1836($\pm$3.4584)
-0.1	0.5	1.19e-13	8.9634	2.03e-12	4.06e-12	608.3907	1216.8	138.4063($\pm$5.7752)	222.3223($\pm$10.0947)

our proposed method IW-PGPE$_{\mathrm{OB}}$. We fix the initial deviation parameter at $\tau = 1$, and test the three different initial mean parameters: $\eta = -1.6$, -0.8, and -0.1. Figure 4.3 depicts the contour of the expected return, where the maximum of the return surface is located at the middle bottom.

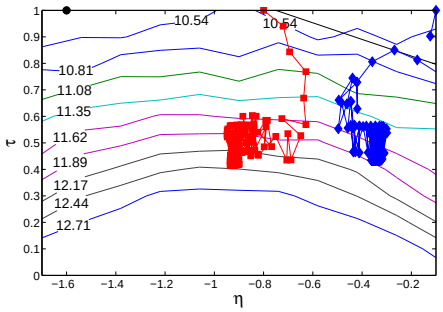
First, let us investigate how the hyper-parameters change over 20 iterations in a large-sample case with $N = 10$. From Figure 4.3(a), we can see that NIW-PGPE can not properly update the solutions, which means that the inconsistency can not be overcome by increasing the number of samples. On the other hand, Figure 4.3(c) shows that IW-PGPE can lead the solutions to an area with large returns sometimes, but can not always reach an area with large returns after 20 iterations. This indicates that the consistency of importance weighting tends to be helpful when the number of samples is large, but it can not converge rapidly because of the large variance. Figure 4.3(e) shows that IW-PGPE$_{\mathrm{OB}}$ gives the reliable update directions and the three paths converge rapidly to the vicinity of the maximum point without detours. This shows that the optimal constant baseline highly contributes to improving the convergence property of IW-PGPE.

Next, we investigate the performance over 200 iterations with only $N = 1$. Figure 4.3(b) shows that NIW-PGPE can not properly update the solutions to the maximum point because of the inconsistency, and Figure 4.3(d) shows that the IW-PGPE solutions can not always reach an area with large returns (middle bottom) after 200 iterations, which is because the variance in IW-PGPE is crucial in this extreme scenario. However, Figure 4.3(e) shows that the proposed IW-PGPE$_{\mathrm{OB}}$ can still find fairly reliable update directions with only $N = 1$.

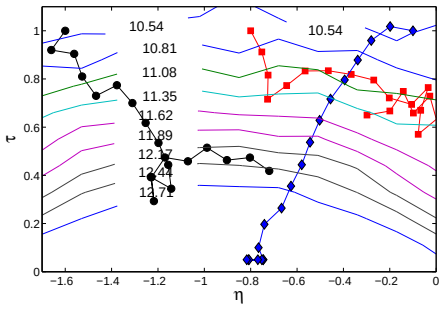
Next, we investigate the directions of estimated gradients more systematically. We fix the starting point at $\eta = -0.8$ and $\tau = 0.5$. The true gradient direction is calculated by the plain PGPE method with 10000 on-policy samples. In this experiment, we first collect $N' = 10$ off-policy samples, which are drawn from $\mathcal{N}(-1.6, 1)$. We then reuse these off-policy samples to estimate the gradients in the data-reuse methods. We calculate the gradients 20 times with different random seeds, and investigate the angle between the true gradient and the estimated gradients. The results are summarized in Figure 4.4. In Figure 4.4(a), the red line denotes the true gradient and blue lines are the estimated gradients by the NIW-PGPE method. The histograms of angles between the true gradient and the



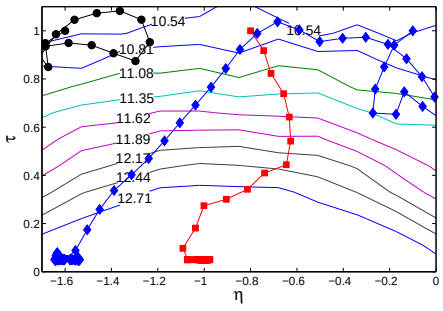
(a) NIW-PGPE ($N = 10$)



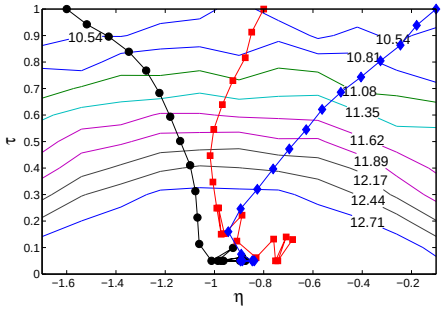
(b) NIW-PGPE ($N = 1$)



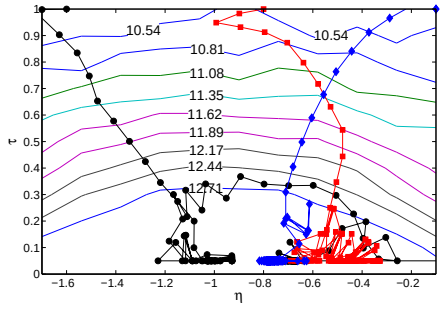
(c) IW-PGPE ($N = 10$)



(d) IW-PGPE ($N = 1$)



(e) IW-PGPE$_{\text{OB}}$ ($N = 10$)



(f) IW-PGPE$_{\text{OB}}$ ($N = 1$)

Figure 4.3: Trajectories of policy hyper-parameters over iterations.

estimated gradients are plotted in Figure 4.4(b). The graph shows that the angles are concentrated in $[-150, -90]$, which further explains the inconsistent property of the NIW-PGPE method. Observing the angle distribution for IW-PGPE in Figure 4.4(d), we can see that the angles are widely distributed in $[-180, 180]$, which clearly illustrates the large variance problem of IW-PGPE. On the other hand, the angles for the IW-PGPE$_{\mathrm{OB}}$ method are concentrated in $[-60, 60]$, which highlights the small variance and consistent properties of IW-PGPE$_{\mathrm{OB}}$.

Performance of Learned Policies

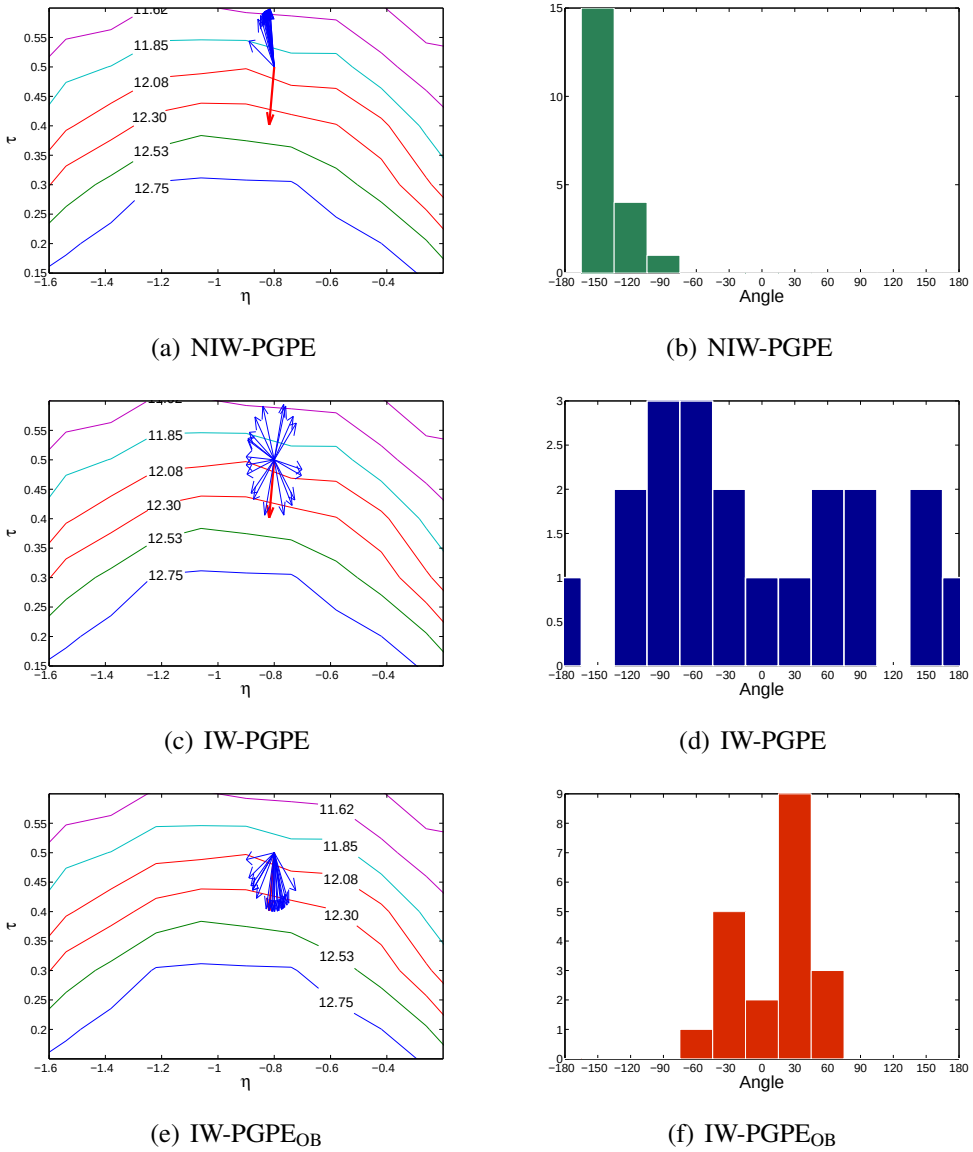
Finally, we evaluate average expected returns obtained by each method over 20 runs. The expected return at each trial is approximated using 100 newly-drawn test episodic data (which are not used for policy learning). The initial mean parameter η is chosen randomly from the standard normal distribution, and the deviation parameter is fixed at $\tau = 1$.

Figure 4.5 shows that IW-PGPE$_{\mathrm{OB}}$ improves the performance over iterations and converges very fast. The performance of NIW-PGPE is not largely improved over iterations, which is caused by biased gradient estimates (see Figure 4.3(a) again). IW-PGPE works better than NIW-PGPE, but the performance is saturated after 9 iterations. IW-PGPE$_{\mathrm{OB}}$ does not outperform NIW-PGPE$_{\mathrm{OB}}$ that much at the first several iterations, because the difference between the target distribution and a sampling distribution is not that large at the beginning. However, the upper bound of importance weights tends to become larger over iterations (see Figure 4.2(b) again), which makes IW-PGPE$_{\mathrm{OB}}$ more reliable than NIW-PGPE$_{\mathrm{OB}}$ in the latter iterations. The plain PGPE$_{\mathrm{OB}}$ method works fairly well with $N = 10$ on-policy samples, but it is still not as good as IW-PGPE$_{\mathrm{OB}}$.

4.3.2 Mountain Car

Next, we evaluate our proposed method in the *mountain car* task, which is illustrated in Figure 4.6. The task consists of a car and two hills whose landscape is described as $\sin(3x)$. The top of the right hill is the goal to which we want to guide the car.

We compare the following 7 methods:



(a) NIW-PGPE (b) NIW-PGPE

(c) IW-PGPE (d) IW-PGPE

(e) IW-PGPE$_{\mathrm{OB}}$ (f) IW-PGPE$_{\mathrm{OB}}$

Figure 4.4: Directions of estimated gradients.

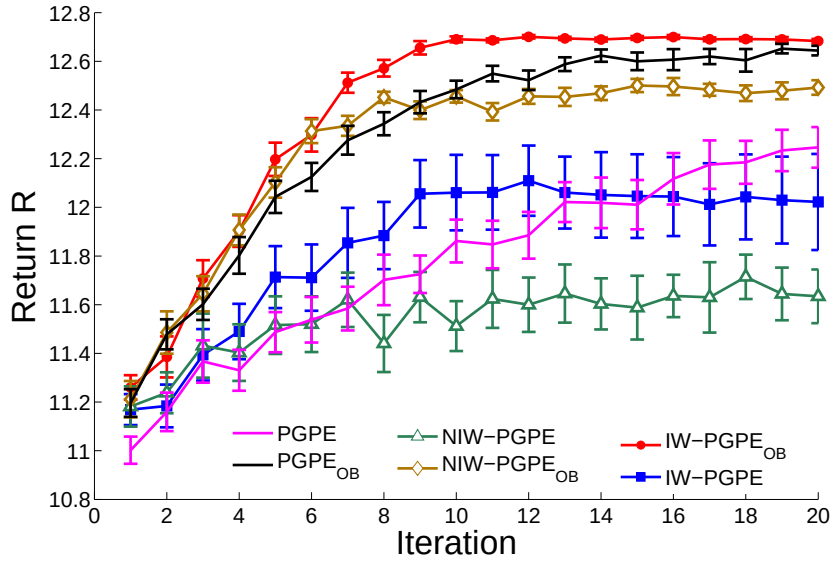


Figure 4.5: Average expected returns through policy update iterations over 20 runs for toy data. Error bars denote standard errors.

- **TIW-eNAC:** Truncated importance-weight episodic natural actor-critic, which is an episodic version of the sample-reuse NAC method (Wawrzynski, 2009; Peters and Schaal, 2008). Following the same line as Wawrzynski (2009), we truncate the importance weight as $w = \min\{w, 2\}$.

- **IW-REINFORCE$_{\mathbf{OB}}$:** Importance-weighted REINFORCE with the optimal baseline, which is basically a combination of the off-policy implementation of the episodic REINFORCE method (Meuleau et al., 2001) and the optimal baseline (Peters and Schaal, 2006), although we could not exactly find this method in literature.

- **R$^{\mathbf{3}}$:** Reward-weighted regression with sample reuse (Hachiya et al., 2011).

- **PGPE$_{\mathbf{OB}}$:** Plain PGPE$_{\mathrm{OB}}$ without data reuse.

- **NIW-PGPE$_{\mathbf{OB}}$:** Data-reuse PGPE$_{\mathrm{OB}}$ without importance weighting.

- **IW-PGPE:** Importance-weighted PGPE.

- **IW-PGPE$_{\mathrm{OB}}$:** Importance-weighted PGPE with the optimal baseline.

The state space $\mathcal{S}$ is two-dimensional and continuous, which consists of the horizontal position $x[m] \in [-1.2, 0.5]$ and the velocity $\dot{x}[m/s] \in [-1.5, 1.5]$, i.e., $\boldsymbol{s} = (x, \dot{x})^\top$. This is non-linearly transformed to a feature space via a basis function vector $\boldsymbol{\phi}(\boldsymbol{s})$. We use 12 Gaussian kernels with mean $\boldsymbol{c}$ and standard deviation $\kappa = 1$ as the basis functions,

$$\boldsymbol{\phi}(\boldsymbol{s}) = \exp\left(-\frac{\|\boldsymbol{s} - \boldsymbol{c}\|^2}{2\kappa^2}\right),$$

where the kernel centers $\boldsymbol{c}$ are distributed over the following grid points:

$$\{-1.2, -0.35, 0.5\} \times \{-1.5, -0.5, 0.5, 1.5\}.$$

The action space $\mathcal{A}$ is one-dimensional and continuous, which corresponds to the force applied to the car (note that the force of the car is not strong enough to climb up the slope to directly reach the goal). We use the Gaussian policy model for IW-REINFORCE$_{\mathrm{OB}}$, TIW-eNAC, and R^3:

$$\pi(a|\boldsymbol{s}, \boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a - \boldsymbol{\mu}^\top \boldsymbol{\phi}(\boldsymbol{s}))^2}{2\sigma^2}\right), \tag{4.3}$$

where $\boldsymbol{\mu}$ is the mean policy parameter and σ is the deviation policy parameter. We employ a linear deterministic policy model (2.3) for the PGPE methods, which corresponds to Eq.(4.3) with $\sigma \to 0$.

The dynamics of the car (i.e., the update rules of the position and the velocity) are given by

$$\begin{aligned} x_{t+1} &= x_t + \dot{x}_{t+1}\Delta t, \\ \dot{x}_{t+1} &= \dot{x}_t + (-9.8w\cos(3x_t) + \frac{a_t}{w} - k\dot{x}_t)\Delta t, \end{aligned}$$

where a_t is the action taken at time t. We set the problem parameters as follows: The mass of the car $w = 0.2$[kg], the friction coefficient $k = 0.3$, and the simulation time step $\Delta t = 0.1$[s]. The reward function is defined as

$$r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \begin{cases} 1 & \text{if } x_{t+1} \geq 0.45, \\ -1 & \text{otherwise.} \end{cases}$$

The initial mean parameter $\boldsymbol{\eta}$ is chosen randomly from the standard normal distribution, and the initial deviation parameter is set at $\tau = 1$. The initial state of the car is set at the bottom of the mountain with the velocity $\dot{x} = 0$. The agent collects $N = 10$ episodic samples with trajectory length $T = 40$ at each iteration. In the data reuse methods, we reuses all previous data at later iterations. In the plain PGPE$_{\mathrm{OB}}$ method, we just use $N = 10$ on-policy samples at each iteration to estimate policy gradients. The discount factor is set at $\gamma = 0.95$. The learning rate is $\varepsilon = 1/\|\nabla_{\rho}\hat{\mathcal{J}}(\boldsymbol{\rho})\|$.

We investigate average expected returns over 10 trials as functions of policy-update iterations. The expected return at each trial is computed over 100 newly-drawn test episodic samples (which are not used for policy learning). The experimental results are plotted in Figure 4.7. This shows that IW-PGPE$_{\mathrm{OB}}$ improves the performance very fast over policy-update iterations, and it achieves superior performance improvement than all other methods. IW-PGPE can also improve the performance over iterations well, implying that the consistency of the IW estimator is useful in this task. However, it is outperformed by the proposed IW-PGPE$_{\mathrm{OB}}$, perhaps because the estimation variance in IW-PGPE is large. NIW-PGPE$_{\mathrm{OB}}$ performs fairly well, which maybe because the bias of policy gradient estimators is not that crucial in this experiment. The plain PGPE$_{\mathrm{OB}}$ can improve the performance throughout the iterations, which indicates that $N = 10$ on-policy samples is enough for this mountain-car task. Other data-reuse methods can improve the performance over iterations, but slowly, and they are outperformed by the compared PGPE methods. IW-REINFORCE$_{\mathrm{OB}}$ outperforms TIW-eNAC, which maybe because the optimal constant baseline contributes significantly in IW-REINFORCE$_{\mathrm{OB}}$ and truncating the importance weights can lead to a larger bias over iterations in TIW-eNAC. R^3 can not improve the performance over iterations. Overall, thanks to the low variance, IW-PGPE$_{\mathrm{OB}}$ achieves smooth and fast policy improvement throughout iterations, and its performance is the best among the compared methods.

4.3.3 Upper-body Humanoid Control

Finally, we evaluate the performance of our proposed method on a highly nonlinear dynamic control problem of the simulated upper-body model of the humanoid

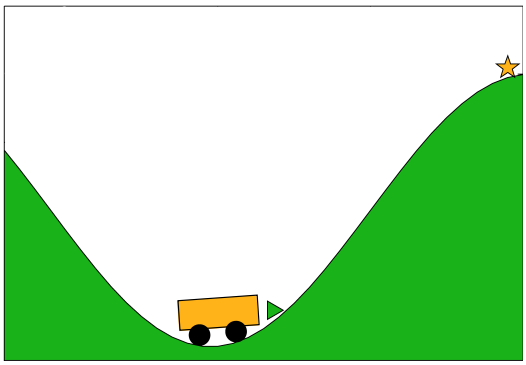


Figure 4.6: Mountain car.

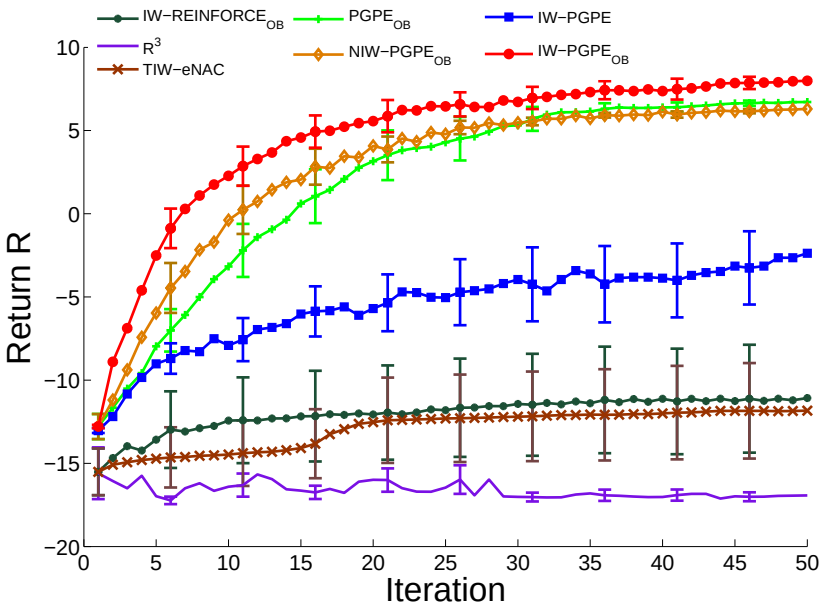
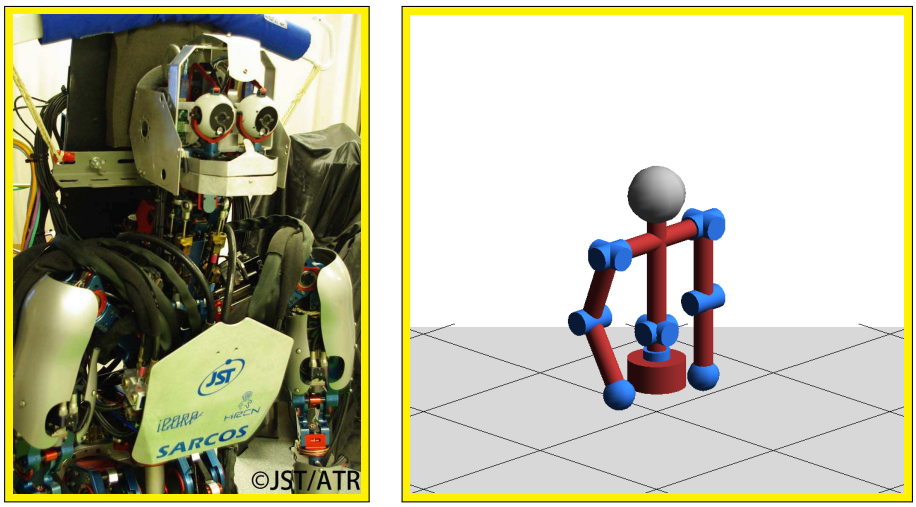


Figure 4.7: Average expected returns over 10 runs as functions of the number of iterations for the mountain-car task. Error bars are standard errors.



(a) CB-i (b) Simulated upper-body model

Figure 4.8: Humanoid robot CB-i and its upper-body model.

robot *CB-i* (Cheng et al., 2007) (see Figure 4.8(a)). We use its simulator in our experiments (see Figure 4.8(b)). The goal is to lead the end-effector of the right arm (right hand) to a target object.

Setup

We compare the performance of the following 4 methods:

- **IW-REINFORCE$_{\mathbf{OB}}$:** Importance-weighted REINFORCE with the optimal baseline.
- **NIW-PGPE$_{\mathbf{OB}}$:** Data-reuse PGPE$_{\mathrm{OB}}$ without importance weighting.
- **PGPE$_{\mathbf{OB}}$:** Plain PGPE$_{\mathrm{OB}}$ without data reuse.
- **IW-PGPE$_{\mathbf{OB}}$:** Importance-weighted PGPE with the optimal baseline.

The simulation is based on the upper body of the CB-i humanoid robot illustrated in Figure 4.8(b), which has 9 degrees of freedom corresponding to main joints of the upper body: The shoulder pitch, shoulder roll, elbow pitch of the right arm, shoulder pitch, shoulder roll, elbow pitch of the left arm, waist yaw, torso roll, and torso pitch.

At each time step, the controller receives states from the system and sends out actions. The state space is 18-dimensional, which corresponds to the current angle and the current angular velocity of each joint. The action space is 9-dimensional, which corresponds to the target angle of each joint. Both states and actions are continuous.

The initial positions of the robot and an object are fixed, where the initial position of the robot is set at the state of standing up straight with the arms down, and the position of the target object depends on the task. Note that the position of the target object is only used in the designing of the reward function. The reward function is given by

$$r_t = k_1 \exp(-10 d_t) - k_2 \min\{c_t, 10000\},$$

where $k_1 = 1$, $k_2 = 0.0005$, d_t is the distance between the robot's right hand and the target object at the time step t, and c_t is the sum of control costs for each joint. Note that the results may change with different k_1 and k_2 for the reward function. In order to keep the value of $\exp(-10 d_t)$ and c_t in the reward function to the same order of magnitude, we need to choose k_1 and k_2 reasonably. We use the same policy model as the mountain car experiment, i.e., the linear deterministic policy for PGPE and the Gaussian policy for IW-REINFORCE$_{\mathrm{OB}}$ with the basis function $\phi(\boldsymbol{s}) = \boldsymbol{s}$.

The initial mean parameter η is randomly chosen from the standard normal distribution, and the initial standard deviation parameter τ is set to 1. To evaluate the usefulness of the data reuse methods with a small number of samples, the agent collects only $N = 3$ on-policy samples with trajectory length $T = 100$ at each iteration. In the data reuse methods, we reuse all previous data at later iterations. In the plain PGPE$_{\mathrm{OB}}$, we just use the on-policy samples to estimate the gradients. The discount factor is set at $\gamma = 0.9$, and the learning rate is set at $\varepsilon = 0.1/\|\nabla_\rho \hat{\mathcal{J}}(\boldsymbol{\rho})\|$.

Reaching Task with 2 Degrees of Freedom

First, we investigate the performance on the reaching task with only 2 degrees of freedom. We fix the body of the robot and use only the right shoulder pitch and right elbow pitch. Figure 4.9 depicts the averaged expected return over 10

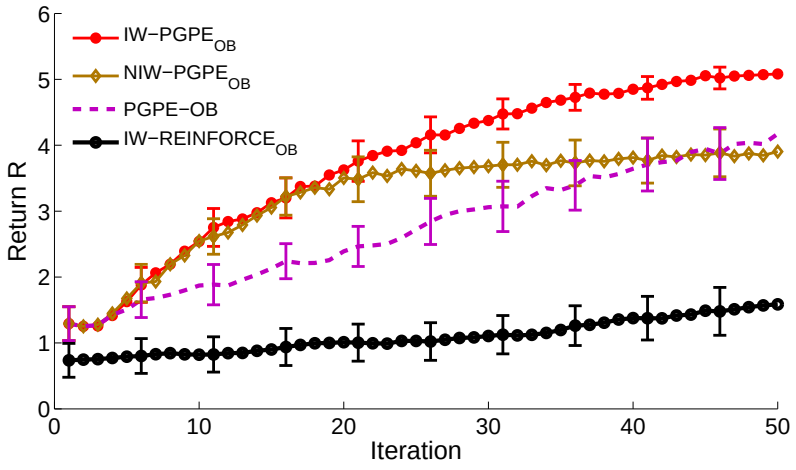
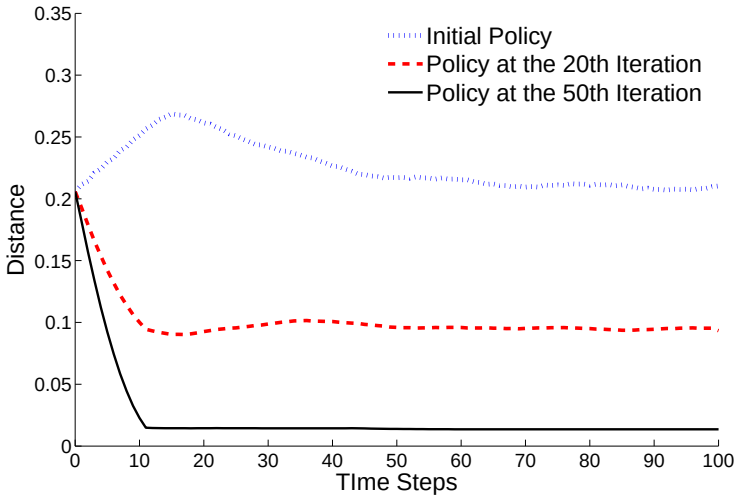


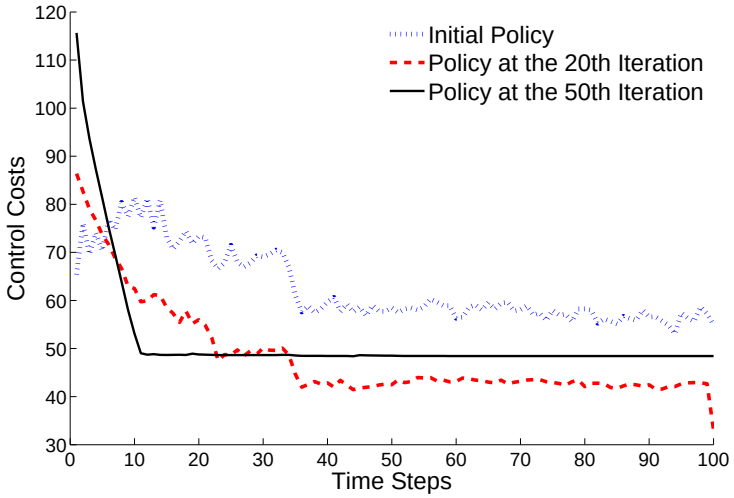
Figure 4.9: Average expected returns over 10 runs as functions of the number of iterations for the reaching task with 2 degrees of freedom (right shoulder pitch and right elbow pitch).

trials as a function of the number of iterations. The expected return at each trial is computed from 50 newly-drawn test episodic data (which are not used for policy learning). The graph shows that IW-PGPE$_{\mathrm{OB}}$ nicely improves the performance over iterations only with a small number of on-policy samples. The plain PGPE$_{\mathrm{OB}}$ can also improve the performance over iterations, but slowly. NIW-PGPE$_{\mathrm{OB}}$ is not as good as IW-PGPE$_{\mathrm{OB}}$ especially at the later iterations, which is because of the inconsistent property of the NIW estimator. The initial mean parameter is randomly chosen in this experiment, which makes IW-REINFORCE$_{\mathrm{OB}}$ not able to improve the performance significantly over iterations. This result is consistent with the observation that the REINFORCE method is sensitive to the initial parameter values (Zhao et al., 2012).

The distance from the right hand to the object and the control costs along the trajectory are also investigated. We test the initial policy, the policy obtained at the 20th iteration by IW-PGPE$_{\mathrm{OB}}$, and the policy obtained at the 50th iteration by IW-PGPE$_{\mathrm{OB}}$. The results are shown in Figure 4.10. From Figure 4.10(a), it is clear to see that the policy obtained at the 50th iteration decreases the distance fastest compared with the initial policy and the policy obtained at the 20th iteration. This



(a) Distance



(b) Control costs

Figure 4.10: Distance and control costs of arm reaching with 2 degrees of freedom using the policy learned by IW-PGPE$_{\text{OB}}$.

means the robot can reach the object fast by using the learned policy. On the other hand, Figure 4.10(b) shows that the control cost required for executing the policy obtained at the 50th iteration decreases steadily until the reaching task is completed. This is because the robot mainly adjusts the shoulder pitch in the beginning, which consumes a larger amount of energy than the energy required for controlling the elbow pitch. Then, once the right hand gets closer to the target object, the robot starts to adjust the elbow pitch reach the target object. The policy obtained at the 20th iteration actually consumes less control costs, but it cannot move the arm to the target object.

Figure 4.11 shows a typical solution of the reaching task with 2 degrees of freedom by IW-PGPE$_{\mathrm{OB}}$ (with the policy obtained at the 50th iteration). The images show that the policy learned by our proposed method successfully leads the right hand to the target object within only 10 time steps.

Reaching Task with 4 Degrees of Freedom

Next, we evaluate the performance on the reaching task with 4 degrees of freedom. We use the right shoulder pitch, right elbow pitch, right shoulder roll, and torso yaw joint. By using the torso yaw joint, the robot can reach a distant object which can not be achieved by only using the right arm. The results are shown in Figure 4.12. The graph shows that IW-PGPE$_{\mathrm{OB}}$ achieves fast policy improvement throughout iterations, and the performance is the best among the compared methods.

Figure 4.13 depicts a representative example of object reaching with 4 degrees of freedom by IW-PGPE$_{\mathrm{OB}}$. Note that the object is distant from the robot and it can not be reached by only using the right arm. The robot first adjusts the torso yaw joint, and then uses the right arm to reach the object. The images show that the policy learned by our proposed method successfully leads the right hand to the distant object.

Reaching Task with All Degrees of Freedom

At last, we evaluate the performance on the reaching task with all degrees of freedom. The position of the target object is the same as the task in the 4-degrees-of-freedom setting.

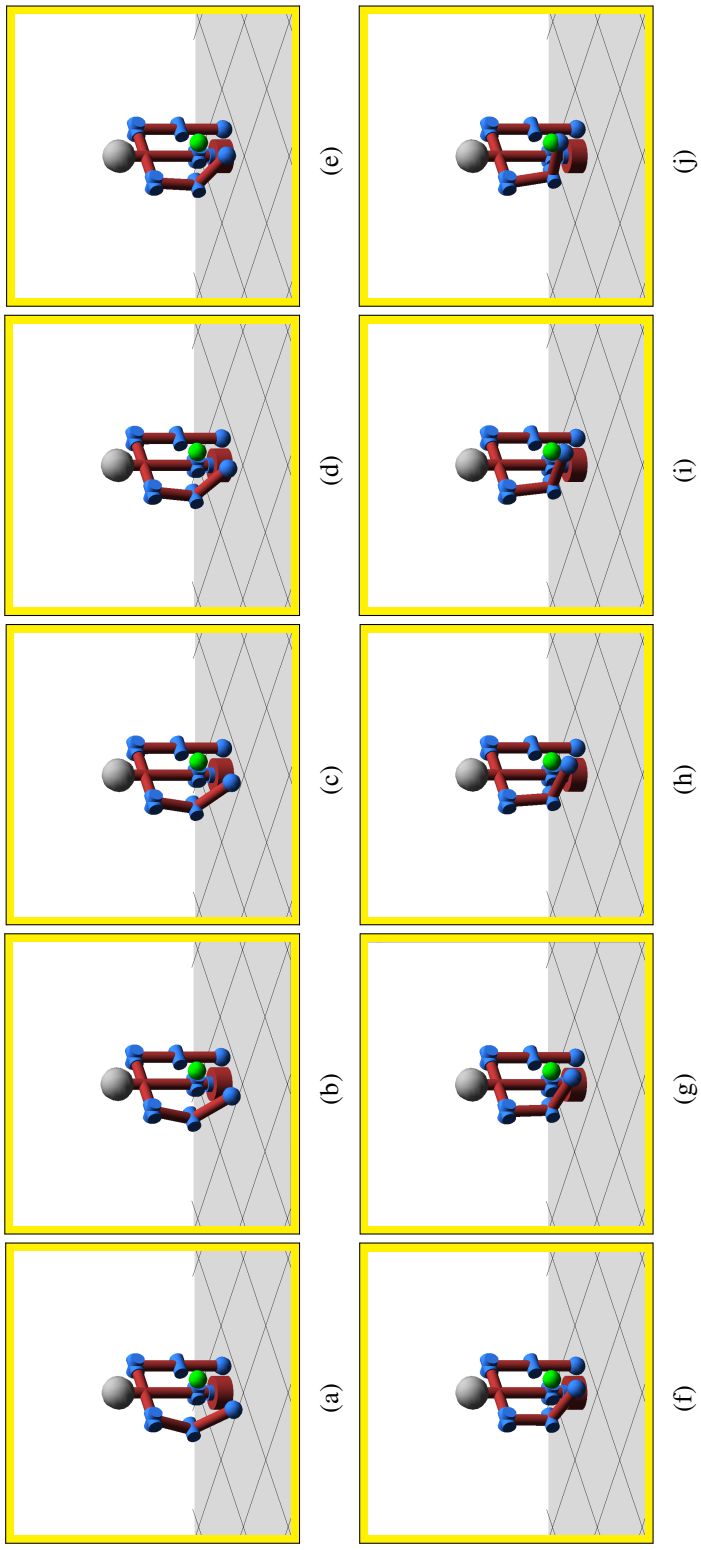


Figure 4.11: Typical example of arm reaching with 2 degrees of freedom using the policy obtained by IW-PGPE$_{\text{OB}}$ at the 50th iteration.

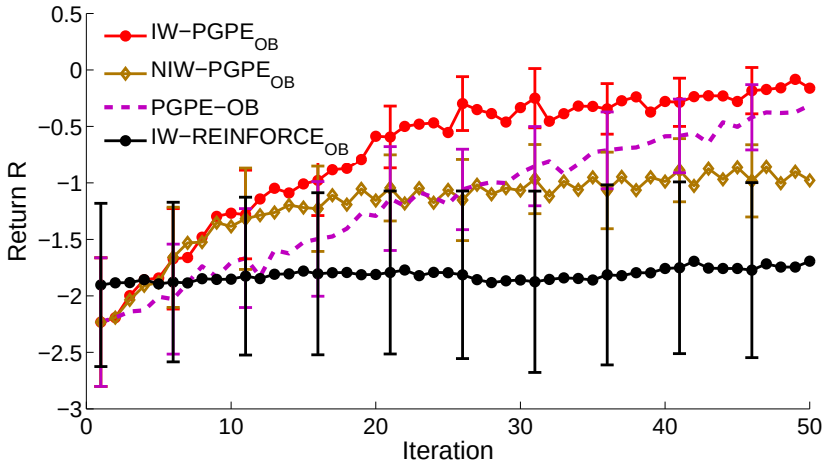


Figure 4.12: Average expected returns over 10 runs as functions of the number of iterations for the reaching task with 4 degrees of freedom (right shoulder pitch, right elbow pitch, right shoulder roll, and torso yaw joint).

In this experiment, we use all degrees of freedom to reach the object. This increases the dimensionality of the state space, which actually may grow the values of importance weights exponentially (Shimodaira, 2000; Cortes et al., 2010). In order to mitigate the large values of importance weights, we decided not to reuse all previously collected samples, but only samples collected in the last 5 iterations. This allows us to keep the difference between the sampling distribution and the target distribution reasonably small, and thus the values of importance weights can be suppressed to some extent. Furthermore, following Wawrzynski (2009), we truncate the importance weights as $w = \min\{w, 2\}$. This version of IW-PGPE$_{\mathrm{OB}}$ is denoted as Truncated IW-PGPE$_{\mathrm{OB}}$ below.

The results are shown in Figure 4.14. The graph shows that the performance of Truncated IW-PGPE$_{\mathrm{OB}}$ is the best, which implies that the truncation of importance weights is helpful when applying our proposed method to high-dimensional problems.

Through all the arm-reaching experiments, we can see that the returns tend to be lower as the dimension is increased, even though we run the higher-dimensional

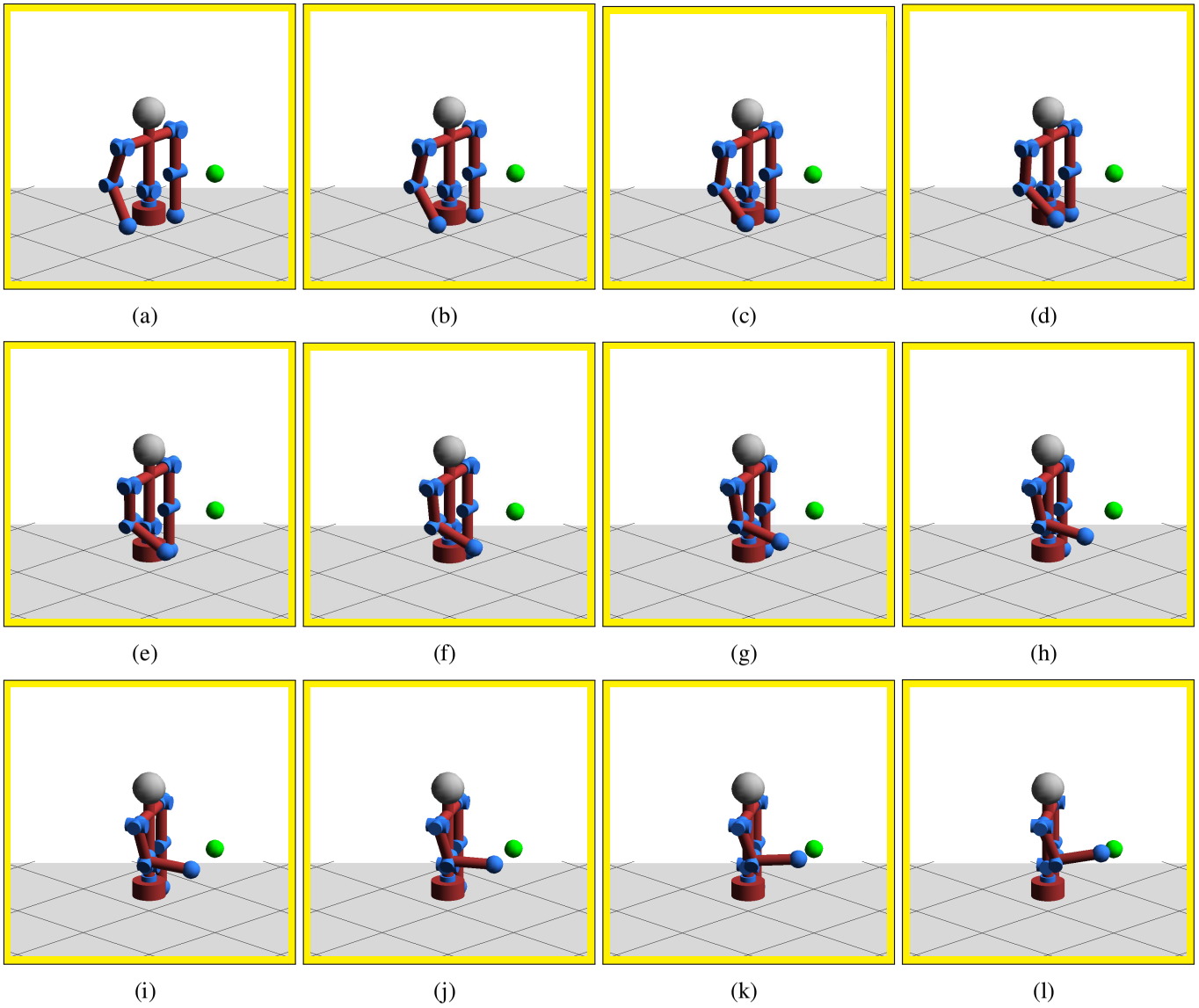


Figure 4.13: Typical example of arm reaching with 4 degrees of freedom using the policy obtained by IW-PGPE$_{\text{OB}}$ at the 50th iteration.

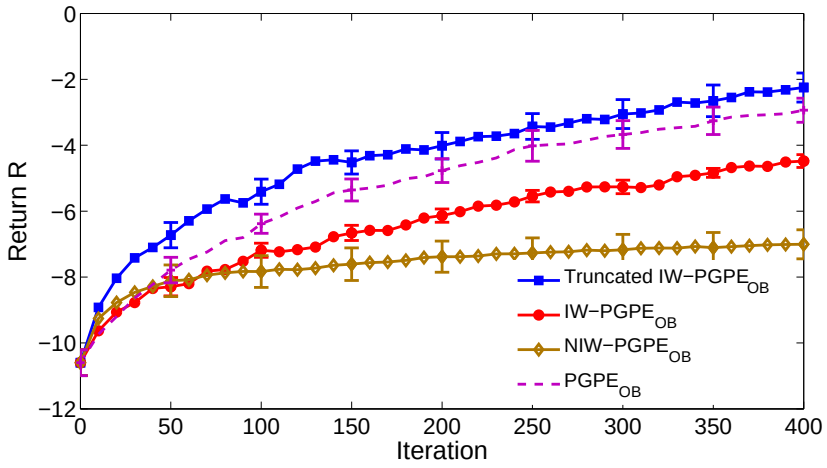


Figure 4.14: Average expected returns over 10 runs as functions of the number of iterations for the reaching task with all degrees of freedom.

experiment for a larger number of iterations. In the task with all degrees of freedom (Figure 4.14), the largest number of iteration is 400. If we continue the experiment for more iterations, the returns may slightly increase, but are still less than the returns in the low-dimensional experiments. This is because the more joints the robot uses, the larger energy will be consumed, and thus the returns tend to be lower in high-dimensional cases.

Overall, the proposed IW-PGPE$_{\mathrm{OB}}$ is shown to be a promising method, although in the last experiment it is obvious that just like other importance weight-based methods, the performance degrades in high-dimensional problems without the use of additional correction techniques such as weight truncation.

4.4 Proofs of Theoretical Results

In the section, we give proofs of all the theorems in this chapter.

Figure 4.15: Typical example of arm reaching with all degrees of freedom using the policy obtained by Truncated IW-PGPE$_{\text{OB}}$ at the 400th iteration.

4.4.1 Proof of Theorem 4.1

Proof. Due to the fact that the sampled data $\{(\boldsymbol{\theta}'_n, h'_n)\}_{n=1}^{N'}$ are independent and identically distributed, we have

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] = \frac{1}{N'}\mathbf{Var}\left[w(\boldsymbol{\theta})\nabla_{\boldsymbol{\eta}}\log p(\boldsymbol{\theta}|\boldsymbol{\rho})R(h)\right], \tag{4.4}$$

where h and $\boldsymbol{\theta}$ are random variables and follow the distributions $p(h, \boldsymbol{\theta}|\boldsymbol{\rho}')$.

Note that we consider the trace of the covariance matrix of gradient vectors, that is, the sum of the variance of the components of the vector. Then by upper-bounding the variance with the second moment, we have the following upper bound:

$$\begin{aligned}
&\mathbf{Var}\left[w(\boldsymbol{\theta})R(h)\nabla_{\boldsymbol{\eta}}\log p(\boldsymbol{\theta}|\boldsymbol{\rho})\right] \\
&\leq \sum_{i=1}^{\ell}\mathbb{E}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho}')}\left[(w(\boldsymbol{\theta})R(h)\nabla_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2\right] \\
&= \sum_{i=1}^{\ell}\iint p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho}')\left(\frac{p(\boldsymbol{\theta}|\boldsymbol{\rho})}{p(\boldsymbol{\theta}|\boldsymbol{\rho}')}\right)^2(R(h))^2(\nabla_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2\mathrm{d}h\mathrm{d}\boldsymbol{\theta} \\
&= \sum_{i=1}^{\ell}\iint p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho})w(\boldsymbol{\theta})(R(h))^2(\nabla_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2\mathrm{d}h\mathrm{d}\boldsymbol{\theta} \\
&\leq \sum_{i=1}^{\ell}\left(\frac{\beta(1-\gamma^T)}{1-\gamma}\right)^2 w_{\max}\iint p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho})(\nabla_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2\mathrm{d}h\mathrm{d}\boldsymbol{\theta} \\
&= \sum_{i=1}^{\ell}\left(\frac{\beta(1-\gamma^T)}{1-\gamma}\right)^2 w_{\max}\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{\rho})}\left[(\nabla_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2\right],
\end{aligned}$$

where $\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{\rho})}[\cdot]$ denotes the expectation of the function of random variable $\boldsymbol{\theta}$ with respect to $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\boldsymbol{\rho})$. Subsequently, given the proof of the first part in Section 3.5.1, we get the upper bound of $\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right]$.

Similarly, given the same technique and the proof of the later part in Section 3.5.1, we could get the conclusion of the upper bound of $\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right]$.

□

4.4.2 Proof of Theorem 4.2

Proof. First, let us derive some elementary expressions. Let $\boldsymbol{A}$, $\boldsymbol{C}$ be random variables taking values in the ℓ-dimensional space and let b be a scalar. Then,

$$\mathbf{Var}[\boldsymbol{A} - b\boldsymbol{C}] = \mathbf{Var}[\boldsymbol{A}] + b^2\,\mathbf{Var}[\boldsymbol{C}] - b\,\mathbf{Cov}[\boldsymbol{A}, \boldsymbol{C}] - b\,\mathbf{Cov}[\boldsymbol{C}, \boldsymbol{A}].$$

We still consider the trace of the covariance matrix of gradient vectors for multi-dimensional space. Assume that $\mathbb{E}[\boldsymbol{C}] = \mathbf{0}$. Then, we could have

$$\begin{aligned}
\mathbf{Var}[\boldsymbol{A} - b\boldsymbol{C}] &= \mathbf{Var}[\boldsymbol{A}] + b^2\,\mathbf{Var}[\boldsymbol{C}] - 2b\,\mathbf{Cov}[\boldsymbol{A}, \boldsymbol{C}] \\
&= \mathbf{Var}[\boldsymbol{A}] + \mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}] \left\{ b^2 - 2b \frac{\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}]}{\mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}]} \right\} \qquad (4.5) \\
&= \mathbf{Var}[\boldsymbol{A}] + \mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}] \left\{ \left(b - \frac{\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}]}{\mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}]} \right)^2 - \left(\frac{\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}]}{\mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}]} \right)^2 \right\}.
\end{aligned}$$

Simple calculus shows that the foregoing is minimized when

$$b = \frac{\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}]}{\mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}]}.$$

The optimal baseline for IW-PGPE follows immediately by plugging in

$$\boldsymbol{A} = Rw\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})$$

and

$$\boldsymbol{C} = w\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})$$

for $\boldsymbol{A}$ and $\boldsymbol{C}$. Note that Eq.(4.5) uses the conclusion of $\mathbb{E}[w\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})] = \mathbf{0}$, which can be found in Section 3.5.4.

As the sampled data are independent and identically distributed, we have

$$\mathbf{Var}[\nabla_{\boldsymbol{\rho}} \hat{\mathcal{J}}^b_{\mathrm{IW}}(\boldsymbol{\rho})] = \frac{1}{N'}\,\mathbf{Var}[\boldsymbol{A} - b\boldsymbol{C}].$$

Then, according to Eq.(4.5) and the definition of b^*, we could have

$$\begin{aligned}
&\mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^b_{\mathrm{IW}}(\boldsymbol{\rho})] - \mathbf{Var}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})] \\
&= \frac{1}{N'} \left(b^2 \mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}] - 2b\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}] + \frac{(\mathbb{E}[\boldsymbol{A}^\top \boldsymbol{C}])^2}{\mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}]} \right) \\
&= \frac{1}{N'} (b - b^*)^2\, \mathbb{E}[\boldsymbol{C}^\top \boldsymbol{C}],
\end{aligned}$$

where the expectation is over random variables h and $\boldsymbol{\theta}$ such that $(h, \boldsymbol{\theta}) \sim p(h, \boldsymbol{\theta}|\boldsymbol{\rho}')$. This completes the proof of Theorem 4.2. □

4.4.3 Proof of Theorem 4.3

Proof. We define $\nabla_{\boldsymbol{\eta}}$ and $\nabla_{\boldsymbol{\eta}_i}$ as

$$\begin{aligned}\nabla_{\boldsymbol{\eta}} =& \nabla_{\boldsymbol{\eta}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}),\\ \nabla_{\boldsymbol{\eta}_i} =& \nabla_{\boldsymbol{\eta}_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}).\end{aligned}$$

We still denote the subscripts $\boldsymbol{\rho}'$ as $p(h, \boldsymbol{\theta}|\boldsymbol{\rho}')$. According to Theorem 4.2, by setting $b = 0$, it is easy to know that

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] = \frac{\left(\mathbb{E}_{\boldsymbol{\rho}'}[R(h)w^2(\boldsymbol{\theta})\nabla^\top_{\boldsymbol{\eta}}\nabla_{\boldsymbol{\eta}}]\right)^2}{N'\mathbb{E}_{\boldsymbol{\rho}'}[w^2(\boldsymbol{\theta})\nabla^\top_{\boldsymbol{\eta}}\nabla_{\boldsymbol{\eta}}]}.$$

We already know that

$$\mathbb{E}_{\boldsymbol{\rho}'}[R(h)w^2(\boldsymbol{\theta})\nabla^\top_{\boldsymbol{\eta}}\nabla_{\boldsymbol{\eta}}] \leq \frac{\beta(1-\gamma^T)}{(1-\gamma)}\mathbb{E}_{\boldsymbol{\rho}'}\left[w^2(\boldsymbol{\theta})\nabla^\top_{\boldsymbol{\eta}}\nabla_{\boldsymbol{\eta}}\right].$$

Hence,

$$\begin{aligned}&\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right]\\ &\leq \frac{\beta^2(1-\gamma^T)^2}{N'(1-\gamma)^2}\mathbb{E}_{\boldsymbol{\rho}'}\left[w^2(\boldsymbol{\theta})\nabla^\top_{\boldsymbol{\eta}}\nabla_{\boldsymbol{\eta}}\right]\\ &\leq \frac{\beta^2(1-\gamma^T)^2}{N'(1-\gamma)^2}w_{\max}\sum_{i=1}^{\ell}\mathbb{E}_{p(\boldsymbol{\theta}|\boldsymbol{\rho})}\left[(\nabla_{\eta_i})^2\right] \qquad (4.6)\\ &= \frac{\beta^2(1-\gamma^T)^2B}{N'(1-\gamma)^2}w_{\max}, \qquad (4.7)\end{aligned}$$

where Eq.(4.6) is based on the same technique used in Section 4.4.1, and Eq.(4.7) is given by results of the proof in Section 3.5.1.

Similarly, we can have the lower bound as

$$\mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\eta}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] \geq \frac{\alpha^2(1-\gamma^T)^2B}{N'(1-\gamma)^2}w_{\min}.$$

By using the same techniques, we get the bounds of the variance reduction of gradient estimation with respect to the deviation parameter $\boldsymbol{\tau}$,

$$\begin{aligned}\mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] &\leq \frac{2\beta^2(1-\gamma^T)^2B}{N'(1-\gamma)^2}w_{\max},\\ \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{\mathcal{J}}_{\mathrm{IW}}(\boldsymbol{\rho})\right] - \mathbf{Var}\left[\nabla_{\boldsymbol{\tau}}\widehat{\mathcal{J}}^{b^*}_{\mathrm{IW}}(\boldsymbol{\rho})\right] &\geq \frac{2\alpha^2(1-\gamma^T)^2B}{N'(1-\gamma)^2}w_{\min},\end{aligned}$$

which completes the proof. □

4.5 Summary and Discussions

In many real-world reinforcement learning problems, reducing the number of training samples is desirable because the sampling cost is often much higher than the computational cost. In this chapter, we proposed a new policy gradient method equipped with efficient sample reuse, which systematically combines a reliable policy gradient method, PGPE, with importance sampling and the optimal constant baseline. We showed that the introduction of the optimal constant baseline can mitigate the large-variance problem of importance weighting under some conditions. Through experiments with an artificial domain, the usefulness of the proposed method was demonstrated. More over, through robotic experiments, we found that the truncation technique was helpful when applying the proposed method to high-dimensional problems.

The baseline and importance weighting techniques are two independent techniques. More specifically, importance weighting is used in the off-policy scenario to efficiently reuse previously collected samples, by using importance weighting the consistency between the data sampling distribution and the target distribution is kept. On the other hand, the optimal constant baseline is used to reduce the variance of gradient estimates.

The use of a baseline technique has been first proposed in terms of reinforcement comparison in Sutton (1984), which intuitively means the comparison between the expected return R and the baseline b: If $R > b$ we adjust learned parameters ρ so as to increase the probability of θ, and, if $R < b$, we do the opposite. Based on this idea, Williams (Williams, 1988) demonstrated that a baseline technique did not introduce bias, which is because the expectation of the coefficient of b is zero, i.e., $\mathbb{E}\left[\frac{\nabla_\rho p(\theta|\rho)}{p(\theta|\rho)}\right] = 0$. The effect of the baseline on variance is considered in Dayan. The intuition behind the baseline is that subtracting a baseline from the return reduces the magnitude, and thus reduces the variance. Technically, subtracting a baseline can be viewed as a *control variate technique* (Fishman, 1996), which is an effective approach to reducing variance of Monte Carlo estimates of integrals. The experimental results in this chapter suggest that the removal of the baseline is possibly the primary factor in improving performance compared with the importance weighting techniques.

Chapter 5

Conclusions and Future Works

In this chapter, we summarize the major contributions of this thesis and discuss some possible future work.

5.1 Conclusions

Reinforcement learning enables a robot to autonomously discover an optimal behavior in the unknown environment. This thesis was devoted to developing statistical algorithms in reinforcement learning. There are two major paradigms in reinforcement learning: *policy iteration* and *policy search*. Policy search can handle continuous state and actions naturally, it is very suitable for solving the robot control tasks. However, there are some practical problems when existing methods are applied to robot control. To improve the performance of policy search algorithms, this thesis contributed to provide more practical and efficient algorithms with theoretical guarantees. The contributions in this thesis are summarized as follows:

- In Chapter 3, we analyzed and improved the stability of the policy gradient method called PGPE (policy gradients with parameter-based exploration). We theoretically showed that, under a mild condition, PGPE provides more stable gradient estimates than the classical REINFORCE method. We also derived the optimal baseline for PGPE, and theoretically showed that PGPE with the optimal baseline is more preferable than REINFORCE with the

optimal baseline in terms of the variance of gradient estimates. Finally, we demonstrated the usefulness of PGPE with optimal baseline through experiments. We also experimentally showed that the use of symmetric sampling further improves the performance.

- In many real-world reinforcement learning problems, reducing the number of training samples is desirable because the sampling cost is often much higher than the computational cost. In Chapter 4, we proposed a new policy gradient method equipped with efficient sample reuse, which systematically combines a reliable policy gradient method, PGPE, with importance sampling and the optimal constant baseline. We showed that the introduction of the optimal constant baseline can mitigate the large-variance problem of importance weighting under some conditions. Through experiments with an artificial domain, the usefulness of the proposed method was demonstrated. More over, through robotic experiments, we found that the truncation technique was helpful when applying the proposed method to high-dimensional problems.

5.2 Future Works

The algorithms proposed in this thesis can mitigate the problems of algorithms of reinforcement learning to some extent. However, there are still some issues need to be considered when designing the learning algorithms. In this section, some important potential future directions are discussed as follows:

Trade-off between Exploration and Exploitation One of the challenging issues to be discussed in the reinforcement learning field is the trade-off between exploration and exploitation. The agent usually goes through the same environment many times in order to learn how to find the optimal actions. The optimal actions are with high rewards, in order to obtain high rewards, the agent must prefer actions that it has tried in the past and found to be with high reward. But in order to discover the such actions, it has to try actions that has not tried before. Thus, the agent has to *exploit* what it already knows, and also has to *explore* in

order to make better action selections in the future. Balancing exploration and exploitation is particularly important: the agent may have found a good goal on one path, but there may be an even better one on another path. Without exploration, the agent will always return to first goal, and the better goal will not be found. Or, the goal may lie behind very low reward areas, that the agent would avoid without exploration. On the other hand, if the agent explores too much, it cannot stick to a path; in fact, it is not really learning: it cannot exploit its knowledge, and so acts as though it knows nothing. Thus, it is important to find a good balance between the two, to ensure that the agent is really learning to take the optimal actions.

PGPE is not an exception since choosing similar policy parameters many times and collecting data is not efficient especially when data collection is expensive and time consuming. Therefore, in our future work, we will investigate the trade-off between exploration and exploitation in the framework of PGPE.

Partially Observable Markov Decision Processes A key challenge in learning for robots is to deal with sensory information, since sensor data is typically noisy. In such case, the learning systems always use filters to help estimating the true state, and it is essential to maintain the information state of the environment that includes the raw observations and the uncertainty of its estimates.

Moreover, it is often unrealistic to assume that the state is completely observable. The learning system will not be able to know precisely in which state it is. Thus, it is natural to consider *partially observable* settings, i.e., partially observable markov decision processes (POMDP). The term *partially* indicates that the state of the world can not be sensed directly or fully. Instead, the measurements received by the robot are incomplete and usually noisy projection of the state. In such case, the robot has to estimate its belief state in terms of the posterior distribution over possible world states.

Till now, we are considering algorithms in the MDP framework. An important future direction is to formulate the problems in the framework of POMDPs.

Real Time Requirements in Robotics Delay in sensing states happens frequently in the robot's physical systems due to processing and communication delays. Due to these delays, actions may not have instantaneous effects. However,

most RL algorithms assume that the actions are taken effect instantaneously, because the delays would violate the Markov properties. Thus, the algorithm must be capable of dealing with delays in sensing and execution that are inherent in physical system.

The robots require that the algorithm runs in real time. Therefore, considering the real time in algorithms of reinforcement learning is an important future direction.

Inverse Reinforcement Learning A common challenge of applying RL to robotics is the generation of appropriate reward functions. The reward function provides the most succinct and transferable definition of the task. Specifying good reward functions in robotics requires experienced domain knowledge and may often be tough in practice. The reward function is usually designed by the human engineer, however, the engineer may have only a very rough idea of the reward function whose optimization would generate a desirable behavior. In such case, it is not straightforward to use RL.

Inverse reinforcement learning (IRL) is a useful method to solve this reward function problem (Ng and Russell, 2000). More specifically, the reward function is learned from the behaviors of an expert, which is assumed to be given by an optimal policy. Once the reward function is obtained, the RL algorithms can be applied to learn the policy. This would be a promising future direction.

Analysis and Improvement of the Natural PGPE Recently, the combination of parameter-based exploration and natural policy gradient has been proposed to speed up the policy gradient methods (Miyamae et al., 2010). We will extend the current theoretical analysis so that the above *natural PGPE* method can also be analyzed. Furthermore, efficient sample reuse in natural PGPE is also worth to investigate.

Bibliography

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *In Advances in Neural Information Processing Systems 19*. MIT Press, 2007.

N. Abe, P. Melville, C. Pendus, C. K. Reddy, D. L. Jensen, V. P. Thomas, J. J. Bennett, G. F. Anderson, B. R. Cooley, M. Kowalczyk, M. Domick, and T. Gardinier. Optimizing debt collections using constrained reinforcement learning. In *Proceedings of the 16th ACM SGKDD Conference on Knowledge Discovery and Data Mining*, pages 75–84, 2010.

T. Akiyama, H. Hachiya, and M. Sugiyama. Efficient exploration through active learning for value function approximation in reinforcement learning. *Neural Networks*, 23(5):639–648, 2010.

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10 (2):251–276, 1998.

J. Bagnell, S. Kakade, A. Ng, and J. Schneider. Policy search by dynamic programming. In *Advances in Neural Information Processing Systems*, volume 16, pages 831–388. MIT Press, 2004.

J. A. Bagnell and J. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the International Conference on Robotics and Automation 2001*. IEEE, May 2001.

L. C. Baird. Advantage updating. Technical Report WL-TR-93-1146, Wright Lab., 1993.

J. Baxter, P. Bartlett, and L. Weaver. Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381, 2001.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

M. Bugeja. Non-linear swing-up and stabilizing control of an inverted pendulum system. In *Proceedings of IEEE Region 8 EUROCON*, volume 2, pages 437–441, 2003.

L. Buşoniu, R. Babuška, B. De Schutter, and D. Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, Boca Raton, Florida, 2010.

G. Cheng, S. Hyon, J. Morimoto, A. Ude, G.H. Joshua, G. Colvin, W. Scroggin, and C. J. Stephen. Cb: A humanoid research platform for exploring neuroscience. *Advanced Robotics*, 21(10):1097–1114, 2007.

C. Cortes, Y. Mansour, and M. Mohri. Learning bounds for importance weighting. In *Advances in Neural Information Processing Systems 23*, pages 442–450, 2010.

P. Dayan. Reinforcement Comparison. In *Proceedings of the 1990 Connectionist Models Summer School*, San Mateo, CA. Morgan Kaufmann.

P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278, 1997.

M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning*, pages 465–473, 2011.

M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.

G. S. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications*. Springer-Verlag, Berlin, Germany, 1996.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

E. Greensmith, P. L. Bartlett, and J. Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

H. Hachiya, T. Akiyama, M. Sugiyama, and J. Peters. Adaptive importance sampling for value function approximation in off-policy reinforcement learning. *Neural Networks*, 22(10):1399–1410, 2009.

H. Hachiya, J. Peters, and M. Sugiyama. Reward weight regression with sample reuse for direct policy search in reinforcement learning. *Neural Computation*, 23(11):2798–2832, 2011.

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. A natural policy gradient. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 1531–1538, Cambridge, MA, 2002. MIT Press.

D. E. Kirk. *Optimal Control Theory: An Introduction*. Dover Publications, 2012.

J. Kober and J. Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1-2):171–203, July 2011.

J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1238 – 1274, July 2013.

J. Koza, K. Martin, S. Matthew, M. William, Y. Jessen, and L. Guido. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers, 2003.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

P. Larrañaga and J. A. Lozano. *Estimation of distribution algorithms: A new tool for evolutionary computation*. Kluwer Academic Publishers, Boston, 2002.

P. Marbach and J. N. Tsitsiklis. Approximate gradient methods in policy-space optimization of Markov reward processes. *Discrete Event Dynamic Systems*, 13 (1-2):111–148, 2004.

T. Matsubara, T. Morimura, and J. Morimoto. Adaptive step-size policy gradients with average reward metric. *Journal of Machine Learning Research - Proceedings Track*, 13:285–298, 2010.

N. Meuleau, L. Peshkin, and K. E. Kim. Exploration in gradient-based reinforcement learning. Technical Report 2001-003, MIT, 2001.

T. Mitchell. The discipline of machine learning. Technical Report CMU ML-06 108, 2006.

A. Miyamae, Y. Nagata, I. Ono, and S. Kobayashi. Natural policy gradient methods with parameter-based exploration for control tasks. In *Advances in Neural Information Processing Systems*, volume 2, pages 437–441, 2010.

T. Morimura, E. Uchibe, and K. Doya. Natural actor-critic with baseline adjustment for variance reduction. *Artificial Life and Robotics*, 13:275–279, 2008.

K. P. Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA, 2012.

A. Ng and M. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 406–415, 2000.

A. Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

A. Y. Ng, H. J. Kim, M. I. Jordan, and S. Sastry. Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems 16 (NIPS 2003)*, 2004.

L. Peshkin and C. R. Shelton. Learning from scarce experience. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 498–505, 2002.

J. Peters and S. Schaal. Policy gradient methods for robotics. In *Proceedings of the IEEE/RSJ International Conferece on Inatelligent Robots and Systems*, pages 2219–2225, 2006.

J. Peters and S. Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *In: Proceedings of the International Conference on Machine Learning (ICML*, pages 745–750, 2007.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

D. L. Poole and A. K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 2010.

D. Precup, R. S. Sutton, and S. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 759–766, 2000.

S. M. Ross. *Introduction to Stochastic Dynamic Programming*. Academic Press, 1983.

T. Rückstiess, F. Sehnke, T. Schaul, D. Wierstra, Y. Sun, and J. Schmidhuber. Exploring parameter space in reinforcement learning. *Paladyn*, 1(1):14–24, 2010.

S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. Learning movement primitives. In *International Symposium on Robotics Research, year = 2004, publisher = Springer*.

D. L. Schacter, D. T. Gilbert, and D. M. Wegner. *Psychology, 2nd edition*. Worth Publishers, 2011.

F. Sehnke, C. Osendorfer, T. Rückstiess, A. Graves, J. Peters, and J. Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

C. R. Shelton. Policy improvement for POMDPs using normalized importance sampling. In *Proceedings of the Seventeenth International Conference on Uncertainty in Artificial Intelligence*, pages 496–503, 2001.

H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90 (2):227–244, 2000.

O. Sigaud and F. Garcia. *Markov Decision Processes in Artificial Intelligence*. Wiley, John & Sons, Incorporated, 2013.

S. P. Singh, T. Jaakkola, and M. I. Jordan. Learning without state-estimation in partially observable markovian decision processes. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 284–292. Morgan Kaufmann, 1994.

M. Sugiyama, H. Hachiya, C. Towell, and S. Vijayakumar. Geodesic Gaussian kernels for value function approximation. *Autonomous Robots*, 25(3):287–304, 2008.

M. Sugiyama, H. Hachiya, H. Kashima, and T. Morimura. Least absolute policy iteration—A robust approach to value function approximation. *IEICE Transactions on Information and Systems*, E93-D(9):2555–2565, 2010.

R. Sutton. *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts, 1984.

R. S. Sutton and G. A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1998.

R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press, 1999.

R.S. Sutton, A.G. Barto, and R. J. Williams. Reinforcement learning is direct adaptive optimal control. *Control Systems, IEEE*, 12(2):19–22, 1992.

C. Szepesvari. *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers, 2010.

G. Tesauro. TD-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. Intelligent robotics and autonomous agents. The MIT Press, 2005.

E. Uchibe and K. Doya. Competitive-cooperative-concurrent reinforcement learning with importance sampling. In *Proceedings of International Conference on Simulation of Adaptive Behavior: From Animals and Animats*, pages 287–296. MIT Press, 2004.

P. Wawrzynski. Real-time reinforcement learning by sequential actor-critics and experience replay. *Neural Networks*, 22:1484–1497, 2009.

L. Weaver and J. Baxter. Reinforcement learning from state and temporal differences. Technical report, Department of Computer Science, Australian National University, 1999.

L. Weaver and N. Tao. The optimal reward baseline for gradient-based reinforcement learning. In *Processings of the Seventeeth Conference on Uncertainty in Artificial Intelligence*, pages 538–545, 2001.

J. D. Williams and S. Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):231–422, 2007.

R. J. Williams. Toward a theory of reinforcement-learning connectionist systems. Technical Report NU-CCS-88-3, College of Computer Science, Northeastern University, Boston, MA, 1988.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

T. Zhao, H. Hachiya, G. Niu, and M. Sugiyama. Analysis and improvement of policy gradient estimation. *Neural Networks*, 26:118–129, 2012.