POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAMME IN INFORMATION TECHNOLOGY

ALGORITHMS FOR RISK-AVERSE REINFORCEMENT LEARNING

Doctoral Dissertation of:
Lorenzo Bisi

Supervisor:
Prof. Marcello Restelli

Tutor:
Prof. Nicola Gatti

The Chair of the Doctoral Program:
Prof. Barbara Pernici

2022 – Cycle XXXIV

Abstract

KEEPING risk under control is a primary concern in many critical real-world domains, including finance and healthcare. The literature on risk-averse reinforcement learning (RL) has mostly focused on designing ad-hoc algorithms for specific risk measures. As such, most of these algorithms do not easily generalize to measures other than the one they are designed for. Furthermore, it is often unclear whether state-of-the-art risk-neutral RL algorithms can be extended to reduce risk. In this dissertation, we take a step towards overcoming these limitations, by following two different paths.

The first one consists in proposing a single framework to optimize some of the most popular risk measures, including conditional value-at-risk, utility functions, and mean-variance. Leveraging theoretical results on state augmentation, we transform the decision-making process so that optimizing the chosen risk measure in the original environment is equivalent to optimizing the expected return in the transformed one. We then present a risk-sensitive meta-algorithm that transforms the trajectories it collects from the environment and feeds these into any risk-neutral policy optimization method.

The second path we follow consists in considering, for the first time, risk-measures connected to the state-action occupancy distribution, instead of the return one. We define a novel measure of risk, which we call reward volatility, consisting of the variance of the rewards under the state-occupancy measure, and we study the optimization of a trade-off objective called mean-volatility. We provide a monotonic improvement theorem for this objective, which allows then to derive a TRPO-like algorithm for risk-averse optimization.

Finally, in order to understand the impact of mean-volatility optimization on sample-complexity, we study the convergence rate of an actor-critic approach optimizing this criterion. Thus, we extend recent analyses in the risk-neutral actor-critic setting to the mean-volatility case, in order to establish the sample-complexity required to attain an ϵ-accurate stationary point.

All contributions are empirically validated with extensive experimental analyses on challenging benchmarks.

Contents

1 Introduction **1**
- 1.1 The Reinforcement Learning Framework . . . 1
- 1.2 Reinforcement Learning for Real-World Applications . . . 2
- 1.3 Risk-averse Approaches for Reinforcement Learning . . . 3
- 1.4 Original Contribution . . . 4
- 1.5 Overview . . . 5

2 Reinforcement Learning **7**
- 2.1 Introduction . . . 7
- 2.2 Markov Decision Processes . . . 8
  - 2.2.1 Decision Rules and Policies . . . 9
  - 2.2.2 Performance Measures: Value Functions and Expected Return . . . 10
  - 2.2.3 Optimality Criteria . . . 11
  - 2.2.4 Bellman Equations and Operators . . . 11
  - 2.2.5 Exact Solution Methods for Finite MDPs . . . 12
- 2.3 Learning the Optimal Solution with Reinforcement Learning . . . 14
- 2.4 Reinforcement Learning: Tabular Methods . . . 16
  - 2.4.1 Prediction . . . 16
  - 2.4.2 Control . . . 17
- 2.5 Reinforcement Learning: Approximate Methods . . . 19
  - 2.5.1 Prediction with Approximation . . . 19
  - 2.5.2 Value-Based Control . . . 22
  - 2.5.3 Policy-Based Control . . . 22
  - 2.5.4 Actor-Critic Control . . . 24
  - 2.5.5 Safe and Trust-Region Approaches . . . 24
- 2.6 Multi-Objective RL . . . 26
  - 2.6.1 Multi-Objective MDPs . . . 26
  - 2.6.2 Optimization Criteria for MOMDPs . . . 27

3 Risk-Aversion in Reinforcement Learning **29**
- 3.1 Introduction . . . 29

3.2 Risk-Averse Objectives . . . 30
3.2.1 Utility Functions . . . 31
3.2.2 The Mean-Variance Criteria . . . 33
3.2.3 Coherent Risk-Measures . . . 36
3.2.4 Robustness . . . 38
3.2.5 Choosing the risk model . . . 39
3.3 Solving Risk-Sensitive RL Tasks . . . 39
3.3.1 Risk-Sensitive MDPs . . . 40
3.3.2 Policy Gradient Methods for Risk-Averse Optimization . . . 41
3.3.3 Risk-Sensitive Distributional RL . . . 43
3.3.4 Beyond Ad-hoc Techniques . . . 44

4 Risk-averse Optimization through State-Augmentation 45
4.1 Introduction . . . 45
4.2 A unified perspective . . . 46
4.2.1 Inner Objective as an Ordinary MDP . . . 48
4.2.2 Optimizing the Outer Objective . . . 50
4.3 Policy Optimization . . . 50
4.4 Experiments . . . 52
4.4.1 Multi-armed Bandit . . . 52
4.4.2 Point Reacher . . . 53
4.4.3 Trading Environment . . . 55
4.4.4 Robotic Locomotion . . . 56
4.5 Conclusions . . . 58

5 Risk-Averse Trust Region Optimization for Reward-Volatility Reduction 59
5.1 Introduction . . . 59
5.2 Risk Measures . . . 60
5.3 Risk-Averse Policy Gradient . . . 63
5.3.1 Estimating the Risk-Averse Policy Gradient . . . 65
5.4 Trust Region Volatility Optimization . . . 67
5.5 Experiments . . . 70
5.5.1 S&P 500 Trading . . . 70
5.5.2 FX Trading . . . 71
5.6 Conclusions . . . 72

6 Finite Sample Analysis of an Actor-Critic Algorithm for Mean-Volatility Optimization 73
6.1 Introduction . . . 73
6.2 Problem Formulation . . . 74
6.2.1 Mean-Volatility Policy Evaluation Techniques . . . 74
6.2.2 Monte-Carlo Estimation of the Expected Return . . . 75
6.2.3 The Critic Algorithm . . . 75
6.2.4 The Actor Algorithm . . . 76
6.2.5 General Assumptions . . . 78
6.3 Main Results . . . 79
6.3.1 Direct Mini-Batch TD Analysis . . . 79

6.3.2 Mean-Volatility Actor-Critic Analysis 81
6.4 Experiments 82
6.5 Related Works 82
6.6 Conclusions 83

7 Conclusion **85**

A Additional Results and Proofs **93**
A.1 Additional Results of Chapter 4 93
A.1.1 Additional Results 93
A.1.2 State augmentation from batch data 94
A.1.3 Reproducibility Details 96
A.2 Additional Results of Chapter 5 98
A.2.1 Safe Volatility Optimization 98
A.2.2 Exponential Utility applied on the reward 103
A.2.3 Experiments 104
A.3 Additional Results of Chapter 6 107
A.3.1 Auxiliary Lemmas 107
A.3.2 Analyzing the Monte-Carlo Estimation of the Expected Return . 113
A.3.3 Critic's Analysis 115
A.3.4 Actor's Analysis 121

B Mean-Volatility and Multi-Objective Reinforcement Learning **127**
B.1 A Multi-Objective Perspective 127
B.2 The Penalized Criterion as a Sequence of Mean-Second-Moment MDPs 129
B.2.1 Mean-Variance Policy Iteration 129
B.2.2 Markovian Deterministic Policies Are Sufficient for Optimality . 130
B.2.3 Considerations on the Optimal Policies Obtained with MVPI . . 132

CHAPTER 1

Introduction

Everyday life is one of the clearest examples of what a *stochastic* environment is. The decisions that we take at any moment have, indeed, only a limited impact on the dynamics of what happens around us. No matter how accurate our plans are, the presence of random events, which we cannot predict or control, makes impossible for us to foresee a *certain* outcome for them. Thus, we always have to account for a certain degree of variability, which may alter the result of our actions, in either a positive or a negative way. The amount of variability that one can tolerate is a subjective quantity, which depends on how much one is *averse* to the possible *risks*. When the fulfillment of our objectives does not go in the same direction of reducing risk, we are in presence of a *trade-off*, whose solution strongly depends on our degree of risk-aversion.

This dilemma is common in every context where some kind of randomness, noise or uncertainty is present. In all these kinds of situations, taking a decision cannot overlook the possible negative outcomes due to risk. The focus of this dissertation is to build artificial agents which are capable of handling the risk-return trade-off, by means of reinforcement learning.

1.1 The Reinforcement Learning Framework

Reinforcement learning (RL) is a term used to indicate at the same time a *problem*, the *framework* describing it, and the several *techniques* that can be used to provide it with a solution.

What reinforcement learning aims at doing is solving *sequential decision-making problems* by means of a learning process that, in a trial and error fashion, is capable of gradually improving the quality of the produced decisions, guided by a *reinforcement signal*. This learning process resembles what happens when one tries to teach a dog to

bring the stick back. By rewarding the dog for correct actions, the owner can modify the dog's behavior according to the desired task. It is also similar to what we experience every time we need to learn a new skill. At school for instance, when a teacher tries to teach her students how to write: she can reward the students when they wrote a letter with the correct shape, but she cannot tell them how to exactly move the muscles of their hand to do that. Therefore, students have to repeatedly try, fail, and learn from their failures, in order to finally being able to become good writers.

The RL framework features two main actors: the *agent* and the *environment*. The environment is characterized by a *state*, which describes its main properties, and evolves according to a certain *dynamics*. The agent is supposed to be able to observe the environment state, and to interact with the environment by means of *actions*. The environment dynamics, in turn, is influenced by agent actions. Together with the new state, the environment dynamics produces a *reward* which is given as a further feedback to the agent. The *goal* of the agent is, thus, to adapt its behavior to gather as much of these rewards as possible. Thanks to its simplicity, this framework is general enough to be applicable to a wide set of problems. Its main advantage is that it does not require the knowledge of the environment *dynamics*, which in many real-world contexts may be unknown or difficult to simulate.

Solution methods for RL problems are all influenced by a principle called *dynamic programming* (Bellman, 1954). Its main idea consists in exploiting the structure of the main problem to splitting it into sub-problems, which can be solved in a easier way, and then combining the solutions to solve the original task. Thanks to the introduction of *temporal-difference* (TD) learning (Sutton, 1984; Klopf, 1988) it is possible to translate this principle to the aforementioned framework, which is agnostic of the real environment dynamics. This idea was inspired from psychology studies on classical conditioning (Pavlov and Anrep, 1927) and was later found to be at the basis of dopamine neuron activity (Montague et al., 1996). The recent developments in reinforcement learning approaches, coupled with the great approximation power of deep neural networks, have allowed to obtain astonishing results on many challenging fields such as board games (Silver et al., 2016), robotic locomotion (Schulman et al., 2015b), single-player (Mnih et al., 2015) and multi-player video-games (Berner et al., 2019).

1.2 Reinforcement Learning for Real-World Applications

The wide applicability of the RL framework makes it suitable not only for simulated benchmarks, but also for real-world problems. While, in principle, reinforcement learning could fit to almost all settings in which traditional control systems can be developed, in practice, it is particularly interesting to employ it in contexts where standard approaches struggle to succeed. Whenever it is difficult to have a good model for the environment, for instance in finance, agriculture settings, or when it is difficult to conceive a valid control policy from scratch, as it happens for some complex robotics or healthcare tasks, reinforcement learning may represent an interesting alternative. However, despite its success on difficult domains, reinforcement learning is still not ready to become a mature technology. The first issue that prevents the widespread adoption of reinforcement learning techniques is its huge *sample-complexity*. RL approaches, being sample-based, need to interact a large number of times with the environment

before obtaining the superior performance they show on the aforementioned contexts. Furthermore, real-world environments present a number of additional challenges for reinforcement learning methods, such as *non-stationarity* (Even-Dar et al., 2009), *delays* Schuitema et al. (2010) or *partial information* (Monahan, 1982). While the RL community is working on remedies for each of these issues, they are still open problems nowadays.

Stochastic tasks are the ideal target for RL applications. First, they naturally fit the RL formulation, and, second, they are typically difficult to solve with traditional methods, hence, they could benefit more from the use of an RL approach. However, dealing with randomness, one has always to take into account some uncertainty related to the final performance, as already mentioned. Unfortunately, it is not always the case that the solution yielding the best (expected) gain is also the one minimizing its variability. In fact, it often happens that a *trade-off* is present between higher expected performance and *risk*. The financial case represents a good example for these kind of situations. Let's imagine to be trading on a financial market. When the market volatility is high, it's easier to make higher profits, since prices have larger oscillations. Clearly, for the same reason, it is also easier to lose a great amount money. On the other hand, in low volatility situations, profit opportunities are few, but also the risk is low. In those cases, measuring *risk* is not sufficient: stakeholders must be able to choose between a variety of possible trade-offs. To ensure that, reinforcement learning should be able to explicitly direct the learning process towards risk-averse behaviors.

1.3 Risk-averse Approaches for Reinforcement Learning

The goal of standard reinforcement learning is the maximization of the *expected value* of the (possibly discounted) cumulative sum of the rewards, also known as *return*. This means that the usual RL objective ignores the variability connected to the return, hence, it seems to be inappropriate for dealing with the aforementioned trade-off. A common criticism which is opposed to this claim is that: *"reward is not money"*. The meaning of this iconic sentence is the following: what we may naturally consider a reward in the real-world task (for instance money, if we are dealing with a financial setting) does not necessarily need to be the reward of the RL model we employ to solve the problem. In other words, according to this view, it is always possible to conceive some scalar reward which accounts for the desired objective (Silver et al., 2021). To say it with Sutton's words:

"That all of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward)".

While it is hard to say whether this claim is ultimately true or not, it should be noticed that, provided that such reward exists, computing it may be a difficult task (Ng et al., 2000), which could even require having already solved the risk-averse problem beforehand. Therefore, modifying the objective is sometimes the only available option. However, most of the time, this modification does not allow to directly use methods and results from the risk-neutral case, calling, then, for ad-hoc methods.

1.4 Original Contribution

Since there are several possible ways to measure risk or to model risk-aversion, a plethora of risk-averse approaches have been developed in the RL literature in the past years. This means that, in order to transfer the advantages of state-of-the-art developments to the risk-averse setting, one has to explicitly extend (if possible) the considered methods for the target risk-averse objective. This complicates the use of reinforcement learning in risk-averse tasks, thus, limiting its applicability to some relevant real-word settings. This dissertation's main goal is *to study how it is possible to ease the application of state-of-the-art risk-neutral methods to risk-averse objectives*, reducing the performance gap between the two settings. In order to approach this broader goal, two main paths are followed.

The first one is related to the optimization of standard *return-based* risk-measures. Adapting a new technique to each of the classical risk-measures requires a lot of effort and it may be a hard task, as already explained. For this reason, as a first contribution, we study a *unified framework*, that allows to consider under the same formalism three of the most common risk-averse objectives. This work exploits some state-augmentation techniques available from the literature to devise a unified approach which translates a risk-averse problem into a sequence of simpler risk-neutral RL tasks. Solving these simpler tasks, even in an approximated way, allows then to obtain a solution for the risk-averse main problem. Any RL method can be employed to solve the sub-problems, allowing then to directly connect the risk-neutral world to the risk-averse one. The proposed technique is empirically validated with an extensive analysis that involves the application of several recent RL approaches to complex domains such as robotic locomotion and simulated trading.

The other path that we follow is about the study of a novel class of risk-measures, which captures a different kind of risk. These risk-measures are computed by considering as a random variable the *per-step reward* which is distributed according to the state-action occupancy distribution. Our second contribution consists, in particular, in studying the variance of this random variable, which we called *reward-volatility*. We show that this new risk-measure exhibits interesting mathematical properties, that ease the direct application of some important results from the risk-neutral policy optimization literature. Furthermore, we can demonstrate that the minimization of the reward-volatility also bounds the classical return variance. After the formulation of a novel trade-off objective, called mean-volatility, we study how to extend results from the safe RL literature to obtain a *monotonic improvement bound* which is then exploited by a practical TRPO-style algorithm. We also present the results of applying this approach to challenging financial environments based on real-world data.

As already discussed, risk-averse approaches typically add a further layer of complexity to the RL problem. While standard reinforcement learning allows to have some global optimality guarantees, at least in some particular settings, this kind of result is typically harder to obtain when risk is involved (Mannor and Tsitsiklis, 2011). Reaching a local optimum is a more affoardable goal, and it is often enough to obtain reasonable solutions. While many works prove asymptotic convergence guarantees for their approaches (Tamar et al., 2012a, 2015a; Chow et al., 2017), only a few focus on the actual convergence rate (Jiang and Powell, 2018; Fei et al., 2020). Since sam-

ple complexity is one of the main issues of modern reinforcement learning, our third contribution consists in a finite-sample analysis of a mean-volatility actor-critic algorithm. Our goal is to study to which extent a risk-averse optimization may impact on the overall sample-complexity.

1.5 Overview

This dissertation is organized as follows. The first two chapters represents an introduction for the reader to the state-of-the-art of risk-averse reinforcement learning literature:

- In Chapter 2 we present an introduction to the reinforcement learning framework. We first describe the Markov Decision Process formalism, and the exact methods that may be used to solve this kind of problems, by knowing the model. We then introduce the main concepts of model-free reinforcement learning, illustrating the main tools that will be used in the later chapters.
- In Chapter 3 we introduce the risk-averse reinforcement learning literature. In the first part we present the main available approaches to model risk, discussing possible criteria to choose the risk-averse objective in the most suitable way. We then survey the main state-of-the-art approach for solving each of the described problems.

The following chapters represent instead the original contribution of this dissertation:

- In Chapter 4 a unified framework for the optimization of some of the most common return-based risk-measures is presented. We devise a meta-algorithm that, thanks to a state-augmentation, allows the application of risk-neutral approaches to the target risk-averse objective. An experimental analysis is conducted to empirically evaluate the performance of this method, testing several combinations of risk-averse objectives and RL algorithms on a set of challenging domains. The work described in this chapter is, at the time of writing, under revision for the "Special Issue on Risk-aware Autonomous Systems: Theory and Practice" in the *Artificial Intelligence Journal*.
- In Chapter 5 we present a novel risk-averse objective which is a trade-off between the expected return and the variance of the reward under the state-action occupancy distribution. We show interesting properties of this risk-measure and we obtain monotonical improvement bounds. Exploiting these result, we also devise a pratical TRPO-like algorithm, which we show to be effective in deriving approximated Pareto frontiers for the mean-volatility trade-off, on a complex real-world based financial task. The content of this chapter has been published in (Bisi et al., 2020c).
- In Chapter 6 we analyse the finite-sample complexity of an actor-critic algorithm optimizing the mean-volatility trade-off. We propose two general methods for policy evaluation, the *direct* approach and the *factored* one, obtaining a finite sample bound for the critic. By extending the analysis of (Xu et al., 2020b) to the mean-volatility case, we obtain a result for the actor-critic algorithm, bounding the sample complexity needed to reach an ϵ-accurate stationary point. The content of this chapter has been accepted as a conference paper for *AISTATS 2022*.

Finally, Chapter 7 summarizes the contribution of this work, highlighting its main limitations and indicating some possible future directions. Additional results and proofs, which have been omitted from the main work, can be found in Appendix A, while some further preliminary work about mean-volatility can be found in Appendix B.

CHAPTER 2

Reinforcement Learning

2.1 Introduction

A sequential decision process models the *interaction* of an *agent* with an *environment*. Each interaction is composed by the following steps:

- the agent receives an *observation* from the environment;
- the agent chooses an *action*;
- the environment, influenced from the chosen action, *transitions* to a new state, according to its *dynamics*;
- the environment produces a *reward*.

This loop repeats while the agent is in the condition of interacting with the environment. In this chapter, we will review the main theoretical results regarding the mathematical tool used to model these interactions, that is the Markov Decision Process. We will also study the main *exact methods* which can be employed when the model is *known*.

In case the model is *not available* instead, reinforcement learning methods allow to *learn* the best solution directly from the interactions with the environment. We will describe the main RL ideas and settings, surveying some of the most important approaches. This introduction has the purpose to present the fundamental RL tools that will be employed in the next chapters, thus, it does not claim to be complete. We invite the reader to refer to (Sutton and Barto, 2018; Bertsekas, 2019; Agarwal et al., 2019; Szepesvári, 2010) for an extensive review of the main aspects of reinforcement learning.

2.2 Markov Decision Processes

Sequential decision making problems are typically modelled under the formalism of Markov Decision Processes (MDP) (Puterman, 2014). This framework derives from the seminal work by Bellman (1954). The main assumption of this model is that the state is *completely observable*[1] and *markovian*. The latter feature means that dynamics depends only on the current state and on the chosen action, and does not present any dependence on the previous history. We restrict our attention to the case of discrete-time MDPs, where time is conceived as a discrete set of instants called *decision time-steps*.

Definition 2.2.1 (Markov Decision Processes (Puterman, 2014))**.** *A discounted Markov decision process (MDP) is a tuple $M = (\mathcal{S}, \mathcal{A}, P, R, \mu, \gamma)$, where:*

- *$\mathcal{S}$ is a measurable* state-space*, containing all the states the agent may visit;*
- *$\mathcal{A}$ is a measurable* action-space*, containing all the actions the agent may take;*
- *$P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel;*
- *$R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function;*
- *$\mu : \mathcal{S} \to \Delta(\mathcal{S})$ is the initial-state distribution;*
- *$\gamma \in [0, 1)$ is the discount factor;*

Here $\Delta(\mathcal{S})$ denotes the set of probability measures over $\mathcal{S}$. The decision process works as follows. First, the initial state s_0 is drawn from μ. Then, the agent takes an action a_0, it transitions to a new state $s_1 \sim P(\cdot|s_0, a_0)$, it receives the reward $r_1 = R(s_0, a_0)$, and so on. As it may be noticed, the transition depends, once the action is chosen, only on the current state, independently from the previous ones. This memorylessness feature is known as the *Markov property*. The sequence of states, actions and rewards up to a certain time-step t is called the *history* and it can be defined as:

$$H_t \coloneqq (s_0, a_0, r_1, s_1, a_1 \ldots, s_t, r_t).$$

We define the set of such histories as $\mathcal{H}_t$. The interaction between the agent and the environment lasts for a certain horizon T, which can be either *finite* ($T < \infty$) or *infinite* ($T = \infty$): in this thesis we focus on the latter case. Accordingly we define as *trajectory* the (possibly infinite) history $\tau = H_T$, and the set of all possible trajectories as $\mathcal{T}$. Given a certain discount factor γ, we define the return function $G_\gamma : \mathcal{T} \to \mathbb{R}$ of any trajectory τ as:

$$G_\gamma(\tau) \coloneqq \sum_{t=0}^{T} \gamma^t r_{t+1},$$

where γ is dropped when clear from the context. We shall also consider the following standard assumption on the boundedness of the rewards.

Assumption 1 (Bounded Rewards)**.** *The reward function is uniformly bounded, i.e., there exists a finite constant $R_{max} > 0$, such that:*

$$\|R\|_\infty = \sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} |R(s, a)| \leq R_{max}$$

[1] When this is not the case, the correct framework is Partially Observable MDPs (POMDP) (Monahan, 1982)

There is a class of MDPs, which we will refer to in some cases, for which each sequence of states eventually ends in an *absorbing* (self-loop) zero-reward state. These MDPs are called *episodic* and we can consider for them an (effective) horizon $T = \hat{T}$, where $\hat{T}$ is the first time the agent encounters the *absorbing state*. When state and action spaces are discrete sets, we call the MDP *finite*, otherwise we call it *continuous*.

2.2.1 Decision Rules and Policies

A *decision rule* is an action selection procedure for each state at a specified *decision epoch* t. Based on their domain and codomain, decision rules are classified as follows:

- *history-dependent randomized (stochastic)*: $d_{HR} : \mathcal{H} \to \Delta(\mathcal{A})$;
- *history-dependent deterministic*: $d_{HD} : \mathcal{H} \to \mathcal{A}$;
- *Markovian randomized (stochastic)*: $d_{MR} : \mathcal{S} \to \Delta(\mathcal{A})$;
- *Markovian deterministic*: $d_{MD} : \mathcal{S} \to \mathcal{A}$;

where $\Delta(\mathcal{A})$ denotes the set of probability measures over $\mathcal{A}$, and $\mathcal{H}$ is the set of histories at each possible time-step $0 \leq t \leq T$. A *policy* is defined as a sequence of decision rules:

$$\pi = (d_1, d_2, \ldots, d_{T-1}). \tag{2.1}$$

Policies are called *non-stationary* if the decision rule may change at each step, and they are labelled instead as *stationary* if the decision rule is always the same[2]. We further define the following policy sets:

- the set of all non-stationary policies, whose decision rule may be history-dependent and randomized Π^{HR};
- the set of stationary policies whose unique decision rule is Markovian and randomized: Π^{SR};
- the set of stationary policies whose unique decision rule is Markovian and deterministic: Π^{SD}.

Fixing a policy $\pi \in \Pi^{SR}$ induces the *(discounted) state-occupancy measure* or *on-policy distribution* :

$$d_\mu^\pi(s) \coloneqq (1-\gamma) \int_{\mathcal{S}} \mu(s_0) \sum_{t=0}^{\infty} \gamma^t p_\pi(s_0 \xrightarrow{t} s) \, \mathrm{d}s_0, \tag{2.2}$$

where $p_\pi(s_0 \xrightarrow{t} s)$ is the probability of reaching state s in t steps from s_0 following policy π. This distribution measures the discounted probability of visiting a state at some point of the interaction, by employing a certain policy. Moreover, the complete knowledge of the MDP and the current policy is sufficient to determine also the *distribution of the (sub-)trajectories of length* T:

$$p_\pi(\tau) = \mu(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t) P(s_{t+1}|s_t, a_t). \tag{2.3}$$

[2]Since in this work the decision rule is the same in all time-steps, we always use the two terms interchangeably.

2.2.2 Performance Measures: Value Functions and Expected Return

The object of interest of RL is the return, a quantity that each RL agent wants to be as high as possible. In particular, the focus of standard *risk-neutral* reinforcement learning is the *expected return*. Value functions are a core concept in this discipline, since they allow to map a state or a state-action pair to the corresponding expected value of the return, conditioning the trajectory to start from the target state or state-action pair. The *state-value function* (or, simply, value function) is, more formally, defined as follows.

Definition 2.2.2 (State Value function)**.** *Let π be a policy and s be any state. The value function $V_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is:*

$$V_\pi(s) \coloneqq \mathop{\mathbb{E}}_{\substack{a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim P(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) | s_0 = s \right] = \mathop{\mathbb{E}}_{\tau \sim p_\pi(\cdot)} \left[G(\tau) | s_0 = s \right]. \tag{2.4}$$

We define also the *state-action value function*, sometimes simply called *Q-function* or *action-value function*.

Definition 2.2.3 (Action-Value function)**.** *Let π be a policy and s be any state. The action-value function $Q_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is:*

$$Q_\pi(s, a) \coloneqq \mathop{\mathbb{E}}_{\substack{s_{t+1} \sim P(\cdot|s_t, a_t) \\ a_{t+1} \sim \pi(\cdot|s_{t+1})}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, a_0 = a \right]. \tag{2.5}$$

The difference between these two functions for each state-action pair is the so-called *advantage function*:

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s), \tag{2.6}$$

and it measures the advantage of deviating from the current policy for one single step. Value functions allow to measure the value of each state (or state-action pair), however, one may be interested in evaluating some overall performance according to a certain distribution over the state space.

Definition 2.2.4 (Expected Discounted Return)**.** *The* expected discounted return *from a distribution ρ and following a policy π is:*

$$J_\rho(\pi) \coloneqq \int \rho(s_0) V_\pi(s_0) \, \mathrm{d}(s_0). \tag{2.7}$$

Typically, this distribution ρ is the initial state distribution μ. Sometimes we want to *learn* or to *approximate* a value function using some function $f_V : \mathcal{S} \to \mathbb{R}$ or $f_Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: in those contexts we call also those functions, respectively, value or action-value functions, even if they do not actually match the expected return of any real policy. Importantly, value functions have a fundamental role in performance optimization, since they can be the starting point for policy improvement. Any function f_Q, indeed, may be associated to a special kind of policy, called *greedy policy*.

Definition 2.2.5 (Greedy Policy)**.** *Given any function $f_Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we define $\bar{\pi} \in \Pi^{SD}$ as* greedy policy *if:*

$$\forall s \in \mathcal{S} : \bar{\pi}(s) \coloneqq \arg\max_{a \in \mathcal{A}} f_Q(s, a) \tag{2.8}$$

2.2.3 Optimality Criteria

The first optimality criterion we describe for the discounted case is the standard one (Puterman, 2014).

Definition 2.2.6 (Optimality)**.** *A history-dependent policy $\pi^\star \in \Pi^{HR}$ is* optimal *if:*

$$\forall_s \in \mathcal{S} : V_{\pi^\star}(s) = \sup_{\pi \in \Pi^{HR}} V_\pi(s). \tag{2.9}$$

While, according to this definition, optimal policies belong to Π^{HR}, it is possible to restrict our focus to Π^{SR}, since in the discounted setting it is always possible to derive a stochastic Markovian policy that has the same value function as any history-dependent one (Puterman, 2014). It is possible to define also the *optimal value function* and the *optimal action-value function* as:

$$V^*(s) \coloneqq V_{\pi^\star}(s), \tag{2.10}$$

$$Q^*(s,a) \coloneqq Q_{\pi^\star}(s,a). \tag{2.11}$$

A different approach consists in considering as optimal all the policies that optimize the expected discounted return from some distribution.

Definition 2.2.7 (J_ρ-optimality)**.** *A policy $\pi^\star$ is J_ρ-*optimal *if:*

$$\pi^\star = \arg\max_{\pi \in \Pi^{SR}} J_\rho(\pi). \tag{2.12}$$

It is easy to see that optimal policies are also J_ρ-optimal for any distribution ρ, while the converse is not necessarily true. The latter definition allows to measure the optimality trough a single index, hence, it is more suitable for policy search approaches.

2.2.4 Bellman Equations and Operators

Value functions enjoy fundamental recursive relations called *Bellman equations*.

Proposition 2.2.8 (Bellman Equations (Bellman, 1954))**.** *Let $\pi \in \Pi^{SR}$, then:*

$$Q_\pi(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) V_\pi(s'), \tag{2.13}$$

$$V_\pi(s) = \int_{\mathcal{A}} \pi(\mathrm{d}a|s) Q_\pi(s,a). \tag{2.14}$$

To better understand the properties of such equations, it is useful to define some appropriate *operators*.

Definition 2.2.9 (Bellman Expectation Operators)**.** *Let $\pi \in \Pi^{SR}$ and $f_V : \mathcal{S} \to \mathbb{R}$, then the* Bellman expectation operator *for the value function is:*

$$(T^\pi f_V)(s) \coloneqq \int_{\mathcal{A}} \pi(\mathrm{d}a|s) \left(R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) f_V(s') \right), \tag{2.15}$$

while, being $f_Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, the Bellman expectation operator *for the action-value function is:*

$$(T^\pi f_Q)(s,a) \coloneqq R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) \int_{\mathcal{A}} \pi(\mathrm{d}a'|s') f_Q(s',a'), \tag{2.16}$$

These operators are linear and they are γ-*contractions* under the L_∞-norm (Puterman, 2014). Thanks to the Banach fixed point theorem (Banach, 1922), each of them admits a unique fixed-point: they are, respectively, the value and action-value functions. Therefore equations (2.14) and (2.13) can be re-written as the fixed point equations:

$$V_\pi = T^\pi V_\pi, \tag{2.17}$$
$$Q_\pi = T^\pi Q_\pi. \tag{2.18}$$

Optimal value functions also enjoy useful recursive relations and operators.

Proposition 2.2.10 (Bellman Optimality Equations, Bellman (1954))**.** *Let $\pi \in \Pi^{SR}$, then:*

$$Q^\star(s,a) = R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) V^\star, \tag{2.19}$$
$$V^\star(s) = \sup_{a \in \mathcal{A}} Q^\star(s,a). \tag{2.20}$$

Definition 2.2.11 (Bellman Optimality Operators)**.** *Let $\pi \in \Pi^{SR}$ and $f_V : \mathcal{S} \to \mathbb{R}$, then the* Bellman optimality operator *for the value function is:*

$$(T^\star f_V)(s) \coloneqq \sup_{a \in \mathcal{A}} \left(R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) f_V(s')) \right), \tag{2.21}$$

while, being $f_Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, the Bellman optimality operator *for the action-value function is:*

$$(T^\star f_Q)(s,a) \coloneqq R(s,a) + \gamma \int_{\mathcal{S}} P(\mathrm{d}s'|s,a) \sup_{a' \in \mathcal{A}} f_Q(s',a'), \tag{2.22}$$

It can be shown that these (non-linear) operators are γ-contractions for the L_∞ norm too, moreover, they enjoy a *monotonicity property* (Puterman, 2014). In fact, for any pair of value functions (f, f'), defined as above, it holds that:

$$\forall s \in \mathcal{S} : (T^\star f)(s) \geq (T^\star f')(s), \tag{2.23}$$

and similarly for action value functions. The fixed points of these contracting operators are, respectively, $V^\star$ and $Q^\star$.

A policy which is optimal (according to Equation 2.2.6) can be obtained by acting greedily w.r.t. $Q^\star$. These result allow to show that one can always retrieve an optimal policy which is *stationary, Markovian, and deterministic* (Puterman, 2014, Theorem 6.2.7).

2.2.5 Exact Solution Methods for Finite MDPs

In this section we describe some *exact* solution methods for *finite* Markov Decision Processes, that are based on a *complete knowledge* of the MDP. All of these approaches exploit the *dynamic programming principle* by means of the Bellman equations. The tasks these methods aim at solving are two:

- *policy evaluation*, which consists in determining the value function given a certain policy;

- *policy/value optimization*, which amounts to finding the optimal policy (or value function) for the MDP.

The algorithms we show here, together with their properties, are taken from (Puterman, 2014).

Policy Evaluation

As shown in Equations (2.17) and (2.17), value functions of a certain policy $\pi \in \Pi^{SR}$ are the fixed points of the Bellman Expectation Operators T^π. Since we assumed finite MDPs, we can re-write our quantities according to matrix notation with:

- r: the reward vector, of lenght $|\mathcal{S}|$, in which each i-th component corresponds to the expected reward of the i-th state: $\int_{a \in \mathcal{A}} \pi(\mathrm{d}a|s_i) R(s_i, a)$;
- v: the value function vector, in which each component corresponds to the value of a state, which has then length $|\mathcal{S}|$;
- P^π the transition kernel matrix for the action-state transition kernel P and the policy π which has dimension $|\mathcal{S}| \times |\mathcal{S}|$.

Bellman equation (2.14) can be written as:

$$V_\pi = r + \gamma P^\pi V_\pi, \tag{2.24}$$

which amounts to a linear system of equations, hence, it can be solved in closed form[3] $V_\pi = (I - \gamma P^\pi)^{-1} r$. While this approach allows to obtain the exact value function for the policy, matrix inversion may be computationally expensive or unfeasible based on the size of the state space. A possible alternative consists in using an *iterative* approach, which is based on the repeated application of the T^π operator, corresponding to the following update:

$$V_k \leftarrow r + \gamma P^\pi V_{k-1}. \tag{2.25}$$

Since the operator is a γ-contraction, then it is possible to upper bound the error w.r.t. the true value function at each iteration with:

$$\|V_k - V_\pi\|_\infty \leq \frac{1}{1-\gamma} \|V_k - V_{k-1}\|_\infty,$$

starting from any value function V_0 (Puterman, 2014). Therefore, one can obtain a value function approximation to the true value function with the *desired precision* ϵ, by stopping when

$$\|V_k - V_{k-1}\|_\infty \leq \epsilon(1-\gamma),$$

or one can decide to continue until complete convergence, by setting $v_k = v_{k-1}$ as termination criterion. The convergence in a finite number of steps is guaranteed in finite MDPs.

Policy and Value Function Optimization

We present the two main algorithms used for the exact solution of finite MDPs: *value iteration* and *policy iteration*.

[3] It can be shown that $(I - \gamma P^\pi)^{-1}$ exists (Puterman, 2014, Appendix C4).

Value Iteration While the operator T^π offers a straightforward ways to estimate the value function, the Bellman Optimality Operator $T^\star$ can be used to estimate the *optimal value function*, hence, solving the MDP. Unfortunately, since the Bellman Optimality Equations (2.21) and (2.22) are not linear, a solution in closed form does not exists. However, it is still possible to iteratively apply $T^\star$ for approaching the optimal value function up to the desidered accuracy, as for the previous policy evaluation method. With this technique, known as value iteration, one may solve the MDP by repeatedly applying the update rule:

$$Q_k = T^\star Q_{k-1} \text{ (or } V_k = T^\star V_{k-1}).$$

The optimal policy may be retrieved by acting greedily w.r.t. the Q-function[4].

Policy Iteration An alternative approach for optimization consists in alternating policy evaluation and *policy improvement* steps, and takes the name of policy iteration. While policy evaluation may be performed with any of the approaches described in Section 2.2.5, the policy improvement steps simply corresponds to computing the greedy policy (see Definition 2.2.5) w.r.t. the current policy value function. This process yields a sequence of Markovian deterministic policies, with monotonically improving value functions, which is guaranteed to converge to the optimal solution in finite time.

2.3 Learning the Optimal Solution with Reinforcement Learning

Exact approaches assumes the full-kwowledge of the MDP and requires states and action to be finite[5], however this is not always the case, especially in real-world problems. Typically, the state space (or also the action one) may be continuous or just very large. Moreover, some part of the model as the transition kernel or the reward function, may be unknown. These characteristic may prevent the application of exact methods, or make them inefficient. Reinforcement Learning(Sutton and Barto, 1998) deals with the two problems of *learning* how to *predict* value functions for some given policy, and how to *control* in the best possible way MDPs which cannot be solved with exact methods. These complementary tasks can be solved with different approaches based on the contexts the learning task is framed in. A typical classification for the main RL settings can be provided by means of the following dichotomies:

- *Prediction and Control.* The policy evaluation task is also called *prediction* in the RL context, in order to highlight the approximate nature of the estimate. Policy or value function optimization, instead, are known as *control* (Sutton and Barto, 2018).
- *Model-Free and Model-Based.* Model-based approaches (Deisenroth and Rasmussen, 2011; Nagabandi et al., 2018) aims at estimating either the transition model or the reward one, in order to then solve the MDP by using exact solutions or approximated ones. On the other hand, model-free techniques work without the necessity of estimating a complete model, but they have as the object of their learning task the value function or the policy itself.

[4] If the value function is computed instead of the action-value function one, one needs to compute the Q-function with Equation (2.16), which is always possible when we know the model.

[5] Extensions are possible for compact action set, (Puterman, 2014, Section 6.4.3)

- *Online and Offline.* Online approaches (Williams, 1992; Schulman et al., 2017) consider a scenario in which the agent learns *while interacting* with the environment. This means that, at each timestep, the agent may receive new samples from the environment which may be useful for the learning task. In an offline setting[6] (Lange et al., 2012; Ernst et al., 2005b), instead, all the samples are assumed to be available from the beginning of the task and no further interaction with the environment is possible. An intermediate setting is possible, where the interaction is limited, but it allow to periodically produce new batch of samples: this setting is called *semi-batch*.

- *On-Policy and Off-Policy.* The learning task, either prediction or control, does not need necessarily to target the current policy. An agent may interact with the environment with some policy and learn some information on another one. For instance, the policy used to interact may be *explorative*, hence, more effective in gathering the information necessarily for the learning process than the target one. When the *target* and the *behavioral* policies coincide, the setting is called on-policy Williams (1992), otherwise it is called off-policy (Watkins and Dayan, 1992; Ernst et al., 2005b; Silver et al., 2014; Schulman et al., 2017).

- *Tabular and Function Approximation.* When the MDP is finite, the setting is also called *tabular*, since we may imagine value functions and policies as tables (or matrices). In case the state-action space is infinite, this view is not valid anymore, it is not even possible to observe each state-action pair once. In such situations, one has to resort to *function approximation* for estimating the value function, or the policy, in unseen state-action pairs, generalizing what it has been learned for the more similar ones.

- *Value-Based, Policy-Based and Actor-Critic.* Value-based (or *critic-only*) methods (Ernst et al., 2005b; Mnih et al., 2015) tries to learn the optimal value function as it happens in the exact case for value iteration (Section 2.2.5). They usually try to apply the dynamic programming principle by means of *temporal-difference* learning (Sutton and Barto, 1998). Policy based ones (sometimes called *actor-only*), instead, aims at directly finding the optimal policies, typically w.r.t. the J_ρ criterion of Definition 2.2.7 (Williams, 1992; Schulman et al., 2015a, 2017; Papini et al., 2017). They usually do not exploit the value function, but they search instead for the optimal solution in the parameter space (e.g. following a gradient). The combination of the two approaches results in a very effective class of algorithms called *actor-critic*, which try to combine the best from both worlds.

In the following sections we will introduce some of the main approaches for tabular (Section 2.4) and function approximation (Section 2.5) reinforcement learning, for what concerns both the prediction and control tasks. In this thesis work we will only focus on model-free RL, hence, we will not cover model-based approaches in this introduction.

[6] It is also known as *batch* setting.

2.4 Reinforcement Learning: Tabular Methods

Here we review some of the main prediction and control methods for tabular reinforcement learning. This setting considers only finite MDPs and, although restrictive, it is fundamental to understand some basic principles which can then be extended in the function approximation case. Most of the content exposed here is taken from (Sutton and Barto, 1998).

2.4.1 Prediction

In a prediction task, the goal is to learn the expected value of the return by following some fixed policy π, either starting from a single state (the value function V_π), or from a distribution ρ of them ($J_\rho(\pi)$). In either case, the possible approaches are mainly two: *Monte-Carlo* and *Temporal Difference (TD)* learning. The first one allows for *unbiased* estimate, but it suffers large *variance*, while the second one introduces some bias, but permits to reduce variance. This trade-off can be regulated in a smooth way thanks to hybrid methods as the n-step TD and TD(λ) algorithms.

Monte-Carlo Prediction Monte-Carlo methods are based on a straightforward averaging the collected samples to estimate their mean. Let's consider, for instance, the $J_\rho(\pi)$ Monte-Carlo prediction task. Given a batch of N trajectories with length T, obtained starting from a distribution ρ and following the policy π, an estimator for $J_\rho(\pi)$ can be provided by:

$$\hat{J}_\rho(\pi) \coloneqq \sum_{i=0}^{N-1} G(\tau_i) = \sum_{i=0}^{n-1} \sum_{t=0}^{T-1} \gamma^t r_{i,t+1}.$$

Since it is the average of i.i.d. samples of the G random variable, it is an *unbiased* estimate with mean $\mathbb{E}[\hat{J}_\rho(\pi)] = J_\rho$ and variance $\mathbb{V}ar[\hat{J}_\rho(\pi)] = \frac{\mathbb{V}ar[\hat{J}_\rho(\pi)]}{n}$. With $n \to \infty$ the variance goes to zero, hence, the estimator coincides with $J_\rho(\pi)$ and it is, thus, *consistent*. By starting from a single state instead of from a distribution of them, one can find an unbiased and consistent estimate of the value function, called *first-visit* estimator. One might even consider also all the *partial* returns starting from each visit of the state of interest. The resulting estimator is called *every-visit* and it is no longer unbiased, but it enjoys lower variance.

TD prediction Unfortunately, Monte-Carlo estimators cannot learn from incomplete trajectories, hence, they are not suitable for online-learning tasks. Moreover, the large variance they suffer may be an issue when the available samples are few. Temporal difference methods (Sutton, 1984) allow to tackle this issue by estimating the value function in an iterative way, using previous estimates for computing the new ones. This approach, reminiscent of iterative policy evaluation, is also known as *bootstrapping*. All TD prediction methods employ an update rule with this common form:

$$V_{k+1}(s_t) \leftarrow V_{k-1}(s_t) + \alpha[G_t - V_{k-1}(s_t)], \tag{2.26}$$

where G_t is some estimator of the value function starting from s_t, also known as the *target*, t is the current timestep and k is the current iteration of the learning process. Let's τ_{s_t} be the sampled trajectory starting from s_t. By properly setting α, and by

setting $G_t = G(\tau_t)$, the method reduces to Monte Carlo every-visit update. Using instead $G_t^1 = R_{t+1} + \gamma V_{k-1}(s_t)$, one obtains the *TD(0)*, or *one-step TD*, update rule:

$$V_{k+1}(s_t) \leftarrow V_{k-1}(s_t) + \alpha[R_{t+1} + \gamma V_{k-1}(s_{t+1})V_{k-1}(s_t)]. \tag{2.27}$$

The term $G_t - V_k(s_t)$ is called TD-error, and it measures the difference between the old estimate for the value function and the next value, which combines the new available samples to the bootstrap. Another advantage of TD consists in exploiting the *structure* of the problem, by taking advantage of the *Markovianity* of the state (when present) by means of approximated Bellman updates. In order to trade-off between the bias introduced by using TD and the high variance of Monte Carlo estimates, a possible hybrid target is the n*-step TD* one:

$$G_t^n = \sum_{i=0}^{n-1} \gamma^i R_{t+i+1} + \gamma^n V_k(s_{t+n}).$$

One can also exponentially average among n-step TD updates to obtain the TD(λ) target:

$$G_t^\lambda = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^n,$$

for some coefficient λ regulating the trade-off. When $\lambda = 0$ we recover TD(0) updates, while with $\lambda = 1$ we have the Monte Carlo one. While G_t^λ would require in principle complete trajectories, it is possible to show that properly weighting updates using *eligibility traces* allow to apply G_t^λ also in an online way.

2.4.2 Control

Control methods have the goal of finding the optimal policy or the optimal value function. In a tabular context it is still possible to adopt the traditional optimality definition (see Definition 2.2.6), hence, trying to maximize the value function in *each* state. Here we review two of the most relevant tabular control methods: *SARSA* and *Q-Learning*. They can be seen, in a sense, as the TD version of, respectively, policy and value iteration. However, RL methods have some crucial differences w.r.t. their dynamic programming counterparts:

- updates are based on samples and not on expectation, hence, they introduce estimation errors;
- samples needs to be actively gathered by the interaction with the environment, while expectation are taken thanks to the knowledge of the model.

Therefore, RL agents are forced to continuously explore the environment, trying to improve their value function estimates, but possibly renouncing to exploit the information they have to gain some reward. This trade-off they have to face is known as *exploration-exploitation dilemma*. In order to correctly balancing between these two needs, there are two main options:

- in an **on-policy** setting: forcing the policy used by the agent to be exploratory;

- in an **off-policy** setting: use samples gathered from an explorative behavioral policy to make target policy estimates.

The methods presented here are representative of both these categories.

SARSA: On-policy TD Control The most common on-policy control algorithm is SARSA (Rummery and Niranjan, 1994). This methods is an instance of Generalized Policy Iteration, where TD prediction is used for the policy evaluation part. At each iteration i, the agent follows some policy π_i and obtain an interaction tuple $(s_t, a, r_{t+1}, s_{t+1}, a_{t+1})$, from which the algorithm name derives. This tuple is used to update the Q function in the following TD way:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(r_{t+1} + \gamma Q(s_t, a_t) - Q(s_{t+1}, a_{t+1})\right), \tag{2.28}$$

which can be seen as an application of an *empirical* Bellman Expectation Operation, in which next state and reward are represented by samples and not computed by means of a model.

For what concerns the policy improvement step, this one is modified to guarantee that the resulting policy is exploratory enough. Greedy policies used in standard policy iteration are indeed deterministic, hence, they do not allow to actively explore other actions than the greedy one. A first alternative to the pure greedy is to use an ϵ-greedy policy, which select a random action with probability ϵ and the greedy one with probability $1 - \epsilon$. Otherwise, one can resort to some *soft*-greedy policy, as, for instance the *soft-max* policy[7]. This policy gives to each action a probability that is proportional to $\exp\left(\frac{Q(s,a)}{c}\right)$, where $c \in \mathbb{R}_+$ is a *temperature* parameter, which regulates the policy entropy. In this way, action with an higher value function will be chosen more often, but also the other one will have the chance to be chosen. Both these policies can be made Greedy in the Limit with Infinite Exploration (GLIE) by assuring that either $\epsilon \to 0$ or $c \to 0$. As long as all state-action pairs are visited an infinite number of times, the improving policy is GLIE, and the Robbins-Moore conditions on the learning rate are met, SARSA is guaranteed to converge to the optimal policy Singh et al. (2000).

Q-Learning: Off-policy TD Control One of the most known off-policy tabular control algorithm is Q-Learning. Being off-policy, Q-Learning interacts with the environment with a behavioral policy π_b and then uses the sample gathered by the latter to update the target Q function. Each interaction with the environment allows to obtain a tuple $(s_t, a_t, r_{t+1}, s_{t+1})$[8], which is employed according to the following update rule:

$$Q(s_t, a_t) = r_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a'). \tag{2.29}$$

This rule amounts to the application of an *empirical* Bellman Optimality Operator, in which next state and reward are represented by samples and not computed by means of a model. Q-learning is guaranteed to converge under the assumption that every state-action pair is visited infinitely often and under the Robbins-Moore conditions on the learning rate (Singh et al., 2000).

[7] Also known as Boltzmann distribution

[8] Differently from SARSA update, here we do not care about the next action taken by the behavioral policy.

2.5 Reinforcement Learning: Approximate Methods

Despite its usefulness in reasoning on the fundamental challenges of RL, the finite MDP assumption is often restrictive for most real-world problems. In many of the possible RL application one has instead to deal with infinite state spaces, infinite action spaces or both. An infinite state-action space is clearly problematic for a prediction problem because it becomes impossible to visit *infinitely often* any state-action pair, but, on contrary, it may be difficult even to guarantee to visit each pairs once. Possible workarounds consists in using *discretization* or *aggregation* to map back the problem to the finite MDP setting. However, based on how fine the discretization (or the aggregation) needs to be, the resulting space cardinality may make tabular approaches very ineffective. An alternative consists in using instead *function-approximation* to learn the value function. This approach has the disadvantage of constraining the learned value function to a specific function set, however, it has the great advantage of promoting generalization on unseen state-action pairs. The regularity of the chosen function allows, indeed, to predict similar outputs for similar state-action inputs, providing then satisfactory values even for novel state-action tuples.

From the control viewpoint, standard policy improvement is still viable when only the state space is infinite, and it allows to build the so-called *value-based* control approaches. However, applying GPI scheme becomes problematic when also the action one is infinite. In this case, one needs to solve a (possibly *non-concave*) maximization problem at each improvement step, which may be computationally hard, even discretizing. By restricing to *parametric policy classes*, it is possible instead to substitute this step with a search in the parameter space through the minimization of a suitable loss.

Prediction with approximation and policy search benefits can be put together with *actor-critic* architectures, which build optimization targets for policy search with TD predictions. These algorithms allow at the same time to use continuous action space and to employ a temporal difference approach for the generation of the learning targets, which allows to trade-off between bias and variance, and to exploit the Markovianity of the states.

2.5.1 Prediction with Approximation

Learning how to map a some input to a certain output by means of examples of the desired mapping is a type of tasks which is the object of study by the machine learning area called *supervised learning* (Bishop, 2014). While, in principle, any supervised learning algorithm may be used in an approximated RL prediction task, the reinforcement learning context has a peculiar characteristic that need to be addressed: *non-stationarity*. This features, which complicates the direct application of state-of-the-art supervised approaches, arises from two main sources:

- **Policy improvement**: each time the policy is improved, the target value function changes. While it is possible to issue a new learning task at each policy update, this may be costly in practice, and one may want to carry on the two processes at the same time.
- **Temporal difference learning**: if the first issue can be somehow overcome, this one cannot be easily avoided. Temporal difference learning introduces the value

function inside the target, creating a circular dependency which has to be explicitly taken into account during the learning process.

For these reasons, RL prediction task needs ad-hoc strategies preventing non-stationarity to cause *oscillations* or *divergence* in the learning process. In the following, we will focus, for ease of exposition, on the prediction of the value function: the exposed concepts can be extended to the action value function case in a straightforward way.

Mean Squared Value Error

A first difference w.r.t. to tabular prediction methods is that, we cannot hope anymore for an equal level of value function accuracy for each state. Due to the approximated value function generalization capabilities, what we learn from a particular state will affect also other states predictions. Therefore, we need to specify a scalar objective for our optimization, accounting for our preferences on the state space. The most natural choice is the *mean squared value error*:

$$\overline{VE} \coloneqq \int_{s\in\mathcal{S}} \rho(\mathrm{d}s)\left[V_\pi(s) - V_{\boldsymbol{\omega}}(s)\right]^2, \tag{2.30}$$

where $V_{\boldsymbol{\omega}}$ is some parametric value function approximator, with parameter vector $\boldsymbol{\omega}$, and ρ is some distribution on the state space, which weights the states based on our *interest* on them. Typically, ρ is chosen to be the *on-policy distribution* (see Definition 2.2), but it can be any valid distribution. Ideally, one would want to find a global optimum $\overline{VE}$ on the parameter space, however, this is only guaranteed if the objective is convex, as it happens with linear approximators. In all the other cases, obtaining a local optimum is the best one can hope for. In what follows we focus on two categories of approaches for solving the aforementioned minimization: *stochastic gradient descent* and *fitting*. While the first approach is more suitable for an online context, the second one is probably the best choice when learning happens offline. For both cases, we will mainly focus on the linear setting, for which strong guarantees can be proved.

Stochastic-gradient and Semi-gradient Methods

Stochastic gradient descent (SGD) methods are among the most widely used of all function approximation methods and are particularly appropriate for online reinforcement learning. Their principle consists in adjusting the weight vector based on the gradient of the error function:

$$\nabla_{\boldsymbol{\omega}}\overline{VE} = \int_{s\in\mathcal{S}} \nabla_{\boldsymbol{\omega}}\left[V_\pi(s) - V_{\boldsymbol{\omega}}(s)\right]^2 = -\int_{s\in\mathcal{S}} \rho(\mathrm{d}s)2\left[V_\pi(s) - V_{\boldsymbol{\omega}}(s)\right]\nabla_{\boldsymbol{\omega}}V_{\boldsymbol{\omega}}(s). \tag{2.31}$$

However, since the expectation over the states is typically unfeasible to take, one can instead consider an estimate of the gradient based on the available samples (hence *stochastic*), which, in the simplest case of one sample reduces to the following update rule:

$$\boldsymbol{\omega}_{t+1} \leftarrow \boldsymbol{\omega}_t - \frac{1}{2}\alpha\nabla_{\boldsymbol{\omega}}\left[V_\pi(s) - V_{\boldsymbol{\omega}}(s)\right]^2 = \boldsymbol{\omega}_t + \alpha\left[V_\pi(s) - V_{\boldsymbol{\omega}}(s)\right]\nabla_{\boldsymbol{\omega}}V_{\boldsymbol{\omega}}(s) \tag{2.32}$$

where α is *step-size* (or *learning rate*) parameter. The gradient appearing in the update is exact, since it employs the real value function, however, the latter is not available in practice, and a substitute target should be used instead:

$$\boldsymbol{\omega}_{t+1} \leftarrow \boldsymbol{\omega}_t - \alpha \left[U_t - V_{\boldsymbol{\omega}}(s)\right] \nabla_{\boldsymbol{\omega}} V_{\boldsymbol{\omega}}(s). \tag{2.33}$$

The target U_t may be built as for the tabular prediction case, thus, using Monte-Carlo returns or TD ones. While the first one is an unbiased estimate of the true gradient, the second one is not, hence, methods using this approach instantiated with TD are called *semi-gradient* algorithms.

An interesting choice for function approximators concerns linear ones. A linear function approximator $v_{\boldsymbol{\omega}}$ can be defined as:

$$v_{\boldsymbol{\omega}}(s) = \boldsymbol{\omega}^\top \boldsymbol{\phi}(s), \tag{2.34}$$

where $\boldsymbol{\phi}(s) \coloneqq (\phi_0(s), \ldots, \phi_{d-1}(s))$ denotes the *feature vector* of some state s. A semi-gradient linear TD(0) update rule can be derived as:

$$\boldsymbol{\omega}_{t+1} \leftarrow \boldsymbol{\omega}_t - \frac{1}{2}\alpha \left[r_t + \gamma \boldsymbol{\omega}_t^\top \boldsymbol{\phi}(s_{t+1}) - \boldsymbol{\omega}_t^\top \boldsymbol{\phi}(s_t)\right] \boldsymbol{\phi}_t(s). \tag{2.35}$$

Assumed that the system has reached a *steady state*, then we must have that $\mathbb{E}\,\boldsymbol{\omega}_{t+1} = \mathbb{E}\,\boldsymbol{\omega}_t$, hence, $\mathbb{E}\left[\left(r_t + \gamma \boldsymbol{\omega}_t^\top \boldsymbol{\phi}(s_{t+1}) - \boldsymbol{\omega}_t^\top \boldsymbol{\phi}(s_t)\right) \boldsymbol{\phi}(s_t)\right] = 0$. We obtain then that the fixed point of this equation is:

$$\boldsymbol{\omega}_{TD} = \boldsymbol{A}\boldsymbol{b}^{-1}, \tag{2.36}$$

where $\boldsymbol{A} \coloneqq \mathbb{E}\left[\boldsymbol{\phi}(s_t)(\boldsymbol{\phi}(s_t) - \gamma \boldsymbol{\phi}(s_{t+1})^\top)\right]$ and $\boldsymbol{b}^{-1} \coloneqq \mathbb{E}[r_t \boldsymbol{\phi}(s_t)]$. This point, called *TD fixed point* is in general different from the optimal $\boldsymbol{\omega}$ for $\overline{VE}$, which can be reached by using the Monte-Carlo target instead of the TD one. The potential loss can be bounded by:

$$\overline{VE}_{\boldsymbol{\omega}_{TD}} \leq \frac{1}{1-\gamma} \min_{\boldsymbol{\omega}} \overline{VE}_{\boldsymbol{\omega}}. \tag{2.37}$$

With γ near to one, the loss can become large: this *bias* is the price to pay to avoid the *variance* that one would experience using the Monte-Carlo update.

Least Square Temporal Difference

If the learning process takes place offline, one can avoid using gradient methods to gradually minimize the error, and directly fit the approximator according to the target loss function. This principle is the basis of the Least Square Temporal Difference algorithm (LSTD) (Boyan, 1999). Let's consider a trajectory τ, sampled follow some policy π, with lenght T. Generalizing the previous TD fixed point equation (2.36) terms:

$$\hat{\boldsymbol{A}}_t \coloneqq \sum_{k=0}^{T-1} \boldsymbol{\phi}(s_k) \left(\boldsymbol{\phi}(s_k)^\top - \gamma \boldsymbol{\phi}(s_{k+1})^\top\right)$$

$$\hat{\boldsymbol{b}}_t \coloneqq \sum_{k=0}^{T-1} \phi(s_k) R_k,$$

we can directly obtain the fixed point solution

$$\boldsymbol{\omega}_{TD} = \hat{\boldsymbol{A}}\hat{\boldsymbol{b}}^{-1}. \tag{2.38}$$

This algorithm has the advantage of extracting the maximum amount of information from the given samples, and avoids any hyper-parameters tuning. On the other sides, it requires a considerable amount of computation ($\mathcal{O}(d^3)$, due to matrix inversion) and it is not suitable for the online context in which the target may be non-stationary. It is possible to show that this algorithm converges to the optimal solution with the required approximation in finite time (Lazaric et al., 2012).

2.5.2 Value-Based Control

The main idea for approximated value-based control approaches consists in extending policy or value iteration to sample based versions, respectively, Approximated Policy Iteration (Scherrer, 2014, API) and Approximated Value Iteration (Gordon, 1995; Munos, 2005, AVI). Severeal theoretical works studied the properties of these approximated value-based algorithms, trying to determine how errors due to *estimation* and *approximation* impact the solution quality (Munos, 2005; Farahmand, 2011; Antos et al., 2008; Munos and Szepesvári, 2008). Practical algorithms have also been developed both for the online case, as Deep Q-Network (Mnih et al., 2015, 2016) and the offline case, as Fitted Q-Iteration (Ernst et al., 2005b; Riedmiller, 2005). These techniques have proved to be very effective in solving difficult benchmarks as the Atari games (Mnih et al., 2015) or challenging real-world problems (Castelletti et al., 2010; Bisi et al., 2020a; Riva et al., 2021). While usually adopted for *discrete* action spaces only, extensions have been proposed for the *continuous* case (Antos et al., 2007; Gu et al., 2016).

2.5.3 Policy-Based Control

The control approaches that have been reviewed so far were all based on the GPI paradigm. However, as already pointed out, when the state space is continuous, taking the maximum of the action value function over the action space may be computationally hard, even if one decides to discretize. The alternative consists in employing *policy search* techniques (Deisenroth et al., 2013), in order to look for the best policy in a restricted policy space. Policy search objective is to find the optimal J_ρ (see Definition 2.2.7) in a constrained *parametric* policy set:

$$\max_{\pi_{\boldsymbol{\theta}} \in \Pi_\Theta} J_\rho(\pi_{\boldsymbol{\theta}}). \tag{2.39}$$

If Π_Θ is a space of stochastic and differentiable policies in $\boldsymbol{\theta}$, then the expected return $J_\rho(\pi_{\boldsymbol{\theta}})$ is differentiable in $\boldsymbol{\theta}$ as well. Stochasticity is needed for two main reasons:

- *ensuring exploration*, as already discussed in the previous sections;
- deriving a policy gradient update formula which is *independent* from the transition model[9] (Deisenroth et al., 2013, Section 2.4.1.2), hence suitable for model-free RL.

[9] A possible alternative consists in using the Deterministic Policy Gradient (Silver et al., 2014)

Provided these conditions are satisfied, the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ can be computed thanks to the following result.

Theorem 2.5.1 (Policy Gradient Theorem (Sutton et al., 2000b)). *Let $\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\Theta}}$, If $\pi_{\boldsymbol{\theta}}$ is stochastic and differentiable in $\boldsymbol{\theta}$, then the policy gradient can be expressed as*

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \mathbb{E}_{\substack{s \sim d_{\mu}^{\pi_{\boldsymbol{\theta}}}(\cdot) \\ a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[Q^{\pi_{\boldsymbol{\theta}}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) \right], \tag{2.40}$$

where $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\cdot|s)$ is also known as the policy *score* function. Another formulation for the policy gradient, based on trajectories, can be derived (Peters and Schaal, 2008):

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \mathbb{E}_{\tau \sim p_{\pi_{\boldsymbol{\theta}}}(\cdot)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) G_{\gamma}(\tau) \right]. \tag{2.41}$$

Having access to the gradient allow to cast the RL problem as a *stochastic optimization* problem. A simple approach to solve this problem consists in performing plain *gradient ascent* (Peters and Schaal, 2008). However, it is possible to refine this approach, for instance, pre-multipling the gradient with its Fisher Information Matrix (Amari, 1998), hence following the *natural* gradient direction (Kakade, 2002; Peters et al., 2005).

Policy Gradient Estimators

The policy gradient formula depends, luckily, only on quantities which can be estimated from the environment, without having access to the model, but only to samples. Different gradient estimators can be developed by relying on the above formulas.

REINFORCE The first gradient estimator has been formulated in (Williams, 1992). Considering an offline context, in which one has access to a dataset of n trajectories $\tau_i = (s_0^i, a_0^i, r_1^i, \dots, s_n^i)$ of length T, the following estimator can be derived from Equation (2.41):

$$\hat{\nabla}_{\boldsymbol{\theta}}^{RF} J(\boldsymbol{\theta}) \coloneqq \frac{1}{n} \sum_{i=0}^{n-1} \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(a_t^i|s_t^i) G_{\gamma}(\tau_i), \tag{2.42}$$

where the relationship $\nabla \log p_{\pi_{\boldsymbol{\theta}}}(\tau_i) = \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(a_t^i|s_t^i)$ has been used. It can be shown that subtracting a *baseline* to the return allows to consirably reduce the variance of the estimator (Peters and Schaal, 2008).

PGT and G(PO)MDP The REINFORCE estimator, however, contains some redundant terms that increase its variance. In fact, it multiplies action score also to rewards coming from the past. Intuitively, this can be avoided, since there those reward are independent from future actions. This is exactly what is done when the Policy Gradient Theorem (PGT) estimator is used instead:

$$\hat{\nabla}_{\boldsymbol{\theta}}^{PGT} J(\boldsymbol{\theta}) \coloneqq \frac{1}{n} \sum_{i=0}^{n-1} \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(a_t^i|s_t^i) \sum_{k=t}^{T-1} \gamma^k r_{k+1} = \frac{1}{n} \sum_{i=0}^{n-1} \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(a_t^i|s_t^i) G_t, \tag{2.43}$$

which can be derived from Equation (2.40), and where we define the Monte-Carlo estimate of Q as $G_t = \sum_{k=t}^{T-1} \gamma^k r_{k+1}$. An equivalent estimator called G(PO)MDP

can be also derived (Baxter and Bartlett, 2001). As for REINFORCE, it is possible to further reduce the variance of this estimator by subtracting to the returns a state-dependent baseline (Sutton and Barto, 1998). Although the optimal baseline may be computed Peters and Schaal (2006), a common baseline choice is to subtract some estimator of the value function $V_{\boldsymbol{\omega}}$, if available.

Vanilla Policy Gradient

Performing gradient ascent using either REINFORCE or PGT (with some baseline to reduce variance) allows to obtain a policy search algorithm called *vanilla* policy gradient. The learning rate α may be fixed or may follow a particular schedule (Kingma and Ba, 2014).

2.5.4 Actor-Critic Control

PGT is an unbiased Monte-Carlo estimator of the gradient, however, it suffers from the variance due to estimating Q with samples. Following the Temporal Difference principle, one can instead estimate the Q-function at time t with a TD(0) target:

$$G_t^0 \coloneqq r_{t+1} + \gamma V_{\boldsymbol{\omega}}(s_{t+1}),$$

which can be clearly generalized to the TD(λ) case. As it happens for the prediction task, this choice will bias the estimate but it will reduce the variance, hence, giving in practice an advantage in term of convergence speed. Techniques which uses at the same time a policy search approach and a bootstrapped target fall under the name of *actor-critic* algorithms. A typical choice consists in employing at the same time the value function estimate as baseline and as component of the bootstrapped target:

$$G_t^{A2C} \coloneqq r_{t+1} + \gamma V_{\boldsymbol{\omega}}(s_{t+1}) - V_{\boldsymbol{\omega}}(s_t).$$

Since this target is in practice estimating the advantage function (2.6), it is known as *Advantage Actor Critic* (A2C) approach Sutton and Barto (1998). By carefully choosing the value function approximator it is also possible to avoid to bias the gradient estimate. Functions that enjoy this property are called *compatible value functions*. Sufficient condition to be compatible has been shown in (Sutton et al., 2000b) to be:

$$\nabla_{\boldsymbol{\omega}} f_{\boldsymbol{\omega}}(s,a) = \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s),$$

and

$$\mathbb{E}_{\substack{s\sim d_{\mu,\pi_{\boldsymbol{\theta}}}(\cdot) \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left(\left(A^{\pi_{\boldsymbol{\theta}}}(s,a) - f_{\boldsymbol{\omega}}(s,a) \right) \nabla_{\boldsymbol{\omega}} f_{\boldsymbol{\omega}}(s,a) \right) = \mathbf{0}.$$

These principles allows to obtain more refine approaches as the *natural* actor critic algorithms (Peters et al., 2005; Bhatnagar et al., 2009; Melo and Lopes, 2008).

2.5.5 Safe and Trust-Region Approaches

Policy gradient algorithms are *first-order* methods, since they are not aware of any *curvature* information, as the policy Hessian for instance. While it is possible, in principle, to compute it (Papini et al., 2019), the Hessian is typically computationally expensive to obtain. It is, thus, difficult to choose a proper step-size for policy gradient updates

due both to their *locality* and to the *noise* contained in their estimate. A seminal work by Kakade and Langford (2002) though, establish a fundamental relationship between policies performances.

Lemma 2.5.2 (Performance Difference (Kakade and Langford, 2002))**.** *Let $\pi, \widetilde{\pi} \in \Pi^{SR}$, then their performance difference is:*

$$J_{\widetilde{\pi}} - J_{\pi} = \int_{\mathcal{S}} d_{\mu}^{\widetilde{\pi}}(s) \int_{\mathcal{A}} \widetilde{\pi}(a|s) A_{\pi}(s,a) \, \mathrm{d}a \, \mathrm{d}s, \tag{2.44}$$

which, with further derivations, permits to lower bound the performance of some policy $\widetilde{\pi}$ using the performance and the advantage function of π. This result motivated the first safe RL approach, Conservative Policy Iteration (CPI), which optimized a lower bound derived from the lemma, in order to obtain *monotonical improvement guarantees* for approximated policy iteration. Later works extended this approach to policy search an derived guarantees allowing to choose safe step-sizes for policy gradient updates (Pirotta et al., 2013, 2015; Papini et al., 2017, 2019). This lemma has revealed important also in studying the *dominance* property of gradient approches in RL (Agarwal et al., 2021), which are fundamental to devise global optimality guarantees.

Unfortunately, algorithms derived by a rigorous application of theoretical bounds have typically a low learning speed, which make them unsuitable for high-dimensional complex tasks. However, this stream of research revealed some basic principles that inspired the development of more practical algorithms, generically called *trust-region approaches* (Schulman et al., 2015a, 2017; Metelli et al., 2018, 2020). The ratio between these techniques is guaranteeing that the next target policy is *close enough* (from a distributional point of view, thus, in trust-region) to the previous one, in order to exploit the samples previously gathered with the latter to offer a reliable estimate of the target one.

Trust-Region Policy Optimization

Here we provide some details for one of the most successful state-of-the-art algorithms: Trust Region Policy Optimization (TRPO) Schulman et al. (2015a). The main idea consists in optimizing a surrogate of the real objective:

$$L_{\pi}(\widetilde{\pi}) = J(\pi) + \int_{\mathcal{S}} d_{\mu}^{\pi}(s) \int_{\mathcal{A}} \widetilde{\pi}(a|s) A_{\pi}(s,a) \, \mathrm{d}a \, \mathrm{d}s,$$

which directly derives from Lemma 2.5.2. Then the following lower bound holds:

$$J() \geq L_{\pi}(\widetilde{\pi}) - C D_{KL}^{max}(\pi, \widetilde{\pi}),$$

where $C = \frac{2\epsilon\gamma}{(1-\gamma)^2}$, $\epsilon = \max_s |\mathbb{E}_{a \sim \widetilde{\pi}(\cdot|s)}[A_{\pi}(s,a)]|$, and D_{KL}^{max} is the maximum KL divergence over states. The practical algorithm does not exactly optimizes the lower bound, but instead optimizes the surrogate objective with a tunable constraint on the KL between successive policies. In practice, this has revealed to be very effective, producing good results in challenging high-dimensional complex environments (Schulman et al., 2015b).

2.6 Multi-Objective RL

Some class of control problems are difficult to frame as Markov Decision Process because they involve the optimization of multiple objectives. Two frameworks have been developed to deal with this kind of problems, extending the original MDP one:

- the Multi-Objective MDP (MOMDP) framework: which involves the simultaneous maximization of multiple criteria;
- the Constrained MDP (CMDP) framework: which involves the maximization of a single objective, but constraining the other ones.

In this section, we will describe some important results for the first class of problems, which are discuss in depth in (Roijers et al., 2013). For a complete dissertation on the second setting instead, the interested reader can refer to (Altman, 1999).

2.6.1 Multi-Objective MDPs

A multi-objective MDP (MOMDP) is an MDP in which the reward function $\mathbf{R} : S \times A \to \mathbb{R}^n$ describes a vector of n rewards, one for each objective, instead of a scalar. The expected return $\mathbf{J}_\pi$ in an MOMDP specifies the expected cumulative discounted reward vector

$$\mathbf{J}_\pi \coloneqq \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu(\cdot) \\ s_{t+1} \sim P(\cdot|s_t, a_t) \\ a_t \sim \pi(\cdot|s_t)}} \left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}_t \right],$$

and the multi-objective value of a state is described as

$$\mathbf{V}_\pi(s) \coloneqq \mathop{\mathbb{E}}_{\substack{s_{t+1} \sim P(\cdot|s_t, a_t) \\ a_t \sim \pi(\cdot|s_t)}} \left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}_t | s_0 = s \right],$$

While according to the *reward hypothesis* (Silver et al., 2021) a scalar reward function should be adequate for all sequential decision-making tasks, this view does not imply that multi-objective problems do not exist. On contrary, this assumption would imply that MOMDPs can always be converted into single-objective MDPs with additive returns.

Definition 2.6.1 (Scalarization Function)**.** *A scalarization function f is a function that projects the multi-objective reward $\mathbf{R}$ to a scalar value:*

$$V_\pi^{\boldsymbol{\omega}}(s) = f(\mathbf{V}_\pi(s), \boldsymbol{\omega}), \tag{2.45}$$

where $\boldsymbol{\omega}$ is a vector parametrizing f.

An example of scalarization function is the linear one $f(\mathbf{V}, \boldsymbol{\omega}) = \boldsymbol{\omega}^\top \mathbf{V}$. Once defined the scalarization, one could try to identify a single-objective MDP with additive returns such that, its expected reward is equal to $V_\pi^{\boldsymbol{\omega}}$. However, there are scenarios in which this is not always possible or desirable:

- *unknown weights*: in this case, weights are unknown a priori;

- *unknown scalarization*: in this case the scalarization function cannot be easily specified;
- *non-linear scalarization*: in this case function and weights are known, but the non-linearity of scalarization make the process more difficult.

The MDP framework struggles to give an answer to any of those questions. Moreover, since scalarization cannot be even specified in the first two cases, multiple policies should be provided as solution, in order to provide alternatives to the decision maker. We denote this scenario, grouping the first two settings, as the *multiple policies* one.

2.6.2 Optimization Criteria for MOMDPs

In order to select the best policies in the set, an optimization criterion has to be formulated.

Definition 2.6.2 (Undominated Policies)**.** *For an MOMDP $\mathcal{M}$ and a scalarization function f, the set of undominated policies, and a set of policies $\bar{\Pi}$, $U(\bar{\Pi})$, is the subset of all possible policies in $\bar{\Pi}$ for which there exists a $\boldsymbol{\omega}$ such that the scalarized value is maximal.*

This set contains all the *undominated* policies, hence, obtaining this set corresponds to solving the multi-objective problem, when multiple policies are sought. However, as noticed in (Roijers et al., 2013), it may contain redundant policies that, while optimal for some weights, are not the only optimal policy in the set for w. A set containing at least one optimal policy for each value of the scalarization weights is called a *coverage set*.

Definition 2.6.3 (Coverage Set, (Roijers et al., 2013))**.** *For an MOMDP $\mathcal{M}$ and a scalarization function f, and a set of policies $\bar{\Pi}$, a set $CS(\bar{\Pi})$ is a* coverage set *if it is a subset of $U(\bar{\Pi})$ and if, for every $\boldsymbol{\omega}$, it contains a policy with maximal scalarized value.*

Based on the considered scenarios and, possibly, the policy set one is interested in (e.g.: deterministic or stochastic, stationary or non stationary...), it is possible to define a taxonomy of MOMDP problems. While a complete picture is given in (Roijers et al., 2013), here we limit to highlight two interesting results about the *multiple policies* case. The first one is related to the case in which a linear scalarization function is employed $f(\mathbf{V}, \boldsymbol{\omega}) = \boldsymbol{\omega}^\top \mathbf{V}$. Before presenting it, we need to define two other sets.

Definition 2.6.4. *For an MOMDP $\mathcal{M}$, and a set of policies $\bar{\Pi}$, the* convex hull (CH) *is the subset of $\bar{\Pi}$ for which there exists a $\boldsymbol{\omega}$ such that the* linearly *scalarized value is maximal.*

This set contains also redundant policies w.r.t. the linear scalarization criterion, hence, a coverage set can be defined.

Definition 2.6.5. *For an MOMDP $\mathcal{M}$, and a set of policies $\bar{\Pi}$, the set $CCS(\bar{\Pi})$ is a* convex coverage set *if it is a subset of $CS(\bar{\Pi})$ and if, for every $\boldsymbol{\omega}$, it contains a policy such that the* linearly *scalarized value is maximal.*

The following proposition highlight how this set is fundamental for MOMDP optimization.

Proposition 2.6.6 (Linear Scalarization)**.** *When a linear scalarization function is applied $f(\mathbf{V}, \boldsymbol{\omega}) = \boldsymbol{\omega}^\top \mathbf{V}$ to a MOMDP $\mathcal{M}$, in the multiple policy case, $CCS(\Pi_{SD})$ is sufficient for optimality.*

Thus, if a linear scalarization is provided, the convex hull of stationary deterministic policies is enough, but, maybe surprisingly, it is sufficient for a larger set of problems.

Proposition 2.6.7 (Monotonically Increasing Scalarization)**.** *When a monotonically increasing scalarization function is applied to a MOMDP $\mathcal{M}$, in the multiple policy case, if* stochastic *policies can be employed, the* convex hull *of deterministic stationary policies is still sufficient for optimality.*

This result was obtained in (Vamplew et al., 2009), where it was also shown that the convex hull of deterministic stationary policies can be obtained by *mixing* policies from $CCS(\Pi_{SD})$. Therefore, linear scalarization can be used also when the actual scalarization function is non-linear (but monotonically increasing) with the goal of recovering the sufficient set of optimal policies. If instead only deterministic policies are sought, one has to resort to the Pareto front solution concept.

Definition 2.6.8. *A policy π* Pareto-dominates *another policy π' when its value is at least* as high *in all objectives and* strictly higher *in at least one objective. The* Pareto front *is the set of all policies that are not Pareto dominated by any other policy.*

Multi-Objective Reinforcement Learning (MORL) approaches try to find *approximation* of those solution set. Both value based (Castelletti et al., 2011) and policy based (Parisi et al., 2014) approaches are available, with a recent growing interest in applying hybrid solutions involving deep reinforcement learning and evolutionary methods (Xu et al., 2020a).

CHAPTER 3

Risk-Aversion in Reinforcement Learning

3.1 Introduction

The object of interest of reinforcement learning is the discounted cumulated reward, also called *return*, which has to be maximized by the agent. In particular, the standard RL framework has the objective of maximizing the *expected value* of the return. However, there are contexts in which one is not only interested in the mean of the return, but also in the extent to which the return random variable can variate around its mean. This variability, usually studied in statistics by means of *dispersion measures*, is labelled as *risk* in optimization contexts. An agent which is somehow influenced by the return variability is called *risk-sensitive*. In case the agent exhibits a preference towards high variability scenarios it is called *risk-seeking*, otherwise, if it prefers to keep the risk low, it is tipically referred to as *risk-averse*. On the other hand, an agent which is indifferent to risk, but simply wants to optimize the expectation of the return is named *risk-neutral*: these kinds of agents are the focus standard reinforcement learning. While it would be difficult to understand why one should want to explicitly prefer a risk-seeking policy w.r.t. to a risk-neutral one, risk-seeking behaviors may be observed as a result of agent trying to explore the state-action space.

A risk-averse preference is instead more understandable, since peculiar of human (and animal) reasoning. This tendency is a well-established phenomenon, which is fundamental for modelling decision-making in economics (Bernoulli, 2011), ethology (Kacelnik and Bateson, 1996), and neuroscience (Platt and Huettel, 2008). It consists in preferring more certain outcomes w.r.t. uncertain ones, even when uncertainty may be connected to an higher profit on average. Recent studies have shown that such disparity can be even spot at the neural level (Niv et al., 2012), with negative events weighed more than positive ones, and with structures dedicated to distributional learning (Dabney et al., 2020). Risk-aversion can also be seen as a *distorted* way of perceiving

random events (Wang, 1996), giving more *weight* to the *negative* ones. The reason why we study this kind of behavior is that there are contexts where variability may have catastrophic effects. To see this, one can try to think of settings in which a single catastrophic event may preclude any successive trial. For a robot exploring the surface of an unknown planet, a failure is sufficient to invalidate the entire mission. An investment bank that loses a huge amount of money for a single wrong decision is going to fail, no matter how much its past decisions were profitable. An autonomous car is expected to drive safe always, and not "on average". More in general, for all high-stakes applications as health-care, finance or autonomous driving, it is mandatory to select policies that limits performance oscillations, because the price to pay for each deviation from the nominal behavior is high.

While risk and safety may have several different meanings (García and Fernandez, 2015), in this chapter we specifically focus on the *inherent risk*. This kind of risk captures the variability of the performance which is due to the inherent *stochastic nature* of the environment, which is absent in deterministic environments (e.g. in board games as chess), but present in many real applications. In particular, in what follows we will describe the main approaches that have been developed for modelling risk, and we will survey the techniques developed to solve each of the corresponding risk-averse objectives.

3.2 Risk-Averse Objectives

In this section we analyse different approaches that have been developed for dealing with risk-aversion. While finance literature has focused on developing functionals for directly *measuring risk*, the economics one has instead studied how to express this attitude trough utility functions that can map uncertain outcomes to some *certain equivalent*. Finally, control literature has instead explored worst case optimization or robust control, which can be seen as an extreme degree of risk-aversion, in which one wants to protect himself from the worst possible realization. As it will be shown, robust optimization has connections with some risk-measures and utility functions.

Developing the concept of risk-measure has been an object of study for the *financial mathematics* literature. Formally, a *risk-measure*[1] or *risk-averse objective* can be defined as the following mapping:

$$\eta : \mathcal{G} \to \mathbb{R}, \tag{3.1}$$

where $\mathcal{G}$ is the is a set of all real valued function (*return functions*) on T, the set of all possible trajectories. The convention is important here:

- The random variable G may be intended as a cumulated cost to be minimized or as a cumulated reward to be maximized depending on the context. While the control literature usually adopts the cost convention, the finance one usually talks about monetary gains, and considers maximizing them. In reinforcement learning both views are present, with a preference towards the reward one.

[1] Sometimes, to be considered a risk-measure, a mapping of this kind is required also to enjoy a list of additional properties, which we do not take into account in this work. Risk-measures are organized according to a particular taxonomy, based on their mathematical properties, which is connected, but not overlapped to another stream of literature studying *general deviation* (Rockafellar et al., 2006). Since our focus is the application of those concepts to reinforcement learning, in what follows things will be kept simpler, borrowing from this field only the concepts that have been in practice applied to the RL framework.

Table 3.1: *Consider the two payment methods, A and B. With A, we pay \$1,000 today (Day 0). With B, we might pay \$1,000 on each of the consecutive 20 days starting from today, but the payment is needed only with probability 0.0475. The expected amount of total payment with B is \$950, which is smaller than the \$1,000 with A. However, B would require a huge amount of \$20,000 with nonnegligible probability. Hence, decision makers who are averse to risk would prefer A to B. However, no utility function can induce this behavior using DEU.*

	Day 0	Day 1	...	Day 19	Probability
A	\$1K	\$0	...	\$0	1.0
B	\$1K	\$1K	...	\$1K	0.0475
	\$0	\$0	...	\$0	0.9525

- η as well may be seen as an risk to minimize, or as an objective to maximize.

Even inside the same area sometimes different conventions are used[2]. In this manuscript we will keep the *maximization* convention for both the reward and the objective, in order to avoid confusion, translating the results from literature in the appropriate way when necessary[3].

3.2.1 Utility Functions

One way to take risk into account consists in defining a function named *utility* that maps an uncertain outcome to a *certainty-equivalent*. This approach has a long tradition (Muliere and Parmigiani, 1993), and has been formalized in (Morgenstern and Von Neumann, 1953). In these works, the utility framework is obtained as a result of particular axioms, which should guarantee the rationality of the agents (though this aspect has been criticized, for instance in (Allais, 1990)). A utility function may be either a function of the reward or a function of the return U, generating the following risk-averse objectives:

$$\eta_{DEU} \coloneqq \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} U(r_t)\right], \tag{3.2}$$

$$\eta_{EUD} \coloneqq U^{-1}\mathbb{E}_{\pi}\left[U\left(\sum_{t=0}^{\infty} r_t\right)\right], \tag{3.3}$$

where we followed the convention of (Osogami, 2012a), in which objectives in the form of (3.3) are called *Discounted Expected Utility* (DEU), while, if they have (3.3) form, they are called *Expected Utility of the Discounted return* (EUD). In both cases, using as a utility function the identity reduces the problem to the risk-neutral case. In particular, for any utility function, DEUs can be seen as reward transformations of the risk-neutral MDP, thus, their optimization does not require to adopt ad-hoc approaches. On contrary, in general, the application of a utility function to the return outputs a new problem which is not an MDP anymore. As shown in (Osogami, 2012a) and reported in Table 3.1, using DEU is impossible to capture some risk-averse preferences though. In fact, considering only the immediate reward, such functions cannot take into account the utility of the cumulated rewards. The example in Table 3.1 shows that standard MDPs alone struggle to be a good model for risk-averse contexts.

[2]The CVaR criterion is emblematic of this issue: in some case maximized, minimized in other ones.

[3]We notice that the term *risk-measure* is usually employed in minimization contexts from the finance literature.

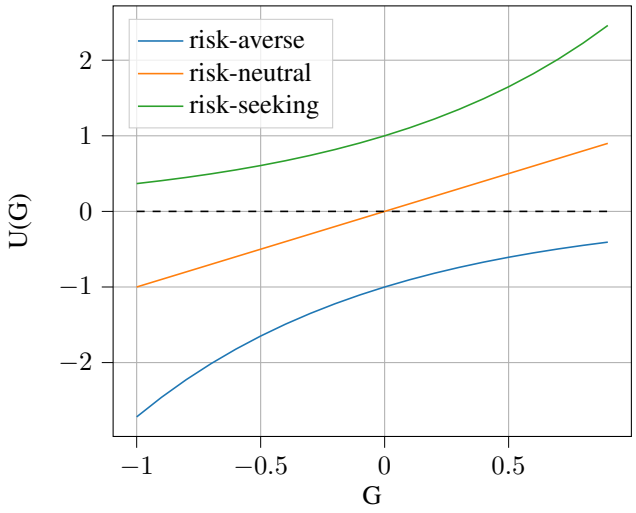

Figure 3.1: *A different concavity in the utility function corresponds to a different attitude toward risk. In a maximization perspective, a (strictly increasing) concave function weights more lower returns, while the opposite happens with a convex one.*

Taking into account the EUD, one can consider as a valid utility any *strictly increasing function* such that its inverse exists. Considering a Taylor expansion of η_{EUD}, it can be shown that (Bäuerle and Rieder, 2013):

$$\eta_{EUD} = U^{-1}\mathbb{E}_\pi[U(G)] \approx \mathbb{E}_\pi[G] - \frac{1}{2} l_U(G)\mathbb{V}ar[G] \tag{3.4}$$

where

$$l_U(G) := -\frac{U''(G)}{U'(G)} \tag{3.5}$$

is the *Arrow-Pratt* function of absolute risk-aversion. The denominator is always positive, since U is increasing, while the numerator sign depends on U being either concave or convex. In a maximization perspective, if U is strictly concave then the agent is risk-averse, while if U is strictly convex then the agent is risk-seeking[4]. Figure 3.1 shows these three kind of attitude towards risk, by means of utility functions with different concavities. In the risk-averse case η_{EUD} is also known as the *certainty-equivalent*.

An equivalent theory has been developed in (Yaari, 1987) and further extended in (Wang, 1996). Under some different axioms, the policy of a risk-averse agent behaves as to maximize a *distortion risk-measure* h, since it distorts the distribution of the return (Dabney et al., 2018b):

$$\pi(s) = \arg\max_{a\in\mathcal{A}} \int_{-\infty}^{\infty} G \frac{\partial}{\partial G}(h \circ F_{G(s,a)})(G)\, \mathrm{d}G \tag{3.6}$$

where h has to be a continuous monotonic function, and $F_{G(s,a)}$ is the cumulative distribution of the returns starting from state s and action a. The choice between these two frameworks amounts to choose whether the behavior should be invariant to mixing with random events or to convex combinations of outcomes (Dabney et al., 2018b; Yaari, 1987).

[4]The two interpretations should be swapped with a minimization perspective

Entropic Risk Measure A typical utility function is the *exponential* one:

$$U(G) = -\frac{1}{\lambda}\exp(-\lambda G). \tag{3.7}$$

In this case we have $l_U = \lambda$, which reduces the Taylor approximation in Equation (3.5) to the *Variance penalized* criterion. This particular version of the EUD criterion has received large attention in the control literature (Howard and Matheson, 1972; Bäuerle and Rieder, 2013; Osogami, 2012a,b; Nass et al., 2019; Marcus et al., 1997; Chung and Sobel, 1986). Optimizing this utility is equivalent to pursuing the maximization of this objective:

$$\eta^{ERM}_{\beta,\boldsymbol{\theta}} = -\frac{1}{\beta}\log \mathop{\mathbb{E}}_{\tau\sim p_{\pi_\theta}(\cdot)}\left[\exp(-\beta G(\tau))\right], \tag{3.8}$$

which is known as the Entropic Risk Measure (Föllmer and Schied, 2011). As pointed out in (Mihatsch and Neuneier, 2002), for infinite horizon tasks with discounted formulation, differently from the risk-neutral case, the optimal policy is **non stationary** in general.

Other Utilities and Distortions Even if the exponential utility is the most common one, many other ones have been developed. For instance, in (Bäuerle and Rieder, 2013) the power utility function $U_P(G) = \frac{1}{\lambda}G^\lambda$ is suggested, which is risk-averse when $\lambda < 1$, and risk-neutral for $\lambda = 1$. The latter is employed also in (Shen et al., 2014) in a financial RL application, though with a modification of the expectation operator. In (Moldovan and Abbeel, 2012) the authors optimize an ad-hoc utility, called *Chernoff functional*, with favourable mathematical properties, and they show that the obtained solutions are optimal also under the exponential utility.

For what concerns the distortion formulation, a possible distortion function is the *cumulative probability weighting* (CPW):

$$U_\lambda(G) \coloneqq \frac{G^\lambda}{(G^\lambda + (1-G)^\lambda)^{\frac{1}{\lambda}}}, \tag{3.9}$$

which has been proposed in cumulative prospect theory (Tversky and Kahneman, 1992). The work in (Wu and Gonzalez, 1996) found that the parameter value $\lambda = 0.71$ is the close to match human subjects. This choice is neither globally convex nor concave: for small values of G it is locally concave and for larger values of G it becomes locally convex. Other possible alternatives consists in a risk-measure proposed in (Wang, 1996)

$$U_W(G)^\lambda \coloneqq \mathcal{N}(\mathcal{N}^{-1}(G) + \lambda),$$

where $\mathcal{N}$ is the normal distribution, and by the CVaR (see Section 3.2.3), which can be also interpreted as a distortion measure.

3.2.2 The Mean-Variance Criteria

Another way to deal with risk consists in taking into account both the expected return J_π and its variance σ_π to evaluate the performance of an agent. This idea traces back to (Markowitz, 1952), a seminal work for modern porfolio theory. The central insight of this work consists in recognizing that investors benefit from diversification because

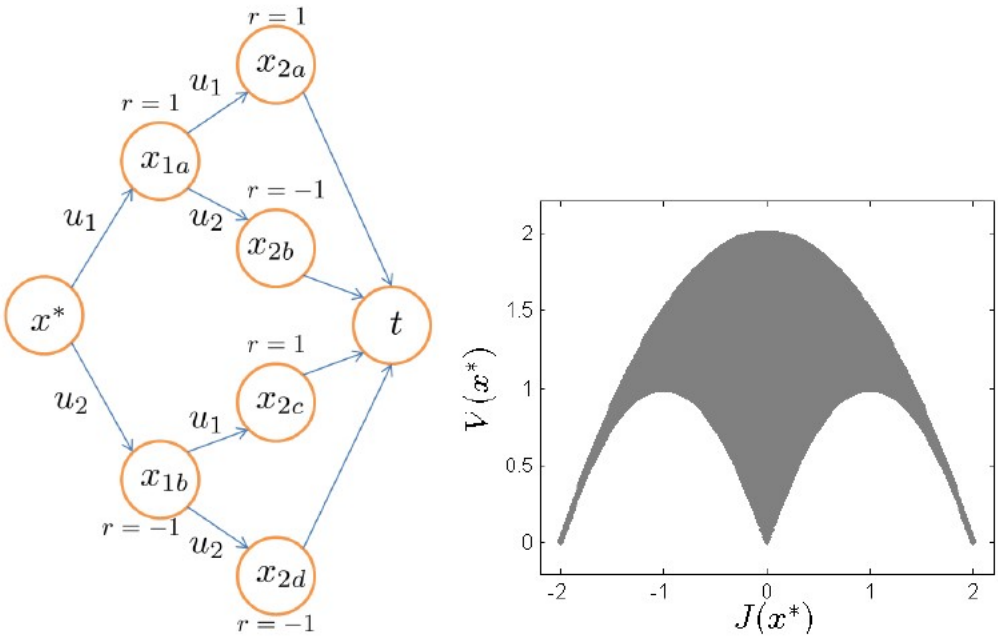

Figure 3.2: *On the left the example reported in (Tamar et al., 2012b). On the right, the variance and expected return values for each possible policy: non-convex regions are present.*

it allows them to minimize variance. This kind of perspective, called *Mean-Variance* analysis, is still widely adopted by the finance community, thanks to its simplicity and ease of interpretation. Formally, with the term *variance of the return* we denote the following quantity:

$$\sigma_\pi^2 := \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu \\ a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) - J_\pi \right)^2 \right]. \tag{3.10}$$

However, the Mean-Variance trade-off does not match a single risk-averse objective, but it represents instead a family of optimization criteria. We list here the most common ones:

- The **multi-objective optimization**[5] of the expected value objective (maximized) and the variance one (minimized). In this case, one can be either interested in solving the problem for a specific preference, or in retrieving the Pareto frontier or an approximation for it.
- One of these **constrained optimization problems**:

$$\max_\pi J_\pi \text{ s.t. } \sigma_\pi^2 \leq c_1,$$
$$\min_\pi \sigma_\pi^2 \text{ s.t. } J_\pi \geq c_2, \text{ or}$$
$$\max_\pi J_\pi \text{ s.t. } \sigma_\pi^2 \leq c_1 \text{ and } J_\pi \geq c_2, \text{ provided a solution exists}$$

[5] This multi-objective problem cannot be mapped to a Multi-Objective MDP as formalized in Section 2.6.1

- The maximization of the **penalized objective**:

$$\eta_{MV}^{\pi,\lambda} \coloneqq J_\pi - \lambda\sigma_\pi^2. \tag{3.11}$$

- The maximization of the Sharpe ratio[6]:

$$\eta_S^\pi \coloneqq \frac{J_\pi}{\sigma_\pi}. \tag{3.12}$$

These objective are not independent from each other of course, and some of them are more general than others. For instance, if one had access to the Pareto frontier, she would be able also to solve all the other problems. Similarly, iteratively solving for many values of c_1 and c_2 one the constrained problems, one could also recover the Pareto frontier with the desired precision, and the same is valid for the problem in which both constraints are present (Mannor and Tsitsiklis, 2011).

Solving for the penalized objective in Equation (3.11) for many values of the parameter λ instead does not always allow us to obtain the complete frontier. To see this consider the MDP shown in Figure 3.2, taken from (Tamar et al., 2012b). It can be noticed that for this problem the Mean-Variance frontier presents concave regions. Thus, differently from the MORL case (see Section 2.6.1), linear utilities, like the ones used for the penalized criterion, cannot be employed to recover the frontier in its entirety. In fact, by varying the parameter λ one can obtain all the points belonging to the convex hull of the frontier, but one cannot recover the un-dominated points belonging to the concave part of frontier, hence, losing some possibly interesting trade-off policies. Finally, optimizing the Sharpe ratio, Equation (3.12), does not allow to regulate the risk-aversion level and returns instead a single policy, which corresponds to a specific point on the Mean-Variance frontier[7].

Policy Evaluation and Improvement. This criterion was early considered also in the MDP literature, taking to the formulation of a Bellman Expectation Operator for the variance value function (Sobel, 1982). While the obtained equation is not linear as the classical one, policy evaluation w.r.t. variance can still be solved efficiently using dynamic programming [8]. In the same work the author also provides the Bellman Expectation Operators for higher moments value functions. We recall here the relationship between the variance and the second moment of any random variable X:

$$\sigma^2[X] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2. \tag{3.13}$$

This equation means that the second moment value function Bellman equation, in conjunction with the standard value function one, can also be used for variance prediction. This approach has been pursued with temporal difference approaches (Tamar et al., 2016b; Sherstan et al., 2018).

However, things are more difficult for policy improvement. As noted in the same work, the variance value function lacks a fundamental property called *consistent choice*

[6] We notice here that the objective involves the *standard deviation* and not the variance as the previous one. In finance, usually this index is defined subtracting to the numerator some reference *risk-free asset* return

[7] This point has a special meaning in portfolio allocation: it is the *tangency portfolio*, which optimizes the allocation when a risk-free asset is present.

[8] The procedure involves the inversion of two matrices instead of one.

which is instead possessed by the standard value function and it is necessary to apply dynamic programming techniques for policy improvement (Denardo, 1967). More recently, it was shown in (Mannor and Tsitsiklis, 2011) that the solution of the mean-variance constrained problem is indeed NP-hard. This result does not leave much hope for the discovery of tractable approaches to obtain Mean-Variance global optima, but does not prevent employing local approaches instead. A stream of risk-averse RL literature developed actor-critic algorithms for the solution of either the penalized criterion (Tamar and Mannor, 2013; Prashanth and Ghavamzadeh, 2013, 2014) in Equation 3.11, or the Sharpe ratio one (Moody and Saffell, 2001) in Equation (3.12).

Mean-Second Moment trade-off To circumvent the difficulties due to the optimization of the Mean-Variance criterion, one could consider instead the trade-off between the first and the second moment of the return (Kato and Nakagawa, 2020). It has been proved in (Baron, 1977) that, under the assumption of an agent is rational, w.r.t von Neumann-Morgenstern axioms, the two criteria are indeed equivalent. The key assumption is the following one: if an agent is indifferent between two strategies (it has the same utility for them), then it must be indifferent also to any *lottery* (convex combination) between them. Removing this assumption, the problems are not equivalent anymore, and it may be shown that a point on the Mean-Second Moment frontier may be dominated in the Mean-Variance one when the latter is concave.

3.2.3 Coherent Risk-Measures

Measures of risk have been developed from financial mathematics to be used as extra capital requirements to regulate the risk assumed by market participants. In simpler terms, a risk-measure can be seen as the necessary amount of cash which is necessary to avoid a failure, or, in general, an undesirable situation, given some underlying probability distribution over the possible outcomes. Therefore, having reserved the correct amount of money to shield against the negative situations may allow to make *accettable* any former *unaccettable* outcome. This notion of acceptable situations has been formalized in (Artzner et al., 1999), where a set of axioms to guarantee the *rationality* of such measures has been derived. Considering G as the usual return random variable, we say that a *risk-measure* η is a *coherent risk-measure* (CRM) only if it satisfies the following properties:

A_1) **Translation Invariance**: *For all* G *and all real numbers* α*, we have* $\eta(G + \alpha) = \eta(G) - \alpha$.

A_2) **Subadditivity**: *For all* G_1 *and* G_2*,* $\eta(G_1 + G_2) \leq \eta(G_1) + \eta(G_2)$.

A_3) **Positive homogeneity**: *For all* $\lambda \geq 0$ *and all* G*,* $\eta(\lambda G) = \lambda\eta(G)$.

A_4) **Monotonicity**: *For all* G_1 *and* G_2*, with* $G_1 \leq G_2$*, we have* $\eta(G_2) \leq \eta(G_1)$.

Such axioms allow any CRM to fullfill some properties, for instance:

- Translation invariance ensures that adding the risk value to G brings the risk to zero.
- Subadditivity avoids that merging creates extra risk.

- Monotonicity tells that an outcome that is always more profitable than another one, cannot be at the same time riskier.

Moreover, CRMs enjoy a *representation theorem* (Artzner et al., 1999), which allows them to be characterized in the following way:

$$\eta_\pi(G) = \min_{\xi:\xi P_\pi \in U(P_\pi)} \mathop{\mathbb{E}}_{\xi}[G], \tag{3.14}$$

where $U(P_\pi)$ is the so-called *risk-envelope* of the CRM, and we have adopted the *maximization* convention, differently from what is usually done in literature. Unfortunately, some of the most common ways to characterize risk such as the Mean-Variance criterion or the utility formulation (3.3), are not coherent according to this framework. We will describe instead in what follows some examples of CRMs.

CVaR The Value-at-Risk (VaR) at level $\alpha \in (0, 1)$ of G is:

$$\rho^\pi_{\mathrm{VaR}}(G; \alpha) \coloneqq \sup\{x : \mathbb{P}^\pi \{G \leq x\} \leq \alpha\}. \tag{3.15}$$

VaR is a popular risk measure, e.g., in finance, which identifies the worst α-quantile of the return distribution. This measure, however, has the disadvantage of overlooking the losses suffered beyond the quantile treshold. Moreover, due to its nature, it is unstable to small variations of the parameter α. To overcome the issues of the VaR risk measure the Conditional Value-at-Risk (CVaR)[9] has been defined in Rockafellar et al. (2000) as:

$$\eta^\pi_{\mathrm{CVaR}}(G; \alpha) = \min_{\eta \in \mathbb{R}} \{\rho - \frac{1}{\alpha}\mathbb{E}^\pi \left[(G - \rho)^-\right]\}, \tag{3.16}$$

where it can be shown that the former quantity is maximized at $\rho = \rho^\pi_{\mathrm{VaR}}(G; \alpha)$. When no probability atoms are present, the previous definition is equivalent to the following one:

$$\eta^\pi_{\mathrm{CVaR}}(G; \alpha) \coloneqq \mathbb{E}^\pi \left[G | G \leq \rho^\pi_{\mathrm{VaR}}(G; \alpha)\right]. \tag{3.17}$$

This measure has the intuitive meaning of quantifying the losses encountered in the tail of the distribution. Moreover, the CVaR, differently from VaR, is a CRM (Artzner et al., 1999), and a distortion risk-measure (Wang, 1996).

Mean-Deviation and Mean-Lower-semideviation It is possible to show that axiom A_4 rules out the mean-variance penalized criterion from the CRMs. Consider two uniform random variables G_1 and G_2, respectively with support $[a, b]$ and $[c, d]$, with $a < b < c < d$. G_2 is always greater than G_1, hence it has a larger mean, but, if $b - a < d - c$, it also has a larger variance. Therefore, one can find a risk-aversion factor that violates the *monotonicity* assumption. One can consider alternatives to them mean-variance penalized criterion, by substituting the variance with another measure. Some candidates are the return deviation σ or the return lower-semideviation σ_-, which can be obtained, respectively, by taking the square root from the variance or the semivariance, defined as:

$$\sigma^2_- \coloneqq \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu \\ a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) - J_\pi \right)^2_- \right].$$

[9] CVaR has many names, it is also known as Expected Shortfall, Average Value at Risk, or Conditional Tail Expectation.

Trade-off risk-measures as:

$$\eta_{MD}^{\pi,\lambda} \coloneqq J_\pi - \lambda\sigma_\pi. \tag{3.18}$$

$$\eta_{ML}^{\pi,\lambda} \coloneqq J_\pi - \lambda\sigma_{\pi,-}. \tag{3.19}$$

are *coherent* only with bounded risk-aversion factor values. For η_{MD} and η_{ML} the valid intervals are, respectively, $[0, \frac{1}{2}]$ and $[0, 1]$. The lower-semideviation, in particular, is a *downside risk-measure* (Danielsson et al., 2006; Dhaene et al., 2004; Spooner and Savani, 2020), since it accounts only for deviations that contribute *unfavourably* to risk.

Dynamic Markov CRM The risk measures taken into account so far are also called *static*, since they do not consider the temporal structure of the random variable. *Dynamic* risk-measures (Iancu et al., 2015) may be defined, by allowing the specification of a different risk mapping for each time-step t: $\eta_t : \mathcal{G} \to \mathbb{R}$. This is particularly useful when one wants to guarantee the *time-consistency* of a risk-measure. Following (Iancu et al., 2015), we define a dynamic risk-measure η as time-consistent when:

$$\forall G_1, G_2,\ \eta_{t+1}(G_1) \geq \eta_{t+1}(G_2) \Rightarrow \eta_t(G_1) \geq \eta_t(G_2). \tag{3.20}$$

Static and time-inconsistent risk-measures do not fulfill this property, which may result in irrational behaviors, as pointed out in (Osogami, 2012a; Iancu et al., 2015). The dynamic *Markov Coherent Risk-Measures*, introduced in (Ruszczyński, 2010), are defined as:

$$\eta_T = \mathop{\mathbb{E}}_{\tau\sim p_\pi(\cdot)}\left[r_0^\tau + \gamma\eta(r_1^\tau + \cdots + \eta(r_{t-1} + \gamma\eta(r_t)))\right], \tag{3.21}$$

where r_t indicates the reward, and η is some static risk-measure which is applied stage-wise. This compositional form allows a recursive estimation of the risk, and time consistency. Moreover, it can be optimized with dynamic programming, by employing the *risk-sensitive Bellman operator* (Ruszczyński, 2010), in contrast to other risk-measures.

3.2.4 Robustness

An algorithm is called *robust* when it can perform well even in the presence of small perturbances in the model. Robust Markov Decision Processes (RMDP) are a worst-case extension of the MDP setting, in which either the reward (Regan and Boutilier, 2012) or the transition model (Iyengar, 2005) (or both) is *ambiguous* or *uncertain*. While the former case can be solved extending the linear program formulation available for standard MDPs (Puterman, 2014), the latter one is more difficult to deal with, hence, most of the robust RL literature focuses on it. The objective takes the following form:

$$\max_{\pi\in\Pi}\min_{P\in\mathcal{P}} \mathop{\mathbb{E}}_{\tau\sim p_\pi^P(\cdot)}[G(\tau)], \tag{3.22}$$

where $\mathcal{P} = \{P : \|P - \bar{P}\| \leq \psi\}$ is the *ambiguity set*, provided a suitable norm. These problems are *NP-hard* to solve in general (Wiesemann et al., 2013), but if one constrains nature separately for each state (Le Tallec, 2007) or each action pair (Nilim and El Ghaoui, 2005a), the problem becomes solvable in polynomial time. These settings are called, respectively, S-rectangular and SA-rectangular, and they can be optimized

using several value and policy iteration (Iyengar, 2005; Kaufman and Schaefer, 2013) methods. However, such assumptions are very conservative, hence, some works tried to relax them, introducing other kind of robustness (Tirinzoni et al., 2018; Mannor et al., 2012; Goyal and Grand-Clement, 2018; Pinto et al., 2017; Nilim and El Ghaoui, 2005b; Xu and Mannor, 2010; Lim et al., 2013). Moreover, the linear programming formulation may scale cubically with the number of states. Therefore, finding methods that better scale with the problem size or that allow for function approximation (Tamar et al., 2014) is still a challenging area of research.

3.2.5 Choosing the risk model

Given the plethora of available methods for modelling risk-aversion and evaluating risk, it is natural to ask whether some approaches are better than other ones. We argue that there is no definitive winner, but instead one should choose the most appropriate solution for the selected context according to multiple criteria:

- **rationality**: if one wants to guarantee a behavior which is rational, under some criterion, then one may choose to implement a utility that respects Von Neumann-Morgensten axioms, or one can consider a CRM;
- **time-consistency**: employing static risk-measures does not allow to consider all the possible risk preferences (Osogami, 2012a), hence, if consistency through difference time-steps is important, one should resort to apply a dynamic Markov CRM;
- **interpretability**: while dynamic CRM enjoy nice mathematical properties, the risk they represent is difficult to interpret and it entails the necessity of specifying single period risk mappings for every future time-point. Classical criteria such as Mean-Variance and ERM have instead a straightforward interpretation.

Finally, one should consider how *hard* is the chosen risk-averse optimization. In the next section, the main techniques developed so far for the described objectives will be described, highlighting the main challenges they pose.

3.3 Solving Risk-Sensitive RL Tasks

In this section we review some approaches which have been derived for optimizing the risk-averse objectives presented in the previous sections. We classify these techniques in four categories:

- **Risk-Sensitive MDPs**: *Ad-hoc* MDPs are developed to solve the particular problem, typically under *robust* constraints.
- **Policy Gradient Optimization**: Due to the impossibility of deriving effective Bellman operators for some of these objectives, policy gradient optimization may be adopted to find local solutions.
- **Distributional RL**: The distributional RL framework allow to directly model the return distribution. This may be helpful for solving for risk-averse objectives that correspond to *distortions* of such distributions.

- **Generalizations to RL framework**: The standard RL framework may sound somehow limited for these complex objective functions. Some works extending the framework to a more general form also contain risk-averse objectives as special cases.

3.3.1 Risk-Sensitive MDPs

Here we describe some dynamic programming or value-based approaches applied to modified MDPs in order to obtain as a result an optimized risk-averse objective. While enjoying some nice mathematical properties, these approaches typically have a high computational complexity, which struggles to scale to large instances.

CVaR MDPs The CVaR risk-measure allows at least two possible formulations as a risk-sensitive MDP. The first one, provided in (Chow et al., 2015), consists in an equivalence of the CVaR problem to a robust MDP (Wiesemann et al., 2013). The robustness in this case is related to transition perturbation. The authors show that perturbations are *budget-constrained*, which is a sufficient condition for the problem to be tractable (Mannor et al., 2012). Chow et al. (2015) also presented a robust Bellman optimality equation and optimized it with an approximated value iteration approach involving a linear interpolation, providing a finite time convergence error bound.

Another approach is instead provided in (Bäuerle and Ott, 2011), where the authors devise a nested optimization whose inner problem can be stated as an MDP. In the inner problem, they fix a quantile value and then solve an augmented MDP in which they add to the state two components: the current cumulated reward, and the current cumulated discount. The transition model and rewards are also modified, in order to have zero reward on ordinary transitions and a final reward corresponding to the cumulated sum to which the estimated α-quantile is subtracted. One has then to solve multiple times this inner problem to find an optimal value, or, as an alternative, one can fix the quantile to determine one's risk-aversion. This approach is described with greater detail in the more general framework of Chapter 4.

ERM MDPs The Entropic risk measure is one of the most common risk-measures in control, and many different techniques have been provided for its optimization. Beside being possible to derive an augmented MDP, as for CVaR, which will be described in Chapter 4, with this measure two other formulations are available. The most classical one consists in an MDP with a modified optimal Bellman equation (Howard and Matheson, 1972):

$$V^{\star}_{\beta,t}(s) = \max_{a} \gamma^t r(s,a) \frac{1}{\beta} \log \sum_{s'} P(s'|s,a) \exp\left(\beta V^{\star}_{\beta,t+1}(s')\right), \tag{3.23}$$

which may be solved with either policy or value iteration (Borkar and Meyn, 2002). However, as pointed out in (Mihatsch and Neuneier, 2002), this methods presents some disadvantages:

- for infinite horizon tasks with discounted formulation the optimal policy is *non-stationary* in general;

- it is impossible to handle *non-deterministic* rewards [10];
- it is not possible to derive model-free RL algorithms as in the risk-neutral settings, starting from the optimality equations.

ERM can also be framed as a robust MDP (Osogami, 2012b) with uncertainty in the transitions, which is regulated by a Kullback-Leibler divergence bonus.

Markov Coherent Risk-Measure MDP A robust formulation is also available for Markov Coherent Risk-Measures, by means of the Risk-Sensitive Optimality equation (Ruszczyński, 2010):

$$V_M^\star(s) = \max_a r(s,a) + \gamma \inf_{\xi : \xi P(\cdot|s,a) \in U(P(\cdot|s,a))} \int_{s \in \mathcal{S}} \xi P(\mathrm{d}s'|s,a) V_M(s), \tag{3.24}$$

where $U(P(\cdot|s,a))$ is the risk-envelope corresponding to the Coherent Risk-Measure which is iterated. The operator is a contraction and is monotonic, so that both value iteration and policy iteration approaches can be derived. Similar results may also be found in (Osogami, 2012a) for a similar composite measure call the *iterated risk-measure* (IRM), which are then specialized for the use of CVaR as a single step measure.

Other Risk-Sensitive MDPs By applying modifications to the standard objective, risk-aversion cannot typically be *exactly* translated to the classical MDP model. This does not correspond to saying that no kind risk-aversion can be injected in standard MDPs. It is possible, for instance, to modify the reward signal r in the two following ways:

$$r_1^\lambda \coloneqq r - \lambda r^2 \tag{3.25}$$

$$r_2^\lambda \coloneqq -exp(-\lambda r) \tag{3.26}$$

Both examples are valid reward function, which induce some kind of risk-aversion, respectively, to the second moment of the per step reward and its variance. While solving these problems does not introduce further complexity, it may be questionable whether the solution they provide is optimal for *any* risk measure, hence valuable for the risk-averse decision maker. This discussion is deferred to Chapter 5 and AppendixB.

3.3.2 Policy Gradient Methods for Risk-Averse Optimization

Policy Gradient Approaches for Mean-Variance The first policy gradient theorem for the Mean-Variance penalized criterion was developed in (Tamar et al., 2012a), where the authors considered the variance w.r.t. the value function:

$$\mathbb{V}ar_{\pi_{\boldsymbol{\theta}}}(s) \coloneqq \mathop{\mathbb{E}}_{\substack{a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1} \sim \mathcal{P}(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R(s_t,a_t) - V_{\pi_{\boldsymbol{\theta}}}(s) \right)^2 \middle| s_0 = s \right]. \tag{3.27}$$

They considered a stochastic shortest path problem starting from a fixed state, hence, in that case their definition coincides with 3.10. In particular they derived the formula

[10] In this thesis we treated deterministic rewards, but all the concepts can be easily extended to the stochastic case.

for the gradient of the variance[11]:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathbb{V}ar_{\boldsymbol{\theta}}(s_0) = & \mathbb{E}_{\substack{s\sim d^{\theta}_{\mu,\gamma^2}(s_0,\cdot) \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s) \\ s'\sim P(\cdot|s,a)}} \left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\ W_{\boldsymbol{\theta}}(s,a)\right] \\ & + 2\gamma \sum_{t=0}^{\infty} \mathbb{E}_{s\sim P^{(t)}_{\boldsymbol{\theta}}(s_0,\cdot)} \left[\gamma^{2t} R(s) \sum_{k=t+1}^{\infty} \mathbb{E}_{\substack{s'\sim P^{(k)}_{\boldsymbol{\theta}}(s,\cdot) \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \left[\gamma^k \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a' \mid s') Q_{\boldsymbol{\theta}}(s',a')\right]\right] \\ & - 2\, V_{\boldsymbol{\theta}}(s_0)\, \nabla_{\boldsymbol{\theta}} J_{\pi_{\boldsymbol{\theta}}},\end{aligned} \tag{3.28}$$

where $d^{\theta}_{\mu,\gamma^2}$ is the occupancy state distribution using γ^2 as discount factor, $P^{(t)}_{\boldsymbol{\theta}}$ is the probability of being the t-step distribution by following $\pi_{\boldsymbol{\theta}}$. Deriving an *unbiased* estimator for this formula is problematic though, since the last term involves multiplication of the gradient of the expected return and the value function, which cannot be estimated with the same samples without introducing bias. The latter is known as the *double sampling* issue: in order to obtain an unbiased estimate one has to estimate the two quantities from independent batches of samples. The alternative consists in using a biased estimate, which is anyway consistent (converging to the true value in the limit of infinite samples). This work was then extended from the actor-only scheme to the actor-critic one in (Tamar and Mannor, 2013), using linear function approximation. However we notice that, for the general case, the objective optimized by following the gradient in Equation 3.27 is:

$$\sigma_V^2 \coloneqq \mathbb{E}_{s\sim\mu(\cdot)}[\mathbb{V}ar_{\pi_{\boldsymbol{\theta}}}(s)] = \mathbb{E}_{s\sim\mu(\cdot)}\left[U_{\pi_{\boldsymbol{\theta}}}(s)\right] - E_{s\sim\mu(\cdot)}\left[V_{\pi_{\boldsymbol{\theta}}}(s)^2\right], \tag{3.29}$$

which is different from

$$\sigma^2 = \mathbb{E}_{s\sim\mu(\cdot)}\left[U_{\pi_{\boldsymbol{\theta}}}(s)\right] - J^2_{\pi_{\boldsymbol{\theta}}}$$

defined in Equation 3.10. In particular, we can see that the difference between the two:

$$\sigma^2 - \sigma_V^2 = E_{s\sim\mu(\cdot)}\left[V_{\pi_{\boldsymbol{\theta}}}(s)^2\right] - J^2_{\pi_{\boldsymbol{\theta}}} = \mathbb{V}ar_{s\sim\mu(\cdot)}[V_{\pi_{\boldsymbol{\theta}}}(s)] \tag{3.30}$$

is the variance of the value function w.r.t. the initial state distribution μ. A simple trajectory based policy gradient optimizing σ^2 can also be derived:

$$\nabla_\theta \mathbb{V}ar_{\boldsymbol{\theta}} = \mathbb{E}_{\tau\sim p_\theta(\cdot)}\left[\nabla_\theta \log p_\theta(\tau)(G_\gamma(\tau) - J_{\boldsymbol{\theta}})^2\right] \tag{3.31}$$

In order to alleviate the problem of variance gradient estimation, two simultaneous perturbation methods were proposed in (Prashanth and Ghavamzadeh, 2013). More recently a method based on block-coordinate optimzation which exploits Fenchel duality was developed in (Xie et al., 2018). The approach allows to write the problem as a nested optimization. The latter algorithm is also generalized from the framework presented in Chapter 4.

More Policy Gradient Methods A policy gradient theorem for CVaR was derived first in (Tamar et al., 2015b):

$$\nabla_{\boldsymbol{\theta}} \eta^{CVaR}_{\alpha,\boldsymbol{\theta}} = \mathbb{E}_{\tau\sim p_\theta(\cdot)}\left[\nabla_\theta \log p_\theta(\tau)(G_\gamma(\tau) - \nu_\alpha)\mathbb{1}_{G_\gamma(\tau)\geq\nu_\alpha}\right]. \tag{3.32}$$

[11] The original formula was for the undiscounted case, for this formula we relied on (Palmisano, 2019)

The previous formula was then generalized from the work in (Tamar et al., 2015a), which provides a unique formula for the coherent risk-measure policy gradient. A different approach is pursued in (Chow et al., 2017), where they optimize a *CVaR-constrained* problem, using a multi-scale actor-critic algorithm for which they were able to prove convergence. A policy gradient for ERM was developed in (Nass et al., 2019):

$$\nabla_{\boldsymbol{\theta}} \eta^{ERM}_{\beta,\boldsymbol{\theta}} = \mathop{\mathbb{E}}_{\tau \sim p_{\pi_\theta}(\cdot)} \left[\nabla_\theta \log p_\theta(\tau) \left(-\frac{1}{\beta} \exp\left(-\beta (G(\tau) - \eta^{ERM}_{\beta,\boldsymbol{\theta}}) \right) \right) \right]. \tag{3.33}$$

Sharing a similar formulation w.r.t. the standard policy gradient, all these gradienst can be estimated from samples. However, since most of them rely on estimating some quantity related to the current performance (ν_α, $\eta^{ERM}_{\beta,\boldsymbol{\theta}}$), they typically needs extra samples to obtain an unbiased estimate. Recently, a natural actor-critic approach has been developed specifically for problems with a constraint on some downside risk-measures (Spooner and Savani, 2020), allowing the authors to exploit state-of-the-art techniques on CMDP (Tessler et al., 2018).

3.3.3 Risk-Sensitive Distributional RL

While distributional approaches to RL have a long history (Jaquette, 1973; Sobel, 1982), the first application to risk-averse RL can be found in (Morimura et al., 2010). This work proposed a non-parametric approach for approximating the distribution of returns with particles, which offers also a way to compute risk-measure on the distribution. They then derived a CVaR version of SARSA. More recently the distributional framework has received renewed attention. In (Bellemare et al., 2017) several theoretical results are provided for distributional versions of Bellman operators. While the policy evaluation one is shown to be a contraction (under a specific metric), unfortunately the optimal operator is not contracting under any metrics. Nevertheless the distributional approach they propose seems to give empirical advantages in the risk-neutral case, providing stability in the learning process and obtaining competitive performance on the Atari benchmark. The main idea consists in estimating a categorical distribution of probability atoms. Further works on the subject (Dabney et al., 2018c,a) proposed instead a different approach: whereas the former used N fixed locations for its approximation distribution and adjusts their probabilities, they assigned fixed, uniform probabilities to N adjustable locations. The first of these approaches (Dabney et al., 2018c), which employs quantile regression, allows to obtain a practical algorithm: QR-DQN. The second one (Dabney et al., 2018a) employs instead the reparametrization trick: a neural network is used to learn the mapping between a certain probability and its quantile. In this way, sampling some probability from a uniform distribution and then using it as input to the mapping is equivalent to sample from the return distribution. Applying a *distortion risk-measure*, as CVaR, to the mapping, it is possible to evaluate distorted expectations. A similar approach is used to learn the critic in (Urpí et al., 2021), where an actor-critic for the offline setting is derived. To learn the actor, pathwise derivatives are used, which allows to sample directly from the distorted distribution, leveraging on the implicit representation.

3.3.4 Beyond Ad-hoc Techniques

Risk-averse solutions may also be provided by more general approaches. An alternative way to optimize risk-averse objectives is offered from the work in (Zheng et al., 2020). Here, a meta-gradient framework is used for learning intrinsic reward functions across multiple lifetimes of experience. The framework is thought of as a way to find the *optimal reward* (Singh et al., 2009), i.e., the reward shaping (Ng et al., 1999) allowing to solve the RL problem in the most effective way possible. The same strategy can be used also to infer the *intrinsic* reward inducing the behavior maximizing the desired risk-averse objective, provided that such a transformation exists. In some cases, the two meta-objectives coincide: risk-averse rewards may be indeed helpful for the learning process itself, avoiding catastrophic events (de Lope et al., 2009).

The necessity of extending RL techniques to explicitly deal with risk-aversion might even suggest that the standard framework is somewhat too restrictive to deal with these kinds of problems. Several extensions to the usual formulation have been developed recently, and some of them are also powerful enough to include some risk-averse objectives as special cases. A first example is represented by the Convex MDP framework (Zahavy et al., 2021). In this work, the classical RL problem

$$\max_{d_\mu^\pi \in \mathcal{K}} \sum_{s,a} R(s,a) d_\mu^\pi(s,a) \tag{3.34}$$

is generalized to the following one:

$$\min_{d_\mu^\pi \in \mathcal{K}} f(d_\mu^\pi(s,a)), \tag{3.35}$$

where f is a convex function, and $\mathcal{K}$ is the set of all state-action occupancy measures which can be induced by some policy π. Thus, all the risk-averse objectives that correspond to convex functions of the occupancy may be framed in this way, and solved with the related methods. For the same class of objectives (Zhang et al., 2020a) derived a variational policy gradient approach, which extends the classical one (Sutton et al., 2000b), and can be employs, thus, in some risk-averse settings.

CHAPTER 4

Risk-averse Optimization through State-Augmentation

4.1 Introduction

Several risk criteria have been taken into consideration in the RL literature, as it is has been illustrated in Chapter 3. We showed that, since these risk criteria possess very different properties, a common approach consists in developing ad-hoc RL algorithms to optimize each risk measure (or class of risk measures), i.e., algorithms that are highly-specialized to the chosen objective function. While this enables a full understanding of the problem at hand, in practice it can be limited for at least two reasons:

1. Given a risk-averse RL algorithm for a specific risk measure, it is often not clear whether the algorithm can be easily adapted to optimize a different measure;
2. Given any state-of-the-art (risk-neutral) RL algorithm, it is often non-trivial to adapt the algorithm to optimize some desired risk measure instead of the expected return.

Intuitively, overcoming these two limitations is highly desirable from a practical viewpoint. Ideally, we would like an algorithm that enables optimization of a *multitude* of risk measures in an almost transparent manner and which, at the same time, can leverage recent advances in risk-neutral RL to improve learning efficiency.

In this chapter, we take a step forward in this direction by proposing a single framework to optimize some of the most popular risk measures, including conditional value-at-risk, entropic risk measure, and mean-variance, by adopting *any* risk-neutral RL algorithm. Instead of focusing on deriving algorithms for optimizing each single risk measure, we transform the underlying Markov decision process (MDP) so that optimizing the chosen risk measure in the original MDP is equivalent to optimizing the

expected return in the transformed one. We achieve this by leveraging previous theoretical results on state augmentation (Bäuerle and Ott, 2011; Bäuerle and Rieder, 2013), which we use to unify the optimization problem for the considered measures. The price we have to pay for this generality is the addition of one or two extra state variables and one extra optimization variable to the original problem, which we show can be easily handled in practice. Overall, our framework enables practitioners to learn risk-averse policies with minimal additional effort beyond learning risk-neutral ones. We believe this to be a significant step towards applying risk-sensitive RL algorithms to complex real-world problems.

The detailed contributions of this chapter are as follows.

i) Using recent results on state augmentation (Bäuerle and Rieder, 2013), we derive a unified objective for the considered risk measures (Section 4.2). In addition to reducing the conditional-value-at-risk and the entropic risk measure to an ordinary MDP, as originally shown by Bäuerle and Rieder (2013), we also show that mean-variance can be treated analogously.

ii) We propose a very simple meta-algorithm, Risk-averse policy Optimization by State Augmentation (ROSA), to optimize the unified objective by exploiting any available risk-neutral RL algorithm (Section 4.3).

iii) We propose extensive empirical results that demonstrate:

  a) the benefits of our single meta-algorithm over existing ad-hoc methodologies

  b) the scalability of our approach

  c) its performance on a real-world trading dataset (Section A.2.3).

4.2 A unified perspective

In these chapter, we will consider a subset of the risk-averse objectives analysed in Chapter 3. In particular, we will take into account:

- **conditional value at risk**, denoted as η^{CVaR} and defined in Equation (3.16), abbreviated as CVaR;
- **penalized Mean-Variance**, denoted as η^{MV} and defined in Equation (3.11), here simply called *mean-variance* and abbreviated as MV;
- **utility-based objectives**, which involve a function of the return as defined in Equation (3.3). We will focus specifically on the **entropic risk-measure**, denoted as η^{ERM} and defined in Equation (3.8), abbreviated as ERM.

The methods we are going to present are based on the following assumption.

Assumption 2 (Bounded hitting times)**.** *We suppose there exists a subset $\bar{\mathcal{S}} \subset \mathcal{S}$ of absorbing (or terminal) states, such that, for all $s, s' \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$, $p(\bar{\mathcal{S}}|s,a) = 1$ and $c(s,a,s') = 0$. Let $T_\pi := \inf\{t \in \mathbb{N}^+ : S_t^\pi \in \bar{\mathcal{S}}\}$ be the hitting time of an absorbing state when executing policy π. We need the following assumption. There exists $T < \infty$ such that, for any $\pi \in \Pi^{\mathrm{HR}}$, $T_\pi \leq T$ almost surely.*

This assumption is standard, e.g., in the policy gradient literature (Peters and Schaal, 2006; Deisenroth et al., 2013), where the trajectories collected by the agent terminate almost surely, no matter what policy is executed. Moreover, differently from what is stated in Definition 2.2.1, we are going to consider here a reward function which depends also on the *next state*, and which assumes only *non-positive* values[1]:

$$R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [-R_{max}, 0].$$

Our first step is to reduce the different risk-averse objectives under a single general objective that unifies all the mentioned risk measures. Later on, we shall see how to design a common framework that optimizes it by leveraging any risk-neutral RL algorithm as a sub-routine.

Definition 4.2.1 (Unified objective)**.** *Let η be a risk-averse objective and $f_\eta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $g_\eta : \mathbb{R} \to \mathbb{R}$ be two functions. The unified optimization problem is:*

$$\max_{\rho \in \mathbb{R}} \left\{ \max_{\pi \in \Pi^{\mathrm{HR}}} \mathbb{E}^\pi \left[f_\eta(G, \rho) \right] + g_\eta(\rho) \right\}. \tag{4.1}$$

The explicit definition of the quantities involved depend on the chosen risk measure and its parameters, as specified in the following proposition.

Proposition 4.2.2. *For any of the risk-averse objective of Section 4.2, an optimal policy computed by solving for an objective η is also optimal for* (4.1) *and viceversa, where*

- *for CVaR at level α, $f_{\mathrm{CVaR}}(G, \rho) = -\frac{1}{\alpha}(G - \rho)^-$ and $g_{\mathrm{CVaR}}(\rho) = \rho$ (see Equation* (**??**)*);*
- *for a utility function U, we have no external parameter ρ, $f_{\mathrm{U}}(G) = U(G)$, and $g = 0$. In particular, for ERM with parameter β, $f_{\mathrm{ERM}}(G) = -e^{-\beta G}$ and $g = 0$;*
- *for MV with parameter λ, $f_{\mathrm{MV}}(G, \rho) = (1 + 2\rho\lambda)G - \lambda G^2$ and $g_{\mathrm{MV}}(\rho) = -\lambda\rho^2$. MVo is analogous with the one-step reward instead of G and the expectation under the state-action occupancy measure.*

Proof. For CVaR at level α, Rockafellar et al. (2000) showed that:

$$\eta^\pi_{\mathrm{CVaR}}(G; \alpha) = \max_{\rho \in \mathbb{R}} \{ \rho - \frac{1}{\alpha} \mathbb{E}^\pi \left[(G - \rho)^- \right] \}.$$

Therefore,

$$\begin{aligned} \max_{\pi \in \Pi^{\mathrm{HR}}} \eta^\pi_{\mathrm{CVaR}}(G; \alpha) &= \max_{\pi \in \Pi^{\mathrm{HR}}} \max_{\rho \in \mathbb{R}} \left\{ \rho - \frac{1}{\alpha} \mathbb{E}^\pi \left[(G - \rho)^- \right] \right\} \\ &= \max_{\rho \in \mathbb{R}} \left\{ \rho - \max_{\pi \in \Pi^{\mathrm{HR}}} \mathbb{E}^\pi \left[\frac{1}{\alpha} (G - \rho)^- \right] \right\}. \end{aligned}$$

In the case of utility functions, the objective is actually equivalent to the one in (4.1) with no outer variable ρ. For ERM, the result simply follows by noting that the problem

[1] This assumption is by no means restrictive, since we can always subtract a constant from our reward function to satisfy the required condition. We notice that variance is not influenced by these translation (it is translation invariant), while the exponential function might be, hence, the β coefficient must be carefully tuned to avoid numerical problems.

is equivalent to optimizing the exponential utility, which in fact does not require any extra variable. For MV with parameter λ, we use the same trick as in (Xie et al., 2018). Starting from $\mathbb{V}\text{ar}^\pi[G] = \mathbb{E}^\pi\left[G^2\right] - \mathbb{E}^\pi\left[G\right]^2$, we use Legendre-Fenchel duality to reduce the squared expectation to a standard expectation,

$$\mathbb{E}^\pi\left[G\right]^2 = \max_{\rho\in\mathbb{R}} \left\{2\rho\mathbb{E}^\pi\left[G\right] - \rho^2\right\}.$$

Thus,

$$\begin{aligned}
\max_{\pi\in\Pi^{\mathrm{HR}}} \eta^\pi_{\mathrm{MV}}(G;\alpha) &= \max_{\pi\in\Pi^{\mathrm{HR}}} \left\{\mathbb{E}^\pi\left[G\right] - \lambda\left(\mathbb{E}^\pi\left[G^2\right] - \mathbb{E}^\pi\left[G\right]^2\right)\right\} \\
&= \max_{\pi\in\Pi^{\mathrm{HR}}} \left\{\mathbb{E}^\pi\left[G\right] - \lambda\mathbb{E}^\pi\left[G^2\right] + \lambda \max_{\rho\in\mathbb{R}} \left\{2\rho\mathbb{E}^\pi\left[G\right] - \rho^2\right\}\right\} \\
&= \max_{\rho\in\mathbb{R}} \left\{\max_{\pi\in\Pi^{\mathrm{HR}}} \mathbb{E}^\pi\left[(1+2\rho\lambda)G - \lambda G^2\right] - \lambda\rho^2\right\}.
\end{aligned}$$

□

In words, the unified objective (4.1), together with Proposition 4.2.2, reveals that computing the optimal risk-averse policy can be reduced to computing a policy minimizing the *expected value* of some (non-linear) function of the total reward. The price we have to pay for moving from the risk operator to the expectation one is the introduction of a single additional optimization variable ρ.

We now discuss how to optimize (4.1) by considering the two variables, π and ρ, separately. Specifically, in Section 4.2.1, we show that optimizing for π when ρ is fixed (inner objective) can be reduced to an ordinary MDP and thus solved by any RL algorithm. In Section 4.2.2, we show that the optimal value of ρ given π (outer objective) can be conveniently found in closed-form for all the considered risk measures. Hence, a natural approach to solve (4.1) is an alternating optimization method, also known as block coordinate descent (Wright, 2015). We present such an approach for our policy optimization framework in Section 4.3.

4.2.1 Inner Objective as an Ordinary MDP

Fix some value $\rho \in \mathbb{R}$ for the outer variables, the inner problem in (4.1) seeks a policy $\pi \in \Pi^{\mathrm{HR}}$ that minimizes $\mathbb{E}^\pi\left[f_\eta(G,\rho)\right]$. Computing its solution is non-trivial for at least two reasons. First, when f_η is a non-linear function of the total reward, as for our risk measures, the existence of an optimal Markovian deterministic policy (analogously to the risk-neutral case) is no longer guaranteed (Puterman, 2014) and we need to look for history-dependent policies. Second, the optimization depends on the specific choice of f_η, i.e., on the underlying risk measure η. Instead of designing ad-hoc methodologies, we address these two complications by reducing the inner objective to an ordinary MDP via state-space augmentation. This will enable the application of any RL algorithm to its optimization. To achieve this, we borrow the state-augmentation proposed by Bäuerle and Rieder (2013). The key intuition is that making an optimal decision at each time only requires keeping track of the cumulative discounted reward suffered so far, instead of the whole sequence of states and actions. Formally, we define the augmented MDP as follows.

Definition 4.2.3 (Augmented MDP (Bäuerle and Rieder, 2013))**.** *Let the original MDP be $M = (\mathcal{S}, \mathcal{A}, P, R, \mu, \gamma)$. The augmented MDP for optimizing $\mathbb{E}^{\pi}\left[f_{\eta}(G, \rho)\right]$ is represented by $\tilde{M} = (\tilde{\mathcal{S}}, \mathcal{A}, \tilde{p}, \tilde{c}, \tilde{\mu}, \tilde{\gamma})$, where:*

- $\tilde{\mathcal{S}} := \mathcal{S} \times [-R_{\max}/(1-\gamma), 0] \times (0, 1]$;
- *for $\tilde{s} = (s, v, w)$, $a \in \mathcal{A}$, and $\tilde{s}' = (s', v', w')$, the transition kernel $\tilde{P}$ is such that*[2] $\tilde{P}(\tilde{s}'|\tilde{s}, a) = P(s'|s, a)\delta\left(\gamma w - w'\right)\delta\left(v + wc(s, a, s') - v'\right)$;
- *the reward function is $\tilde{R}(\tilde{s}, a, \tilde{s}') = f_{\eta}(v', \rho)$ if $s' \in \bar{\mathcal{S}}$ and $s \notin \bar{\mathcal{S}}$, zero otherwise;*
- *the initial state-distribution is $\tilde{\mu}(\tilde{s}) = \mu(s)\delta\left(v\right)\delta\left(w - 1\right)$*
- *and the discount factor is $\tilde{\gamma} = 1$.*

Intuitively, the state-space is augmented with two scalar variables, while the action-space remains unchanged. We denote each augmented state by a tuple $\tilde{s} = (s, v, w)$, where s is the original state of our system, v keeps track of the running cumulative reward, and w keeps track of the discounting.[3] The transition kernel is such that the first state variable evolves according to the original transition dynamics, while the remaining two evolve deterministically. If $(\tilde{s}_0, a_0, \ldots, \tilde{s}_T)$ is a trajectory in the augmented MDP, we have $w_{t+1} = \gamma w_t$ and $v_{t+1} = v_t + w_t R(s_t, a_t, s_{t+1})$. Since $v_0 = 0$ and $w_0 = 1$, unrolling this dynamics, it is easy to see that $w_{t+1} = \gamma^t$ and $v_{t+1} = \sum_{h=0}^{t} \gamma^h R(s_h, a_h, s_{h+1})$, i.e., the two extra state variables have the intended meaning. If a transition to an absorbing state of the original MDP occurs at time t, $\tilde{R}(\tilde{s}_t, a_t, \tilde{s}'_{t+1}) = f_{\eta}(v_{t+1}, \rho)$. Since the reward function is zero everywhere except when entering an absorbing state and $\tilde{\gamma} = 1$, $\sum_{t=0}^{T-1} \tilde{\gamma}^t \tilde{R}(\tilde{S}_t, A_t, \tilde{S}_{t+1}) = f_{\eta}\left(\sum_{t=0}^{T-1} \gamma^t R(S_t, A_t, S_{t+1}), \rho\right) = f_{\eta}(G, \rho)$, that is, the cumulative reward of the augmented trajectories is the same as the application of f_{η} to the cumulative reward of the original ones. This implies that we can solve the augmented MDP as an ordinary one, in which we seek a policy $\tilde{\pi} : \tilde{\mathcal{S}} \to \mathcal{A}$ that minimizes $\mathbb{E}^{\tilde{\pi}}\left[\sum_{t=0}^{T-1} \tilde{\gamma}^t \tilde{R}(\tilde{S}_t, A_t, \tilde{S}_{t+1})\right]$. Bäuerle and Rieder (2013) showed that, under mild conditions, an optimal Markov deterministic policy exists for this (augmented) problem. Furthermore, there is always a corresponding non-Markovian policy that is optimal for the original (non-augmented) problem. Here Markov refers to the fact that $\tilde{\pi}$ directly maps augmented states to actions, though these states depend on the history of the original process. The key result is, thus, that we can solve the inner objective in (4.1) by first augmenting the state space and then applying *any* (risk-neutral) RL algorithm. Clearly, we cannot directly build the augmented MDP as in Bäuerle and Rieder (2013) since the dynamics are unknown. However, we note that it is simple to perform this augmentation on given samples, such as trajectories collected by the chosen RL method. In fact, all we have to do is keep track of the running rewards and discounting and progressively add them to the original state samples. This procedure is summarized in Algorithm 1.

[2] Here $\delta(x)$ denotes the Dirac delta function at x.

[3] Keeping track of the discounting can be avoided, but the augmented MDP would have a non-stationary transition kernel. Clearly, when $\gamma = 1$ this extra state variable can be neglected.

Algorithm 1 Trajectory-based State Augmentation

Require: Trajectories $\{\tau_1, \tau_2, \ldots, \tau_n\}$
where $\tau_i = (S_{0,i}, A_{0,i}, S_{1,i}, R_{1,i} \ldots, S_{T_i,i}, R_{T_i,i})$,
function $f_\eta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, parameter ρ
Ensure: Augmented trajectories $\{\tilde{\tau}_1, \tilde{\tau}_2, \ldots, \tilde{\tau}_n\}$
for $i = 1, 2, \ldots, n$ **do**
  for $t = 0, 1, \ldots, T_i$ **do**
    $\tilde{S}_{t,i} \leftarrow (S_{t,i}, \sum_{h=0}^{t} \gamma^h R_{h,i+1}, \gamma^t)$
  end for
  $\tilde{R}_{T_i,i} \leftarrow f_\eta(\sum_{t=0}^{T_i-1} \gamma^t R_{t,i+1}, \rho)$
  $\tilde{\tau}_i = (\tilde{S}_{0,i}, A_{0,i}, \tilde{S}_{1,i}, 0 \ldots, \tilde{S}_{T_i,i}, \tilde{R}_{T_i,i})$
end for

4.2.2 Optimizing the Outer Objective

Now that we have a convenient way to solve the inner objective in (4.1) for any fixed ρ, it only remains to specify how to compute the optimal value of ρ for any fixed policy. We now see that this is actually simple and can be done in closed-form for all the risk measures that we consider. Formally, let $\pi \in \Pi^{\text{HR}}$ be any policy and $\rho^\star(\pi) := \arg\max_{\rho \in \mathbb{R}} \{\mathbb{E}^\pi [f_\eta(G, \rho)] + g_\eta(\rho)\}$. Starting from the definitions of the functions f_η, g_η for the various risk measures, we have what follows.

- For CVaR:
$$\rho^\star_{\text{CVaR}}(\pi) = \arg\max_{\rho \in \mathbb{R}} \left\{ -\frac{1}{\alpha} \mathbb{E}^\pi \left[(G - \rho)^- \right] + \rho \right\}.$$
This was shown by Rockafellar et al. (2000) to be exactly the value-at-risk at level α, i.e., $\rho^\star_{\text{CVaR}}(\pi) = \eta^\pi_{\text{VaR}}(G; \alpha)$.

- For mean-variance, we have:
$$\rho^\star_{\text{MV}}(\pi) = \arg\max_{\rho \in \mathbb{R}} \left\{ \mathbb{E}^\pi \left[(1 + 2\rho\lambda)G - \lambda G^2 \right] - \lambda \rho^2 \right\}.$$
This is a simple concave quadratic function of ρ. Taking its derivative and equating it to zero, we obtain $\rho^\star_{\text{MV}}(\pi) = \mathbb{E}^\pi [G]$, i.e., $\rho^\star_{\text{MV}}(\pi)$ is the expected total reward of π.

- Finally, the ERM has no outer parameter and thus it can be optimized by solving exclusively the inner objective via state-augmentation.

4.3 Policy Optimization

We now present our general approach to risk-averse RL. Our meta-algorithm, called ROSA (Risk-averse policy Optimization by State Augmentation), is shown in Algorithm 2. ROSA takes as input a risk measure η among those of Section 4.2 and a risk-neutral RL algorithm A (hence the name meta-algorithm). No requirement on A is imposed and, in principle, it can be any RL algorithm. We shall elaborate more on the its choice later on. Before learning starts, ROSA casts η into the unified objective (4.1) by finding the corresponding functions f_η and g_η. At each iteration $j = 1, \ldots, k$, ROSA collects a batch of n trajectories using the current policy π_j. Then, it computes

Algorithm 2 Risk-averse policy Optimization by State Augmentation (ROSA)

Require: Risk measure η, risk-neutral RL algorithm A (e.g., PPO, TRPO, etc.), batch size n, number of iterations k

1: Compute functions f_η and g_η as in Proposition 4.2.2
2: Initialize policy π_1
3: **for** $j = 1, 2, \ldots, k$ **do**
4: Collect a batch $\{\tau_i\}_{i=1}^n$of n trajectories using π_j
5: Compute $\rho_j \leftarrow \arg\max_{\rho \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^n f_\eta \left(\sum_{t=0}^{T_i - 1} \gamma^t R_{t+1,i}, \rho \right) + g_\eta(\rho)$
6: Get augmented trajectories $\{\tilde{\tau}_i\}_{i=1}^n$ with Alg. 1
7: Feed $\{\tilde{\tau}_i\}_{i=1}^n$ into A, optimize π_j and obtain π_{j+1}
8: **end for**

the next value of ρ_j by using the closed-form expression as mentioned in Section 4.2.2. Since this involves the expected value of $f(G; \rho_j)$ under π_j, an unbiased estimator is built using the collected trajectories. Finally, ROSA augments the trajectories by using Algorithm 1 and feeds them into the policy optimization algorithm A. The latter performs one or more updates to the current policy. The overall procedure is an incremental block coordinate descent method (Wright, 2015), whose convergence has been proven for general convex (Beck and Tetruashvili, 2013) and non-convex (Tseng, 2001) settings.

Discussion The possibility to adopt any risk-neutral RL algorithm A to optimize a risk measure is the key component of ROSA. Such an algorithm can be freely chosen among those available in the literature. For instance, it can be an on-policy (Schulman et al., 2015a, 2017) or off-policy (Haarnoja et al., 2018; Munos et al., 2016) policy search algorithm or even a value-based method (Mnih et al., 2016). In our experiments, we shall indeed combine ROSA with both on-policy and off-policy methods. Regardless of the chosen algorithm A, ROSA interacts with the environment in an online on-policy fashion, collecting, at each step, a batch of trajectories under the current policy and updating the latter by means of A. This is required to compute the outer variables, whose update requires the estimation of some statistics of the current policy (e.g., the expected return). While this is the solution that we consider in this work, we note that it is not restrictive and ROSA can be generalized to a fully off-policy setting by employing, e.g., importance sampling (Owen, 2013).

A possible concern in using the proposed approach regards the state augmentation's negative impact to the underlying RL problem. While it is true that adding state variables might increase the sample complexity, we note that this augmentation has been shown as a sufficient condition for representing optimal *Markov* policies (Bäuerle and Ott, 2011; Bäuerle and Rieder, 2013). On the other hand, existing ad-hoc approaches typically consider only Markov policies in the original state space and, thus, while solving simpler problems, might not necessarily converge to near-optimal risk-averse behavior.

Handling non-stationarity We note that, when using alternated incremental updates (in a block-coordinate descent fashion) as in ROSA, the reward function optimized by the risk-neutral RL algorithm becomes non-stationary. This is due to the fact that the

reward at each iteration depends on ρ through f, and the value of ρ is repeatedly updated by the outer optimizer. While this might not be an issue for, e.g., policy gradient algorithms, whose convergence could still be guaranteed using analyses from the block-coordinate literature, it could become problematic for certain RL methods (e.g., value-based ones). In practice, to avoid any undesirable behavior, it is important to include the outer variable ρ as part of the state or, more simply, as an input to the policy and/or value function.

4.4 Experiments

We conducted an empirical analysis of the proposed approach on three domains: two toy problems (a multi-armed bandit problem and a more complex MDP problem), a trading environment based on real financial data, and standard robotics benchmarks, with the last two being contexts where risk aversion plays a fundamental role. The purpose of our experiments is three-fold:

1. to show that ROSA can be successfully combined with different risk-neutral RL algorithms;
2. to show that ROSA outperforms existing ad-hoc methodologies;
3. to show that ROSA scales to high-dimensional continuous domains which have received little to no attention in the risk-averse literature.

We focused on three risk measures: mean-variance, ERM and CVaR, which are representative of all the transformations we propose. ERM is indeed a particular case of utility function, while we did not test the mean-volatility since thorough experiments, for an algorithm that is conceptually equivalent to ROSA, have been recently provided by (Zhang et al., 2020b). We compared our algorithm with baselines from the risk-averse RL literature for each of the chosen risk measures. We employ, respectively, a policy search approach (Nass et al., 2019) for ERM, a block-coordinate approach (Xie et al., 2018) for mean-variance, and GCVaR Tamar et al. (2015b) for CVaR. The implementation details, together with additional results, can be found in the appendix.

4.4.1 Multi-armed Bandit

We consider a multi-armed bandit problem with a continuous space of actions. More precisely, the agent can take any action in the interval $[-1, 1]$. When taking an action $a \in [-1, 1]$, the agent receives a reward R distributed as $\mathcal{N}(1-|a|, (1-|a|)^2)$. Clearly, the optimal risk-neutral policy is to take action $a = 0$ (which has maximum expected value equal to 1). However, this action has also the largest variance and is thus risky. Therefore, depending on the chosen risk measure, the agent needs to trade off between taking small actions (in absolute value) to maximize the expected return, and taking large actions to reduce risk. Since the reward is Gaussian, we are able to compute the optimal trade-off for mean-variance and CVaR in closed form, which allows us to perfectly evaluate the solutions learned by ROSA.

Results We combine ROSA with four different risk-neutral algorithms: TRPO (Schulman et al., 2015a), PPO (Schulman et al., 2017), SAC (Haarnoja et al., 2018), and

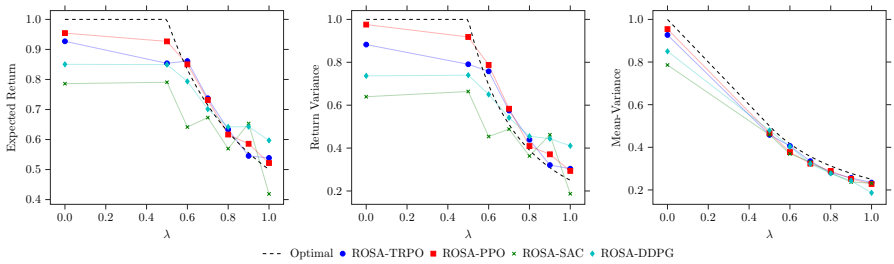

Figure 4.1: *Results of ROSA optimizing mean-variance when combined with different risk-neutral algorithms compared to the optimal solutions.*

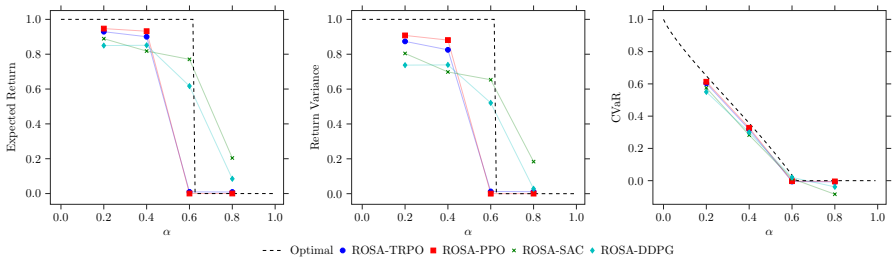

Figure 4.2: *Results of ROSA optimizing CVaR when combined with different risk-neutral algorithms compared to the optimal solutions.*

DDPG (Lillicrap et al., 2015). We report the results for mean-variance in Figure 4.1. More precisely, the three figures show the optimal values of expected return, return variance, and mean-variance as function of the risk-aversion parameter λ compared with the solutions learned by ROSA combined with the four base algorithms. Among these, the right-most plot is clearly the most indicative since it reports the actual objective function optimized by ROSA. Notably, all the algorithms almost perfectly learn the optimal mean-variance curve. The fact that expected returns and return variances of the learned policies are not as close to the optimal curve seems to indicate that mean-variance objective function is nearly flat in a neighborhood of the optimal points. Moreover, the slight sub-optimality of the learned risk-neutral solutions (for $\lambda = 0$) is probably due to the fact that online RL algorithms tend to be sensible to return variances (especially when learning with small batch sizes), thus converging to slightly risk-averse solutions.

The results for CVaR are shown in Figure 4.2 in the same format as those for MV. Consistently with MV, ROSA learns almost perfectly the optimal CVaR curve when combined with all algorithms. We report the results for ERM in Figure 4.3. Differently from before, for ERM we cannot compute the optimal solution in closed-form, so we simply report the learned pareto frontiers. We can appreciate that ROSA combined with all algorithms achieves very clear pareto frontiers, where solutions with high expectation/variance correspond to low risk-aversion parameters and viceversa.

4.4.2 Point Reacher

In the second toy problem, the agent controls a point mass that moves along the real line in order to bring it to a target location in the minimum number of steps. The state of the system is described by the position of the mass in the interval $[-10, 10]$, while the agent chooses (continuous) actions in $[-2, 2]$. If the system is in state s and the

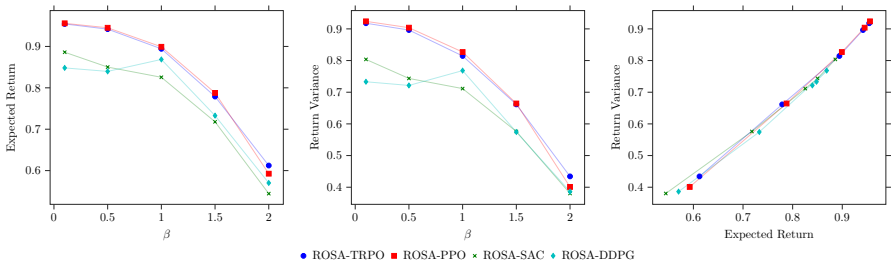

Figure 4.3: *Pareto frontiers for ROSA optimizing ERM when combined with different risk-neutral algorithms.*

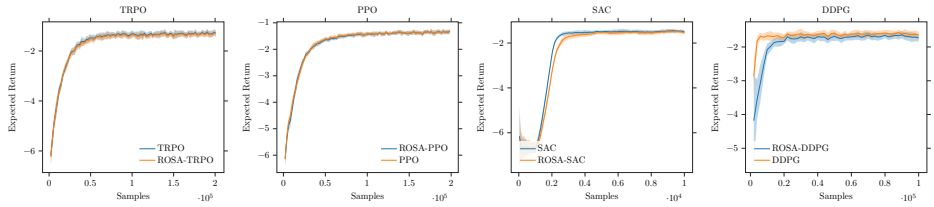

Figure 4.4: *Comparison between the base risk-neutral RL algorithms with and without ROSA's state augmentation in the Point Reacher domain.*

agent takes action a, the new state is $s' \sim \mathcal{N}(s + a, a^2)$ and the immediate reward is $r = -0.1|s'| + a^2$. The goal is the ball of radius 0.05 around the origin. Episodes have length at most 10 and terminate whenever the agent reaches a goal state. The initial state is drawn uniformly in $[-5.1, -5] \cup [5, 5.1]$.

Results First, we investigate the effects of the state augmentation on the learning process of the standard risk-neutral objective function. To this purpose, we run the original version of our four base algorithms for optimizing the expected return and compare it to their ROSA counterparts with state augmentation (i.e., with rewards delayed to the end of the episode and state augmented by the running cumulative return). Figure 4.4 shows the results. While it is true that the state augmentation slightly slows down the learning process (especially for the off-policy algorithms), we notice that this performance degradation is never too severe. Moreover, convergence seems unaffected.

The results of ROSA optimizing the different risk measures are shown in Figure 4.5 for MV, Figure 4.6 for CVaR, and Figure 4.7 for ERM. Since we cannot evaluate the optimal solutions in closed-form as before, here we plot the mean-variance Pareto frontier achieved by the learned policies for all risk measures. As expected, all the algorithms achieve a clear Pareto frontier when optimizing the mean-variance. Good results are also obtained for the CVaR, while a clear frontier is not achieved in ERM. The latter result is probably due to the fact that the adopted risk aversion parameters are all very similar and do not encode much risk aversion. All the algorithms seem to keep an almost constant expected return but, interestingly, they manage to slightly reduce variance with higher levels of risk aversion.

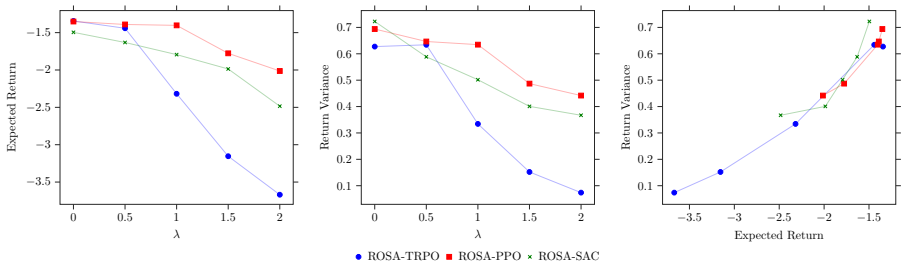

Figure 4.5: *Results of ROSA optimizing MV when combined with different risk-neutral algorithms on the Point Reacher domain.*

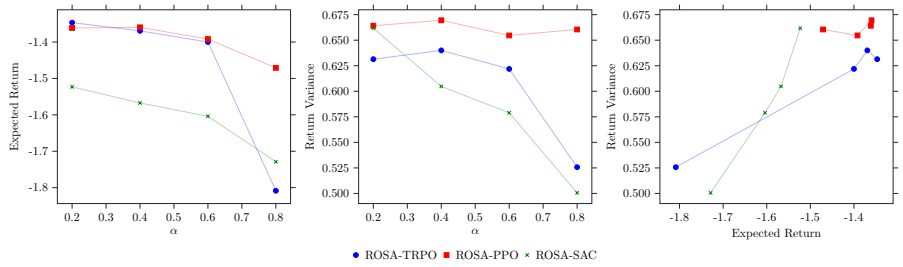

Figure 4.6: *Results of ROSA optimizing CVaR when combined with different risk-neutral algorithms on the Point Reacher domain.*

4.4.3 Trading Environment

The S&P trading environment simulates a trading scenario in which an agent has to trade a single asset, whose price follows the daily S&P index values from the '80s until 2019. In each episode, the agent starts its trajectory from a random day of the S&P time-series and observes the ordered sequence of historical prices for 49 steps (two months). Its state is composed of its current portfolio, the price, and the time left until the end of the episode, plus the 10 previous prices. The action space in this task is discrete, and the three possible actions are: buy, sell or stay flat. The reward is equal to $R_t = a_t(p_t - p_{t-1}) - f|a_t - a_{t-1}|$, where the first term is the profit or loss given by action a_t, and the second term represents the transaction costs, where f is set to $7 \cdot 10^{-5}$. See Bisi et al. (2020c) for further details.

Results In Figure 4.8, we report the results obtained in the Trading environment for all the selected risk measures. For this task, we instantiated ROSA with TRPO. A mean-variance Pareto frontier is plotted for both our approach and the corresponding baseline when optimizing mean-variance and ERM.[4] The algorithms were trained with the same budget of 15M samples. It can be noticed that ROSA learns solutions that dominate those of the baselines. For ERM (Figure 4.8b) the baselines cannot even obtain a clear frontier, while in Figure 4.8a it is clear that the learning process is still far from convergence. The improved learning speed for mean-variance and CVaR can be noticed from Figure 4.8c-d, which show the learning curve for two levels of risk aversion. Notably, ROSA achieves faster and more stable learning behavior.

[4]We recall that ERM is an approximation to the mean-variance objective, so it makes sense to plot the same Pareto frontier.

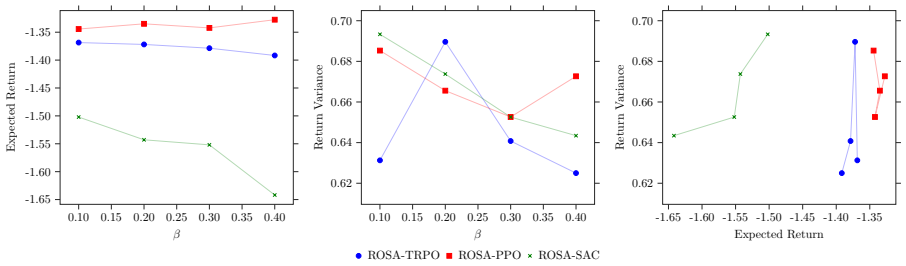

Figure 4.7: *Results of ROSA optimizing ERM when combined with different risk-neutral algorithms on the Point Reacher domain.*

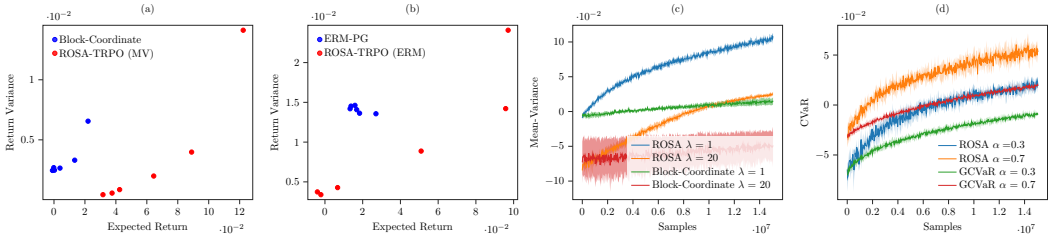

Figure 4.8: *The results obtained on the Trading environment by instantiating ROSA for mean-variance, ERM, and CVaR in comparison to the considered baselines. ROSA's optimization was performed by employing TRPO (Schulman et al., 2015a). Figures a-b show the mean-variance trade-off when optimizing mean-variance and ERM, respectively. Figures c-d report the learning curves for mean-variance and CVaR, respectively, with two different values of risk aversion.*

4.4.4 Robotic Locomotion

For the robotic setting, two challenging environments from the MuJoCo simulator (Todorov et al., 2012) were evaluated: Walker and Hopper[5]. The state of the robot is composed by its generalized position and velocity, while the controls are torques applicable to various joints. Both the state and the action spaces are continuous and high-dimensional. The reward is a linear combination of the following components:

1. a bonus for being alive;
2. a penalization for large action torques;
3. a bonus for moving forward;
4. a bonus for high speed.

Since these environments have deterministic dynamics, it can be difficult to understand the meaning of a risk-averse optimization. Therefore, we modified the task by introducing a perturbation to the action chosen by the agent. In particular, we added a white Gaussian *noise* to each action, with zero mean and a standard deviation proportional to the action magnitude. Intuitively, this models the fact that high torques have typically more unpredictable effects on the resulting system states. These environments, presenting high-dimensional continuous actions and states, are out of reach for the aforementioned baselines which performed very poorly in all our experiments. Their

[5]We employed the refined version of these environments from Pybullet (E. Coumans, 2016). Moreover we set the maximum length of each episode to 500 instead of 1000.

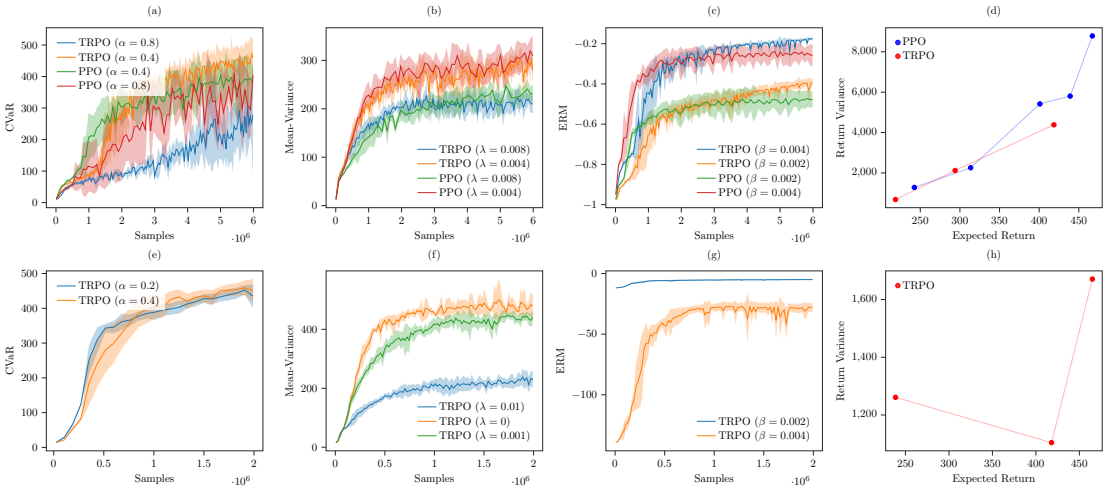

Figure 4.9: *The figures report the results obtained for the Walker and the Hopper environments on the first and the second row, respectively. Figures (a-c) and (e-g) display the learning curves obtained by employing ROSA to optimize, respectively, mean-variance, ERM, and CVaR. For each risk-measure, two levels of risk-aversion are shown. The inner optimization was performed by employing both TRPO and PPO for the Walker case, and only TRPO for the Hopper one. Shaded areas represent the standard deviation between* 5 *independent runs, while solid lines represent their means. Figures (d) and (h) show the trade-off between expected return and return variance obtained when optimizing mean-variance.*

results have thus been neglected from our plots to ease readability. The experiments were run with a fixed budget of 6M and 2M of samples for the Walker and the Hopper environments, respectively. All the reported results are the average of 5 independent runs with shaded areas representing plus-minus standard deviation.

Results Figure 4.9 (a-d) shows the results we obtained on the Walker environment, where we optimized the three risk measures under consideration while instantiating ROSA with both PPO an TRPO. In particular, we report the learning curves of both algorithms for two of the risk-aversion coefficients we trained the agents with. It can be noticed that the learning process is stable and improving for all the objectives and for both risk-neutral algorithms. This empirically demonstrates that ROSA successfully optimizes the considered risk measures even in high-dimensional tasks when combined with state-of-the-art RL approaches. As expected, the most critical risk measure to be optimized seems to be the CVaR, which is known to pose many estimation issues (Prashanth and Fu, 2018). In fact, both PPO and TRPO seems to struggle in optimizing CVaR with the higher level of risk aversion, while they perform well with the lower level. In Figure 4.9(d), we report the approximated Pareto frontier obtained for the mean-variance criterion. It can be noticed that, independently from the base algorithm chosen, ROSA obtains nice trade-offs between expected return and return variance by varying the risk-aversion coefficient.

In Figure 4.9 (e-h), we reported the results of ROSA optimizing the three risk measures on the Hopper environment, obtained using TRPO as base risk-neutral algorithm. Consistently with the Walker environment, ROSA successfully optimizes the different objectives with a stable and improving learning process. The mean-variance Pareto

frontier in Figure 4.9 (h), generated from the policies learned with three different values of λ, is less clear than before since the solution associated with $\lambda = 0.01$ is dominated by the one of $\lambda = 0.001$. This is probably due to the fact that the optimization process of the former did not reach convergence in 2M steps. However, as desired, both risk-averse solutions achieve a clear variance reduction with respect to the risk-neutral counterpart.

4.5 Conclusions

We presented a unified framework for risk-averse RL which captures many of the most popular risk measures. Our simple meta-algorithm, ROSA, allows to optimize risk-sensitive policies by using any risk-neutral RL algorithm. We tested our approach on both a financial and a robotic setting. The empirical results presented reveal that our method combined with state-of-the-art policy optimization approaches scales to complex problems and outperforms ad-hoc risk-sensitive algorithms, while requiring minimal additional efforts, both in terms of computation and implementation, with respect to learning risk-neutral policies. A relevant direction for future work is to extend ROSA to the batch RL setting, which would further increase its applicability. Moreover, we could investigate whether our framework generalizes to a larger class of risk measures, such as the coherent ones. Empirically, we noticed that risk-averse agents tend to under-explore the environment, occasionally converging to poor local optima. As a possible workaround, it would be interesting to run ROSA starting from some good pre-trained risk-neutral policy.

CHAPTER 5

Risk-Averse Trust Region Optimization for Reward-Volatility Reduction

5.1 Introduction

In Chapter 4 we have provided a unified framework that allows the direct application of risk-neutral methods for the optimization of some of the most important risk-averse objectives. Nevertheless, all the approaches seen so far take into account only the minimization of the *long-term* risk, since they consider only the variability related to the *return*. Limiting variations w.r.t. the *reward* may also be interesting though, since it captures per-step oscillations, which might be important in high-stakes tasks. For instance, in financial trading interim results are also fundamental, and keeping a low-varying intermediate P&L (Profit and Loss) becomes crucial. This chapter formally defines and analyzes for the first time, to the best of our knowledge, the variance of the reward at each time step w.r.t. state visitation probabilities. We call this quantity *reward volatility*. Intuitively, the return variance measures the variation of cumulated rewards among trajectories, while reward volatility is concerned with the variation of single-step rewards among visited states. We derive a Bellman equation for the reward-volatility that is exploited to obtain a policy gradient theorem for this novel objective. In addition, we also show that this new measure upper bounds the return variance (albeit for a normalization term). This is an interesting outcome, indicating that it is possible to use the analytic results we derived for the reward volatility to keep under control the return variance. Reward volatility is used to define a new risk-averse performance objective, called *mean-volatility*, which is a trade-off between the maximization of the expected return and the minimization of short-term risk. This trade-off can be customized in order to meet the specific needs of each individual trader, by tuning the risk aversion parameter. Optimizing the mean-volatility objective allows to limit the *inherent risk*

due to the stochastic nature of the environment. However, the imperfect knowledge of the model parameters, and the consequent imprecise optimization process, is another relevant source of risk, known as *model risk*. This is especially important when the optimization is performed on-line, as may happen for an autonomous, adaptive trading system. To avoid any kind of performance oscillation, the intermediate solutions implemented by the learning algorithm must guarantee continuing improvement. The TRPO algorithm (Schulman et al., 2015a) provides this kind of guarantees (at least in its ideal formulation) for the risk-neutral objective, based on the conservative bounds proven in (Kakade and Langford, 2002). Thanks to the linearity of the corresponding Bellman equation, we can show that the same bound still holds under the mean-volatility formulation. Hence, we derive the Trust Region Volatility Optimization (TRVO) algorithm, a TRPO-style algorithm for the new mean-volatility objective.

This chapter is organized as follows: the volatility measure is introduced in Section 5.2 and compared to the return variance. The Policy Gradient Theorem for the mean-volatility objective is provided in Section 5.3. In Section 5.3.1, we introduce an estimator for the gradient which is based on sample trajectories obtained from direct interaction with the environment. In Section 5.4, the monotonic improvement guarantees are presented and discussed, and the TRVO algorithm is introduced. Finally, in Section A.2.3, we test our algorithms on two financial environments, where the agents must learn to trade on real assets using historical data.

5.2 Risk Measures

For this chapter and the following one, we will refer to the *normalized* version of the expected return as:

$$J_\pi \coloneqq (1-\gamma) \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t,a_t)\right], \tag{5.1}$$

which allows us to write it as the expected value of the reward under the occupancy distribution:

$$J_\pi = \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi(\cdot|s)}} \left[R(s,a)\right]. \tag{5.2}$$

We recall here the definition of return variance (see Section 3.10), re-written according to the new notation:

$$\sigma_\pi^2 \coloneqq \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R(s_t,a_t) - \frac{J_\pi}{1-\gamma}\right)^2\right]. \tag{5.3}$$

We define the *variance of the per-step reward* as follows.

Definition 5.2.1 (Reward-Volatility)**.** *Let π be some policy, then variance of the reward w.r.t. the occupancy distribution is defined as:*

$$\nu_\pi^2 \coloneqq \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[\left(R(s,a) - J_\pi\right)^2\right]. \tag{5.4}$$

Reward-volatility can also be written in another form:

$$\nu_\pi^2 = (1-\gamma) \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t,a_t) - J_\pi \right)^2 \right]. \tag{5.5}$$

By setting a risk-aversion parameter λ, a novel risk-averse objective can be defined as:

$$\eta_\pi \coloneqq J_\pi - \lambda \nu_\pi^2, \tag{5.6}$$

called *mean-volatility* hereafter, where $\lambda \geq 0$ allows to trade-off expected return maximization with risk minimization, in a similar way respect to the Mean-Variance penalized objective (see Section 3.2.2). An important result on the relationship between the two variance measures is the following:

Lemma 5.2.2. *Consider the return variance σ_π^2 defined in Equation* (5.3) *and the reward volatility ν_π^2 defined in Equation* (5.4)*. The following inequality holds:*

$$\sigma_\pi^2 \leq \frac{\nu_\pi^2}{(1-\gamma)^2},$$

Proof. Taking the left hand side (Equation 3.10) and expanding the square we obtain[1]:

$$\begin{aligned}
\sigma_\pi^2 &= \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R_t \right)^2 \right] + \frac{J_\pi^2}{(1-\gamma)^2} - \frac{2J_\pi}{(1-\gamma)} \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R_t \right] \\
&= \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R_t \right)^2 \right] - \frac{J_\pi^2}{(1-\gamma)^2}.
\end{aligned}$$

Similarly, for the right hand side of the inequality:

$$\begin{aligned}
\nu_\pi^2 &= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[\big(R(s,a) - J_\pi \big)^2 \right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[R(s,a)^2 \right] + J_\pi^2 - 2J_\pi \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[R(s,a) \right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[R(s,a)^2 \right] - J_\pi^2.
\end{aligned}$$

Thus, the inequality we want to prove reduces to:

$$(1-\gamma)^2 \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty} \gamma^t R_t \right)^2 \right] \leq \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[R(s,a)^2 \right].$$

[1] To shorten the notation, R_t is used instead of $R(s_t,a_t)$.

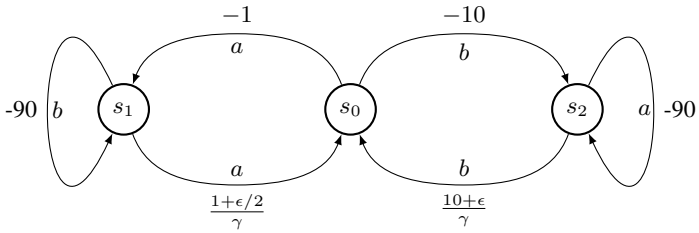

Figure 5.1: *A deterministic MDP. The available actions are a and b, s_0 is the initial state and rewards are reported on the arrows.*

Consider the left hand side. By the Cauchy-Schwarz inequality, it reduces to:

$$
\begin{aligned}
(1-\gamma)^2 \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty}\gamma^t R_t\right)^2\right] &\leq (1-\gamma)^2 \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\left(\sum_{t=0}^{\infty}\gamma^t\right)\left(\sum_{t=0}^{\infty}\gamma^t R_t^2\right)\right] \\
&= (1-\gamma) \mathop{\mathbb{E}}_{\substack{s_0\sim\mu \\ a_t\sim\pi_{\boldsymbol{\theta}}(\cdot|s_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t)}} \left[\sum_{t=0}^{\infty}\gamma^t R_t^2\right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \left[R(s,a)^2\right].
\end{aligned}
$$

□

It is important to notice that the factor $(1-\gamma)^2$ comes from the fact that the return variance is not normalized, unlike the reward volatility (intuitively, volatility measures risk on a shorter time scale). What is lost in the reward volatility compared to the return variance are the inter-temporal correlations between the rewards. However, Lemma 5.2.2 shows that the minimization of the reward volatility yields a *low return variance*. The opposite is clearly not true: as counterexample it is possible to think of the following example from the financial field. Consider a stock price, having the same value at the beginning and at the end of the investment period, but making complex movements in-between. A policy which simply holds the stock is going to have zero variance, but an high reward-volatility.

To better understand the difference between the two types of variance, consider the deterministic MDP in Figure 5.1. First assume $\epsilon = 0$. Every optimal policy (thus avoiding the -90 rewards) yields an expected return $J_\pi = 0$. The reward volatility of a deterministic policy that repeats the action a is $\nu_a^2 = {}^1\!/\!_\gamma$ while the reward volatility of repeating the action b is $\nu_b^2 = {}^{100}\!/\!_\gamma$. If we were minimizing the reward volatility, we would prefer the first policy, while we would be indifferent between the two policies based on the return variance (σ_π^2 is 0 in both cases). Now let us complicate the example, setting $\epsilon \in (0,1)$. The returns are now $J_b = \frac{\epsilon}{1+\gamma} > \frac{\epsilon/2}{1+\gamma} = J_a$. As a consequence, a mean-variance objective would always choose action b, since the return variance is still 0, while the mean-volatility objective may choose the other path, depending on the value of the risk-aversion parameter λ. This simple example shows how the mean-variance objective can be insensitive to short-term risk (the -10 reward), even when the gain in expected return is very small in comparison ($\epsilon \simeq 0$). Instead, the mean-volatility objective correctly captures this kind of trade-off.

5.3 Risk-Averse Policy Gradient

In this section, we derive a policy gradient theorem for the reward volatility ν_π^2 and propose an unbiased gradient estimator. This will allow us to solve the optimization problem $\max_{\boldsymbol{\theta}\in\Theta} \eta_{\pi_{\boldsymbol{\theta}}}$ via stochastic gradient ascent. We introduce a volatility equivalent of the action-value function Q_π (Equation (2.5)) called *action-volatility* function, which is the volatility observed by starting from state s, taking action a, and following policy π thereafter:

$$X_\pi(s,a) \coloneqq \mathop{\mathbb{E}}_{\substack{s_{t+1}\sim P(\cdot|s_t,a_t)\\ a_{t+1}\sim\pi(\cdot|s_{t+1})}} \left[\sum_{t=0}^{\infty} \gamma^t (R(s_t,a_t) - J_\pi)^2 | s,a\right], \tag{5.7}$$

Like the Q function, this can be written recursively by means of a Bellman equation:

$$X_\pi(s,a) = \big(R(s,a) - J_\pi\big)^2 + \gamma \mathop{\mathbb{E}}_{\substack{s'\sim P(\cdot|s,a)\\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[X_\pi(s',a')\big]. \tag{5.8}$$

We define also the *state-volatility* function W_π as the expected value of X_π under the policy $\pi_{\boldsymbol{\theta}}$, i.e. the equivalent of the V function (Equation 2.4) for volatility. The linearity of this Bellman equation allows an alternative interpretation of the mean-volatility objective. In fact, by applying a reward transformation

$$R_\pi^\lambda(s_t,a_t) = R(s_t,a_t) - \lambda(R(s_t,a_t) - J_\pi)^2, \tag{5.9}$$

it is possible to formulate the problem as a standard RL problem, where X and W functions are reduced to Q and V. Nonetheless, R_π^λ is a non-stationary policy-dependent reward, hence it is not compliant with the usual MDP framework and it is not possible to apply standard value-based algorithms to it. In general, even policy gradient approaches cannot be used with this kind of rewards. However, with the obtained Bellman equation we can derive a Policy Gradient Theorem (PGT) that holds for both ν_π^2 and the transformed reward case, as done in (Sutton et al., 2000b) for the expected return:

Theorem 5.3.1 (Reward Volatility PGT)**.** *Using the definitions of action-volatility and state-volatility function, the variance term ν_π^2 can be rewritten as:*

$$\nu_\pi^2 = (1-\gamma)\int_{\mathcal{S}} \mu(s) W_\pi(s) ds. \tag{5.10}$$

Moreover, for a given policy $\pi_{\boldsymbol{\theta}}$, $\boldsymbol{\theta}\in\Theta$:

$$\nabla\nu_\pi^2 = \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \Big[\nabla \log \pi_{\boldsymbol{\theta}}(a|s) X_\pi(s,a)\Big].$$

Proof. First, we need the following property (see e.g., Lemma 1 in Papini et al. (2019)):

Lemma 5.3.2. *Any integrable function $f : \mathcal{S} \to \mathbb{R}$ that can be recursively defined as:*

$$f(s) = g(s) + \gamma \int_{\mathcal{S}} P_\pi(s'|s) f(s')\, \mathrm{d}s',$$

where $g : \mathcal{S} \to \mathbb{R}$ is any integrable function, is equal to:

$$f(s) = \frac{1}{1-\gamma} \int_{\mathcal{S}} d_\pi(s'|s) g(s') \, \mathrm{d}s'.$$

From Equation 5.8 and the definition of W_π, we have[2]:

$$\begin{aligned}
X_\pi(s,a) &= \big(R(s,a) - J_\pi\big)^2 + \gamma \mathop{\mathbb{E}}_{\substack{s'\sim P(\cdot|s,a)\\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[X_\pi(s',a')\big] \\
&= \big(R(s,a) - J_\pi\big)^2 + \gamma \mathop{\mathbb{E}}_{s'\sim P(\cdot|s,a)} \big[W_\pi(s')\big], \\
W_\pi(s) &= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)} \big[\big(R(s,a) - J_\pi\big)^2\big] + \gamma \mathop{\mathbb{E}}_{s'\sim P^\pi(\cdot|s)} \big[W_\pi(s')\big] \\
&= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s'\sim d_\pi(\cdot|s)\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \Big[\big(R(s',a) - J_\pi\big)^2\Big], \\
\nu_\pi^2 &= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} \Big[\big(R(s,a) - J_\pi\big)^2\Big] \\
&= (1-\gamma) \mathop{\mathbb{E}}_{s\sim\mu} \Big[W_\pi(s)\Big].
\end{aligned}$$

For the second part, we follow a similar argument as in (Sutton et al., 2000b). We first consider the gradient of $X_\pi(s,a)$ and $W_\pi(s)$ $\forall s, a \in \mathcal{S} \times \mathcal{A}$:

$$\begin{aligned}
\nabla X_\pi(s,a) &= -2\big(R(s,a) - J_\pi\big)\nabla J_\pi + \gamma \mathop{\mathbb{E}}_{s'\sim P} \big[\nabla W_\pi(s')\big], \\
\nabla W_\pi(s) &= \nabla \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a|s) X_\pi(s,a) \, \mathrm{d}a \\
&= \int_{\mathcal{A}} \big[\nabla \pi_{\boldsymbol{\theta}}(a|s) X_\pi(s,a) + \pi_{\boldsymbol{\theta}}(a|s) \nabla X_\pi(s,a)\big] \, \mathrm{d}a \\
&= \int_{\mathcal{A}} \big[\nabla \pi_{\boldsymbol{\theta}}(a|s) X_\pi(s,a) - 2\pi_{\boldsymbol{\theta}}(a|s)\big(R(s,a) - J_\pi\big)\nabla J_\pi\big] \, \mathrm{d}a \\
&\quad + \gamma \int_{\mathcal{S}} p^\pi(s'|s) \nabla W_\pi(s') \, \mathrm{d}s' \\
&= \frac{1}{1-\gamma} \int_{\mathcal{S}} d_\pi(s'|s) \int_{\mathcal{A}} \big[\nabla \pi_{\boldsymbol{\theta}}(a|s') X_\pi(s,a) - 2\pi_{\boldsymbol{\theta}}(a|s)\big(R(s,a) - J_\pi\big)\nabla J_\pi\big] \, \mathrm{d}a \, \mathrm{d}s',
\end{aligned}$$

[2]To simplify the notation, the dependency on $\boldsymbol{\theta}$ is left implicit.

where the last equality is from Lemma 5.3.2. Finally:

$$\begin{aligned}\nabla\nu_\pi^2 &= (1-\gamma)\int_{\mathcal{S}}\mu(s)\nabla W_\pi(s)\,\mathrm{d}s \\ &= \int_{\mathcal{S}} d_{\mu,\pi}(s)\int_{\mathcal{A}}\left[\nabla\pi_{\boldsymbol{\theta}}(a|s)X_\pi(s,a) - 2\pi_{\boldsymbol{\theta}}(a|s)\big(R(s,a)-J_\pi\big)\nabla J_\pi\right]\mathrm{d}a\,\mathrm{d}s \\ &= \mathop{\mathbb{E}}_{\substack{s'\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}}\left[\nabla\log\pi_{\boldsymbol{\theta}}(a|s)X_\pi(s,a)\right] - 2\nabla J_\pi \mathop{\mathbb{E}}_{\substack{s'\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}}\left[R(s,a)-J_\pi\right] \\ &= \mathop{\mathbb{E}}_{\substack{s'\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}}\left[\nabla\log\pi_{\boldsymbol{\theta}}(a|s)X_\pi(s,a)\right].\end{aligned}$$

□

The term that becomes null in the proof corresponds to the policy-dependent component of the reward. Therefore, we also proved that, in this special case, the PGT still applies after the transformation. With a simple extension it is possible to obtain the policy gradient theorem for the mean-volatility objective defined in equation (5.6). The action value and state value functions are obtained by combining the action value functions of the expected return and of the volatility:

$$\begin{aligned}Q_\pi^\lambda(s,a) &\coloneqq Q_\pi(s,a) - \lambda X_\pi(s,a) \\ V_\pi^\lambda(s) &= V_\pi(s) - \lambda W_\pi(s).\end{aligned}$$

The policy gradient theorem thus states:

$$\nabla\eta_\pi = \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}}\left[\nabla\log\pi_{\boldsymbol{\theta}}(a|s)Q_\pi^\lambda(s,a)\right]. \tag{5.11}$$

5.3.1 Estimating the Risk-Averse Policy Gradient

To design a practical actor-only policy gradient algorithm, the action-value function Q_π needs to be estimated as in (Sutton and Barto, 1998; Peters and Schaal, 2008). Similarly, we need an estimator for X_π. In this approximate framework, we consider to collect N finite trajectories $s_0^i, a_0^i, ..., s_{T-1}^i, a_{T-1}^i,\ i = 0,\dots,N-1$ per each policy update. An unbiased estimator of J_π can be defined as:

$$\hat{J} = \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\sum_{t=0}^{T-1}\gamma^t R_t^i, \tag{5.12}$$

where rewards are denoted as $R_t^i = R(s_t^i, a_t^i)$. This can be used to compute an estimator for the action-volatility function:

Lemma 5.3.3. *Let $\widehat{X}$ be the following estimator for the action-volatility function:*

$$\widehat{X} = \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\sum_{t=0}^{T-1}\gamma^t\left[(R_t^i-\hat{J}_1)(R_t^i-\hat{J}_2)\right], \tag{5.13}$$

where $\hat{J}_1$ and $\hat{J}_2$, defined as in Equation (5.12)*, are taken from two different sets of trajectories $\mathcal{D}_1$ and $\mathcal{D}_2$, and a third set of samples $\mathcal{D}_3$ is used for the rewards R_t^i in Equation* (5.13)*. Then, $\widehat{X}$ is an unbiased estimator of X.*

Algorithm 3 Volatility-Averse Policy Gradient (VOLA-PG)

Input: initial policy parameter $\boldsymbol{\theta}_0$, batch size N, number of iterations K, learning-rate α.
for $k = 0, \dots, K-1$ **do**
  Collect $3N$ trajectories with $\boldsymbol{\theta}_k$ to obtain dataset $\mathcal{D}_{3N}$
  Split $\mathcal{D}_{3N}$ into three independent subsets, $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ of N trajectories each
  Compute estimates $\widehat{J}_1, \widehat{J}_2$ using $\mathcal{D}_1$, $\mathcal{D}_2$ as in Equation (5.12)
  Estimate gradient $\widehat{\nabla}_N \eta_{\boldsymbol{\theta}_k}$ using $\mathcal{D}_3$, $\widehat{J}_1$ and $\widehat{J}_2$ as in Equation (5.14)
  Update policy parameters as $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha \widehat{\nabla}_N \eta_{\boldsymbol{\theta}_k}$
end for

Proof. First of all, we recall that

$$\mathop{\mathbb{E}}_{\tau}[\hat{J}] = J_\pi.$$

Thus:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\tau_1}\mathop{\mathbb{E}}_{\tau_2}\mathop{\mathbb{E}}_{\substack{s'\sim d_\pi \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}}[\widehat{X}] &= \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\mathop{\mathbb{E}}_{\tau_1}\mathop{\mathbb{E}}_{\tau_2}\mathop{\mathbb{E}}_{\substack{s'\sim d_\pi \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}}\left[\sum_{t=0}^{T-1}\gamma^t(\mathcal{R}(s_t^i,a_t^i)-\hat{J}_1)(\mathcal{R}(s_t^i,a_t^i)-\hat{J}_2)\right] \\
&= \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\mathop{\mathbb{E}}_{\substack{s'\sim d_\pi \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}}\sum_{t=0}^{T-1}\gamma^t\left[\mathop{\mathbb{E}}_{\tau_1}(\mathcal{R}(s_t^i,a_t^i)-\hat{J}_1)\mathop{\mathbb{E}}_{\tau_2}(\mathcal{R}(s_t^i,a_t^i)-\hat{J}_2)\right] \\
&= \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\sum_{t=0}^{T-1}\gamma^t\mathop{\mathbb{E}}_{\substack{s'\sim d_\pi \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}}\left[(\mathcal{R}(s_t^i,a_t^i)-J_\pi)(\mathcal{R}(s_t^i,a_t^i)-J_\pi)\right] \\
&= \frac{1-\gamma}{1-\gamma^T}\frac{1}{N}\sum_{i=0}^{N-1}\sum_{t=0}^{T-1}\gamma^t\mathop{\mathbb{E}}_{\substack{s'\sim d_\pi \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}}\left[(\mathcal{R}(s_t',a_t')-J_\pi)^2\right] \\
&= X_\pi.
\end{aligned}$$

□

Note that, in order to obtain an unbiased estimator for X, a *triple sampling* procedure is needed. This may be very restrictive. However, by adopting single sampling instead, the bias introduced is equivalent to the variance of $\widehat{J}$, so the estimator is still consistent. This result can be used to build a consistent estimator for the policy gradient $\nabla\eta_\pi$, as an extension of the PGT estimator (Sutton et al., 2000b):

$$\widehat{\nabla}_N\eta_\pi = \frac{1}{N}\sum_{i=0}^{N-1}\sum_{t=0}^{T-1}\gamma^t\left(\sum_{t'=t}^{T-1}\gamma^{t'-t}\left[R_{t'}^i - \lambda\frac{1-\gamma}{1-\gamma^T}(R_{t'}^i-\widehat{J})^2\right]\right)\nabla\log\pi_{\boldsymbol{\theta}}(a_t^i|s_t^i). \tag{5.14}$$

As shown in (Peters and Schaal, 2008), this can be turned into a GPOMDP-like estimator (Baxter and Bartlett, 2001) (a refinement of REINFORCE (Williams, 1992)), for which variance-minimizing baselines can be easily computed. Pseudocode for the resulting actor-only policy gradient method is reported in Algorithm 3.

5.4 Trust Region Volatility Optimization

In this section, we go beyond the standard policy gradient theorem and show it is possible to guarantee a monotonic improvement of the mean-volatility performance measure (5.6) at each policy update. Safe (in the sense of non-detrimental) updates are of fundamental importance when learning online on a real system; but also helps speeding up offline training by dynamically choosing the optimal step size. While the mean-volatility objective ensures a risk-averse *behavior* of the policy, the safe update ensures a risk-averse *update* of the parameters of the policy. Thus, if we care about the agent's performance within the learning process, we must consider the importance of the step sizes at each update of the parameters. Adapting the approach in (Schulman et al., 2015a) to our mean-volatility objective, we show it is possible to obtain a learning rate that guarantees that the performance of the updated policy is bounded with respect to the previous policy. An alternative analysis, based instead on the work in (Papini et al., 2019), is provided in Appendix A.2.

The safe update is based on the advantage function, defined as the difference between the action value and state value function. From the linearity of the new Bellman equations, we can extend the definitions of advantage $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$ to their λ-versions, to obtain the mean-volatility advantage function:

$$A^\lambda_\pi(s,a) = Q^\lambda_\pi(s,a) - V^\lambda_\pi(s). \tag{5.15}$$

Furthermore, with the mean-volatility objective all the theoretical results leading to the *TRPO* algorithm hold. In particular, Theorem 5.4.1 is a λ-extension of Lemma 6.1 in (Kakade and Langford, 2002), with an interesting extra additive term[3]:

Theorem 5.4.1 (Performance Difference)**.** *The performance difference between two policies π and $\widetilde{\pi}$ is equal to the sum of the expected mean-volatility advantage and a bonus term, related to the squared expected advantage:*

$$\eta_{\widetilde{\pi}} - \eta_\pi = \int_{\mathcal{S}} d_{\mu,\widetilde{\pi}}(s) \int_{\mathcal{A}} \widetilde{\pi}(a|s) A^\lambda_\pi(s,a)\,\mathrm{d}a\,\mathrm{d}s + \lambda(J_{\widetilde{\pi}} - J_\pi)^2. \tag{5.16}$$

Proof. [4]

$$\begin{aligned}
\eta_{\widetilde{\pi}} &= (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t (R(s_t,a_t) - \lambda(R(s_t,a_t) - J_{\widetilde{\pi}})^2) \big] \\
&= (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[V^\lambda_\pi(s_0) - V^\lambda_\pi(s_0) + \sum_t \gamma^t (R(s_t,a_t) - \lambda(R(s_t,a_t) - J_{\widetilde{\pi}})^2) \big] \\
&= \eta_\pi + (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t (R(s_t,a_t) - \lambda(R(s_t,a_t) - J_{\widetilde{\pi}})^2 + \gamma V_\pi(s_{t+1}) - V_\pi(s_t)) \big]
\end{aligned}$$

Now, the goal is to obtain the discounted sum of the mean-volatility advantages defined in Equation (5.15); however, it must be evaluated using policy π instead of $J_{\widetilde{\pi}}$. Hence, we recall the result in (Kakade and Langford, 2002)[5]:

$$J_{\widetilde{\pi}} = J_\pi + (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi(s_t,a_t) \big] \tag{5.17}$$

[3] Different definitions result in different normalization terms.

[4] For the sake of clarity, we use the notation $\tau|\pi$ to denote the expectation over trajectories: $s_0 \sim \mu, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t,a_t)$.

[5] the difference is only in the normalization terms, which are used accordingly to the definitions above.

Hence, by using R_t to denote $R(s_t, a_t)$:

$$\begin{aligned}(R_t - J_{\widetilde{\pi}})^2 =&(R_t - J_\pi - (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi(s_t, a_t)\big])^2 \\ =&(R_t - J_\pi)^2 + (1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi(s_t, a_t)\big]^2 \\ &- 2(1-\gamma)(R_t - J_\pi) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi(s_t, a_t)\big]\end{aligned}$$

In this way, it is possible to separate the mean-volatility advantage function from the standard advantage function, since the performance difference becomes:

$$\begin{aligned}\eta_{\widetilde{\pi}} - \eta_\pi =&(1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t (R_t - \lambda(R_t - J_\pi)^2 + \gamma V_\pi^\lambda(s_{t+1}) - V_\pi^\lambda(s_t))\big] \\ &- \lambda(1-\gamma)^3 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)]^2\big] \\ &+ 2\lambda(1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t (R_t - J_\pi) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)]\big] \\ =&(1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi^\lambda(s_t, a_t)] - \lambda(1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)]^2 \\ &+ 2\lambda(1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)] \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t (R_t - J_\pi)]\end{aligned}$$

But, if we consider that

$$\begin{aligned}\mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t (R_t - J_\pi)] &= \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t R_t] - \frac{J_\pi}{1-\gamma} \\ &= \frac{J_{\widetilde{\pi}} - J_\pi}{1-\gamma} = \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)]\end{aligned}$$

Then, collecting everything together, we obtain

$$\eta_{\widetilde{\pi}} - \eta_\pi = (1-\gamma) \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} \big[\sum_t \gamma^t A_\pi^\lambda(s_t, a_t)] + \lambda(1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}} [\sum_t \gamma^t A_\pi(s_t, a_t)]^2$$

By applying the risk-neutral Performance Difference lemma (5.17) to the last term, the squared normalization factor cancel out and we obtain the thesis. □

Neglecting the last term, the bound becomes the same that could be obtained considering the transformed reward R_π^λ. In practice, it corresponds to considering the volatility of the previous policy rather than approximating the next one. This is, in general, the main issue that arises with the reward transformation: it works well for the on-policy case, but it cannot be handled with the same ease in the off-policy one. The aforementioned term adds a gain related to the square of the difference in the expected returns of the policies; therefore there is always a bonus w.r.t the reward transformation approach if the expected return of the second policy is either higher or lower than the first one. Following the approach proposed in (Schulman et al., 2015a), it is then possible to adopt an approximation $L_\pi^\lambda(\widetilde{\pi})$ of the surrogate function, which provides monotonic improvement guarantees by considering the KL divergence between the policies:

Theorem 5.4.2 (Safe Improvement Bound)**.** *Consider the following approximation of $\eta_{\widetilde{\pi}}$, replacing the state-occupancy density of the old policy $d_{\mu,\pi}$:*

$$L_\pi^\lambda(\widetilde{\pi}) \coloneqq \eta_\pi + \int_{\mathcal{S}} d_{\mu,\pi}(s) \int_{\mathcal{A}} \widetilde{\pi}(a|s) A_\pi^\lambda(s,a)\,\mathrm{d}a\,\mathrm{d}s; \tag{5.18}$$

Let

$$\alpha = D_{KL}^{max}(\pi,\widetilde{\pi}) = \max_s D_{KL}(\pi(\cdot|s),\widetilde{\pi}(\cdot|s))$$
$$\epsilon_\lambda = \max_s \big|\mathop{\mathbb{E}}_{a\sim\widetilde{\pi}}\big[A_\pi^\lambda(s,a)\big]\big|, \quad \epsilon = \max_s \big|\mathop{\mathbb{E}}_{a\sim\widetilde{\pi}}\big[A_\pi(s,a)\big]\big|$$

Then, the performance of $\widetilde{\pi}$ can be bounded as follows:[6]

$$\eta_{\widetilde{\pi}} \geq L_\pi^\lambda(\widetilde{\pi}) - \frac{2\gamma\epsilon_\lambda}{1-\gamma}\alpha + \lambda(1-\gamma)^2 M^2, \tag{5.19}$$

where

$$M \coloneqq \max(0, A_\pi^{\widetilde{\pi}} - \frac{2\epsilon\gamma}{1-\gamma}\alpha, -A_\pi^{\widetilde{\pi}} - \frac{\gamma}{1-\gamma}\alpha R_{\max}),$$
$$A_\pi^{\widetilde{\pi}} \coloneqq \int_{\mathcal{S}} d_{\mu,\pi}(s) \int_{\mathcal{A}} \widetilde{\pi}(a|s) A_\pi(s,a)\,\mathrm{d}a\,\mathrm{d}s.$$

Proof. By applying Theorem 1 from (Schulman et al., 2015a) to the Mean-Volatility version of the Performance Difference Lemma 5.4.1 we obtain

$$\eta_{\widetilde{\pi}} \geq L_\pi^\lambda(\widetilde{\pi}) - \frac{2\gamma\epsilon_\lambda}{1-\gamma}\alpha + \lambda(1-\gamma)^2 \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}}\left[\sum_t \gamma^t A_\pi(s_t,a_t)\right]^2,$$

In order to further bound the last term, we want to find a quantity $M \geq 0$ such that:

$$M^2 \leq \mathop{\mathbb{E}}_{\tau|\widetilde{\pi}}\left[\sum_t \gamma^t A_\pi(s_t,a_t)\right]^2.$$

The square function can be lower bounded by 0, or by the square of a lower bound of its argument, if the latter is greater than 0. Due to its convexity, the square function can have a lower bound which is greater than 0 in two cases: when an upper bound of its argument is lower than 0, or when a lower bound of its argument is larger than 0. Therefore, we need to compute both:

$$\mathop{\mathbb{E}}_{\tau|\widetilde{\pi}}\left[\sum_t \gamma^t A_\pi(s_t,a_t)\right] \geq A_\pi^{\widetilde{\pi}} - \frac{2\epsilon\gamma}{1-\gamma}\alpha$$

and:

$$\mathop{\mathbb{E}}_{\tau|\widetilde{\pi}}\left[\sum_t \gamma^t A_\pi(s_t,a_t)\right] \leq A_\pi^{\widetilde{\pi}} + \frac{\gamma}{1-\gamma}\alpha R_{\max}.$$

We then obtain the best lower bound by taking the maximum among the argument lower bound, the opposite of the argument upper bound and 0, finally taking the square of this quantity. □

[6] Comparing this bound to the results shown in the original paper, the denominator term is not squared due to return normalization.

Algorithm 4 Trust Region Volatility Optimization (TRVO)

Input: initial policy parameter $\boldsymbol{\theta}_0$, batch size N, number of iterations K, discount factor γ.
for $k = 0, \ldots, K-1$ **do**
Collect N trajectories with $\boldsymbol{\theta}_k$ to obtain dataset $\mathcal{D}_N$
Compute estimates $\widehat{J}$ as in Equation (5.12)
Estimate advantage values $A^{\lambda}_{\boldsymbol{\theta}_k}(s,a)$
Solve the constrained optimization problem

$$\boldsymbol{\theta}_{k+1} = \arg\max_{\boldsymbol{\theta}\in\Theta} \left[L^{\lambda}_{k}(\boldsymbol{\theta}) - \frac{2\epsilon\gamma}{1-\gamma} D^{max}_{KL}(\pi_{\boldsymbol{\theta}_k}, \pi_{\boldsymbol{\theta}}) \right]$$
$$\text{where } \epsilon = \max_{s} \max_{a} |A^{\lambda}_{\boldsymbol{\theta}_k}(s,a)|$$
$$L^{\lambda}_{k}(\boldsymbol{\theta}) = \eta_{\boldsymbol{\theta}_k} + \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi_k} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}} A^{\lambda}_{\boldsymbol{\theta}_k}(s,a)$$

end for

Finally, we can devise the first risk-averse trust-region optimization algorithm (to the best of our knowledge), which is called TRVO (Trust Region Volatility Optimization) and is outlined in Algorithm 4. The reader should notice that this extension is highly dependent on the the risk-measure chosen, and could not be easily applied to the other ones, which lack a linear Bellman equation (Sobel, 1982).

5.5 Experiments

In this section, we show an empirical analysis of the performance of TRVO (Algorithm 4) applied in two financial trading tasks: the first on an equity index, the S&P 500, and the second on spot Foreign Exchange (FX): USD/EUR and USD/JPY. The first baseline we compare to is a mean-variance policy gradient approach presented in (Tamar and Mannor, 2013) (indicated as MV-PG), which we adjusted to take into account discounting. The second one is Direct Reinforcement Learning (DRL) (Moody and Saffell, 2001). Finally we consider a risk averse transformation of the rewards, $\widetilde{R}_t \coloneqq (1 - \exp\{-cR_t\})/c$, in the original *TRPO* algorithm (indicated as TRPO-exp). It represents a first-order approximation of mean-volatility, but it is sound only for small values of the risk-aversion coefficient, since negative rewards can generate strong instabilities of the learning process. As shown below, TRVO is capable of obtaining a complete Pareto frontier on both these environments and it converges sooner than the baselines.

5.5.1 S&P 500 Trading

This first environment considers the daily prices of the S&P index from the 1980s, until 2019. The possible actions are $a_t \in \{-1, 0, 1\}$, where -1 indicates a short, 1 a long, and 0 a flat position (thus, short selling is possible). We assume that at each time-step we go long or short of the same unitary amount, thus the profits (and losses) are not reinvested, which means that the final gain is the sum of all the rewards. The value of the asset at time t is p_t, and the reward is equal to $R_t = a_t(p_t - p_{t-1}) - f|a_t - a_{t-1}|$, where the first term is the profit or loss given by the action a_t, and the second term represents the transaction costs, where f is the proportionality constant, set to $7 \cdot 10^{-5}$. The policy

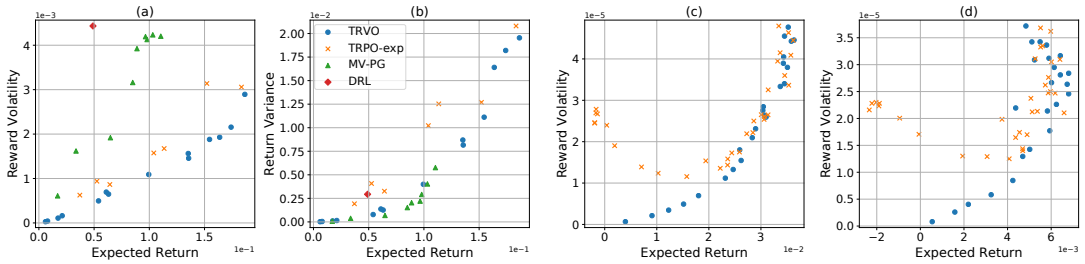

Figure 5.2: *(a) and (b): expected return, reward volatility, return variance in the S&P 500 environment with: TRVO, TRPO-exp, MV-PG, DRL; (c) and (d): expected return, mean volatility in the FX environment in training (c) testing (d). Performance is on 3 months and not normalized.*

we used is a neural network with two hidden layers and 64 neurons per hidden layer. The state consists of the last 10 days of percentage price changes, the previous portfolio position and the fraction of episode left (50 days long).

Results. The relevant plots for this environment are the first two in Figure 5.2, obtained on in-sample data. Plot (a) shows the Pareto frontier obtained with the four different algorithms by changing the risk aversion coefficient in the mean-volatility space, plot (b) in the mean-variance space. It is evident that the frontier generated by TRVO dominates the naive approach (TRPO-exp). Also, TRPO-exp becomes unstable for high levels of the risk-aversion parameter c, so it is not possible to find the value for which the risk aversion is maximal, which is why there are no points in the bottom left. The same figure includes also the results obtained with MV-PG, trained with the same number of iterations as TRVO (and TRPO-exp). In the mean-volatility space (Figure 5.2.a), the frontier generated by TRVO is clearly dominating. Instead, in the mean-variance space (Figure 5.2.b), the frontiers generated by MV-PG and TRVO are overlapping, but while the points generated by TRVO span a wide part of the space, those generated by MV-PG are concentrated in the lower-leftmost part of the graph even though they are trained with different risk aversions. This is due in part to the fact that MV-PG has not reached convergence even though it was given the same number of steps as TRVO, and reflects the faster convergence of TRPO w.r.t. GPOMDP. For DRL it is not possible to set the risk-aversion, hence it consists in a single point, which is on the Mean-Variance frontier, but it is instead dominated w.r.t. the Mean-Volatility criterion.

5.5.2 FX Trading

In the second experiment, actions and rewards are defined in the same way as before, but two different assets are considered: the FX rates USD/EUR and USD/JPY. The dataset has a much higher frequency (one datapoint per minute), hence also the agent can act every minute for a total of 1170 steps per episode (a trading day). The possible actions correspond to the position to keep for each asset, and the fee for each transaction is $f = 10^{-6}$. The training has been performed for a total of $5 \cdot 10^7$ steps on the 2017 dataset, while the testing was applied on 2018.

Results. The results for this environment can be found in the last two plots in Figure 5.2. We can see that TRPO-exp obtains the same results as TRVO for small risk-

aversion coefficients, both in training (c) and in testing (d). However, higher coefficients lead to instability in the exponential reward, that is gradually dominated by TRVO. It is interesting to notice that the settings having small or null risk-aversion coefficients (top right of the plots) are on the edge of the frontier in training, but are dominated in testing by more risk-averse policies. In this environment, MV-PG converges to a sub-optimal policy with null expected return, while DRL does not improve. Hence, they are not shown in the figures.

5.6 Conclusions

We proposed a novel methodology for risk-averse RL, exploiting, for the first time, a safe improvement bound. This was possible thanks to the definition of a risk measure called reward volatility that captures the variability of the rewards between steps. Optimizing this measure allows to obtain smoother trajectories that avoid shocks, which is a fundamental feature in a trading setting, and has never been considered by other risk measures so far. We showed interesting theoretical properties of reward-volatility: it bounds the variance of the returns and, differently from other risk measures, it has a linear Bellman equation. A policy gradient theorem for the mean-volatility objective was derived and, thanks to the aforementioned linearity, we obtained TRVO, a trust region algorithm that exploits a monotonic improvement bound of our objective. The proposed algorithm was tested on two financial trading environments where it was shown to outperform the baselines, obtaining better Pareto frontiers in shorter time. This work lays the foundation for extensions to both off-policy and online settings. To conclude, the developed framework is the first to take into account two kinds of safety, as it is capable of keeping risk under control while maintaining the same training and convergence properties as state-of-the-art risk-neutral approaches.

CHAPTER 6

Finite Sample Analysis of an Actor-Critic Algorithm for Mean-Volatility Optimization

6.1 Introduction

The development of novel analysis techniques has allowed the RL literature to produce a number of interesting results on the *finite-sample* complexity of many RL algorithms (Lazaric et al., 2012; Farahmand, 2011; Liu et al., 2020). Establishing the correct sample complexity of state-of-the-art algorithms such as, for instance, the well-known actor-critic scheme is a hot topic, which is receiving growing attention (Yang et al., 2018; Wu et al., 2020; Chen et al., 2021; Wang et al., 2019; Kumar et al., 2019; Xu et al., 2020b). On the other hand, few works have been dedicated to derive the complexity of risk-averse approaches (Jiang and Powell, 2018; Fei et al., 2020). Penalized risk-averse objectives as Mean-Variance and Mean-Volatility (Tamar et al., 2012a; Bisi et al., 2020b) need to estimate the expected return to compute the policy gradient. However, how the consequent estimation error translates in terms of convergence rate is an issue which has not been investigated yet. How do the various error sources compound in the gradient estimation? Is it possible to obtain the guarantees of risk-neutral algorithms in this risk-averse setting? This chapter tries to answer to some of those questions by means of a finite-sample analysis of a mean-volatility actor-critic algorithm.

Our contributions are as follows: (i) We propose two alternative methods (the *direct* one and the *factored* one) for the policy evaluation problem for the mean-volatility. We provide a finite sample bound for a semi-gradient TD(0) approach applied to the direct case. (ii) The previous contribution is used as input for an analysis on an actor-critic algorithm for which we bound the sample complexity necessary for reaching a ϵ-accurate stationary point. All provided bounds are valid in expectation. (iii) We validate

our theoretical results by means of an empirical study on a stochastic environment.

6.2 Problem Formulation

6.2.1 Mean-Volatility Policy Evaluation Techniques

If we wish to estimate the transformed value function V_π^λ for a given policy π, the most immediate idea is to transform the rewards using R_π^λ, and then use any risk-neutral policy evaluation algorithm. We call this approach the *direct-method*. In practice, performing the aforementioned reward transformation requires one to first estimate (via sampling) the (normalized) expected return J_π of the policy under evaluation. Denote by $\hat{J}_\pi$ our estimate of J_π, and by $\hat{R}_\pi^\lambda$ the resulting reward transformation when using $\hat{J}_\pi$ instead of J_π. One can then see $\hat{R}_\pi^\lambda$ as an estimator for the true reward transformation R_π^λ. Clearly, this estimator is biased if $\hat{J}_\pi$ is biased. Even if $\hat{J}_\pi$ was unbiased, $\hat{R}_\pi^\lambda$ would still be biased if we do not use two independently estimated versions of $\hat{J}_\pi$ since it is involved in a squared term. It is then natural to wonder how using such an approximate reward transformation affects the adopted policy evaluation algorithm. One may ask whether this kind of algorithm converges, and how distant (according to some measure) the obtained solution is from the exact one. Answering this question could enable us to show that the overall algorithm is consistent when $\hat{J}_\pi$ is consistent. This could also enable us to infer the order of the number of samples (used either by the policy evaluation algorithm or the sampling process for estimating J_π) needed to keep the estimation error below some given level ϵ. An alternative approach, which we will call the *factored-method*, relies on the following alternative expression for V_π^λ:

$$V_\pi^\lambda(s) = (1 + 2\lambda J_\pi)V^\pi(s) - \lambda M^\pi(s) - \frac{\lambda}{1-\gamma} J_\pi^2, \tag{6.1}$$

where we call $M^\pi : \mathcal{S} \to \mathbb{R}$ the *second moment value function*[1], which is defined as follows:

$$M^\pi(s) := \mathbb{E}_{\substack{a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim P(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)^2 \middle| s_0 = s \right]. \tag{6.2}$$

Squaring the rewards can be seen as a deterministic (policy-independent) reward transformation. Thus, M^π can be learned by adapting any algorithm that can be used for learning V^π, and any accuracy guarantees (e.g. finite-time bounds) on the learned estimate of V^π can be adapted for M^π; we would just need to consider that the range of values of the step-reward is different. A natural choice would be to learn both functions in parallel using the same algorithm and the same data. The factored method involves estimating V^π, M^π, and J_π, and then plugging them in (6.1) to obtain an estimate of V_π^λ. Note that this approach bears some resemblance to the approach adopted in (Tamar et al., 2016a) for estimating the variance of the reward to go. However, the approach in our case is simpler. This is mainly because the three quantities to be estimated (namely V^π, M^π, and J_π) can be learned *separately* using standard methods, and only combined at the end via (6.1).

[1] It is the second moment of the step reward $R(s', a')$ (where $s' \sim d_\pi(\cdot|s)$ and $a' \sim \pi(\cdot|s')$), not of the return when starting from s.

6.2.2 Monte-Carlo Estimation of the Expected Return

No matter which policy evaluation method one chooses to use, the estimation of the expected return J_π is a crucial step. We will adopt a simple Monte-Carlo procedure for estimating $\hat{J}_\pi$ (see Algorithm 5). In this procedure, we simulate L trajectories each truncated at a fixed horizon of T_J steps, and then average the (normalized) truncated returns from these trajectories. That is, if $G_i := \sum_{t=0}^{T_J-1} \gamma^t R(s_{i,t}, a_{i,t})$ denotes the truncated return from trajectory i, then $\hat{J}_\pi$ is given by:

$$\hat{J}_\pi := \frac{1}{L} \sum_{i=0}^{L-1} (1-\gamma) G_i.$$

Note that $\hat{J}_\pi$ is not necessarily an unbiased estimate of J_π since we are truncating the returns. This bias, however, can be arbitrarily reduced by making the trajectories long enough. Also, we will use only a single estimate of J_π in our algorithms, which can introduce bias due to the involvement of J_π in squared terms in both policy evaluation methods. These issues will be taken into account in our analysis.

Algorithm 5 Monte-Carlo-J

Input: π, γ, L, T_J
Initialize: $G_0, \ldots, G_{L-1} = 0$
for $i = 0, \ldots, L-1$ **do**
  $s_0 \sim \mu_0(\cdot)$
  for $t = 0, \ldots, T_J - 1$ **do**
    $a_t \sim \pi(s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$
    $G_i = G_i + \gamma^t R(s_t, a_t)$
  end for
end for
$\hat{J} = \frac{1}{L} \sum_{i=0}^{L-1} (1-\gamma) G_i$
Output: $\hat{J}$

6.2.3 The Critic Algorithm

We will extend the analysis in (Xu et al., 2020b) conducted over the actor-critic algorithm in the risk-neutral setting to our mean-volatility problem. We start by describing our extension of the critic. In (Xu et al., 2020b), the critic is a temporal difference algorithm that uses linear function approximation. More specifically, they use a mini-batch version of linear TD(0) in which a mini-batch of samples is used to performs the updates instead of just a single sample. Their motivation for adopting mini-batch updates is that the iterates can be driven arbitrarily close, in expectation, to the TD fixed point by increasing the mini-batch size while using a fixed step-size. Using this approach, they were able prove a better sample complexity than that provided in other works in the literature (e.g. (Bhandari et al., 2018)).

If we are to use the direct method, our aim will be to use that algorithm to learn V_π^λ by transforming the rewards using $\hat{R}_\pi^\lambda$ (which depends on $\hat{J}_\pi$). If we are to use the factored method instead, we can learn V^π by directly using the algorithm, and learn M^π in the same manner except that we square the rewards. For all three functions, we will consider a linear approximation scheme where candidate functions belong to

the function space $\{f_\omega : \omega \in \mathbb{R}^{d_\omega} \text{ and } f_\omega(.) = \omega^\intercal \phi(.)\}$, where $\varphi_i : S \to \mathbb{R}, i = 1, \ldots, d_\omega$. are basis functions defined over the states, and $\phi(.) := (\varphi_1(.), \ldots, \varphi_{d_\omega}(.))^\intercal$ is the corresponding feature mapping.

Algorithm 6 is a generalization of Algorithm 2 in (Xu et al., 2020b)[2], where the difference is that we get to choose the reward function f_R to be used in the algorithm. This could be:

- $f_R(s,a) = R(s,a)$, if we are learning V^π.
- $f_R(s,a) = R^2(s,a)$, if we are learning M^π.
- $f_R(s,a) = R(s,a) - \lambda(R(s,a) - \hat{J}_\pi)^2$, if we are learning V_π^λ using the direct method.[3]

Note that in the last case, f_R is a function of $\hat{J}_\pi$ and λ, which subsequently become parameters of the algorithm. As for the rest of the parameters, T_c is the number of iterations, M is the mini-batch size, and β is the step-size.

Algorithm 6 Mini-batch TD

Input: $s_{\text{ini}}, \theta, \phi(\cdot), \gamma, \beta, T_c, M, f_R$
Initialize: ω_0
Set $s_{-1,M} = s_{\text{ini}}$
for $k = 0, \ldots, T_c - 1$ **do**
  $s_{k,0} = s_{k-1,M}$.
  for $j = 0, \ldots, M-1$ **do**
    $a_{k,j} \sim \pi_\theta(s_{k,j})$, $s_{k,j+1} \sim P(.|s_{k,j}, a_{k,j})$
    $\widetilde{R}_{k,j} = f_R(s_{k,j}, a_{k,j})$
    $\delta_{k,j} = \widetilde{R}_{k,j} + \gamma\phi(s_{k,j+1})^\intercal\omega_k - \phi(s_{k,j})^\intercal\omega_k$
  end for
  $\omega_{k+1} = \omega_k + \beta\frac{1}{M}\sum_{j=0}^{M-1}\delta_{k,j}\phi(s_{k,j})$
end for
Output: $\omega_{T_c}, s_{k,M}$

6.2.4 The Actor Algorithm

In (Xu et al., 2020b), they adopt an *advantage actor critic* (A2C) approach, where they also use mini-batches to perform the stochastic gradient ascent updates. This means that the policy updates take the following from:

$$\theta_{t+1} = \theta_t + \alpha\frac{1}{B}\sum_{i=0}^{B-1}\delta_{t,i}\nabla_\theta \log \pi_{\theta_t}(a_{t,i}|s_{t,i}), \tag{6.3}$$

where B is the mini-batch size, α is the step-size, $\delta_{t,i} = R(s_{t,i}, a_{t,i}) + \gamma\hat{V}_t(s_{t,i+1}) - \hat{V}_t(s_{t,i})$ is the *temporal difference (TD) error* at the i$^{\text{th}}$ step, and $\hat{V}_t$ is the critic learned at iteration t. Note that $\delta_{t,i}$ is, in effect, an estimate of the advantage function at $(s_{t,i}, a_{t,i})$.

[2]Note that, unlike in (Xu et al., 2020b), we use ω for the critic's parameters and the more common choice of θ for the policy's parameters.

[3]We refer to this version of the algorithm as direct mini-batch TD.

When using parameterized policies, the gradient of η_θ with respect to θ has been derived in Theorem 5.11. Interestingly, this gradient has the same form as the risk-neutral policy gradient (Sutton et al., 2000a), but here we have the transformed action-value function instead of the normal one. Also note that as V_π^λ is a function of only the states (and not the actions), it satisfies $\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\big[V_\pi^\lambda(s)\nabla_\theta \log \pi_\theta(a|s)\big] = \mathbf{0}$, and hence can be used as a baseline in the mean-volatility gradient in the same way that baselines are used in the risk-neutral setting (Sutton et al., 2000a). In other words,

$$\nabla_\theta \eta_\theta = \mathbb{E}_{\substack{s\sim d_{\mu_0,\pi_\theta}(\cdot)\\ a\sim\pi_\theta(\cdot|s)}}\big[A_{\pi_\theta}^\lambda(s,a)\nabla_\theta \log \pi_\theta(a|s)\big], \tag{6.4}$$

where $A_\pi^\lambda(s,a) := Q_\pi^\lambda(s,a) - V_\pi^\lambda(s)$ is the transformed advantage function. Thanks to this policy gradient expression for the mean-volatility, adapting this approach to our case would just involve using the (estimated) transformed reward and the (estimated) transformed value function in place of their risk-neutral counterparts in the TD-error. To collect the samples of the mini-batch, the agent interacts with a slightly modified MDP characterized by the following transition kernel:

$$\tilde{P}(\cdot|s,a) = \gamma P(\cdot|s,a) + (1-\gamma)\mu_0(\cdot),$$

where P is the transition kernel of the original MDP. That is, at each step, the next state is sampled according to the original kernel with probability γ, while we draw the next state from the initial state distribution (i.e. restart) with probability $1-\gamma$. This sampling process causes the encountered states to be distributed, at steady-state, according to the discounted state distribution (Thomas, 2014). While this is indeed the desired distribution of states (see (6.4)), a side effect is that the next state (s_{t+1}) utilized in the TD-error expression is now sampled from $\tilde{P}(\cdot|s_t,a_t)$, whereas it should be sampled from $P(\cdot|s_t,a_t)$. This introduces a subtle bias in the algorithm, which is not accounted for in the analysis of (Xu et al., 2020b). To remedy this, we employ a slightly altered sampling process. At any time step t, consider two different random variables for the next state, namely, s_{t+1} and s'_{t+1}, with different distributions. The latter is distributed according to the standard kernel (i.e., $s'_{t+1} \sim P(\cdot|s_t,a_t)$), while s_{t+1} is sampled from the following variant of the modified kernel[4] $\tilde{P}(\cdot|s_t,a_t,s'_{t+1}) := \gamma\boldsymbol{\delta}_{s'_{t+1}}(\cdot) + (1-\gamma)\mu_0(\cdot)$. That is, with probability γ, s_{t+1} is the same as s'_{t+1}, and with probability $1-\gamma$, s_{t+1} is drawn from the initial state distribution. In any case, s'_{t+1} is the one we use as the next state in the TD-error, whereas s_{t+1} is the state from which we resume sampling the rest of the actor mini-batch[5]. With the proposed modification, the analysis of (Xu et al., 2020b) remains largely applicable, we just need to account for the extra performed sampling when we consider the sample complexity of the algorithm.

Algorithm 7 demonstrates our proposed adaptation of the mini-batch actor-critic algorithm to the mean-volatility setting. In the algorithm description, we used that (for any state-action pair) $\psi_\theta(s,a) := \nabla \log \pi_\theta(a|s)$, which is referred to as the *score function* of policy π_θ. Note that the algorithm leaves the choice of the critic procedure open. In particular, if we want to use the direct method, then we can call the mini-batch TD algorithm with $f_R(s,a) = R(s,a) - \lambda(R(s,a) - \hat{J}_t)^2$. If we name the learned

[4]Here, $\boldsymbol{\delta}$ is the Dirac delta function.

[5]Note that the proposed sampling process does not require a generative model, it only requires that we can halt the trajectory at any time and restart from the initial state distribution.

parameter vector ω_t, then we can set $\hat{V}_t^\lambda(s) := \phi(s)^\intercal \omega_t, \forall s \in \mathcal{S}$. If we want to use the factored method, we can call the mini-batch TD algorithm with $f_R(s,a) = R(s,a)$ and $f_R(s,a) = R^2(s,a)$ for learning $\hat{V}_t$ and $\hat{M}_t$ respectively[6]. If we then denote by ω_t^v and ω_t^m the learned parameter vectors for $\hat{V}_t$ and $\hat{M}_t$ respectively, we can set ($\forall s \in \mathcal{S}$):

$$\hat{V}_t^\lambda(s) = (1 + 2\lambda \hat{J}_t)\phi(s)^\intercal \omega_t^v - \lambda \phi(s)^\intercal \omega_t^m - \frac{\lambda}{1-\gamma}\hat{J}_t^2.$$

Note that the algorithm takes L and T_J as parameters, which denote the number of

Algorithm 7 Mini-batch Mean-Volatility Actor-Critic (Mini-batch MVAC)

Input: Policy Class π_θ, $\phi(\cdot)$, $\mu_0(\cdot)$, λ, γ, L, T_J, T, B, α.
Initialize: θ_0, $s_{-1,B} \sim \mu_0(\cdot)$
for $t = 0, \ldots, T-1$ **do**
  $s_{\text{ini}} = s_{t-1,B}$
  Estimate J:
    $\hat{J}_t = \text{Monte-Carlo-J}(\pi_{\theta_t}, \gamma, L, T_J)$.
  Estimate the Critic $\hat{V}_t^\lambda$ (utilizing $\hat{J}_t$, s_{ini}).
  Set $s_{t,0}$ as last state from the critic sampling.
  Actor mini-batch sampling:
  for $i = 0, \ldots, B-1$ **do**
    $a_{t,i} \sim \pi_\theta(s_{t,i})$
    $s'_{t,i+1} \sim P(\cdot|s_{t,i}, a_{t,i})$
    $s_{t,i+1} \sim \tilde{P}(\cdot|s_{t,i}, a_{t,i}, s'_{t,i+1})$
    $\tilde{R}_{t,i} = R(s_{t,i}, a_{t,i}) - \lambda(R(s_{t,i}, a_{t,i}) - \hat{J}_t)^2$
    $\delta_{t,i} = \tilde{R}_{t,i} + \gamma V_t^\lambda(s'_{t,i+1}) - V_t^\lambda(s_{t,i})$
  end for
  Actor update:
  $\theta_{t+1} = \theta_t + \alpha \frac{1}{B} \sum_{i=0}^{B-1} \delta_{t,i}\, \psi_{\theta_t}(s_{t,i}, a_{t,i})$
end for
Output: $\theta_{\hat{T}}$ with $\hat{T}$ chosen uniformly from $\{1, \ldots, T\}$.

trajectories and the number steps per trajectory used in the Monte-Carlo estimation of the expected return, which we have described before.

6.2.5 General Assumptions

Before describing our results, we highlight the main required technical assumptions.

Assumption 3. $\forall (s,a) \in S \times A$*:*

(i) $|R(s,a)| \leq R_{max}$.

(ii) $\pi_\theta(a|s)$ *is differentiable w.r.t.* θ.

(iii) $\exists\, C_\psi > 0 : \forall \theta \; \|\psi_\theta(s,a)\|_2 \leq C_\psi$.

(iv) $\exists\, L_\psi > 0 : \forall \theta_1, \theta_2 \; \|\psi_{\theta_1}(s,a) - \psi_{\theta_2}(s,a)\|_2 \leq L_\psi \|\theta_1 - \theta_2\|_2$.

(v) $\exists\, C_\pi > 0 : \forall \theta_1, \theta_2 \; \|\pi_{\theta_1}(.|s) - \pi_{\theta_2}(.|s)\|_{TV} \leq C_\pi \|\theta_1 - \theta_2\|_2$,

where, for a probability density function q(.), $\|q(.)\|_{TV} := \frac{1}{2}\int_s |q(ds)|$.

[6] In practice, one would use the same sample path for learning both functions.

Assumptions 3.iii and 3.iv assert that, for any policy in our class of policies, the score function is bounded and smooth, while assumption 3.v asserts that the chosen class of policies is smooth in the described sense. Note that by Assumption 3.i and the definition of J_π, $\forall (s,a) \in S \times A$ and $\lambda \geq 0$, we have that

$$\left|R(s,a) - \lambda(R(s,a) - J_\pi)^2\right| \leq R_{\lambda,\max},$$

where $R_{\lambda,\max} := R_{\max} + 4\lambda R_{\max}^2$. We also make the following assumption on the basis functions and the feature mapping that we use to learn V_π^λ.

Assumption 4. $\exists\, C_\phi > 0 : \forall s \in S\ \|\phi(s)\|_2 \leq C_\phi$. *Furthermore, the basis functions* $\varphi_i(\cdot), i = 1, ..., d_\omega$ *are mutually linearly independent.*

The following assumption serves to simplify the expressions of the bounds.

Assumption 5. *W.L.O.G.*

(i) $C_\psi = 1$.

(ii) $C_\phi = 1$.

The following is an assumption on the regularity of the MDP.

Assumption 6 (Uniform Ergodicity, Adapted from (Xu et al., 2020b))**.** *For any* $\theta \in \mathbb{R}^{d_\theta}$*, consider the MDP with policy* π_θ *and the transition kernel* $P(\cdot|s,a)$ *or* $\tilde{P}(\cdot|s,a) = \gamma P(\cdot|s,a) + (1-\gamma)\xi(\cdot)$*, where* $\xi(\cdot)$ *can be* μ_0 *or* $P(\cdot|\hat{s},\hat{a})$ *for any* $(\hat{s},\hat{a}) \in \mathcal{S} \times \mathcal{A}$*. Let* μ_{π_θ} *be the stationary state distribution of the MDP when acting with policy* π_θ*. There exists constants* $\kappa > 0$ *and* $\rho \in (0,1)$ *such that:*

$$\sup_{s \in \mathcal{S}} \|\mathbb{P}(s_t \in \cdot | s_0 = s) - \mu_{\pi_\theta}(\cdot)\|_{TV} \leq \kappa \rho^t, \forall t \geq 0.$$

Assumption 7. *For any triple* $(s_{i,t}, a_{i,t}, s_{i,t+1}) \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ *and any* $\hat{J}$ *estimate bounded, in absolute value, by* $R_{\max}$*, there exists real constants* C_A *and* C_b *such that* $\|A_{i,t}\|_F \leq C_A$ *and* $\|b_{i,t}(\hat{J})\|_2 \leq C_b$*, where* $\|\cdot\|_F$ *is the Frobenius norm*[7] *of a matrix.*

6.3 Main Results

In this section, we present the the main finite sample analysis results. We will first consider the analysis of the direct mini-batch TD algorithm for learning the transformed value function, and then we will consider the full mean-volatility actor-critic procedure, where the critic is learned using direct mini-batch TD.

6.3.1 Direct Mini-Batch TD Analysis

In the direct method, if $\hat{J}$ (and subsequently, the estimated transformed reward function) is fixed, we can invoke the results from (Tsitsiklis and Van Roy, 1997) about the convergence of TD learning with linear function approximation. In particular, if we define[8] $b(\hat{J}) := \mathbb{E}_{\mu_\theta}\left[\phi(s_t) R^\lambda(s_t, a_t, \hat{J})\right]$, and $A := \mathbb{E}_{\mu_\theta}[\phi(s_t)(\gamma\phi(s_{t+1}) - \phi(s_t))^\intercal]$, then

[7] For an $m \times n$ matrix X, its Frobenius norm (Golub and Van Loan, 1996) is defined as $\|X\|_F := \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2(A)}$, where $\sigma_i(A)$ are the singular value of A.

[8] Here, μ_θ is the stationary distribution of policy π_θ.

the algorithm converges to a point $\omega^*_{\hat{J}}$ such that $A\omega^*_{\hat{J}} + b(\hat{J}) = 0$. However, our goal is to describe the convergence rate, in expectation, of the critic to ω^*_J, where J is the true expected return of the policy under evaluation, not to $\omega^*_{\hat{J}}$. Moreover, $\hat{J}$ is not fixed; it is a random variable whose properties depend on the number of trajectories L (and their length T_J) used to estimate it. The main idea of the analysis is thus to bound how far we expect $\omega^*_{\hat{J}}$ to be from ω^*_J in terms of L and T_J. Before presenting the bound, we state the following result[9], which is an adaptation of a similar statement in (Xu et al., 2020b). There exists a positive constant χ_A such that, for any $\omega \in \mathbb{R}^{d_\omega}$ and any value of our (bounded) estimate $\hat{J}$, we have that

$$\langle(\omega - \omega^*_{\hat{J}}), A(\omega - \omega^*_{\hat{J}})\rangle \leq -\frac{\chi_A}{2}\left\|\omega - \omega^*_{\hat{J}}\right\|_2^2.$$

Theorem 6.3.1 (Critic's Bound)**.** *Suppose Assumptions 3, 4 and 6 hold, and suppose we are given a policy π_θ (with normalized expected return J) and risk parameter λ. Suppose that a Monte-Carlo estimate $\hat{J}$ is obtained for π_θ as described before, and then Algorithm 6 is run for T_c steps using $f_R(s,a) = R(s,a) - \lambda(R(s,a) - \hat{J})^2$. Then, for $\beta \leq \min\{\mathcal{O}(\chi_A), \mathcal{O}(\chi_A^{-1})\}$, we have that*

$$\begin{aligned}
&\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_J\right\|_2^2\right] \leq \\
&\quad 4\|\omega_0 - \omega^*_J\|_2^2(1 - \mathcal{O}(\chi_A\beta))^{T_c} + \mathcal{O}\left(\frac{\chi_A^{-1} + \beta}{\chi_A M}\right) \\
&\quad + \frac{2}{\bar{\sigma}^2}\left[1 + 2(1 - \mathcal{O}(\chi_A\beta))^{T_c}\right]\mathcal{O}\left(\lambda^2\left(\gamma^{2T_J} + \frac{1}{L}\right)\right),
\end{aligned}$$

where $\omega^{\hat{J}}_{T_c}$ is the parameter vector obtained after T_c iterations of the algorithm while using $\hat{J}$ to perform the reward transformation, $\bar{\sigma}$ is the smallest singular value of the matrix A, and the expectation is over both the Monte-Carlo estimation of $\hat{J}$ and the TD algorithm. Furthermore, for a sufficiently small $\epsilon > 0$, to achieve an ϵ-accurate solution, that is,

$$\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_J\right\|_2^2\right] \leq \epsilon,$$

the sample complexity of the algorithm is

$$T_c M + LT_J = \mathcal{O}\big(\epsilon^{-1} log\big(\epsilon^{-1}\big)\big).$$

The proof of this theorem can be found in Appendix A.3.3. The first two terms of the bound are (up to constants) the risk-neutral bound of (Xu et al., 2020b). The third term primarily quantifies the inaccuracy of $\hat{J}$, and decays by increasing L and T_J. Interestingly, the obtained sample complexity is the same as the risk-neutral version in (Xu et al., 2020b). In fact, only the third term depends on λ, and upon setting it to zero, the risk-neutral bound is recovered. Although a higher degree of risk-aversion (i.e., greater λ) has a negative impact on the bound, it does not affect the order of the required number of samples. It is important to note that the conducted analysis requires that the transformed value function and the expected return are estimated using different data.

[9] This can be seen as a consequence of Lemmas 1 and 3 in (Bhandari et al., 2018).

6.3.2 Mean-Volatility Actor-Critic Analysis

Since $\eta(\theta)$[10] is, in general, a non-concave function of θ, we do not expect that we reach a global maximum using a gradient ascent algorithm. Instead, we strive to reach a stationary point of $\eta(\theta)$, and the goal of the analysis is thus to bound $\mathbb{E}\left[\|\nabla\eta(\theta_{\hat{T}})\|_2^2\right]$ in terms of the number of used samples. Crucial to the analysis of the actor is for the gradient of $\eta(\theta)$ to be Lipschitz continuous. That is, for any $\theta_1, \theta_2 \in \mathbb{R}^{d_\theta}$, there exists a real constant $L_\eta \geq 0$ such that

$$\|\nabla\eta(\theta_1) - \nabla\eta(\theta_2)\|_2 \leq L_\eta\|\theta_1 - \theta_2\|_2.$$

The proof of this statement can be found in Appendix A.3.10. Since the actor relies on the critic for the estimation of the gradient, the convergence of the actor naturally relies on the accuracy of the critic. However, the analysis of the last section was only concerned with how far the critic was from the TD fixed point. We will thus need an additional notion to describe the approximation error incurred due to not only using a linear function, but also for using TD learning, which, in general, leads to a fixed point different from the best approximation in our space of candidate function (Tsitsiklis and Van Roy, 1997). Thus, we define the following quantity to be used in the actor's bound:

$$\xi_{appr} := \max_{\theta\in\mathbb{R}^{d_\theta}} \mathbb{E}_{s\sim d_{\mu_0,\pi_\theta}(\cdot)}\left[\left|V^\lambda_{\pi_\theta}(s) - \phi(s)^\intercal\omega^*_{J_\theta}\right|^2\right].$$

Theorem 6.3.2 (Actor's Bound)**.** *Suppose Assumptions 3, 4 and 6 hold, and suppose we run Algorithm 7 for T iterations with the critic learned as described in Theorem 6.3.1, then if $\alpha = \frac{1}{8L_\eta}$, we have:*

$$\begin{aligned}\mathbb{E}\left[\|\nabla\eta(\theta_{\hat{T}})\|_2^2\right] \leq &\sum_{t=0}^{T-1}\mathbb{E}\left[\|\omega^*_{J_t} - \omega_t\|_2^2\right]\mathcal{O}\left(\frac{1}{T}\right)\\ &+ \mathcal{O}\left(\frac{1}{B}\right) + \mathcal{O}(\xi_{appr}) + \mathcal{O}\left(\frac{L_\eta}{T}\right)\\ &+ \mathcal{O}\left(\lambda^2\left(\gamma^{2T_J} + \frac{1}{L}\right)\right),\end{aligned}$$

where ω_t is the parameter vector of the learned critic at the t^{th} iteration, and $\omega^*_{J_t}$ is the TD fixed point for the true transformed value function of policy π_{θ_t}. Furthermore, for a sufficiently small $\epsilon > 0$, to achieve an ϵ-accurate stationary point, that is,

$$\mathbb{E}\left[\|\nabla\eta(\theta_{\hat{T}})\|_2^2\right] \leq \epsilon + \mathcal{O}(\xi_{appr}),$$

the total sample complexity is:

$$T((2-\gamma)B + MT_c + LT_J) = \mathcal{O}\left(\epsilon^{-2}\log\left(\epsilon^{-1}\right)\right).$$

The proof of this theorem can be found in Appendix A.3.4. The bound in Theorem 6.3.2 is made up of five terms. The first is proportional to the average (across iterations) of how far we expect the critic to be from the TD fixed point of the true transformed

[10] $\eta(\theta) := \eta_\theta$.

value function. This is precisely the quantity bounded in Theorem 6.3.1. The second term is related to the error due to the variance of the mini-batch estimates of the gradient, and it decays by increasing the mini-batch size. The third term is the approximation error discussed before. The fourth one is an error term that decays as the number of actor iterations increases. The last term, much like the third term in Theorem 6.3.2, represents the error due to the inaccuracy of $\hat{J}$, and it decays by increasing L and T_J. Compared to the bound in (Xu et al., 2020b), our bound assumes a similar form (albeit some of the quantities are naturally defined differently in our setting), with the exception of the last term. Although estimating the expected return requires extra sampling, the theorem asserts that the sample complexity is still not worsened compared to the risk neutral case.

6.4 Experiments

In this section we empirically validate our algorithms by means of an experimental analysis on an environment called Point Reacher.

The Point Reacher Environment The agent controls a point mass that moves along the real line in order to bring it to a target location in the minimum number of steps. The state of the system is described by the position of the mass in the interval $[-10, 10]$, while the agent chooses (continuous) actions in $[-2, 2]$. If the system is in state s and the agent takes action a, the new state is $s' \sim \mathcal{N}(s + a, a^2)$ and the immediate reward is $r = -0.1|s'| + a^2$. The goal is the ball of radius 0.05 around the origin. Episodes have length at most 10 and terminate whenever the agent reaches a goal state. The initial state is drawn uniformly in $[-5.1, -5] \cup [5, 5.1]$.

Results We tested the performance of Algorithm 7 in this environment, where the used critic is again direct mini-batch TD. We considered Gaussian policies, where the mean and standard deviation are linear functions of the state. The features we used for the states are Gaussian radial basis functions. The critic also used these same features. Figure 6.1 show the performance of Algorithm 7 applied to the Point-Reacher environment. The rightmost plot shows the approximated Pareto frontier w.r.t. the expected return and the reward-volatility. Different points have been obtained by selecting different values of the risk-aversion parameter λ. It can be noticed that the algorithm can obtain different trade-offs between the two criteria, as desired. Curves in the central and the leftmost plots show smooth return and volatility curves, without signs of instability.

6.5 Related Works

The mean-volatility objective has been first introduced in (Bisi et al., 2020b), where it was optimized through a trust-region approach, but without providing any finite sample analysis results. The technique has been extended in (Zhang et al., 2021), where the optimization of the same objective was pursued by means of a framework, which allows to decompose the problem into a series of standard MDPs. The authors were able to show the asymptotic convergence of the method to a local optimum (stationary point), but they did not provide any convergence rate. Convergence to a local optimum is typically

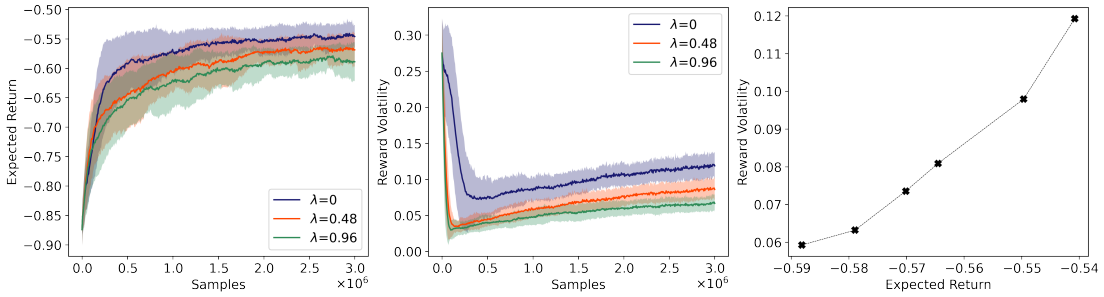

Figure 6.1: *This figures reports the performance of the mini-batch MVAC algorithm in the Point-Reacher environment. The first two plots show the progress of the expected return and the reward volatility as the number of samples increases for three values of λ. The last plot shows the approximated Pareto frontier using six values of λ chosen uniformly between 0 and 1.2.*

the best that one can hope for even for risk-neutral policy optimization approaches, unless the problem presents particular favourable features (Agarwal et al., 2021). For what concerns the risk-neutral side, several finite sample analyses have been recently developed for the actor-critic approach (Yang et al., 2018; Wu et al., 2020; Chen et al., 2021; Wang et al., 2019; Kumar et al., 2019; Xu et al., 2020b). In this work, we follow the approach suggested in (Xu et al., 2020b) to analyse the mean-volatility case, evaluating to which extent our risk-averse extension impacts the risk-neutral convergence rate. This analysis is interesting because, differently from (Chen et al., 2021) for instance, it allows to considers the continuous action case. This is important because enabling the access to continuous actions without losing the advantages of TD-learning is one of the main advantages of actor-critic schemes.

As it is the case for (Zhang et al., 2021), many risk-averse policy gradient approaches offer asymptotic convergence guarantees (Tamar et al., 2012a, 2015a; Chow et al., 2017), but they do not provide finite sample analyses. There are only few works focusing on this kind of analysis on algorithms optimizing risk-averse objective. The work in (Jiang and Powell, 2018) optimizes a dynamic coherent risk-measure that involves as static conditional risk-measure either CVaR or VaR, by means of an approximated dynamic programming approach, similar in spirit to Q-learning. The authors provide the convergence rate in terms of the expected deviation from the optimal Q-function. Recently, in (Fei et al., 2020), the authors analysed the model-free optimization of the Entropic Risk-Measure, through two different value-based algorithms. They prove a sub-linear regret bound which can be used to derive the finite-sample complexity of the approaches. While being interesting methods, we remark that these works are not directly comparable to our analysis, since they involve value-based approaches and different objectives.

6.6 Conclusions

The goal of this chapter was to shed light on the impact of risk-aversion on the sample complexity of RL algorithms. We analysed the mean-volatility case, focusing, in particular, on an actor-critic algorithm. We developed two different methods for mean volatility policy evaluation: the factored method and the direct method. Firstly, we provided a finite-sample bound for the critic algorithm, which applied the direct method to

a mini-batch TD algorithm. Secondly, we extended the analysis to the actor procedure, deriving the sample-complexity of the whole algorithm. Our results show that while increasing risk-aversion negatively affects the error bounds, the sample complexity of the algorithms remains the same as that of their risk-neutral counterparts. Finally, we tested the proposed algorithms on a stochastic environment to assess its soundness. We showed that the algorithm is effective in obtaining different trade-offs between the expected return and the reward-volatility according to the desired level of risk-aversion. A challenging future research direction could be analysing the case in which a single batch of samples is used for each iteration, in order to discover the impact of the resulting bias.

CHAPTER 7

Conclusion

In this dissertation we developed novel approaches to deal with the risk-averse reinforcement learning setting. Our goal was to make a step forward the use of RL methods in real-world contexts, by allowing to optimize risk-averse objectives with standard techniques. We contributed to this ambition through the development of algorithms, the definition of a novel risk-measure and by conducting theoretical and experimental analyses. In what follows, we will review the main contributions of the dissertation, discussing their main limitations and some possible future works.

Risk-Averse Optimization via Risk-Neutral Optimization

In Chapter 4 we presented a unified framework for risk-averse reinforcement learning, capturing several of the most popular risk measures: the conditional value at risk (CVaR) (Rockafellar and Uryasev, 2002), the Mean-Variance penalized criterion (Tamar et al., 2012b), and the whole family of concave utility functions, which includes also the entropic risk-measure (Howard and Matheson, 1972). Leveraging on previous works (Bäuerle and Rieder, 2011, 2013; Xie et al., 2018), we mapped each of the aforementioned risk-averse objectives to a nested optimization, which gives as a final result a valid solution for the original problem. We showed that the inner problem is equivalent to a standard MDP with an augmented state space, and can be solved with the usual tools. The outer problem consists, instead, in a closed form optimization which can be easily carried on for each of the risk-measures. By properly defining two characteristic functional for each of the risk-averse objectives, it is possible to unify the optimization of this set of measures under the same framework.

We developed a simple meta-algorithm, called ROSA (Risk-averse Optimization via State Augmentation), which allows to transform the trajectories obtained in the original problem to effective samples for the transformed task. The optimization is

carried on in a block coordinate fashion, alternating the optimization of the inner and the outer problem. Since the inner problem is an MDP, it can be optimized with any RL algorithm.

We conducted an extensive experimental campaign in order to validate the soundness of our method. For each of the analysed risk-measures we instantiated ROSA with different state-of-the-art algorithms. This procedure has been repeated for each of the chosen environments, allowing in every case to correctly optimize the target objective. We considered both toy problems, which allowed us to analyse a wider spectrum of base risk-neutral algorithms, and complex domains as robotic locomotion and simulated trading, which allowed us instead to test the limits of the approach. Empirical results revealed that our method combined with state-of-the-art policy optimization scales to complex domains and outperforms ad-hoc risk-sensitive algorithms, while requiring minimal additional efforts, both in terms of computation and implementation, w.r.t. learning risk-neutral policies.

Limitations and Future Directions One limitation of the state-augmentation presented in Section 4.2.1 is that it requires to have access to trajectories. This is due to the fact that we need to propagate the cumulative cost and discounting through time. However, in the offline case, this is not always possible, since data may be available only in the form of tuples, as it happens for Fitted Q-Iteration (FQI Ernst et al., 2005a). An important future direction consists, then, in extending the proposed approach to the offline RL setting, which would further increase its applicability.

The presented unified framework can be applied only to a fixed set of risk-measures. It is still unclear whether the same formulation can be extended to a wider group of them. Investigating the possibility of applying the same result to other risk-averse objective can be another possible future direction. In particular, it would be interesting to understand the reason why for some risk-measures this conversion can be done, and why for other one this seems not possible. Furthermore, the decomposition of an MDP with a non-conventional objective into a nested optimization, where the inner problem is a standard MDP, is a recurrent pattern for optimizing MDP with a modified objective (Zhang et al., 2021; Zahavy et al., 2021; Zhang et al., 2020a). This probably hides a more general formulation, which would be interesting to investigate.

From the empirical side, the extensive work that we did allowed to notice one complication with risk-averse agents. When learning from scratch, risk-averse agents tend to under-explore the environment, occasionally converging to poor local optima. However, when the risk-aversion degree is small, but not null, we also experienced an improved convergence, which surprisingly resulted in better expected return than the risk-neutral case. This suggests that risk-aversion play an important role in the learning process, hence, varying the risk-aversion during training may be beneficial for either risk-neutral or risk-averse tasks. Thus, it would be interesting to deepen this relationship under the lens of *curriculum learning* (Bengio et al., 2009).

Risk-Averse Trust Region Optimization for Reward-Volatility Reduction

In Chapter 5 we proposed a novel methodology for risk-averse reinforcement learning, exploiting, for the first time in a risk-averse setting, a safe improvement bound. This was possible thanks to the definition of a novel risk measure called reward volatility that accounts for the variability of the rewards between steps. Minimizing this risk-measure allows to obtain smoother trajectories that avoid shocks, which is a fundamental feature in a trading setting, never considered before. This measure consider a different target random variable, the per-step reward, which is distributed according to the state-action occupancy distribution. We showed interesting theoretical properties of reward-volatility. This measure allows to upper bound the variance of the return, thus, minimizing the former we also constraint the latter, making the reward volatility a proxy for variance minimization. Differently from other risk measures, we showed that it possible to derive for it a linear Bellman equation. Thanks to this property, it was possible to obtain a policy gradient theorem for the mean-volatility trade-off objective, and to extend some monotonic improvement bounds available in the risk-neutral case to the risk-averse one. Exploiting the safe bound, we obtained TRVO, a practical TRPO-like algorithm, which, thanks to a dynamic tuning of the step-size, allows to obtain an empirically fast learning speed. The proposed algorithm was tested on two financial trading environments where it was shown to outperform the baselines, obtaining better Pareto frontiers in shorter time.

Limitations and Future Directions The main limitation of this work, which does not impact the proposed approach but could prevent the application of other techniques, is the presence of a *policy-dependent* transformed reward. A first-step to better understand the role of this component has been done in (Zhang et al., 2021). In this work, the authors showed that, decomposing the mean-volatility objective with a Fenchel duality, it is possible to reduce the original problem to a nested optimization. The inner problem is an MDP which features the aforementioned reward, which is not policy dependent anymore, since computed from a fixed expected return. The outer problem instead can be solved in closed form, and it simply amounts at computing the expected return of the policy given as output from the inner MDP. This work allows to apply any RL algorithm to the inner optimization problem, thus, addressing the issue of off-policy optimization. This algorithm is called Mean-Variance Policy Iteration (MVPI). The authors presented a monotonic improvement bound for their approach and they showed convergence to local optima. MVPI include as a special case our approach, TRVO, by instantiating TRPO for the solution of the inner problem.

An interesting research direction would consists in bounding the error found by MVPI solution. Unfortunately, as for the mean-variance penalized objective, mean-volatility in inherently a non-concave objective w.r.t. the expected return. This means that the initial value of the expected return will determine the attraction basin for the optimization, making difficult any attempt to constraint the distance with the mean-volatility optimal value. It would be interesting, however, to study which regularity conditions are needed to develop efficient algorithms, capable to approach the optimal solution with the desired precision.

This work studied for the first- time a novel class of risk-measures, based on the per-step reward instead of the return. While only the variance of this random variable

was deeply analysed, here called reward-volatility, it would make sense to understand the meaning of applying other traditional risk-measure to this setting. In particular, it would be important to understand whether such reward-based risk-measure have any relationship with their return-based counterparts, and if they can be optimized in a more efficient way.

Finite Sample Analysis of Mean-Volatility Actor-Critic for Risk-Averse Reinforcement Learning

In Chapter 6 we tried to better understand the impact of risk-aversion on RL algorithms sample-complexity. For this purpose, we developed a finite-sample analysis of an actor-critic approach applied to the novel risk trade-off we defined in the previous chapter. We developed two different methods for the policy evaluation of the mean-volatility objective. The *direct* method features the use of the policy-dependent transformed reward, that has to be computed by estimating the expected return, and it can then be plugged to standard policy evaluation approaches. The *factored* method, instead, allows to estimate the the same quantity by computing two value function, the standard one and the one corresponding to the reward second moment: by composing the two functions one can obtain the mean-volatility value. We analysed the sample-complexity of estimating a critic using the direct method, employing a mini-batch TD approach with linear function approximation. To do that, we extended the analysis presented in (Xu et al., 2020b). We then completed the analysis with the study of the actor part, deriving a finite-sample bound for reaching an ϵ-accurate stationary point of the mean-volatility policy gradient. It was possible to notice that the batch size is influenced by the risk-aversion degree of the agent: the higher is λ the coefficient, the higher is the number of samples required to obtain a solution with some fixed precision. However, the results we obtained showed that the sample-complexity order is the same of the risk-neutral case, thus, allowing us to conclude that the negative impact on the batch size due to the introduction of risk-aversion is limited.

Limitations and Future Directions The main limitation of this work consists in assuming the use of separate batches of samples to estimate the value function and the expected return, which is needed to compute the transformed reward. While our analysis showed the limited impact that this approach has on the overall sample complexity, it is still an annoying aspect. Moreover, empirical experiments shows that using a single batch for computing both quantities allows for a faster convergence, suggesting that the effect of the bias may be not severe.

Extending the analysis to take explicitly into account this bias is one of the most interesting, while complex, future research directions. A possible workaround solution to the issue could consists in employing the expected return of the previous policy in place of the current one, in order to break the correlation between samples. This approach would clearly introduce a bias, but it is guaranteed to converge, being an instance of MVPI ((Zhang et al., 2021)), where an actor-critic approach in used to solve the inner optimization, which is protracted for just one iteration.

Actor-critic algorithms allows to deal with continuous action-spaces without renouncing to the advantages of temporal difference learning. This is particularly important for real-world tasks, in which finite action spaces are often just the product

of physical quantities discretization. What is also fundamental in real-world applications is to limit the interaction with the environment, which can be an expensive and time-consuming activity. Off-policy methods are, thus, the main candidate for real-life applications, since they can also exploits samples gathered by other policies (Ernst et al., 2005a; Watkins and Dayan, 1992; Mnih et al., 2015). A prospective work could consist in extending our analysis to the off-policy case (Chen et al., 2021), to obtain a batch actor-critic architecture in the style of (Melo and Lopes, 2008). Studying how the effect of being off-policy compounds with the risk-aversion one in an actor-critic algorithm (Urpí et al., 2021), is definitely a challenging but important direction for future research.

Final Remarks

Risk-averse reinforcement learning poses several questions and challenges. Furthermore, its deep connection with real-world problems makes this setting one of the most interesting lines of work for reinforcement learning research. In this dissertation we provided methods for a more efficient optimization of some risk-averse objectives, and we offered insights on the risk-averse learning process by means of theoretical analyses and experimental evaluations. The journey to make reinforcement learning a mature technology for real life applications is still at the beginning, and we just took a few more steps on this long path, but we hope that our effort could be inspiring for further and even better developments.

Appendices

APPENDIX $\mathcal{A}$

Additional Results and Proofs

In this appendix, we report additional results and proofs we have omitted in the main text of the dissertation.

A.1 Additional Results of Chapter 4

A.1.1 Additional Results

The goal of this subsection is to complement the results on the toy environments exposed in A.2.3 with the respective learning curves. All the results reported in the remaining are the average of 20 independent runs. Plots with shaded error bars report plus/minus the standard deviation. Figure A.1 and A.2 report the learning curves, respectively, for the Multi-armed bandit and the Point Reacher environments for all risk measures and base algorithms. Here we notice that MV seems the simplest risk measure to optimize as all algorithms converge quickly quickly with a stable learning behavior. On the other hand, ERM seems the most difficult and, due to its exponentiated nature, makes some algorithms (especially the off-policy ones) more unstable. Nonetheless, all curves converge to good solutions as we have already seen in the previous plots.

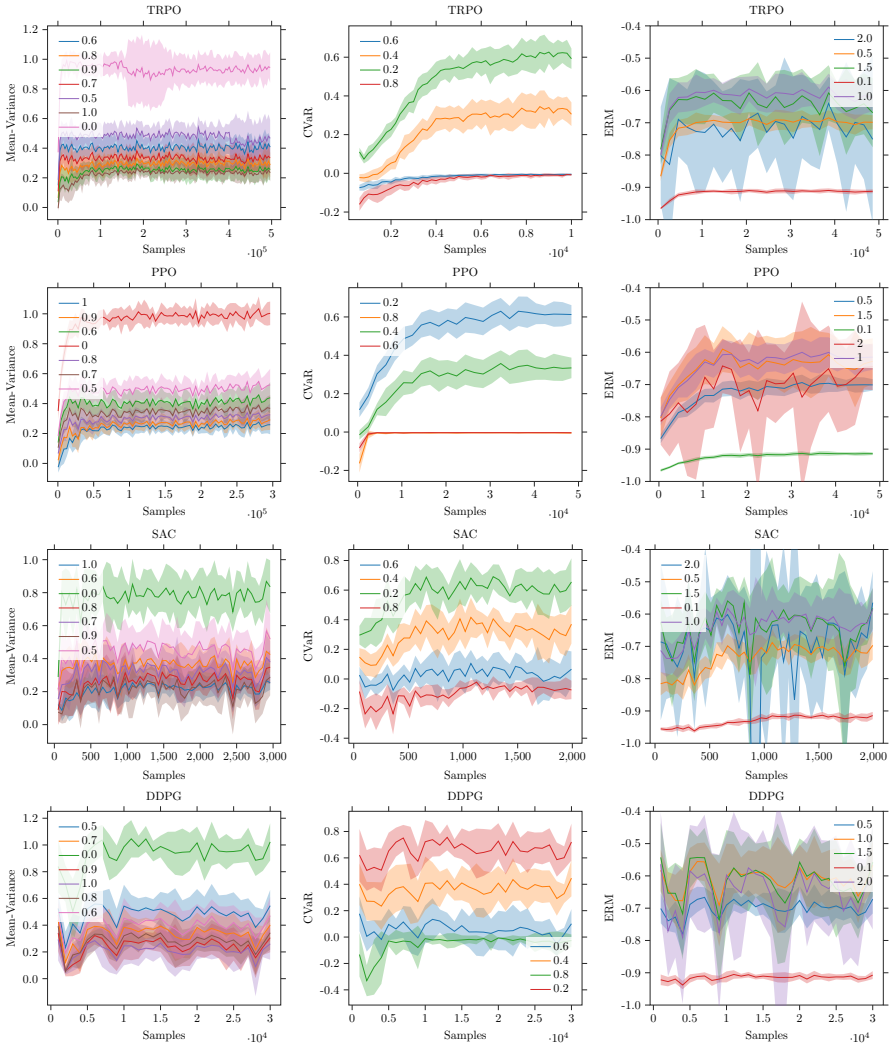

Figure A.1: *Learning curves for ROSA optimizing MV, CVaR, and ERM when combined with TRPO, PPO, SAC, and DDPG.*

A.1.2 State augmentation from batch data

One limitation of the state-augmentation derived in subsection 4.2.1 is that it requires data in the form of trajectories in order to be computed from samples. This is due to the fact that we need to propagate the cumulative cost and discounting through time. In some real-world scenarios, we may have only access to batch data in the form of transition tuples $\{(S_i, A_i, S'_i, R_i)\}_{i=1}^n$, where S_i, A_i are collected under some arbitrary distribution, $S'_i \sim P(\cdot|S_i, A_i)$, and $R_i = R(S_i, A_i, S'_i)$. Furthermore, no further interaction with the environment is allowed. Typical RL approaches for this setting include batch value-based algorithms, such as Fitted Q-Iteration (FQI Ernst et al., 2005a). Unfortunately, in this case our state augmentation cannot be exactly performed from samples as in Algorithm 1. Although we left the extension to batch data as a challenging direction for future work, here we provide some intuition on how this augmentation could be performed.

Consider the augmented MDP of Definition 4.2.3 for a fixed $\rho \in \mathbb{R}$. Using the

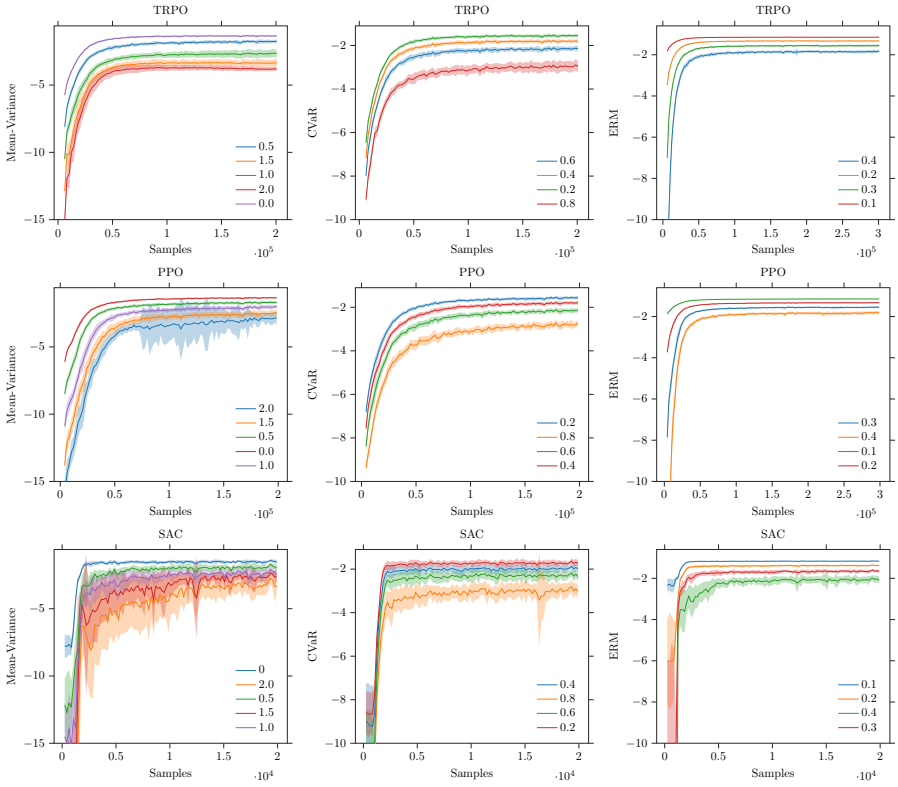

Figure A.2: *Learning curves for ROSA optimizing MV, CVaR, and ERM when combined with different risk neutral algorithms on the Point Reacher domain.*

results of (Bäuerle and Rieder, 2013), we have the following recursion for computing the optimal Q function in the augmented MDP. Starting from

$$\tilde{Q}_0(s, v, w, a) = f(v, \rho),$$

we have that, for $k \geq 0$,

$$\tilde{Q}_{k+1}(s, v, w, a) = \int_{\mathcal{S}} \max_{a' \in \mathcal{A}} Q_k(s', wR(s, a, s') + v, \gamma w, a') P(\mathrm{d}s' | s, a).$$

In order to approximate this from samples as in FQI, we samples of transitions in the augmented MDP. Fortunately, the only unknown components are the dynamics of s through the original transition model P and the single-step cost, for which we have access to samples. The dynamics of v and w, on the other hand, are known and deterministic. Therefore, a (probably naive) solution is augment the observed states with random values for v, w and compute their transition exactly given the observed cost. As a concrete example, take a transition tuple (s_i, a_i, s'_i, R'_i) that we intend to augment. First, we randomly generate $v_i \in [-R_{\max}/(1-\gamma), 0]$ and $w_i \in (0, 1]$. Then, we compute $v'_i = w_i R_i + v_i$ and $w'_i = \gamma w_i$. Finally, we set the augmented sample to $((s_i, v_i, w_i), a_i, (s'_i, v'_i, w'_i), 0)$. We can then run FQI to approximate the optimal value function by using the iterates reported above.

A.1.3 Reproducibility Details

Here we provide the configurations and hyperparameters that we adopted for all the considered algorithms in our experiments. We implemented ROSA on top of Stable Baselines (Raffin et al., 2019). For each algorithm and domain, we used the hyperparameters suggested in the library or slight variations of them.

Multi-armed Bandit

TRPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 200, entropy coefficient: 0.

PPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, clip range: 0.2, generalized advantage estimation factor: 0.95, learning rate: 0.003, batch size: 200, entropy coefficient: 0, number of mini-batches: 1.

SAC We used the custom SAC policy from Stable Baselines. The main parameters are: γ: 0.999, learning rate: 0.001, batch size: 50, buffer size: 1000, entropy coefficient: automatically learned, number of gradient steps: 5. Every 500 time steps, we collected 50 samples under the current policy to update the outer variables.

DDPG We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, batch size: 50, memory limit: 30000, number of rollouts per iteration: 10, number of training steps per iteration: 5, noise type: Ornstein-Uhlenbeck with 0.1 std. Every 500 time steps, we collected 50 samples under the current policy to update the outer variables.

Point Reacher

TRPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 2048, entropy coefficient: 0.

PPO We used the default MLP policy of Stable Baselines. The main parameters are: γ: 0.999, clip range: 0.2, generalized advantage estimation factor: 0.95, learning rate: 0.005, batch size: 2048, entropy coefficient: 0, number of mini-batches: 1.

SAC We used the custom SAC policy from Stable Baselines. The main parameters are: γ: 0.999, learning rate: 0.001, batch size: 100, buffer size: 20000, entropy coefficient: automatically learned, number of gradient steps: 5. Every 500 time steps, we collected 50 episodes (i.e., 500 additional steps) under the current policy to update the outer variables.

Trading

For this environment we used a Boltzmann policy on top of a neural-network architecture composed of 2 layers with 64 hidden neurons each. We used the following parameters for TRPO: γ: 1, generalized advantage estimation factor: 1, maximum

KL: 0.001, batch size: 700, entropy coefficient: 0, conjugate-gradient iterations: 10, conjugate-gradient damping: 0.01, value-function step size: 0.0003, value-function update iterations: 3. The episodes in this setting have a fixed length of 49 steps. The total number of iterations was 400.

The risk-aversion coefficients used for the two risk measures are $\lambda \in \{1, 5, 10, 20, 25, 30\}$ for mean-variance and $\beta \in \{-0.1, -2, -3, -3.5, -4, -5\}$ for ERM.

Walker and Hopper

We used a Gaussian policy on top of a neural-network architecture composed of 2 layers with 64 hidden neurons each. We used the following parameters for TRPO: γ: 0.999, generalized advantage estimation factor: 0.95, maximum KL: 0.01, batch size: 2048, entropy coefficient: 0. The episodes in this setting have a maximum length of 500. For PPO instead, the main parameters were: γ: 0.999, generalized advantage estimation factor: 0.95, batch size: 4096, minibatches: 32 and a learning rate starting from 0.0002 and decreasing with a linear schedule.

A.2 Additional Results of Chapter 5

A.2.1 Safe Volatility Optimization

In this section, we provide a more rigorous alternative to TRVO. To do so, we simply adapt the Safe Policy Gradient approach from (Papini et al., 2019) to our mean-volatility objective and find safe, adaptive values for the step size α and the batch size N in Algorithm 3. We restrict our analysis to *smoothing* policies:

Definition A.2.1 (Smoothing policies)**.** *Let $\Pi_\Theta = \{\pi_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^m\}$ be a class of twice-differentiable parametric policies. We call it **smoothing** if the parameter space Θ is convex and there exists a set of non-negative constants $(\psi; \kappa; \xi)$ such that, for each state and in expectation over actions, the Euclidean norm of the score function, its square Euclidean norm and the spectral norm of the observed information are upper-bounded, i.e., $\forall s \in \mathcal{S}$:*

$$\mathop{\mathbb{E}}_{a}\left[\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\|\right] \leq \psi; \quad \mathop{\mathbb{E}}_{a}\left[\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\|^2\right] \leq \kappa; \quad \mathop{\mathbb{E}}_{a}\left[\|\nabla\nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s)\|\right] \leq \xi.$$

Smoothing policies include Gaussians with fixed variance and Softmax policies (Papini et al., 2019). For smoothing policies, the performance improvement yielded by a generic parameter update is lower bounded by:

Theorem A.2.2. *Let Π_Θ be a smoothing policy class, $\boldsymbol{\theta} \in \Theta$ and $\boldsymbol{\theta}' = \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$. For any $\Delta\boldsymbol{\theta} \in \mathbb{R}^m$:*

$$\eta_{\boldsymbol{\theta}'} - \eta_{\boldsymbol{\theta}} \geq \langle \Delta\boldsymbol{\theta}, \nabla\eta_{\boldsymbol{\theta}} \rangle - \frac{L}{2}\|\Delta\boldsymbol{\theta}\|^2, \tag{A.1}$$

where:

$$L = \frac{c}{(1-\gamma)^2}\left(\frac{2\gamma\psi^2}{1-\gamma} + \kappa + \xi\right) + \frac{2R_{max}^2\psi^2}{(1-\gamma)^3}.$$

To prove this result, we need some additional Lemmas. The challenging part is bounding the spectral norm of the Hessian Matrix of η. First, we derive a compact expression for the Hessian of ν:

Lemma A.2.3. *Given a twice-differentiable parametric policy $\pi_{\boldsymbol{\theta}}$, the policy Hessian is:*

$$\begin{aligned}\mathcal{H}\nu_\pi^2 = \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \Big[&\big(\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s) + \mathcal{H} \log \pi_{\boldsymbol{\theta}}(a|s)\big) X_\pi(s,a) \\ &+ \nabla \log \pi_{\boldsymbol{\theta}}(a|s)\nabla^\top X_\pi(s,a) + \nabla X_\pi(s,a)\nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s)\Big] \\ &+ \frac{2}{1-\gamma}\nabla J \nabla^\top J.\end{aligned}$$

Proof. First note that:

$$\mathcal{H}(\pi_{\boldsymbol{\theta}} X_\pi) = X_\pi \mathcal{H}\pi_{\boldsymbol{\theta}} + \nabla\pi_{\boldsymbol{\theta}}\nabla^\top X_\pi + \nabla X_\pi \nabla^\top \pi_{\boldsymbol{\theta}} + \pi_{\boldsymbol{\theta}} \mathcal{H} X_\pi.$$

Then[1]:

$$\begin{aligned}
\mathcal{H}W_\pi(s) &= \mathcal{H}\int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a|s) X_\pi(s,a) da \\
&= \int_{\mathcal{A}} \Big[X_\pi(s,a)\mathcal{H}\pi_{\boldsymbol{\theta}}(a|s) + \nabla\pi_{\boldsymbol{\theta}}(a|s)\nabla^\top X_\pi(s,a) \\
&\qquad + \nabla X_\pi(s,a)\nabla^\top \pi_{\boldsymbol{\theta}}(a|s) + \pi(a|s)\mathcal{H}X_\pi(s,a)\Big] da \\
&= \int_{\mathcal{A}} \Big[X\mathcal{H}\pi_{\boldsymbol{\theta}} + \nabla\pi_{\boldsymbol{\theta}}\nabla^\top X + \nabla X\nabla^\top \pi_{\boldsymbol{\theta}} + \pi_{\boldsymbol{\theta}}\mathcal{H}\big((R-J_\pi)^2\big)\Big] da \\
&\qquad + \gamma \int_{\mathcal{S}} P(s'|s) W_\pi(s') ds'.
\end{aligned}$$

Since:

$$\begin{aligned}
\mathcal{H}\Big((R(s,a)-J_\pi)^2\Big) &= 2\big(R(s,a)-J_\pi\big)\mathcal{H}[R(s,a)-J_\pi] + 2\nabla J_\pi \nabla^\top J_\pi \\
&= -2\mathcal{H}J_\pi\big(R(s,a)-J_\pi\big) + 2\nabla J_\pi\nabla^\top J_\pi,
\end{aligned}$$

then, using Lemma 5.3.2 and the log-trick:

$$\begin{aligned}
\mathcal{H}W_\pi(s) &= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{s'\sim d_\pi(\cdot|s)} \Big[\int_{\mathcal{A}} [X\mathcal{H}\pi_{\boldsymbol{\theta}} + \nabla\pi_{\boldsymbol{\theta}}\nabla^\top X + \nabla X\nabla^\top\pi_{\boldsymbol{\theta}} - 2\pi_{\boldsymbol{\theta}}\mathcal{H}J_\pi\big(R-J_\pi\big)] da\Big] \\
&\qquad + 2\nabla J_\pi\nabla^\top J_\pi \int_{\mathcal{S}} d(s'|s) \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(a|s')\,\mathrm{d}a\,\mathrm{d}s' \\
&= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s'\sim d_\pi(\cdot|s) \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[(\nabla\log\pi_{\boldsymbol{\theta}}\nabla^\top\log\pi_{\boldsymbol{\theta}} + \mathcal{H}\log\pi_{\boldsymbol{\theta}})X + \nabla\log\pi_{\boldsymbol{\theta}}\nabla^\top X + \nabla X\nabla^\top\log\pi_{\boldsymbol{\theta}}\big] \\
&\qquad - \frac{2}{1-\gamma}\mathcal{H}J_\pi \mathop{\mathbb{E}}_{\substack{s\sim d_\pi(\cdot|s) \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[R-J_\pi\big] + \frac{2}{1-\gamma}\nabla J_\pi\nabla^\top J_\pi, \\
\mathcal{H}\nu_\pi^2 &= \mathop{\mathbb{E}}_{s\sim\mu}\big[\mathcal{H}W_\pi(s)\big] \\
&= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}}} \big[(\nabla\log\pi_{\boldsymbol{\theta}}\nabla^\top\log\pi_{\boldsymbol{\theta}} + \mathcal{H}\log\pi_{\boldsymbol{\theta}})X + \nabla\log\pi_{\boldsymbol{\theta}}\nabla^\top X + \nabla X\nabla^\top\log\pi_{\boldsymbol{\theta}}\big] \\
&\qquad - \frac{2}{1-\gamma}\mathcal{H}J_\pi \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}(\cdot|s) \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[R-J_\pi\big] + \frac{2}{1-\gamma}\nabla J_\pi\nabla^\top J_\pi.
\end{aligned}$$

The second term is null, from the definition (5.12) of J_π:

$$\mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[R(s,a)-J_\pi\big] = 0.$$

□

[1]For the sake of brevity, the dependence on s, a is often omitted.

It is now possible to use the results proven in (Papini et al., 2019) for the Hessian of J_π:

$$\mathcal{H}J_\pi = \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \Big[\big(\nabla \log \pi_{\boldsymbol{\theta}}(a|s) \nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s) + \mathcal{H} \log \pi_{\boldsymbol{\theta}}(a|s)\big) Q_\pi(s,a) \\ + \nabla \log \pi_{\boldsymbol{\theta}}(a|s) \nabla^\top Q_\pi(s,a) + \nabla Q_\pi(s,a) \nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s) \Big].$$

Putting everything together, the following holds:

$$\begin{aligned} \mathcal{H}\eta_\pi =& \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi} \\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \Big[\big(\nabla \log \pi_{\boldsymbol{\theta}}(a|s) \nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s) + \mathcal{H} \log \pi_{\boldsymbol{\theta}}(a|s)\big) \big[Q_\pi(s,a) - \lambda X_\pi(s,a)\big] \\ &+ \nabla \log \pi_{\boldsymbol{\theta}}(a|s) \nabla^\top \big[Q_\pi(s,a) - \lambda X_\pi(s,a)\big] + \nabla \big[Q_\pi(s,a) - \lambda X_\pi(s,a)\big] \nabla^\top \log \pi_{\boldsymbol{\theta}}(a|s) \Big] \\ &+ \frac{2}{1-\gamma} \nabla J_\pi \nabla^\top J_\pi. \end{aligned} \tag{A.2}$$

This expression allows to upper bound the spectral norm of the Hessian:

Lemma A.2.4. *Given a $(\psi;\kappa;\xi)$-smoothing policy $\pi_{\boldsymbol{\theta}}$, the spectral norm of the policy Hessian can be upper-bounded as follows:*

$$\|\mathcal{H}\eta_\pi\| \leq \frac{c}{(1-\gamma)^2} \left(\frac{2\gamma\psi^2}{1-\gamma} + \kappa + \xi \right) + \frac{2R_{max}^2 \psi^2}{(1-\gamma)^3},$$

where

$$\begin{aligned} c &\coloneqq \sup_{s\in\mathcal{S}, a\in\mathcal{S}} |\mathcal{R}(s,a) - \lambda(\mathcal{R}(s,a) - J_\pi)^2| \\ &= \max \left\{ \min \left\{ \frac{1}{4\lambda} + J_\pi; R_{max} + 4\lambda R_{max}^2 \right\}; \pm \big[R_{max} - \lambda (R_{max} - J_\pi)^2 \big] \right\}. \end{aligned}$$

Proof. First we note that:

$$\begin{aligned} |Q_\pi(s,a) - \lambda X_\pi(s,a)| &= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s'\sim d_\pi(\cdot|s) \\ a'\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \big[|R(s,a) - \lambda (R(s,a) - J_\pi)^2| \big] \\ &\leq \frac{c}{1-\gamma} \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{aligned}$$

Then, using the same argument as in Lemma 6 from (Papini et al., 2019), the following upper bounds hold:

$$\begin{aligned} \|\nabla J_\pi\| &\leq \frac{R_{max}\psi}{1-\gamma}, \\ \|\nabla (Q_\pi - \lambda X_\pi)\| &\leq \frac{\gamma}{(1-\gamma)^2} c\psi. \end{aligned}$$

Finally, applying triangle and Jensen inequalities on Equation A.2:

$$\begin{aligned}\|\mathcal{H}\eta_\pi\| \leq & \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \left[\|\nabla \log \pi_{\boldsymbol{\theta}} \nabla^\top (Q_\pi - \lambda X_\pi)\|\right] + \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \left[\|\nabla (Q_\pi - \lambda X_\pi) \nabla^\top \log \pi_{\boldsymbol{\theta}}\|\right] \\ & + \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \left[\|\nabla \log \pi_{\boldsymbol{\theta}} \nabla^\top \log \pi_{\boldsymbol{\theta}} (Q_\pi - \lambda X_\pi)\|\right] \\ & + \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu,\pi}\\ a\sim\pi_{\boldsymbol{\theta}}(\cdot|s')}} \left[\|\nabla \nabla^\top \log \pi_{\boldsymbol{\theta}} (Q_\pi - \lambda X_\pi)\|\right] \\ & + \frac{2}{1-\gamma} \|\nabla J_\pi\|^2.\end{aligned}$$

The application of the previous bounds and the smoothing assumption give the thesis. □

We can now see that Theorem A.2.2 is just an adaptation of Theorem 9 from (Papini et al., 2019) to the mean-volatility objective η, using the Hessian-norm bound from Lemma A.2.4.

In the case of stochastic gradient-ascent updates, as the ones employed in Algorithm 3, this result can be directly used to derive optimal, safe meta-parameters for Algorithm 3:

Corollary A.2.5. *Let Π_Θ be a smoothing policy class, $\boldsymbol{\theta} \in \Theta$ and $\delta \in (0, 1)$. Given a δ-confidence bound on the gradient estimation error, i.e., an $\epsilon_\delta > 0$ such that:*

$$\mathbb{P}\left(\|\widehat{\nabla}_N \eta_{\boldsymbol{\theta}} - \nabla \eta_{\boldsymbol{\theta}}\| \leq \frac{\epsilon_\delta}{\sqrt{N}}\right) \geq 1 - \delta \qquad \forall \boldsymbol{\theta} \in \Theta\, N \geq 1, \tag{A.3}$$

the guaranteed performance improvement of the stochastic gradient-ascent update $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \widehat{\nabla}_N \eta_{\boldsymbol{\theta}_k}$ is maximized by step size $\alpha^* = \frac{1}{2L}$ and batch size $N^* = \left\lceil \frac{4\epsilon_\delta^2}{\|\widehat{\nabla}_N \eta_{\boldsymbol{\theta}_k}\|^2} \right\rceil$. Moreover, with probability at least $1 - \delta$, the following non-negative performance improvement is guaranteed:

$$\eta_{\boldsymbol{\theta}_{k+1}} - \eta_{\boldsymbol{\theta}_k} \geq \frac{\|\widehat{\nabla}_N \eta_{\boldsymbol{\theta}_k}\|^2}{8L}.$$

Again, this is just an adaptation of Corollary 14 from (Papini et al., 2019) to the mean-volatility objective η, using the Hessian-norm bound from Lemma A.2.4.

Under a Gaussianity assumption on $\nabla \eta_{\boldsymbol{\theta}}$, which is reasonable for a sufficiently large batch size N, the error bound ϵ_δ can be derived from an F-distribution ellipsoidal confidence region:

Theorem A.2.6. *Let $\widehat{\nabla}_N \eta_{\boldsymbol{\theta}}$ the mean of N independent samples drawn from $\nabla \eta_{\boldsymbol{\theta}} \sim \mathcal{N}_m(\boldsymbol{\mu}, \Sigma)$. Then:*

$$\mathbb{P}(\|\widehat{\nabla}_N \eta_{\boldsymbol{\theta}} - \nabla \eta_{\boldsymbol{\theta}}\| \leq \epsilon_\delta) \geq 1 - \delta,$$

where

$$\epsilon_\delta \leq \sqrt{\frac{Nm}{N-m} \|S\| F_{1-\delta}(m; n-m)},$$

N is the batch size, m is the dimension of the parameters space Θ, $F_{1-\delta}(a;b)$ is the δ quantile of a F-distribution with a and b degrees of freedom, and $\|S\|$ is spectral norm of the sample variance matrix S generated by the gradient samples.

Proof. We recall Corollary 5.3 and Theorem 5.9 in (Härdle and Simar, 2012), adopting the same notation[2]. Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \sim \mathcal{N}_m(\boldsymbol{\mu}, \Sigma)$. The sample mean and sample variance are defined as:

$$\overline{\boldsymbol{x}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i,$$
$$S = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T.$$

Then:

$$(\overline{\boldsymbol{x}} - \boldsymbol{\mu})^\top S^{-1} (\overline{\boldsymbol{x}} - \boldsymbol{\mu}) \sim \frac{1}{N-1} T^2(m; N-1) = \frac{m}{N-m} F(m; N-m),$$

where $T^2(a;b)$ and $F(a;b)$ are respectively the Hotelling's T^2 and F distribution with a and b degrees of freedom.
Consequently, the following standard result provides a confidence region for $\boldsymbol{\mu}$:

Proposition A.2.7. *For all $\delta \in (0,1)$, $\mathbb{P}(\boldsymbol{\mu} \in E) \geq 1-\delta$, where E is the following set:*

$$E = \left\{ \boldsymbol{x} \in \mathbb{R}^m : (\overline{\boldsymbol{x}} - \boldsymbol{x})^\top S^{-1} (\overline{\boldsymbol{x}} - \boldsymbol{x}) < \frac{m}{N-m} F_{1-\delta}(m; N-m) \right\},$$

and $F_{1-\delta}(a;b)$ is the $(1-\delta)$-quantile of the F-distribution with a and b degrees of freedom.

Hence, the estimation error $\boldsymbol{\mu} - \overline{\boldsymbol{x}}$ is contained, with probability at least $1-\delta$, in the following set:

$$E_0 = \left\{ \boldsymbol{x} \in \mathbb{R}^m : \boldsymbol{x}^\top S^{-1} \boldsymbol{x} < \frac{m}{N-m} F_{1-\delta}(m; N-m) \right\},$$

which is bounded by the following ellipsoid:

$$\mathcal{E} = \left\{ \boldsymbol{x} \in \mathbb{R}^m : \boldsymbol{x}^\top A \boldsymbol{x} = 1 \right\},$$

where $A = \left(\frac{m F_{1-\delta}(m; N-m)}{N-m} S \right)^{-1}$. As a consequence, the euclidean norm of the estimation error is upper bounded by the largest semiaxis of the ellipsoid:

$$\zeta_\delta \coloneqq \|\boldsymbol{\mu} - \overline{\boldsymbol{x}}\| \leq \max_{i \in \{1,\dots,m\}} \{c_i\},$$

where $c_1, \dots, c_m$ are the semiaxes of $\mathcal{E}$. The semiaxes can be derived from the matrix A using the following equalities:

$$\mathrm{eig}_i(A) = \frac{1}{c_i^2} \qquad \text{for } i = 1, \dots, m,$$

[2]This means that we will use x to refer to $\nabla_{\boldsymbol{\eta}_{\boldsymbol{\theta}}}$ and x_i for its i-th estimation.

where $\text{eig}_i(A)$ denotes the i-th eigenvalue of A (the order does not matter). Thus, we can bound the estimation error norm as:

$$\epsilon \leq \frac{1}{\sqrt{\min_{i\in\{1,\dots,m\}}\{\text{eig}_i(A)\}}}.$$

Finally, we can just compute the largest eigenvalue of S:

$$\begin{aligned}\min_{i\in\{1,\dots,m\}}\{\text{eig}_i(A)\} &= \min_{i\in\{1,\dots,m\}}\left\{\text{eig}_i\left(\left(\frac{mF_{1-\delta}(m;N-m)}{N-m}S\right)^{-1}\right)\right\}\\ &= \frac{N-m}{mF_{1-\delta}(m;N-m)}\max_{i\in\{1,\dots,m\}}\left\{\text{eig}_i\left(S\right)\right\},\end{aligned}$$

Leading to:

$$\zeta_\delta \leq \sqrt{\frac{m}{N-m}F_{1-\delta}(m;N-m)\|S\|},$$

with probability at least $1-\delta$, where $\|S\|$ denotes the spectral norm of the sample variance matrix S (equal to the largest eigenvalue since S is positive semi-definite). Equation A.3 is verified by defining

$$\epsilon_\delta \coloneqq \sqrt{N}\zeta_\delta.$$

□

A.2.2 Exponential Utility applied on the reward

In this section we show that the exponential utility applied on the reward approximates the mean-volatility objective.
We consider the following measures:

$$\begin{aligned}J_\pi &= (1-\gamma)\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t R_t\big]\\ M_\pi &= (1-\gamma)\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t R_t^2\big]\\ \nu_\pi^2 &= (1-\gamma)\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t (R_t - J_\pi)^2\big],\end{aligned}$$

where $R_t = R(S_t, A_t)$ is the reward obtained at time t.
Let's take into account now the exponential utility applied on the reward, and its second-order Taylor expansion:

$$U(R) = e^{-cR} = 1 - cR + \frac{c^2}{2}R^2 + o(c^3)$$

Hence, if we sum all the discounted utilities of the rewards and take the expected value, we obtain:

$$\begin{aligned}\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t e^{-cR_t}\big] &= \mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t(1 - cR_t + \frac{c^2}{2}R_t^2) + o(c^3)\big] \\ &= \frac{1}{1-\gamma} - c\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t R_t] + \frac{c^2}{2}\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t R_t^2\big] + o(c^3) \\ &= \frac{1}{1-\gamma} - \frac{c}{1-\gamma}J_\pi + \frac{c^2}{2(1-\gamma)}M_\pi + o(c^3)\end{aligned}$$

Again, consider the Taylor expansion applied to the logarithm:

$$\log(\alpha + x) \approx \log(\alpha) + \frac{x}{\alpha} - \frac{x^2}{2\alpha^2}$$

As a consequence the following loose approximation holds:

$$\begin{aligned}\log(\mathop{\mathbb{E}}_{\tau|\pi}\big[\sum_t \gamma^t e^{-cR_t}\big]) &= -\log(1-\gamma) - cJ_\pi + \frac{c^2}{2}M_\pi \\ &\quad - \frac{(1-\gamma)^2}{2}[(-\frac{cJ_\pi}{1-\gamma} + \frac{c^2}{2(1-\gamma)}M_\pi)^2] + o(c^3) \\ &\approx -\log(1-\gamma) - cJ_\pi + \frac{c^2}{2}(M_\pi - J_\pi^2)\end{aligned}$$

Consequently:

$$\max_\pi -\frac{1}{c}\log(\mathop{\mathbb{E}}_{\tau|\pi}[\sum_t \gamma^t e^{cR_t}]) \approx \max_\pi J_\pi - \frac{c}{2}[M_\pi - J_\pi^2]$$

Finally, following the definition of ν_π^2, we can obtain the first-order approximation.

$$\begin{aligned}\nu_\pi^2 &= (1-\gamma)\mathop{\mathbb{E}}_{\tau|\pi}[\sum_t \gamma^t(R_t - J_\pi)^2] \\ &= (1-\gamma)\mathop{\mathbb{E}}_{\tau|\pi}[\sum_t \gamma^t(R_t^2 + J_\pi^2 - 2R_tJ_\pi)] = M_\pi - J_\pi^2\end{aligned}$$

A.2.3 Experiments

In this section, we show an empirical analysis of the performance of the VOLA-PG (Algorithm 3) and its safe version in a simplified portfolio management task taken from (Tamar et al., 2012a). We compare these results with a mean-variance policy gradient presented in the same article, which we adjusted to take into account the discounting. We first evaluate the performance of our algorithm in terms of average return during the learning phase (Figure A.3). We then compare the solutions obtained with the two algorithms, visualizing them in the mean-volatility and the mean-variance objective spaces (Figure A.4). Finally, we compare the behavior of Algorithm 3 and its safe version (Figure A.5).

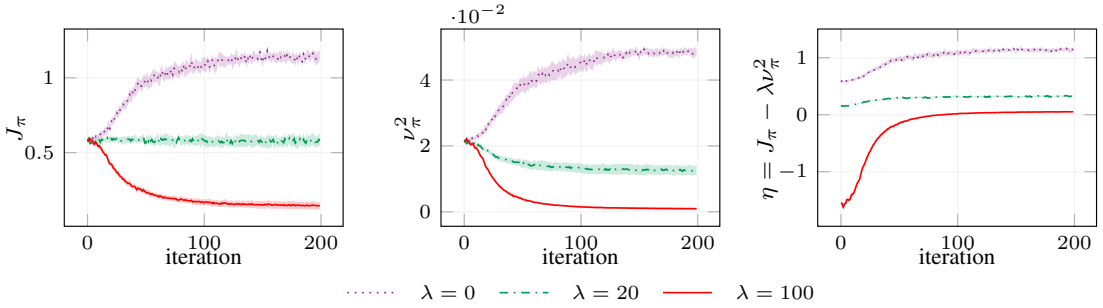

Figure A.3: *Performance of VOLA-PG for three different values of λ (without safety constraints). From left to right we can see the expected return J_π, the reward-volatility ν_π^2 and the overall performance η_π at each update of the policy during training (the shaded areas are 95% confidence intervals).*

Setting Our portfolio domain is composed of two assets: liquid and non-liquid. The liquid asset has a fixed interest rate r_l, while the non-liquid asset has a stochastic interest rate that switches between two values, r_{nl}^{low} and r_{nl}^{high}, with probability p_{switch}. If the non-liquid asset does not default (a default can happen with probability p_{risk}), it is sold when it reaches maturity after a fixed period of N time steps. At $t = 0$, the portfolio is composed only of the liquid asset. At every time step, the investor can choose to buy an amount of non-liquid assets (with a maximum of M assets), each one with a fixed cost α. Let us denote the state at time t as $\mathbf{x}(t) \in \mathbb{R}^{N+2}$, where x_1 is the allocation in the liquid asset, $x_2, \ldots, x_{N+1} \in [0, 1]$ are the allocations in the non-liquid assets (with time to maturity respectively equal to $1, \ldots, N$ time-steps), and $x_{N+2}(t) = r_{nl}(t) - \mathbb{E}_{t'<t}[r_{nl}(t')]$. When the non-liquid asset is sold, the gain is added to the liquid asset. The reward at time t is computed (unlike (Tamar et al., 2012a)) as the liquid P&L, i.e., the difference between the liquid assets at time t and $t-1$. The task parameters we used are specified below:

$$\begin{array}{lll} T = 50; & r_l = 1.001; & N = 4; \\ r_{nl}^{high} = 2; & r_{nl}^{low} = 1.1; & M = 10; \\ p_{risk} = 0.05; & p_{switch} = 0.1; & \alpha = \frac{0.2}{M}. \end{array}$$

There are $M + 1$ possible actions, and the policy we used is a neural network with two hidden layers and 10 neuron per hidden layer.

Results Figure A.3 shows three different training plots, with different values of λ. We can see that, as λ grows, the algorithm moves from the maximization of the expected return to the minimization of the reward volatility. Specifically, $\lambda = 0$ maximizes J_π, $\lambda = 100$ minimizes the variance, and the intermediate values of λ show the possible trade-offs between the two.

Figure A.4 shows how the optimal solutions of the two algorithms trade off between the maximization of the expected return and the minimization of one of the two forms of variability when changing the risk aversion parameter λ. The plot on the right shows that the mean-variance frontier obtained by Algorithm 3 almost coincides with the frontier obtained using the algorithm proposed in (Tamar et al., 2012a). This means that, at least in this domain, the return variance is equally reduced by optimizing either the mean-volatility or the mean-variance objectives. Instead, from the plot on the left, we

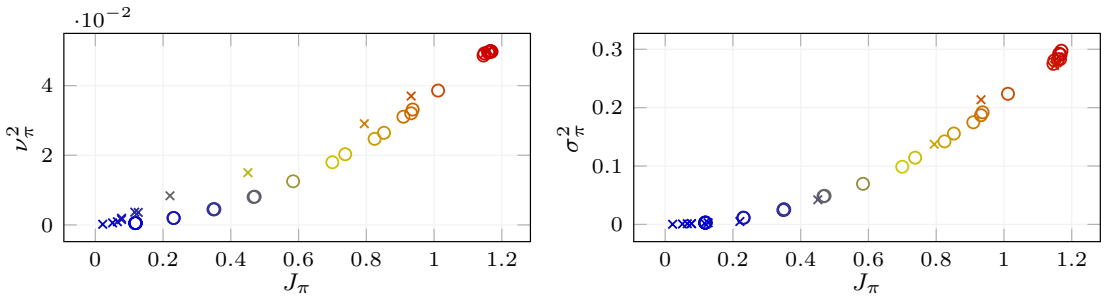

Figure A.4: *Analysis of the approximated Pareto frontier obtained by varying λ from low (top-right) to high (bottom-right) values. The circles are obtained using VOLA-PG, the crosses are obtained using the algorithm proposed in (Tamar et al., 2012a).*

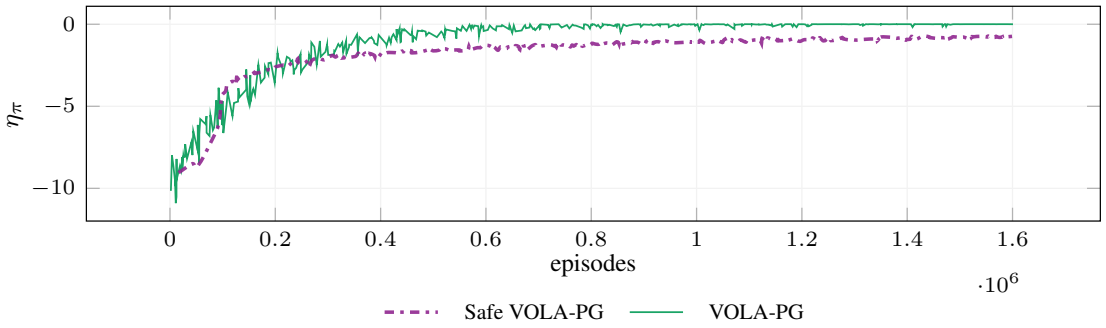

Figure A.5: *Comparison of the (unnormalized) objective improvement for each episode between the safe and the standard VOLA-PG algorithm ($\lambda = 60$).*

can notice that the reward-volatility is better optimized with VOLA-PG. These results are consistent with Lemma 5.2.2.

In Figure A.5 we can see that the safe variant of VOLA-PG shows monotonic improvement, but a slow convergence. The non-safe version, despite the initially faster learning, shows policy oscillations. It is easy to understand why in a real-world environment the learning plot obtained with the safe algorithm is preferable.

A.3 Additional Results of Chapter 6

A.3.1 Auxiliary Lemmas

Lemma A.3.1. *Suppose A is an $n \times n$ invertible matrix, then*

$$\left\|A^{-1}\right\|_2 = \frac{1}{\min_i \sigma_i},$$

where, for a matrix, $\|\cdot\|_2$ denotes its spectral norm, and σ_i is the i^{th} singular value of A.

Proof. (The following proof is due to (Grant, 2014).) By Theorem 4.3 in (Dahleh et al., 2004), we have that

$$\min_i \sigma_i = \inf_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}.$$

And since A is invertible, $\min_i \sigma_i > 0$. We then have that

$$\begin{aligned}
\frac{1}{\min_i \sigma_i} &= \sup_{x \neq 0} \frac{\|x\|_2}{\|Ax\|_2} \\
&= \sup_{A^{-1}y \neq 0} \frac{\|A^{-1}y\|_2}{\|y\|_2} \\
&= \sup_{y \neq 0} \frac{\|A^{-1}y\|_2}{\|y\|_2} \\
&= \|A^{-1}\|_2,
\end{aligned}$$

where we have made the substitution $Ax = y$ and utilized the fact that $A^{-1}y = 0$ iff $y = 0$ since A is invertible. □

Lemma A.3.2. *With ζ_i denoting the i^{th} central moment of $G_{0:T_J-1}$, we have*

i. $\zeta_2 \leq R_{max}^2$.

ii. $|\zeta_3| \leq \frac{4\sqrt{3}}{9} R_{max}^3 \leq R_{max}^3$.

iii. $\zeta_4 \leq \frac{4}{3} R_{max}^4 \leq 2R_{max}^4$.

Proof. For a random variable X upper-bounded by M and lower bounded by m, with μ_i denoting its ith central moment, we have by Popoviciu's inequality that

$$\mu_2 \leq \frac{(M-m)^2}{4}.$$

Thus, we have that

$$\zeta_2 \leq \frac{(2R_{max})^2}{4} = R_{max}^2,$$

since we can take $M = R_{max}$ and $m = -R_{max}$, proving the first item. For the second item we have from Theorem (2.3) in (Sharma et al., 2015) that

$$|\mu_3| \leq \frac{(M-m)^3}{6\sqrt{3}}.$$

Which means that in our case we shall have that

$$|\zeta_3| \leq \frac{(2R_{max})^3}{6\sqrt{3}} = \frac{8R_{max}^3}{6\sqrt{3}} = \frac{4\sqrt{3}}{9}R_{max}^3 \leq R_{max}^3.$$

Finally, for the third item, Theorem (2.1) in (Sharma et al., 2015) states that

$$\mu_4 \leq \frac{(M-m)^4}{12}.$$

And for us,

$$\zeta_4 \leq \frac{(2R_{max})^4}{12} \leq \frac{4}{3}R_{max}^4 \leq 2R_{max}^4.$$

□

Lemma A.3.3. *The following holds when $\hat{J}$ is learned using Algorithm 5:*

$$|J - \mathbb{E}[\hat{J}]| \leq \gamma^{T_J} R_{max}.$$

Proof. Since $\hat{J}$ is a sample average of instances of $G_{0:T_J-1}$, its expected value is the same as that of $G_{0:T_J-1}$, which is $\overline{G}_{0:T_J-1}$. Moreover, we remarked earlier that $J = \overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty}$, this then means that

$$|J - \mathbb{E}[\hat{J}]| = |J - \overline{G}_{0:T_J-1}| = |\overline{G}_{T_J:\infty}| \leq \gamma^{T_J} R_{max},$$

which concludes the proof. □

Lemma A.3.4. *With $\hat{J}$ learned using Algorithm 5, we have that*

$$|J^2 - \mathbb{E}[\hat{J}^2]| \leq R_{max}^2 \left(2\gamma^{T_J} + \frac{1}{L}\right).$$

Proof. Recall that for a random variable X, $\text{Var}(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2$. Which means that $\mathbb{E}[\hat{J}^2] = \mathbb{E}[\hat{J}]^2 + \text{Var}(\hat{J})$. Consequently,

$$\begin{aligned}
|J^2 - \mathbb{E}[\hat{J}^2]| &= |J^2 - \mathbb{E}[\hat{J}]^2 + \text{Var}(\hat{J})| \\
&\overset{(1)}{=} \left| J^2 - \mathbb{E}[\hat{J}]^2 + \frac{\zeta_2}{L} \right| \\
&\leq |J^2 - \mathbb{E}[\hat{J}]^2| + \frac{\zeta_2}{L} \\
&\overset{(2)}{\leq} |J^2 - \mathbb{E}[\hat{J}]^2| + \frac{R_{max}^2}{L},
\end{aligned}$$

where (1) holds since $\hat{J}$ is an empirical mean of L instances of $G_{0:T_J-1}$, and thus $\text{Var}(\hat{J}) = \frac{\text{Var}(G_{0:T_J-1})}{L} = \frac{\zeta_2}{L}$. The step labelled (2) then follows by Lemma A.3.2.i. To proceed, remember that $\mathbb{E}[\hat{J}] = \overline{G}_{0:T_J-1}$, and that $J = \overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty}$. This

means that

$$\begin{aligned}
|J^2 - \mathbb{E}[\hat{J}^2]| &\leq \left|(\overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty})^2 - \overline{G}^2_{0:T_J-1}\right| + \frac{R^2_{max}}{L} \\
&= \left|\overline{G}_{T_J:\infty}(2\overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty})\right| + \frac{R^2_{max}}{L} \\
&\leq \left|\overline{G}_{T_J:\infty}\right|(2\left|\overline{G}_{0:T_J-1}\right| + \left|\overline{G}_{T_J:\infty}\right|) + \frac{R^2_{max}}{L} \\
&\leq \gamma^{T_J} R_{max}\left(2(1-\gamma^{T_J})R_{max} + \gamma^{T_J}R_{max}\right) + \frac{R^2_{max}}{L} \\
&= \gamma^{T_J}(2-\gamma^{T_J})R^2_{max} + \frac{R^2_{max}}{L} \\
&\leq 2\gamma^{T_J}R^2_{max} + \frac{R^2_{max}}{L} \\
&= R^2_{max}\left(2\gamma^{T_J} + \frac{1}{L}\right),
\end{aligned}$$

where we used in the second inequality the fact that $|\overline{G}_{0:T_J-1}| \leq (1-\gamma^{T_J})R_{max}$ and that $|\overline{G}_{T_J:\infty}| \leq \gamma^{T_J}R_{max}$. □

Lemma A.3.5. *For a generic random variable X with mean $\mathbb{E}[X]$ and sample mean $\hat{X} = \frac{1}{N}\sum_{i=1}^N X_i$, where $X_1 \dots X_N$ are i.i.d. copies of X, we have that*

$$\mathit{Var}[\hat{X}^2] = 4\mathbb{E}[X]^2\frac{\mu_2}{N} + \frac{2\mu_2^2 + 4\mu_3\mathbb{E}[X]}{N^2} + \frac{\mu_4 - 3\mu_2^2}{N^3},$$

where μ_i is $X's$ ith central moment defined as: $\mu_i = \mathbb{E}[(X - \mathbb{E}[X])^i]$.

Proof.

$$\begin{aligned}
\text{Var}[\hat{X}^2] &= \mathbb{E}[\hat{X}^4] - \mathbb{E}[\hat{X}^2]^2 \\
&= \mathbb{E}[\hat{X}^4] - (\mathbb{E}[X]^2 + \text{Var}[\hat{X}])^2 \\
&= \mathbb{E}[\hat{X}^4] - \mathbb{E}[X]^4 - 2\mathbb{E}[X]^2\text{Var}[\hat{X}] - \text{Var}[\hat{X}]^2 \\
&= \mathbb{E}[\hat{X}^4] - \mathbb{E}[X]^4 - 2\mathbb{E}[X]^2\frac{\text{Var}[X]}{N} - \frac{\text{Var}[X]^2}{N^2} \\
&= \mathbb{E}[\hat{X}^4] - \mathbb{E}[X]^4 - 2\mathbb{E}[X]^2\frac{\mu_2}{N} - \frac{\mu_2^2}{N^2}
\end{aligned}$$

From (Angelova, 2012), we have:

$$\mathbb{E}[\hat{X}^4] = \mathbb{E}[X]^4 + 6\mathbb{E}[X]^2\frac{\mu_2}{N} + \frac{3\mu_2^2 + 4\mu_3\mathbb{E}[X]}{N^2} + \frac{\mu_4 - 3\mu_2^2}{N^3}.$$

The result follows by plugging this back in the previous equation. □

Lemma A.3.6. *Consider d real valued vectors $a_1, \dots, a_d \in \mathbb{R}^n$, we have that:*

(i) $\forall i, j \in \{1, \dots, d\} : \ |\langle a_i, b_j\rangle| \leq \frac{1}{2}\left(\|a_i\|_2^2 + \|b_i\|_2^2\right).$

(ii) $\left\|\sum_{i=1}^d a_i\right\|_2^2 \leq d\sum_{i=1}^d \|a_i\|_2^2$

Proof. For any i, j we have:

$\|a_i + a_j\|_2^2 = \|a_i\|_2^2 + \|a_j\|_2^2 + 2\langle a_i, a_j\rangle \geq 0$, and $\|a_i - a_j\|_2^2 = \|a_i\|_2^2 + \|a_j\|_2^2 - 2\langle a_i, a_j\rangle \geq 0$,

hence, trivially:

$$-\frac{1}{2}\|a_i\|_2^2 - \frac{1}{2}\|a_j\|_2^2 \leq \langle a_i, a_j\rangle, \text{ and } \frac{1}{2}\|a_i\|_2^2 + \frac{1}{2}\|a_j\|_2^2 \geq \langle a_i, a_j\rangle,$$

which proves **(i)**.

By repeatedly applying **(i)** to the cross-terms of $\left\|\sum_{i=1}^d a_i\right\|_2^2$, we obtain:

$$\left\|\sum_{i=1}^d a_i\right\|_2^2 = \sum_{i=1}^d \|a_i\|_2^2 + 2\sum_{i>j}^d \langle a_i, a_j\rangle \leq \sum_{i=1}^d \|a_i\|_2^2 + \sum_{i>j}^d \left(\|a_i\|_2^2 + \|a_j\|_2^2\right) = d\sum_{i=1}^d \|a_i\|_2^2,$$

since each index is counted $d-1$ times in the summation. □

Lemma A.3.7. *Suppose Assumptions 3 and 6 hold, then $\forall\, \theta_1, \theta_2 \in \mathbb{R}^{d_\theta}$, we have*

$$\|d_{I,\theta_1}(.,.) - d_{I,\theta_2}(.,.)\|_{TV} \leq C_d\|\theta_1 - \theta_2\|_2,$$

where $C_d := C_\pi\left(1 + \lceil log_\rho \kappa^{-1}\rceil + \frac{1}{1-\rho}\right)$, and $d_{I,\theta}(s,a) := d_{I,\theta}(s)\pi(a|s)$, where $d_{I,\theta}(\cdot)$ is the (normalized) discounted state distribution when using policy π_θ and starting from $I(\cdot)$, which is an initialization distribution over the states; it can be taken as $\mu_0(.)$ (the initial state distribution) or $P(.|s', a')$ for any fixed state-action pair (s', a').

Proof. See Lemma 3 in (Xu et al., 2020b). □

Lemma A.3.8. *Suppose Assumptions 3 and 6 hold, then $\forall\, \theta_1, \theta_2 \in \mathbb{R}^{d_\theta}$, we have*

$$|J_{\theta_1} - J_{\theta_2}| \leq L_J\|\theta_1 - \theta_2\|_2,$$

where $L_J := 2R_{\max}(C_d + C_\pi)$.

Proof.

$$\begin{aligned}
&|J_{\theta_1} - J_{\theta_2}| \\
&= \left|(1-\gamma)\int_s (V_{\theta_1}(s) - V_{\theta_2}(s))\mu(ds)\right| \\
&\leq (1-\gamma)\int_s |V_{\theta_1}(s) - V_{\theta_2}(s)|\mu(ds) \\
&\leq (1-\gamma)\int_s \left|\int_a Q_{\theta_1}(a,s)\pi_{\theta_1}(da|s) - \int_a Q_{\theta_2}(a,s)\pi_{\theta_2}(da|s)\right|\mu(ds) \\
&\leq (1-\gamma)\int_s \left|\int_a Q_{\theta_1}(a,s)\pi_{\theta_1}(da|s) \pm \int_a Q_{\theta_2}(a,s)\pi_{\theta_1}(da|s) - \int_a Q_{\theta_2}(a,s)\pi_{\theta_2}(da|s)\right|\mu(ds) \\
&\leq (1-\gamma)\int_s\int_a |(Q_{\theta_1}(a,s) - Q_{\theta_2}(a,s))|\pi_{\theta_1}(da|s)\mu(ds) \\
&\quad + (1-\gamma)\int_s\int_a |Q_{\theta_2}(a,s)|\,|\pi_{\theta_1}(da|s) - \pi_{\theta_2}(da|s)|\mu(ds)
\end{aligned}$$

By Lemma 4 in (Xu et al., 2020b), $|Q_{\theta_1}(s,a) - Q_{\theta_2}(s,a)| \leq \frac{2R_{\max}C_d}{1-\gamma}\|\theta_1 - \theta_2\|_2 \, \forall (s,a) \in S \times A$. Using this, and assumption 3.v, we have that

$$
\begin{aligned}
&|J_{\theta_1} - J_{\theta_2}| \\
&\leq 2R_{\max}C_d\|\theta_1 - \theta_2\|_2 + R_{\max}\int_s\int_a |\pi_{\theta_1}(da|s) - \pi_{\theta_2}(da|s)|\mu(ds) \\
&\leq 2R_{\max}C_d\|\theta_1 - \theta_2\|_2 + 2R_{\max}C_\pi\|\theta_1 - \theta_2\|_2 \\
&= 2R_{\max}(C_d + C_\pi)\|\theta_1 - \theta_2\|_2
\end{aligned}
$$

□

Lemma A.3.9. *Suppose Assumptions 3 and 6 hold, then* $\forall\, \theta_1, \theta_2 \in \mathbb{R}^{d_\theta}$ *and* $\forall\, (s,a) \in S \times A$*, we have*

$$\left|Q^\lambda_{\theta_1}(s,a) - Q^\lambda_{\theta_2}(s,a)\right| \leq L_{Q^\lambda}\|\theta_1 - \theta_2\|_2,$$

where $L_{Q^\lambda} := \frac{2C_d R_{\lambda,\max} + 4\lambda L_J R_{\max}}{1-\gamma} = \frac{2C_d R_{\max} + 8\lambda R^2_{\max}(2C_d + C_\pi)}{1-\gamma}$*, and* $\lambda \geq 0$.

Proof. By definition,

$$
\begin{aligned}
Q^\lambda_\theta(s,a) &= \frac{1}{1-\gamma}\mathop{\mathbb{E}}_{\substack{s'\sim d_\theta(\cdot|s,a)\\ a'\sim\pi_\theta(\cdot|s')}}\left[R^\lambda_\theta(s,a)\right] \\
&= \frac{1}{1-\gamma}\int_{s'}\int_{a'} R^\lambda_\theta(s',a')d_\theta(ds'|s,a)\pi_\theta(da'|s') \\
&= \frac{1}{1-\gamma}\int_{(s',a')} R^\lambda_\theta(s',a')d_\theta(ds',da'|s,a),
\end{aligned}
$$

where $d_\theta(s',a'|s,a) := d_\theta(s'|s,a)\pi_\theta(a'|s')$, and $d_\theta(\cdot|s,a)$ is the (normalized) discounted state distribution when using policy π_θ after taking action a in state s. We then have

that

$$
\begin{aligned}
&(1-\gamma)\left|Q^{\lambda}_{\theta_1}(s,a) - Q^{\lambda}_{\theta_2}(s,a)\right| \\
&= \left| \int_{(s',a')} \left[R^{\lambda}_{\theta_1}(s',a') d_{\theta_1}(ds',da'|s,a) - R^{\lambda}_{\theta_2}(s',a') d_{\theta_2}(ds',da'|s,a) \right] \right| \\
&= \left| \int_{(s',a')} \left[R^{\lambda}_{\theta_1}(s',a') d_{\theta_1}(ds',da'|s,a) \pm R^{\lambda}_{\theta_1}(s',a') d_{\theta_2}(ds',da'|s,a) - R^{\lambda}_{\theta_2}(s',a') d_{\theta_2}(ds',da'|s,a) \right] \right| \\
&= \left| \int_{(s',a')} R^{\lambda}_{\theta_1}(s',a') (d_{\theta_1}(ds',da'|s,a) - d_{\theta_2}(ds',da'|s,a)) \right| \\
&\quad + \left| \int_{(s',a')} (R^{\lambda}_{\theta_1}(s',a') - R^{\lambda}_{\theta_2}(s',a')) d_{\theta_2}(ds',da'|s,a) \right| \\
&\leq \int_{(s',a')} \left| R^{\lambda}_{\theta_1}(s',a') \right| \left| d_{\theta_1}(ds',da'|s,a) - d_{\theta_2}(ds',da'|s,a) \right| \\
&\quad + \int_{(s',a')} \left| R^{\lambda}_{\theta_1}(s',a') - R^{\lambda}_{\theta_2}(s',a') \right| d_{\theta_2}(ds',da'|s,a) \\
&\leq R_{\lambda,\max} \int_{(s',a')} \left| d_{\theta_1}(ds',da'|s,a) - d_{\theta_2}(ds',da'|s,a) \right| \\
&\quad + \int_{(s',a')} \left| 2\lambda R(s',a')(J_{\theta_1} - J_{\theta_2}) - \lambda (J^2_{\theta_1} - J^2_{\theta_2}) \right| d_{\theta_2}(ds',da'|s,a) \\
&\leq 2C_d R_{\lambda,\max} \|\theta_1 - \theta_2\|_2 \\
&\quad + \int_{(s',a')} \left| 2\lambda R(s',a')(J_{\theta_1} - J_{\theta_2}) - \lambda (J_{\theta_1} + J_{\theta_2})(J_{\theta_1} - J_{\theta_2}) \right| d_{\theta_2}(ds',da'|s,a) \\
&\leq 2C_d R_{\lambda,\max} \|\theta_1 - \theta_2\|_2 \\
&\quad + \int_{(s',a')} \left| \lambda (2R(s',a') - (J_{\theta_1} + J_{\theta_2})) \right| \left| J_{\theta_1} - J_{\theta_2} \right| d_{\theta_2}(ds',da'|s,a) \\
&\leq 2C_d R_{\lambda,\max} \|\theta_1 - \theta_2\|_2 + 4\lambda R_{\max} |J_{\theta_1} - J_{\theta_2}| \\
&\leq 2C_d R_{\lambda,\max} \|\theta_1 - \theta_2\|_2 + 4\lambda L_J R_{\max} \|\theta_1 - \theta_2\|_2 \\
&= \left(2C_d R_{\lambda,\max} + 4\lambda L_J R_{\max}\right) \|\theta_1 - \theta_2\|_2 \\
&= \left(2C_d R_{\max} + 8\lambda R^2_{\max}(2C_d + C_\pi)\right) \|\theta_1 - \theta_2\|_2.
\end{aligned}
$$

□

Lemma A.3.10. *Suppose Assumptions 3 and 6 hold, then* $\forall\, \theta_1, \theta_2$*, we have*

$$\|\nabla \eta_{\theta_1} - \nabla \eta_{\theta_2}\|_2 \leq L_\eta \|\theta_1 - \theta_2\|_2,$$

where $L_\eta := \frac{2R_{\lambda,\max} C_\psi C_d}{1-\gamma} + C_\psi L_{Q^\lambda} + \frac{R_{\lambda,\max} L_\psi}{1-\gamma}$*, and* $\lambda \geq 0$.

Proof.

$$
\begin{aligned}
&\|\nabla\eta_{\theta_1} - \nabla\eta_{\theta_2}\|_2 \\
&= \left\| \int_{(s,a)} \left[\psi_{\theta_1}(s,a) Q^{\lambda}_{\theta_1}(s,a) d_{\mu,\theta_1}(ds,da) - \psi_{\theta_2}(s,a) Q^{\lambda}_{\theta_2}(s,a) d_{\mu,\theta_2}(ds,da) \right] \right\|_2 \\
&\leq \int_{(s,a)} \left\| Q^{\lambda}_{\theta_1}(s,a) \psi_{\theta_1}(s,a) \right\|_2 \left| d_{\mu,\theta_1}(ds,da) - d_{\mu,\theta_2}(ds,da) \right| \\
&\quad + \int_{(s,a)} \left| Q^{\lambda}_{\theta_1}(s,a) - Q^{\lambda}_{\theta_2}(s,a) \right| \|\psi_{\theta_1}(s,a)\|_2 \, d_{\mu,\theta_2}(ds,da) \\
&\quad + \int_{(s,a)} \left| Q^{\lambda}_{\theta_2}(s,a) \right| \|\psi_{\theta_1}(s,a) - \psi_{\theta_2}(s,a)\|_2 \, d_{\mu,\theta_2}(ds,da) \\
&\leq \frac{2R_{\lambda,\max} C_\psi C_d}{1-\gamma} \|\theta_1 - \theta_2\|_2 + C_\psi L_{Q^\lambda} \|\theta_1 - \theta_2\|_2 + \frac{R_{\lambda,\max} L_\psi}{1-\gamma} \|\theta_1 - \theta_2\|_2 \\
&= \left(\frac{2R_{\lambda,\max} C_\psi C_d}{1-\gamma} + C_\psi L_{Q^\lambda} + \frac{R_{\lambda,\max} L_\psi}{1-\gamma} \right) \|\theta_1 - \theta_2\|_2,
\end{aligned}
$$

where the last inequality follows from Assumption 3, Lemma A.3.7, and Lemma A.3.9. □

A.3.2 Analyzing the Monte-Carlo Estimation of the Expected Return

Proposition A.3.11. *Suppose, for a given policy, an estimate $\hat{J}$ is obtained using Algorithm 5, and that Assumption 3.i holds, then we have*

$$
\mathbb{E}\left[\left(J - \hat{J} \right)^2 \right] \leq \gamma^{2T_J} R^2_{\max} + \frac{R^2_{\max}}{L}.
$$

Proof. We begin with a bias-variance decomposition:

$$
\begin{aligned}
\mathbb{E}\left[\left(J - \hat{J} \right)^2 \right] &= \mathbb{E}\left[\left(J - \mathbb{E}[\hat{J}] + \mathbb{E}[\hat{J}] - \hat{J} \right)^2 \right] \\
&= \mathbb{E}\left[(J - \mathbb{E}[\hat{J}])^2 \right] + \mathbb{E}\left[\left(\mathbb{E}[\hat{J}] - \hat{J} \right)^2 \right] \\
&= \left(J - \mathbb{E}[\hat{J}] \right)^2 + Var(\hat{J}),
\end{aligned}
$$

where the second equality holds since $2\Big(J - \mathbb{E}[\hat{J}]\Big)\mathbb{E}\Big[\Big(\mathbb{E}[\hat{J}] - \hat{J}\Big)\Big] = 0$. For the bias term, we know from Lemma A.3.3 that

$$
\left| J - \mathbb{E}[\hat{J}] \right| \leq \gamma^{T_J} R_{\max}.
$$

Thus, $\Big(J - \mathbb{E}[\hat{J}]\Big)^2 \leq \gamma^{2T_J} R^2_{\max}$. As for the variance, since $\hat{J}$ is a sample mean of $G_0, ..., G_{L-1}$, then $Var(\hat{J}) = \frac{\zeta_2}{L}$. Combining both terms and applying Lemma A.3.2, we get

$$
\mathbb{E}\left[\left(J - \hat{J} \right)^2 \right] \leq \gamma^{2T_J} R^2_{\max} + \frac{\zeta_2}{L} \leq \gamma^{2T_J} R^2_{\max} + \frac{R^2_{\max}}{L}.
$$

□

Proposition A.3.12. *Suppose, for a given policy, an estimate $\hat{J}$ is obtained using Algorithm 5, and that Assumption 3.i holds, then we have*

$$\mathbb{E}\left[\left(J^2 - \hat{J}^2\right)^2\right] \leq 4R_{\max}^4\gamma^{2T_J} + 4R_{\max}^2\frac{\zeta_2}{L} + \frac{3\zeta_2^2 + 4|\zeta_3|R_{\max}}{L^2} + \frac{\zeta_4 - 3\zeta_2^2}{L^3}$$
$$\leq 4R_{\max}^4\gamma^{2T_J} + R_{\max}^4\left(\frac{4}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right).$$

Proof. We, again, start with a bias-variance decomposition:

$$\begin{aligned}\mathbb{E}\left[\left(J^2 - \hat{J}^2\right)^2\right] &= \mathbb{E}\left[\left(J^2 - \mathbb{E}\left[\hat{J}^2\right] + \mathbb{E}\left[\hat{J}^2\right] - \hat{J}^2\right)^2\right] \\ &= \mathbb{E}\left[\left(J^2 - \mathbb{E}\left[\hat{J}^2\right]\right)^2\right] + \mathbb{E}\left[\left(\mathbb{E}\left[\hat{J}^2\right] - \hat{J}^2\right)^2\right] \\ &= \left(J^2 - \mathbb{E}\left[\hat{J}^2\right]\right)^2 + Var(\hat{J}^2).\end{aligned}$$

For the bias term, similar to what we did in Lemma A.3.4, we have that

$$\begin{aligned}\left|J^2 - \mathbb{E}[\hat{J}^2]\right| &= \left|J^2 - \mathbb{E}[\hat{J}]^2 - \frac{\zeta_2}{L}\right| \\ &\leq \left|J^2 - \mathbb{E}[\hat{J}]^2\right| + \frac{\zeta_2}{L} \\ &= \left|(\overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty})^2 - \overline{G}_{0:T_J-1}^2\right| + \frac{\zeta_2}{L} \\ &= \left|\overline{G}_{T_J:\infty}(2\overline{G}_{0:T_J-1} + \overline{G}_{T_J:\infty})\right| + \frac{\zeta_2}{L} \\ &\leq \gamma^{T_J}R_{\max}\left(2(1-\gamma^{T_J})R_{\max} + \gamma^{T_J}R_{\max}\right) + \frac{\zeta_2}{L} \\ &= \gamma^{T_J}(2-\gamma^{T_J})R_{\max}^2 + \frac{\zeta_2}{L} \\ &\leq 2\gamma^{T_J}R_{\max}^2 + \frac{\zeta_2}{L}.\end{aligned}$$

Thus,

$$\left(J^2 - \mathbb{E}\left[\hat{J}^2\right]\right)^2 \leq \left(2\gamma^{T_J}R_{\max}^2 + \frac{\zeta_2}{L}\right)^2 = 4\gamma^{2T_J}R_{\max}^4 + 4\gamma^{T_J}R_{\max}^2\frac{\zeta_2}{L} + \frac{\zeta_2^2}{L^2}.$$

For the variance, we apply Lemma A.3.5:

$$Var(\hat{J}^2) = 4\overline{G}_{0:T_J-1}^2\frac{\zeta_2}{L} + \frac{2\zeta_2^2 + 4\zeta_3\overline{G}_{0:T_J-1}}{L^2} + \frac{\zeta_4 - 3\zeta_2^2}{L^3}.$$

Putting everything together, we have

$$
\begin{aligned}
&\mathbb{E}\left[\left(J^2-\hat{J}^2\right)^2\right] \\
&\leq 4\gamma^{2T_J}R_{\max}^4+4\gamma^{T_J}R_{\max}^2\frac{\zeta_2}{L}+\frac{\zeta_2^2}{L^2} \\
&\quad+4\overline{G}_{0:T_J-1}^2\frac{\zeta_2}{L}+\frac{2\zeta_2^2+4\zeta_3\overline{G}_{0:T_J-1}}{L^2}+\frac{\zeta_4-3\zeta_2^2}{L^3} \\
&\leq 4\gamma^{2T_J}R_{\max}^4+4(\overline{G}_{0:T_J-1}^2+\gamma^{T_J}R_{\max}^2)\frac{\zeta_2}{L}+\frac{3\zeta_2^2+4|\zeta_3|R_{\max}}{L^2}+\frac{\zeta_4-3\zeta_2^2}{L^3} \\
&\leq 4\gamma^{2T_J}R_{\max}^4+4R_{\max}^2((1-\gamma^{T_J})^2+\gamma^{T_J})\frac{\zeta_2}{L}+\frac{3\zeta_2^2+4|\zeta_3|R_{\max}}{L^2}+\frac{\zeta_4-3\zeta_2^2}{L^3} \\
&= 4\gamma^{2T_J}R_{\max}^4+4R_{\max}^2(1+\gamma^{2T_J}-\gamma^{T_J})\frac{\zeta_2}{L}+\frac{3\zeta_2^2+4|\zeta_3|R_{\max}}{L^2}+\frac{\zeta_4-3\zeta_2^2}{L^3} \\
&\leq 4\gamma^{2T_J}R_{\max}^4+4R_{\max}^2\frac{\zeta_2}{L}+\frac{3\zeta_2^2+4|\zeta_3|R_{\max}}{L^2}+\frac{\zeta_4-3\zeta_2^2}{L^3}.
\end{aligned}
$$

Furthermore, we can apply Lemma A.3.2 and get

$$
\begin{aligned}
\mathbb{E}\left[\left(J^2-\hat{J}^2\right)^2\right] &\leq 4\gamma^{2T_J}R_{\max}^4+\frac{4R_{\max}^4}{L}+\frac{3R_{\max}^4+4R_{\max}^4}{L^2}+\frac{2R_{\max}^4+3R_{\max}^4}{L^3} \\
&= 4\gamma^{2T_J}R_{\max}^4+\left(\frac{4}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)R_{\max}^4.
\end{aligned}
$$

□

A.3.3 Critic's Analysis

Theorem A.3.13 (Critic's Bound)**.** *Suppose Assumptions 3 to 7 hold, and suppose we are given a policy π_θ (with normalized expected return J) and risk parameter λ. Suppose that a Monte-Carlo estimate $\hat{J}$ is obtained for π_θ using Algorithm 5, and then plugged into the Algorithm 6 which is run for T_c steps. Then, for $M \geq \left(\frac{2}{\lambda_A}+2\beta\right)\frac{192C_A^2[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A}$ and $\beta \leq \min\left\{\frac{\lambda_A}{8C_A^2}, \frac{4}{\lambda_A}\right\}$, we have that*

$$
\begin{aligned}
\mathbb{E}\left[\left\|\omega_{T_c}^{\hat{J}}-\omega_J^*\right\|_2^2\right] &\leq 4\|\omega_0-\omega_J^*\|_2^2\left(1-\frac{\lambda_A}{8}\beta\right)^{T_c} \\
&\quad+\left(\frac{2}{\lambda_A}+2\beta\right)\frac{384(C_A^2C_\omega^2+C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\
&\quad+\frac{2}{\bar{\sigma}^2}\left[1+2\left(1-\frac{\lambda_A}{8}\beta\right)^{T_c}\right]\xi_J,
\end{aligned}
$$

where $\omega_{T_c}^{\hat{J}}$ is the parameter vector obtained after T_c iterations of the algorithm while using $\hat{J}$ to perform the reward transformation, $\xi_J := 2\lambda^2 R_{\max}^4\left(8\gamma^{2T_J}+\frac{8}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)$, $\bar{\sigma}$ is the smallest singular value of the matrix A, and the expectation is over both the Monte-Carlo estimation of $\hat{J}$ and the TD algorithm.

Proof. We begin by adding and subtracting $\omega^*_{\hat{J}}$, which is the TD fixed point when using $\hat{J}$. Note that, at this point, $\omega^*_{\hat{J}}$ is a random variable due to its dependence on $\hat{J}$.

$$\begin{aligned}\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_J\right\|_2^2\right] &= \mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}} + \omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right] \\ &\leq 2\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}}\right\|_2^2\right] + 2\mathbb{E}\left[\left\|\omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right]. \end{aligned} \tag{A.4}$$

where the inequality follows from Lemma A.3.6.ii. Focusing on the first term, we have

$$\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}}\right\|_2^2\right] = \mathbb{E}\left[\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}}\right\|_2^2 \middle| \hat{J}\right]\right]. \tag{A.5}$$

For the inner expectation, as remarked before, we can apply the risk-neutral bound from theorem 4 in (Xu et al., 2020b). Namely for $M \geq \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{192C_A^2[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A}$ and $\beta \leq \min\left\{\frac{\lambda_A}{8C_A^2}, \frac{4}{\lambda_A}\right\}$, we have

$$\begin{aligned}&\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}}\right\|_2^2 \middle| \hat{J}\right] \\ &\leq \left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\left\|\omega_0 - \omega^*_{\hat{J}}\right\|_2^2 + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{192(C_A^2C_\omega^2 + C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M}.\end{aligned}$$

Note that $\|\omega_0 - \omega^*_{\hat{J}}\|_2^2$ is only part that depends on $\hat{J}$ in the previous bound. Plugging back in (A.5), we get that

$$\begin{aligned}&\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_{\hat{J}}\right\|_2^2\right] \\ &\leq \left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\mathbb{E}\left[\left\|\omega_0 - \omega^*_{\hat{J}}\right\|_2^2\right] + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{192(C_A^2C_\omega^2 + C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ &\leq \left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\mathbb{E}\left[\left\|\omega_0 - \omega^*_J + \omega^*_J - \omega^*_{\hat{J}}\right\|_2^2\right] + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{192(C_A^2C_\omega^2 + C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ &\leq 2\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\|\omega_0 - \omega^*_J\|_2^2 + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{192(C_A^2C_\omega^2 + C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ &\quad + 2\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\mathbb{E}\left[\left\|\omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right],\end{aligned}$$

where the last inequality again follows from Lemma A.3.6.ii. Plugging back in (A.4), we get that

$$\begin{aligned}&\mathbb{E}\left[\left\|\omega^{\hat{J}}_{T_c} - \omega^*_J\right\|_2^2\right] \\ &\leq 4\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\|\omega_0 - \omega^*_J\|_2^2 + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{384(C_A^2C_\omega^2 + C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ &\quad + \left[2 + 4\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}\right]\mathbb{E}\left[\left\|\omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right].\end{aligned} \tag{A.6}$$

Thus, we only need to bound $\mathbb{E}\left[\left\|\omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right]$. We proceed as follows:

$$\begin{aligned}
\mathbb{E}\left[\left\|\omega^*_{\hat{J}} - \omega^*_J\right\|_2^2\right] &= \mathbb{E}\left[\left\|A^{-1}b(J) - A^{-1}b(\hat{J})\right\|_2^2\right] \\
&= \mathbb{E}\left[\left\|A^{-1}\Big(b(J) - b(\hat{J})\Big)\right\|_2^2\right] \\
&\leq \frac{1}{\bar{\sigma}^2}\mathbb{E}\left[\left\|b(J) - b(\hat{J})\right\|_2^2\right], \qquad \text{(A.7)}
\end{aligned}$$

where $\bar{\sigma}$ is the smallest singular value of A, and the last inequality holds since, as demonstrated before, for an $m \times n$ matrix X and a vector $y \in \mathbb{R}^n$, $\|Xy\|_2^2 \leq \|X\|_2^2 \|y\|_2^2$, where $\|X\|_2$ is the spectral norm of X. Furthermore, we used that, by Lemma A.3.1, $\|A^{-1}\|_2 = \frac{1}{\bar{\sigma}}$. Moving on, recall that μ_θ is the stationary distribution of the MDP when using policy π_θ. We then have that

$$\begin{aligned}
&\mathbb{E}\left[\left\|b(J) - b(\hat{J})\right\|\right]_2^2 \\
&= \mathbb{E}\left[\left\|\mathbb{E}_{\mu_\theta}\left[\phi(s_t)R^\lambda(s_t,a_t,J)\right] - \mathbb{E}_{\mu_\theta}\left[\phi(s_t)R^\lambda(s_t,a_t,\hat{J})\right]\right\|_2^2\right] \\
&= \mathbb{E}\left[\left\|\mathbb{E}_{\mu_\theta}\left[\phi(s_t)\Big(R^\lambda(s_t,a_t,J) - R^\lambda(s_t,a_t,\hat{J})\Big)\right]\right\|_2^2\right] \\
&= \mathbb{E}\left[\left\|\mathbb{E}_{\mu_\theta}\left[\phi(s_t)\Big(2\lambda R(s_t,a_t)\Big(J - \hat{J}\Big) + \lambda\Big(\hat{J}^2 - J^2\Big)\Big)\right]\right\|_2^2\right] \\
&= \mathbb{E}\left[\left\|2\lambda\Big(J - \hat{J}\Big)\mathbb{E}_{\mu_\theta}[\phi(s_t)R(s_t,a_t)] + \lambda\Big(\hat{J}^2 - J^2\Big)\mathbb{E}_{\mu_\theta}[\phi(s_t)]\right\|_2^2\right] \\
&\leq \mathbb{E}\left[2\left\|2\lambda\Big(J - \hat{J}\Big)\mathbb{E}_{\mu_\theta}[\phi(s_t)R(s_t,a_t)]\right\|_2^2 + 2\left\|\lambda\Big(\hat{J}^2 - J^2\Big)\mathbb{E}_{\mu_\theta}[\phi(s_t)]\right\|_2^2\right] \\
&= 8\lambda^2\mathbb{E}\left[\Big(J - \hat{J}\Big)^2\|\mathbb{E}_{\mu_\theta}[\phi(s_t)R(s_t,a_t)]\|_2^2\right] + 2\lambda^2\mathbb{E}\left[\Big(\hat{J}^2 - J^2\Big)^2\|\mathbb{E}_{\mu_\theta}[\phi(s_t)]\|_2^2\right] \\
&\leq 8\lambda^2 R_{\max}^2\mathbb{E}\left[\Big(J - \hat{J}\Big)^2\right] + 2\lambda^2\mathbb{E}\left[\Big(\hat{J}^2 - J^2\Big)^2\right], \qquad \text{(A.8)}
\end{aligned}$$

where the first inequality follows from Lemma A.3.6, and the last inequality follows (keeping in mind Assumptions 3.i, 4, and 5) since

$$\|\mathbb{E}_{\mu_\theta}[\phi(s_t)R(s_t,a_t)]\|_2^2 \leq \mathbb{E}_{\mu_\theta}\left[\|\phi(s_t)R(s_t,a_t)\|_2^2\right] \leq R_{\max}^2,$$

and

$$\|\mathbb{E}_{\mu_\theta}[\phi(s_t)]\|_2^2 \leq \mathbb{E}_{\mu_\theta}\left[\|\phi(s_t)\|_2^2\right] \leq 1.$$

Now, we can plug the results of Propositions A.3.11 and A.3.12 in inequality (A.8) to

get

$$\begin{aligned}
\mathbb{E}\Big[\Big\|b(J)-b(\hat{J})\Big\|\Big]_2^2 &\leq 8\lambda^2 R_{\max}^2 \mathbb{E}\Big[\Big(J-\hat{J}\Big)^2\Big] + 2\lambda^2 \mathbb{E}\Big[\Big(\hat{J}^2-J^2\Big)^2\Big] \\
&\leq 8\lambda^2 R_{\max}^2\left(\gamma^{2T_J} R_{\max}^2 + \frac{R_{\max}^2}{L}\right) \\
&\quad + 2\lambda^2\left(4\gamma^{2T_J} R_{\max}^4 + \left(\frac{4}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)R_{\max}^4\right) \\
&= 8\lambda^2\gamma^{2T_J} R_{\max}^4 + \frac{8\lambda^2}{L} R_{\max}^4 \\
&\quad + 8\lambda^2\gamma^{2T_J} R_{\max}^4 + 2\lambda^2\left(\frac{4}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)R_{\max}^4 \\
&= 16\lambda^2\gamma^{2T_J} R_{\max}^4 + 2\lambda^2\left(\frac{8}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)R_{\max}^4.
\end{aligned}$$

Plugging back in (A.7), we get

$$\begin{aligned}
\mathbb{E}\Big[\big\|\omega_{\hat{J}}^* - \omega_J^*\big\|_2^2\Big] &\leq \frac{2\lambda^2}{\bar{\sigma}^2}\left(8\gamma^{2T_J} R_{\max}^4 + \left(\frac{8}{L}+\frac{7}{L^2}+\frac{5}{L^3}\right)R_{\max}^4\right) \\
&= \frac{\xi_J}{\bar{\sigma}^2},
\end{aligned}$$

where $\xi_J := 2\lambda^2 R_{\max}^4\big(8\gamma^{2T_J} + \frac{8}{L} + \frac{7}{L^2} + \frac{5}{L^3}\big)$. We can now plug back the last result into (A.6) to get

$$\begin{aligned}
\mathbb{E}\left[\left\|\omega_{T_c}^{\hat{J}} - \omega_J^*\right\|_2^2\right] &\leq 4\|\omega_0-\omega_J^*\|_2^2\left(1-\frac{\lambda_A}{8}\beta\right)^{T_c} \\
&\quad + \left(\frac{2}{\lambda_A}+2\beta\right)\frac{384(C_A^2C_\omega^2+C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\
&\quad + \frac{2}{\bar{\sigma}^2}\left[1+2\left(1-\frac{\lambda_A}{8}\beta\right)^{T_c}\right]\xi_J.
\end{aligned}$$

□

Corollary A.3.14 (Critic's Complexity)**.** *Suppose we are again in the same setting of Theorem A.3.13, and suppose the assumptions mentioned therein hold. Then, for a sufficiently small $\epsilon > 0$, if $\beta \leq \min\Big\{\frac{\lambda_A}{8C_A^2}, \frac{4}{\lambda_A}\Big\}$, $T_J \geq \frac{\log\left(\frac{192\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}\right)}{2(1-\gamma)}$, and $L \geq \frac{576\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}$, $T_c \geq \frac{8\log\left(\frac{24}{\epsilon}\left\|\omega_0-\omega_J^*\right\|_2^2\right)}{\lambda_A\beta}$, $M \geq \left(\frac{2}{\lambda_A}+2\beta\right)\frac{2304(C_A^2C_\omega^2+C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A\epsilon}$, then*

$$\mathbb{E}\left[\left\|\omega_{T_c}^{\hat{J}} - \omega_J^*\right\|_2^2\right] \leq \epsilon,$$

and the total sample complexity is

$$T_c M + LT_J = \mathcal{O}\big(\epsilon^{-1}\log\big(\epsilon^{-1}\big)\big).$$

Proof. By expanding and rearranging the bound in Theorem A.3.13, we have that

$$\begin{aligned}\mathbb{E}\left[\left\|\omega_{T_c}^{\hat{J}} - \omega_J^*\right\|_2^2\right] \leq\ & 4\|\omega_0 - \omega_J^*\|_2^2\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c} \\ & + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{384(C_A^2C_\omega^2 + C_b^2)[1 + (\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ & + \frac{32\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\gamma^{2T_J} \\ & + \frac{4\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\left(\frac{8}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right) \\ & + \frac{64\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\gamma^{2T_J}\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c} \\ & + \frac{8\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\left(\frac{8}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right)\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c}.\end{aligned}$$

Note that $\left(1 - \frac{\lambda_A}{8}\beta\right)^{T_c} \leq e^{-\frac{\lambda_A}{8}\beta T_c}$. This holds since $(1-x) \leq e^{-x}$, and if $x \leq 1$, then $(1-x)^r \leq e^{-rx}$ for $r \geq 0$. The claim then follows since $\beta < \frac{8}{\lambda_A}$ and $T_c \geq 0$. By a similar argument, $\gamma^{2T_J} = (1-(1-\gamma))^{2T_J} \leq e^{-2(1-\gamma)T_J}$. Plugging back these bounds, we get

$$\begin{aligned}\mathbb{E}\left[\left\|\omega_{T_c}^{\hat{J}} - \omega_J^*\right\|_2^2\right] \leq\ & 4\|\omega_0 - \omega_J^*\|_2^2\, e^{-\frac{\lambda_A}{8}\beta T_c} \\ & + \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{384(C_A^2C_\omega^2 + C_b^2)[1 + (\kappa-1)\rho]}{(1-\rho)\lambda_A M} \\ & + \frac{32\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\, e^{-2(1-\gamma)T_J} \\ & + \frac{4\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\left(\frac{8}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right) \\ & + \frac{64\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\, e^{-2(1-\gamma)T_J}\, e^{-\frac{\lambda_A}{8}\beta T_c} \\ & + \frac{8\lambda^2 R_{\max}^4}{\bar{\sigma}^2}\left(\frac{8}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right) e^{-\frac{\lambda_A}{8}\beta T_c}.\end{aligned}$$

To bound the whole expression by ϵ, we can bound each of the six terms by $\frac{\epsilon}{6}$. This can be achieved for each term if

Term 1

$$T_c \geq \frac{8\log\left(\frac{24}{\epsilon}\|\omega_0 - \omega_J^*\|_2^2\right)}{\lambda_A\beta}$$

Term 2

$$M \geq \left(\frac{2}{\lambda_A} + 2\beta\right)\frac{2304(C_A^2C_\omega^2 + C_b^2)[1 + (\kappa-1)\rho]}{(1-\rho)\lambda_A\epsilon}$$

Term 3

$$T_J \geq \frac{\log\left(\frac{192\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}\right)}{2(1-\gamma)}$$

Term 4

$$L \geq \max\left\{\frac{576\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}, \sqrt{\frac{504\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}}, \sqrt[3]{\frac{360\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}}\right\}$$

Term 5

$$T_c \geq \frac{8\log\left(\frac{\sqrt{6}}{\sqrt{\epsilon}}\right)}{\lambda_A\beta},\ T_J \geq \frac{\log\left(\frac{64\sqrt{6}\lambda^2 R_{\max}^4}{\sqrt{\epsilon}\bar{\sigma}^2}\right)}{2(1-\gamma)}$$

Term 6

$$T_c \geq \frac{8log\left(\frac{\sqrt{6}}{\sqrt{\epsilon}}\right)}{\lambda_A\beta}, L \geq \max\left\{\frac{192\sqrt{6}\lambda^2 R_{\max}^4}{\sqrt{\epsilon}\bar{\sigma}^2}, \sqrt{\frac{168\sqrt{6}\lambda^2 R_{\max}^4}{\sqrt{\epsilon}\bar{\sigma}^2}}, \sqrt[3]{\frac{120\sqrt{6}\lambda^2 R_{\max}^4}{\sqrt{\epsilon}\bar{\sigma}^2}}\right\}$$

Note that there are multiple conditions on some parameters. However, if ϵ is sufficiently small, it is enough that $T_c \geq \frac{8\log\left(\frac{24}{\epsilon}\left\|\omega_0-\omega_J^*\right\|_2^2\right)}{\lambda_A\beta}$, $M \geq \left(\frac{2}{\lambda_A}+2\beta\right)\frac{2304(C_A^2C_\omega^2+C_b^2)[1+(\kappa-1)\rho]}{(1-\rho)\lambda_A\epsilon}$, $T_J \geq \frac{\log\left(\frac{192\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}\right)}{2(1-\gamma)}$, and $L \geq \frac{576\lambda^2 R_{\max}^4}{\epsilon\bar{\sigma}^2}$. Thus, the sample complexity is given by

$$T_cM + LT_J = \mathcal{O}\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right) + \mathcal{O}\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right) = \mathcal{O}\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right).$$

□

A.3.4 Actor's Analysis

We first define the following quantities, which will help us in the analysis:

- the TD-error[3] $\delta_\omega(s,a,s') = R^\lambda(s,a,J) + \gamma\phi(s')^\top\omega - \phi(s)^\top\omega$ which employs the exact expected return J;
- the *approximated* TD-error $\hat{\delta}_\omega(s,a,s') = R^\lambda(s,a,\hat{J}) + \gamma\phi(s')^\top\omega - \phi(s)^\top\omega$ which employs, instead, the the Monte-Carlo current estimate of the expected return $\hat{J}$;
- $v_t(\omega,\theta) = \frac{1}{B}\sum_{i=0}^{B-1}\delta_\omega(s_{t,i},a_{t,i},s'_{t,i+1})\psi_{\theta_t}(s_{t,i},a_{t,i})$, which would have been the estimated gradient at time t (using a critic with parameters ω) if we had access to the true J_θ;
- $\hat{v}_t(\omega,\theta) = \frac{1}{B}\sum_{i=0}^{B-1}\hat{\delta}_\omega(s_{t,i},a_{t,i},s'_{t,i+1})\psi_{\theta_t}(s_{t,i},a_{t,i})$, which is the estimated gradient at time t (using a critic with parameters ω) based on $\hat{J}$;
- $A_\omega(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[\delta_\omega(s,a,s')]$, which is the expected value of the TD-error δ_ω at a given state-action pair when the next state is sampled from the transition kernel of the original MDP;
- $g(\omega,\theta) = \mathbb{E}_{\substack{s\sim d_{\mu_0,\pi_\theta}(\cdot)\\ a\sim\pi_\theta(\cdot|s)}}[A_\omega(s,a)\psi_\theta(s,a)]$, which is the expectation of the estimated gradient when using a critic with parameter vector ω.

Next, we prove two propositions, which will be combined to bound the expectation on the gradient norm.

Proposition A.3.15. *Suppose Assumptions 3 to 7 hold, then:*

$$\left(\frac{\alpha}{2} - 2L_\eta\alpha^2\right)\|\nabla\eta(\theta_t)\|_2^2 \leq \eta(\theta_{t+1}) - \eta(\theta_t) + \left(\frac{\alpha}{2} + 2L_\eta\alpha^2\right)\|\hat{v}_t(\omega_t,\theta_t) - \nabla\eta(\theta_t)\|_2^2.$$

Proof. By applying the Mean-Value Theorem, for some $0 \leq \Delta \leq 1$ there is some $\tilde{\theta} = \Delta\theta_t + (1-\Delta)\theta_{t+1}$ such that:

$$\begin{aligned}\eta(\theta_{t+1}) &= \eta(\theta_t) + (\theta_{t+1}-\theta_t)^\top\nabla\eta(\tilde{\theta}) = \eta(\theta_t) + (\theta_{t+1}-\theta_t)^\top\nabla\eta(\tilde{\theta}) \pm (\theta_{t+1}-\theta_t)^\top\nabla\eta(\theta_t)\\ &= \eta(\theta_t) + (\theta_{t+1}-\theta_t)^\top\Big(\nabla\eta(\tilde{\theta}) - \nabla\eta(\theta_t)\Big) + (\theta_{t+1}-\theta_t)^\top\nabla\eta(\theta_t).\end{aligned}$$

By using Cauchy-Schwarz we also have:

$$\begin{aligned}(\theta_{t+1}-\theta_t)^\top\Big(\nabla\eta(\tilde{\theta}) - \nabla\eta(\theta_t)\Big) &\geq -\|\theta_{t+1}-\theta_t\|_2\|\nabla\eta(\tilde{\theta}) - \nabla\eta(\theta_t)\|_2\\ &\geq -L_\eta\|\theta_{t+1}-\theta_t\|_2\|\tilde{\theta}-\theta_t\|_2\\ &\geq -L_\eta\|\theta_{t+1}-\theta_t\|_2^2\end{aligned}$$

where we also used that the gradient of η is Lipschitz (Lemma A.3.10).

[3]Note that the $\delta_\omega(s,a,s')$ and $\hat{\delta}_\omega(s,a,s')$ do depend on the current policy since they depend on its expected return, or an estimate of it. However, we do not explicitly express this dependence as to not burden the notation since it is usually clear from the context.

We exploit this relationship in the previous equation, together with the definition of the policy parameters update:

$$\begin{aligned}
\eta(\theta_{t+1}) &\geq \eta(\theta_t) - L_\eta\|\theta_{t+1} - \theta_t\|_2^2 + (\theta_{t+1} - \theta_t)^\top \nabla\eta(\theta_t) \\
&= \eta(\theta_t) - \alpha^2 L_\eta\|\hat{v}_t(\omega_t, \theta_t)\|_2^2 + \alpha\hat{v}_t(\omega_t, \theta_t)^\top \nabla\eta(\theta_t) \\
&= \eta(\theta_t) - \alpha^2 L_\eta\|\hat{v}_t(\omega_t, \theta_t) \pm \nabla\eta(\theta_t)\|_2^2 + \alpha\langle\hat{v}_t(\omega_t, \theta_t) \pm \nabla\eta(\theta_t), \nabla\eta(\theta_t)\rangle \\
&\overset{(1)}{\geq} \eta(\theta_t) - 2\alpha^2 L_\eta\|\nabla\eta(\theta_t)\|_2^2 - 2\alpha^2 L_\eta\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2 + \\
&\quad + \alpha\|\nabla\eta(\theta_t)\|_2^2 + \alpha\langle\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t), \nabla\eta(\theta_t)\rangle \\
&\overset{(2)}{\geq} \eta(\theta_t) - 2\alpha^2 L_\eta\|\nabla\eta(\theta_t)\|_2^2 - 2\alpha^2 L_\eta\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2 + \\
&\quad + \alpha\|\nabla\eta(\theta_t)\|_2^2 - \frac{\alpha}{2}\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2 - \frac{\alpha}{2}\|\nabla\eta(\theta_t)\|_2^2,
\end{aligned}$$

where in the last two steps we used, respectively, Lemma A.3.6.ii and Lemma A.3.6.i in (1) and (2). By re-ordering terms we obtain the desired result. □

The last term in the bound of the last proposition represents how far the estimated gradient is from the true one. Mirroring (Xu et al., 2020b), the next proposition bounds the expected value of this quantity.

Proposition A.3.16. *Suppose Assumptions 3 to 7 hold, and let $\mathcal{F}_t$ be the filtration on the samples up to iteration t:*

$$\begin{aligned}
\mathbb{E}\left[\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2|\mathcal{F}_t\right] \leq &\frac{24(R_{\lambda,\max} + 2C_\omega)^2[1 + (k-1)\rho]}{B(1-\rho)} \\
&+ 48\lambda^2 R_{\max}^2\mathbb{E}\left[|J - \hat{J}|^2|\mathcal{F}_t\right] + 12\lambda^2\mathbb{E}\left[|\hat{J}^2 - J^2|^2|\mathcal{F}_t\right] \\
&+ 24\|\omega_{J_t}^* - \omega_t\|_2^2 + 12\,\xi_{appr},
\end{aligned}$$

where J_t is short for $J_{\pi_{\theta_t}}$, and $\omega_{J_t}^*$ is the TD fixed point for the transformed value function of policy π_{θ_t}.

Proof. Consider $\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2$, we can decompose it in the following way (followed by an application of Lemma A.3.6.ii):

$$\begin{aligned}
&\|\hat{v}_t(\omega_t, \theta_t) - \nabla\eta(\theta_t)\|_2^2 \\
&= \left\|\hat{v}_t(\omega_t, \theta_t) \pm v_t(\omega_{J_t}^*, \theta_t) \pm g(\omega_{J_t}^*, \theta_t) - \nabla\eta(\theta_t)\right\|_2^2 \\
&\leq 3\underbrace{\left\|\hat{v}_t(\omega_t, \theta_t) - v_t(\omega_{J_t}^*, \theta_t)\right\|_2^2}_{(a)} + 3\left\|v_t(\omega_{J_t}^*, \theta_t) - g(\omega_{J_t}^*, \theta_t)\right\|_2^2 + 3\underbrace{\left\|g(\omega_{J_t}^*, \theta_t) - \nabla\eta(\theta_t)\right\|_2^2}_{(b)}.
\end{aligned} \tag{A.9}$$

We now focus on (a):

$$
\begin{aligned}
&\left\|\hat{v}_t(\omega_t,\theta_t) - v_t(\omega^*_{J_t},\theta_t)\right\|_2^2 \\
=&\left\|\frac{1}{B}\sum_{i=0}^{B-1}\psi_{\theta_t}(s_{t,i},a_{t,i})\left[\hat{\delta}_{\omega_t}(s_{t,i},a_{t,i},s'_{t,i+1}) - \delta_{\omega^*_{J_t}}(s_{t,i},a_{t,i},s'_{t,i+1})\right]\right\|_2^2 \\
\leq&\frac{1}{B}\sum_{i=0}^{B-1}\underbrace{\|\psi_{\theta_t}(s_{t,i},a_{t,i})\|_2^2}_{\leq C_\psi=1}\left|\hat{\delta}_{\omega_t}(s_{t,i},a_{t,i},s'_{t,i+1}) - \delta_{\omega^*_{J_t}}(s_{t,i},a_{t,i},s'_{t,i+1})\right|^2 \\
\leq&\frac{1}{B}\sum_{i=0}^{B-1}\Big|R^\lambda(s_{t,i},a_{t,i},\hat{J}) - R^\lambda(s_{t,i},a_{t,i},J) + \gamma\left(\phi(s'_{t,i+1})^\top\omega_t - \phi(s'_{t,i+1})^\top\omega^*_{J_t}\right) + \\
&\qquad + \left(\phi(s_{t,i})^\top\omega^*_{J_t} - \phi(s_{t,i})^\top\omega_t\right)\Big|^2 \\
\overset{(1)}{\leq}&\frac{1}{B}\sum_{i=0}^{B-1}2\left|R^\lambda(s_{t,i},a_{t,i},\hat{J}) - R^\lambda(s_{t,i},a_{t,i},J)\right|^2 + 2\left|(\gamma\phi(s'_{t,i+1}) - \phi(s_{t,i}))^\top(\omega_t - \omega^*_{J_t})\right|^2 \\
\overset{(2)}{\leq}&\frac{1}{B}\sum_{i=0}^{B-1}2\left|R^\lambda(s_{t,i},a_{t,i},\hat{J}) - R^\lambda(s_{t,i},a_{t,i},J)\right|^2 + 8\left\|\omega^*_{J_t} - \omega_t\right\|_2^2 \\
\overset{(3)}{=}&\frac{1}{B}\sum_{i=0}^{B-1}2\lambda^2\left|2R(s_{t,i},a_{t,i})(J-\hat{J}) + \hat{J}^2 - J^2\right|^2 + 8\left\|\omega^*_{J_t} - \omega_t\right\|_2^2 \\
\overset{(4)}{\leq}&16\lambda^2R_{\max}^2|J-\hat{J}|^2 + 4\lambda^2|\hat{J}^2 - J^2|^2 + 8\|\omega^*_{J_t} - \omega_t\|_2^2.
\end{aligned}
$$

where (1) is an application of Lemma A.3.6.ii, (2) is due to Cauchy-Schwarz, Lemma A.3.6.ii, and Assumption 5.ii, (3) to definition of R^λ, and in (4) Lemma A.3.6.ii is applied again.

We can then exploit results from Theorem 5 in Xu et al. (2020b), to bound (b) as:

$$\left\|g(\omega^*_{J_t},\theta_t) - \nabla\eta(\theta_t)\right\|_2^2 \leq 4\xi_{appr}.$$

Substituting back to inequality (A.9) and taking the expectation w.r.t. the filtration $\mathcal{F}_t$, we get:

$$
\begin{aligned}
\mathbb{E}\left[\|\hat{v}_t(\omega_t,\theta_t) - \nabla\eta(\theta_t)\|_2^2|\mathcal{F}_t\right] \leq\ & 3\mathbb{E}\left[\|v_t(\omega^*,\theta_t) - g(\omega^*,\theta_t)\|_2^2|\mathcal{F}_t\right] \\
&+ 48\lambda^2R_{max}^2\mathbb{E}\left[|J-\hat{J}|^2|\mathcal{F}_t\right] + 12\lambda^2\mathbb{E}\left[|\hat{J}^2 - J^2|^2|\mathcal{F}_t\right] \\
&+ 24\|\omega^*_{J_t} - \omega_t\|_2^2 + 12\,\xi_{appr}.
\end{aligned}
$$

To bound the conditional expectation on the RHS, we follow again the proof in Xu et al. (2020b) to have:

$$\mathbb{E}\left[\|v_t(\omega^*,\theta_t) - g(\omega^*,\theta_t)\|_2^2|\mathcal{F}_t\right] \leq \frac{8(R_{\lambda,\max} + 2C_\omega)^2(1+(k-1)\rho)}{B(1-\rho)}. \tag{A.10}$$

□

Theorem A.3.17 (Actor's Bound)**.** *Suppose Assumptions 3 to 7 hold and let* $\alpha = \frac{1}{8L_\eta}$*, then we have:*

$$\mathbb{E}\left[\|\nabla\eta(\theta_{\hat{T}})\|_2^2\right] \leq \frac{64L_\eta R_{\lambda,\max}}{T} + \xi_{distr} + 18\xi_J + 72\frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\omega_{J_t}^* - \omega_t\|_2^2]}{T} + 36\xi_{appr},$$

where

$$\xi_{distr} \coloneqq \frac{72(R_{\lambda,\max} + 2C_\omega)^2(1 + (k-1)\rho)}{B(1-\rho)}.$$

Proof. Taking the conditioned expectation on the result of Proposition A.3.15 and plugging what we obtained with Proposition A.3.16 we obtain the following:

$$\begin{aligned}
&\left(\frac{\alpha}{2} - 2L_\eta\alpha^2\right)\mathbb{E}\left[\|\nabla\eta(\theta_t)\|_2^2|\mathcal{F}_t\right] \\
&\leq \mathbb{E}\left[\eta(\theta_{t+1})|\,\mathcal{F}_t\right]] - \eta(\theta_t) + \left(\frac{\alpha}{2} + 2L_\eta\alpha^2\right)\left[\frac{24(R_{\lambda,\max} + 2C_\omega)^2[1 + (k-1)\rho]}{B(1-\rho)} + \right. \\
&\quad \left. + 48\lambda^2 R_{\max}^2\mathbb{E}\left[|J - \hat{J}|^2|\mathcal{F}_t\right] + 12\lambda^2\mathbb{E}\left[|\hat{J}^2 - J^2|^2|\mathcal{F}_t\right] + 24\|\omega_{J_t}^* - \omega_t\|_2^2 + 12\,\xi_{appr}\right] \\
&\leq \mathbb{E}\left[\eta(\theta_{t+1})|\,\mathcal{F}_t\right]] - \eta(\theta_t) + \left(\frac{\alpha}{2} + 2L_\eta\alpha^2\right)\left[\frac{24(R_{\lambda,\max} + 2C_\omega)^2(1 + (k-1)\rho)}{B(1-\rho)} + \right. \\
&\quad + 48\lambda^2 R_{\max}^2\left(\gamma^{2T_J}R_{\max}^2 + \frac{R_{\max}^2}{L}\right) + 12\lambda^2\left(4\gamma^{2T_J}R_{\max}^4 + \left(\frac{4}{L} + \frac{7}{L^2} + \frac{5}{L^3}\right)R_{\max}^4\right) + \\
&\quad \left. + 24\|\omega_{J_t}^* - \omega_t\|_2^2 + 12\,\xi_{appr}\right],
\end{aligned}$$

We let $\alpha = \frac{1}{8L_\eta}$ and we multiply both sides by $32L_\eta$ to get:

$$\begin{aligned}
\mathbb{E}\left[\|\nabla\eta(\theta_t)\|_2^2|\mathcal{F}_t\right] \leq\; & 32L_\eta\left(\mathbb{E}\left[\eta(\theta_{t+1})|\mathcal{F}_t\right]] - \eta(\theta_t)\right) + \xi_{distr} + 18\xi_J \\
& + 72\|\omega_{J_t}^* - \omega_t\|_2^2 + 36\xi_{appr},
\end{aligned}$$

with

$$\xi_{distr} \coloneqq \frac{72(R_{\lambda,\max} + 2C_\omega)^2(1 + (k-1)\rho)}{B(1-\rho)},$$

which bounds the variance of the mini-batch estimate of the gradient if the critic was at the TD fixed point, while ξ_J, the error arising from the expected return estimation, has been already defined in Theorem A.3.13.

We take the expectation w.r.t. $\mathcal{F}_t$ to both sides to yield:

$$\begin{aligned}
\mathbb{E}\left[\|\nabla\eta(\theta_t)\|_2^2\right] \leq\; & 32L_\eta(\mathbb{E}\left[\eta(\theta_{t+1})\right] - \mathbb{E}\left[\eta(\theta_t)\right]) + \xi_{distr} + 18\xi_J \\
& + 72\mathbb{E}[\|\omega_{J_t}^* - \omega_t\|_2^2] + 36\xi_{appr}.
\end{aligned}$$

Taking the summation of the last result over $t = 0, \ldots, T-1$ and dividing both

sides by T gives:

$$\begin{aligned}\mathbb{E}\left[\|\nabla\eta(\theta_{\hat{T}})\|_2^2\right] &= \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla\eta(\theta_t)\|_2^2\right] \\ &\leq 32L_\eta\frac{\mathbb{E}\left[\eta(\theta_T)\right]-\eta(\theta_0)}{T}+\xi_{distr}+18\xi_J \\ &\quad+72\frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\omega^*_{J_t}-\omega_t\|_2^2]}{T}+36\xi_{appr} \\ &\leq \frac{64L_\eta R_{\lambda,\max}}{T}+\xi_{distr}+18\xi_J+72\frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\omega^*_{J_t}-\omega_t\|_2^2]}{T}+36\xi_{appr}.\end{aligned}$$

□

Corollary A.3.18 (Actor's Complexity)**.** *Suppose we are in the same setting of Theorem A.3.17, and assume that the parameters used in the critic are conditioned as to make* $\mathbb{E}\left[\left\|\omega_t-\omega^*_{J_t}\right\|_2^2\right]\leq\frac{\epsilon}{360}$ *for all* $t=0,\ldots,T-1$*. Then, if additionally*

- $T\geq\frac{320L_\eta(R_{\max}+4\lambda R_{\max}^2)}{\epsilon}$,
- $B\geq\frac{360((R_{\max}+4\lambda R_{\max}^2)+2C_\omega)^2(1+(k-1)\rho)}{(1-\rho)\epsilon}$,
- $T_J\geq\frac{\log\left(\frac{1440\lambda^2R_{max}^4}{\epsilon}\right)}{2(1-\gamma)}$,
- $L\geq\frac{3600\lambda^2R_{\max}^4}{\epsilon}$,

we have that

$$\mathbb{E}\left[\|\nabla\eta(\omega_{\hat{T}})\|_2^2\right]\leq\epsilon+\mathcal{O}(\xi_{appr}),$$

with the total sample complexity given by:

$$T((2-\gamma)B+MT_c+LT_J)=\mathcal{O}\left(\epsilon^{-2}\log\left(\epsilon^{-1}\right)\right).$$

Proof. In order to compute the different contributions to sample complexity, we will split the error bound obtained in Theorem A.3.17 in its components. We then bound the components in the following way:

- $\frac{64L_\eta(R_{\max}+4\lambda R_{\max}^2)}{T}\leq\epsilon_1$,
- $\frac{72(R_{\max}+4\lambda R_{\max}^2+2C_\omega)^2(1+(k-1)\rho)}{B(1-\rho)}\leq\epsilon_2$,
- $288\lambda^2R_{max}^4\gamma^{2T_J}\leq288\lambda^2R_{max}^4e^{-2(1-\gamma)T_J}\leq\epsilon_3$,
- $36\lambda^2R_{max}^4(\frac{8}{L}+\frac{7}{L^2}+\frac{5}{L^3})\overset{L>1}{\leq}36\lambda^2R_{max}^4(\frac{20}{L})\leq\epsilon_4$,
- $72\frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\omega^*_{J_t}-\omega_t\|_2^2]}{T}\leq\epsilon_5$,

where we have split $18\xi_J$ in two parts, and we have ignored the approximation error ξ_{appr}, which cannot be reduced with more samples. We set, then, each ϵ_i to $\frac{\epsilon}{5}$. Rearranging terms in each inequality, we obtain then, by the conditions on the parameters indicated in the statement[4], the desired error. In order to obtain it, the following sample complexity is needed:

$$\begin{aligned} T((2-\gamma)B + MT_c + LT_J) &= \mathcal{O}\left(\frac{1}{\epsilon}\left(\frac{1}{\epsilon} + \frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right) + \frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)\right) \\ &= \mathcal{O}\left(\epsilon^{-2}\log\left(\epsilon^{-1}\right)\right) \end{aligned}$$

where the $(2-\gamma)$ extra factor is due to the actor sampling process, which needs to sample twice at each restart, which can happen at each step with probability $1-\gamma$. □

[4] And the conditions adapted from Corollary A.3.14 needed to make $\mathbb{E}[\|\omega_t - \omega^*_{J_t}\|_2^2] \leq \frac{\epsilon}{360}$ (instead of just ϵ) for all $t = 0, \ldots, T-1$.

APPENDIX $\mathcal{B}$

Mean-Volatility and Multi-Objective Reinforcement Learning

In this appendix, we show some preliminary results about the connections between the Mean-Volatility trade-off and Multi-Objective Reinforcement Learning.

B.1 A Multi-Objective Perspective

In Chapters 5 and 6, we focused on the penalized version of the Mean-Volatility trade-off, however, as for the return variance (see Section 3.2.2), many other related optimality criteria can be defined. We now formally define the Pareto-optimality criterion for Mean-Volatility.

Definition B.1.1 (Mean-Volatility Pareto Front)**.** *We say that a policy π Pareto-dominates w.r.t. the Mean-Volatility trade-off another policy π', if its value is at least as high w.r.t. one of the two objectives, and strictly higher w.r.t. the other one:*

$$\pi \succ_{MV} \pi' \Leftrightarrow (\nu^2_\pi < \nu^2_{\pi'} \wedge J_\pi \geq J_{\pi'}) \vee (\nu^2_\pi \leq \nu^2_{\pi'} \wedge J_\pi > J_{\pi'}).$$

We, accordingly, define the Pareto front w.r.t. these criteria as the set of policies which are not Pareto-dominated:

$$PF_{MV}(\Pi^{SR}) \coloneqq \left\{\pi : \pi \in \Pi^{SR} \wedge \nexists(\pi' \in \Pi^{SR}), \pi' \succ_{MV} \pi\right\}.$$

It is possible to draw some connections between the Mean-Volatility problem and another multi-objective problem with involves the maximization of the expected return and the minimization of the reward second moment. Recalling the second moment value function $M(s)$, defined in Equation (6.2), we overload the notation by defining

the objective:

$$M_\pi \coloneqq \int_{\mathcal{S}} M_\pi(s) d_\mu^\pi(\mathrm{d}s). \tag{B.1}$$

Considering the trade-off between the risk-neutral expected return and the latter risk-measure, we define the Mean-Second-Moment Pareto front as follows.

Definition B.1.2 (Mean-Second-Moment Pareto Front)**.** *We say that a policy π Pareto-dominates w.r.t. the Mean-Second-Moment trade-off another policy π', if its value is at least as high w.r.t. one of the two objectives, and strictly higher w.r.t. the other one:*

$$\pi \succ_{MS} \pi' \Leftrightarrow (M_\pi < M_{\pi'} \wedge J_\pi \geq J_{\pi'}) \vee (M_\pi \leq M_{\pi'} \wedge J_\pi > J_{\pi'}).$$

We, accordingly, define the Pareto front w.r.t. these criteria as the set of policies which are not Pareto-dominated:

$$PF_{MS}(\Pi^{SR}) \coloneqq \left\{\pi : \pi \in \Pi^{SR} \wedge \nexists(\pi' \in \Pi^{SR}), \pi' \succ_{MS} \pi\right\}.$$

Differently from the Mean-Volatility case, this trade-off can be described by a MOMDP (See Section 2.6.1). The MOMDP $\mathcal{M}_{MS}$ inherits all its components from $\mathcal{M}$, a part from the reward vector that is constituted by the original reward and its square: $\boldsymbol{r} = [r, r^2]$. Policies which are Pareto-optimal for $\mathcal{M}_{MS}$ are indeed also contained in the pareto front $PF_{MS}(\bar{\Pi})$, while the converse is not necessary true, as it is the case for standard MDP risk-neutral objectives (see Section 2.2.3). It is possible then to extend the result of Proposition 2.6.7, establishing the equivalence between the $PF_{MS}(\Pi^{SR})$ and the convex hull of the $CCS(\Pi^{SD})$, to the MS Pareto front, if we allow for stochastic policies, contained in Π^{SR}.

We now need to assume rewards to be *non-negative*, in order to prove the next result. This assumption is justified by the *translation invariance* property of the reward-volatility, and by the linearity of the expected return. In practice, adding a positive constant to the reward, increases the expected return by exactly that amount and does not impact on the reward-volatility.

Proposition B.1.3. *Let the reward function R to be non-negative, then we have:*

$$PF_{MV} \subseteq PF_{MS}.$$

Proof. We prove the claim by contradiction. Consider the thesis as false, thus, it must exists a policy π in PF_{MV}, which is Pareto-dominated w.r.t. the Mean-Second-Moment trade-off:

$$\exists \pi, (\pi \in PF_{MV} \wedge \exists \bar{\pi} \succ_{MS} \pi).$$

We will now distinguish three cases: $J_{\bar{\pi}} < J_\pi$, $J_{\bar{\pi}} = J_\pi$, and $J_{\bar{\pi}} > J_\pi$. If $J_{\bar{\pi}} < J_\pi$, then $\bar{\pi}$ is dominated. Consider now $J_{\bar{\pi}} = J_\pi$, then, since $\pi \in PF_{MV}$, we need to have $\nu_{\bar{\pi}}^2 \geq \nu_\pi^2$, but this means that

$$M_{\bar{\pi}} = \nu_{\bar{\pi}}^2 + J_{\bar{\pi}}^2 \geq \nu_\pi^2 + J_\pi^2 = M_\pi,$$

thus, π cannot be dominated by $\bar{\pi}$ in PF_{MV}. Consider now $J_{\bar{\pi}} > J_\pi$, in this case we need $\nu_{\bar{\pi}}^2 > \nu_\pi^2$ to hold, given that $\pi \in PF_{MV}$. Since we have assumed rewards

to be non negative, we also have $J_{\bar{\pi}}^2 > J_{\pi}^2$. Therefore, using once more the variance decomposition we have:

$$M_{\bar{\pi}} = \nu_{\bar{\pi}}^2 + J_{\bar{\pi}}^2 > \nu_{\pi}^2 + J_{\pi}^2 = M_{\pi},$$

thus, π is not dominated by $\bar{\pi}$. Consequently, the thesis is proven, since its negation always brings to a contradiction. □

This proposition states that all the Pareto-optimal policies for the Mean-Volatility trade-off are also optimal for the Mean-Second-Moment one. However, solving the latter problem is easier than the former one, since it is a MOMDP. Unfortunately, the converse relationship is not true, hence, some policies in PF_{MS} can be dominated in PF_{MV}.

B.2 The Penalized Criterion as a Sequence of Mean-Second-Moment MDPs

In this section we analyse a recent work that focuses on the penalized criterion for Mean-Volatility, and we derive some original theoretical results based on their constructions.

B.2.1 Mean-Variance Policy Iteration

In (Zhang et al., 2021), the authors noticed that, applying the the Fenchel duality, the *penalized mean-volatility* objective can be reformulated as:

$$\begin{aligned}
\eta_{\lambda}^{\pi} &= J_{\pi} - \lambda \nu_{\pi}^2 \overset{(a)}{=} J_{\pi} - \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a)^2\right] + \lambda J_{\pi}^2 \\
&= J_{\pi} - \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a)^2\right] + \lambda \max_{y} 2J(\pi)y - y^2 \\
&= \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a)\right] - \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a)^2\right] + \lambda \max_{y} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[2R(s,a)\right] y - y^2,
\end{aligned}$$

where (a) derive from the well-known variance decomposition. They, hence, stated the follow equivalence between optimization problems:

$$\max_{\pi} \eta_{\pi}^{\lambda} = \max_{\pi, y} \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[r(s,a)\right] - \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[r(s,a)^2\right] + \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_{\mu}^{\pi}(\cdot) \\ a\sim\pi(\cdot|s)}} \left[2r(s,a)\right] y - \lambda y^2, \tag{B.2}$$

and optimized it using a *block-coordinate* approach, iteratively maximizing one of the two variables, as shown in Algorithm 8. Once y is fixed, the optimization over π

Algorithm 8 Mean-Variance Policy Iteration from (Zhang et al., 2021)

for $k = 1, \dots$ **do**
Step 1: $y_{k+1} \coloneqq (1-\gamma) J_{\pi_k}$
Step 2:

$$\pi_k \coloneqq \arg\max_{\pi} \sum_{s,a} d_\pi(s,a) \Big((1 + 2\lambda y_{k+1}) r(s,a) - \lambda r^2(s,a) \Big) - \lambda y_{k+1}^2$$

end for

becomes equivalent to:

$$\max_{\pi} \mathop{\mathbb{E}}_{\substack{s\sim d_\mu^\pi(\cdot) \\ a\sim\pi(\cdot|s)}} [R(s,a)] - \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_\mu^\pi(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a)^2\right] + \lambda \mathop{\mathbb{E}}_{\substack{s\sim d_\mu^\pi(\cdot) \\ a\sim\pi(\cdot|s)}} [2R(s,a)]\, y \tag{B.3}$$

$$= \mathop{\mathbb{E}}_{\substack{s\sim d_\mu^\pi(\cdot) \\ a\sim\pi(\cdot|s)}} \left[R(s,a) - \lambda R(s,a)^2 + 2\lambda R(s,a) y\right], \tag{B.4}$$

where we dropped the $-\lambda y^2$ term, since it is a constant, hence, it has no influence in the optimization process. This problem is equivalent to a modified MDP with a transformed reward.

$$R_\nu(s,a) = R(s,a) - \lambda R(s,a)^2 + 2\lambda R(s,a) y \tag{B.5}$$

On the other hand, maximizing over y consists in an evaluation step by construction: the Fenchel duality tells us that the best value for this variable is, indeed, the current expected return itself. This is, indeed, what happens also when optimizing the Mean-Variance objective under the ROSA framework (see Chapter 4). This means that the reward optimized in the inner problem is indeed the same policy-based reward that we obtained in Chapter 5. The MVPI framework, thanks to its block-coordinate approach, allows to drop the dependence on the policy by solving a sequence of MDPs. In particular, these MDPs are all linear scalarizations of the MOMDP $\mathcal{M}_{MS}$ defined above.

B.2.2 Markovian Deterministic Policies Are Sufficient for Optimality

Exploiting the same Fenchel duality used in MVPI, it is possible to derive a important result about the class of policies that are sufficient to solve the the penalized Mean-Volatility criterion.

Proposition B.2.1. *Given a risk-aversion coefficient $\lambda > 0$, and an MDP $\mathcal{M}$, consider a pair $(J_\star, \nu_\star^2)$, such that $J_\star - \lambda \nu_\star^2 = \eta_\star = \max_\pi \eta_\pi^\lambda$. A policy $\pi^\star$ attains $J_{\pi^\star} = J_\star$ and $\nu_{\pi^\star}^2 = \nu_\star^2$ if and only if it is also a J-optimal policy for the transfomed MDP $\widetilde{\mathcal{M}}$ with reward function $\widetilde{R}(s,a) \coloneqq (1 + 2\lambda J_\star) R(s,a) - \lambda R(s,a)^2$.*

Proof. We will start proving ($\Rightarrow$), i.e., a policy $\pi^\star$ attaining $J_{\pi^\star} = J_\star$ and $\nu_{\pi^\star}^2 = \nu_\star^2$ is also J-optimal for the aforementioned MDP.
We recall that, due to the variance decomposition (see (3.13)):

$$\eta_\lambda^\pi = J_\pi - \lambda \nu_\pi^2 = J_\pi - \lambda M_\pi + \lambda J_\pi^2,$$

hence, we will define also $M_\star = \nu_\star^2 + J_\star^2$. Since $\pi^\star$ is an η^λ-optimal policy by construction, we have:

$$\begin{aligned}\forall \pi, \quad & J_{\pi^\star} - \lambda M_{\pi^\star} + \lambda J_{\pi^\star}^2 \geq J_\pi - \lambda M_\pi + \lambda J_\pi^2 \overset{(*)}{=} J_\pi - \lambda M_\pi + \lambda \left(\max_y 2y J_\pi - y^2 \right) \\ & = \max_y (1 + 2\lambda y) J_\pi - \lambda M_\pi - \lambda y^2 \geq (1 + 2\lambda J_{\pi^\star}) J_\pi - \lambda M_\pi - \lambda J_{\pi^\star}^2,\end{aligned}$$

where in $(*)$ we used the Fenchel dual of the square function: $x^2 = \max_y yx - y^2$. Since we have:

$$J_{\pi^\star} - \lambda M_{\pi^\star} + \lambda J_{\pi^\star}^2 \pm \lambda J_{\pi^\star}^2 = (1 + 2\lambda J_{\pi^\star}) J_{\pi^\star} - \lambda M_{\pi^\star} - \lambda J_{\pi^\star}^2,$$

we can affirm:

$$\begin{aligned}\forall \pi, \quad (1 + 2\lambda J_{\pi^\star}) J_{\pi^\star} - \lambda M_{\pi^\star} - \lambda J_{\pi^\star}^2 \geq (1 + 2\lambda J_{\pi^\star}) J_\pi - \lambda M_\pi - \lambda J_{\pi^\star}^2 \\ (1 + 2\lambda J_{\pi^\star}) J_{\pi^\star} - \lambda M_{\pi^\star} \geq (1 + 2\lambda J_{\pi^\star}) J_\pi - \lambda M_\pi,\end{aligned}$$

which proves the first part.
We focus on the the second part ($\Leftarrow$), i.e., any optimal policy $\tilde{\pi}$ in the transformed MDP attains $J_{\tilde{\pi}} = J_\star$ and $\nu_{\tilde{\pi}}^2 = \nu_\star^2$. We will prove the claim by contradiction. Let's define:

$$\tilde{J}_\pi \coloneqq \mathbb{E}_\pi \left[\sum_t \gamma^t \tilde{R}(s_t, a_t) \right] = (1 + 2\lambda J_\star) J_\pi - \lambda M_\pi,$$

and assume that the thesis is false, hence, there exists $\tilde{\pi} \in \arg\max_\pi \tilde{J}_\pi$, such that either $J_{\tilde{\pi}} \neq J_\star$ or $\nu_{\tilde{\pi}}^2 \neq \nu_\star^2$. Since both $\tilde{\pi}$ and $\pi_\star$ are optimal w.r.t. $\tilde{J}$, we have:

$$(1 + 2\lambda J_\star) J_{\tilde{\pi}} - \lambda M_{\tilde{\pi}} = (1 + 2\lambda J_\star) J_\star - \lambda M_\star,$$

thus, we can write $M_{\tilde{\pi}}$ as:

$$M_{\tilde{\pi}} = \frac{(1 + 2\lambda J_\star)(J_{\tilde{\pi}} - J_\star) + \lambda M_\star}{\lambda}. \tag{B.6}$$

This allows us also to compute:

$$\begin{aligned}\eta_{\tilde{\pi}}^\lambda &= J_{\tilde{\pi}} - \lambda M_{\tilde{\pi}} + \lambda J_{\tilde{\pi}}^2 = J_{\tilde{\pi}} - (1 + 2\lambda J_\star)(J_{\tilde{\pi}} - J_\star) - \lambda M_\star + \lambda J_{\tilde{\pi}}^2 \\ &= -2\lambda J_\star J_{\tilde{\pi}} + J_\star + 2\lambda J_\star^2 - \lambda M_\star + \lambda J_{\tilde{\pi}}^2 = \eta_\star^\lambda + \lambda J_\star^2 - 2\lambda J_\star J_{\tilde{\pi}} + \lambda J_{\tilde{\pi}}^2 \\ &= \eta_\star^\lambda + \lambda (J_{\tilde{\pi}} - J_\star)^2.\end{aligned}$$

Since $\lambda > 0$, if $J_{\tilde{\pi}} \neq J_\star$, we have $\eta_{\tilde{\pi}}^\lambda > \eta_\star^\lambda$, which is not possible, since $\eta_\star^\lambda$ is optimal. However, taking $J_{\tilde{\pi}} = J_\star$, and substituting it in Equation (B.6) entails also $M_{\tilde{\pi}} = M_\star$, and so $\nu_{\tilde{\pi}}^2 = \nu_\star^2$, which brings to a contradiction. □

Corollary B.2.2. *Markovian stationary deterministic policies are sufficient to solve the penalized mean-volatility objective η^λ.*

Proof. Thanks to the result of Proposition B.2.1, we known that for any optimal combination $J_\star$ and $\nu_\star^2$ we can build an MDP with a transformed reward $\tilde{R}(s,a) \coloneqq (1 + 2\lambda J_\star) R(s,a) - \lambda R(s,a)^2$, such that any optimal policy for that MDP is also optimal for η_λ. Since for any MDP a Markovian stationary deterministic optimal policy exists, the same holds for η_λ. □

B.2.3 Considerations on the Optimal Policies Obtained with MVPI

Unfortunately, optimizing the penalized criterion with MVPI we cannot hope to reach a global optimal solution, even if we resort to planning for solving the inner problem. This is due to the following fact: in general the Mean-Volatility objective $J_\pi - \lambda M_\pi + \lambda J_\pi^2$ is *non-concave* w.r.t. the expected return J_π. Therefore, we can only expect MVPI to converge to a local optimum, as such as TRVO.

As noted by Zhang et al. (2021), TRVO can be though as an instance of MVPI where only a single optimization step is carried on at each inner iteration by means of TRPO. The exact version of the latter algorithm for the finite actions case has been shown in (Neu and Pike-Burke, 2020) to be equivalent to the MDP-E algorithm (Even-Dar et al., 2009) in a stationary reward case. The MDP-E algorithm has been developed for an adversarial setting called Online MDP, in which the reward is allowed to change (adversarially) at each iteration. We can then interpret TRVO as an instance of MDP-E applied to a particular setting in which rewards are chosen according to a precise schedule. Since this hypothesis is less strong than the adversarial one, we inherits its theoretical guarantees in terms of regret. We have that MDP-E enjoys the following regret bound w.r.t. the best single policy:

$$\max_{\bar{\pi}} \mathop{\mathbb{E}}_{\mu}[V_{\bar{\pi}}] - \mathop{\mathbb{E}}_{\mu}[V_{\text{MDP-E}}] \leq O\left(T^{-\frac{1}{2}}\right). \tag{B.7}$$

By choosing ϵ as the maximum error that we what to achieve, it is straightforward to translate this regret bound to a finite-sample complexity of $O(\epsilon^{-2})$ to obtain a policy which is ϵ-optimal w.r.t. some local optimum. While this is an interesting result, we remark that it is limited to the *exact* case, in which the transition model is known, together with the past rewards. Extending these kind of guarantees in the sample-based case is an interesting direction for future research.

Bibliography

A. Agarwal, N. Jiang, S. M. Kakade, and W. Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 2019.

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

M. Allais. Allais paradox. In *Utility and probability*, pages 3–9. Springer, 1990.

E. Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

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

J. Angelova. On moments of sample mean and variance. *International Journal of Pure and Applied Mathematics*, 79, 01 2012.

A. Antos, R. Munos, and C. Szepesvári. Fitted q-iteration in continuous action-space mdps. 2007.

A. Antos, C. Szepesvári, and R. Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical finance*, 9(3):203–228, 1999. Publisher: Wiley Online Library.

S. Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fund. math*, 3(1):133–181, 1922.

D. P. Baron. On the utility theoretic foundations of mean-variance analysis. *The Journal of Finance*, 32(5):1683–1697, 1977.

N. Bäuerle and J. Ott. Markov decision processes with average-value-at-risk criteria. *Mathematical Methods of Operations Research*, 74(3):361–379, 2011. Publisher: Springer.

N. Bäuerle and U. Rieder. *Markov decision processes with applications to finance*. Springer Science & Business Media, 2011.

N. Bäuerle and U. Rieder. More risk-sensitive Markov decision processes. *Mathematics of Operations Research*, 39(1):105–120, 2013. Publisher: INFORMS.

J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15, 2001.

A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM journal on Optimization*, 23(4):2037–2060, 2013. Publisher: SIAM.

M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

R. Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, and others. Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv preprint arXiv:1912.06680*, 2019.

D. Bernoulli. Exposition of a new theory on the measurement of risk. In *The Kelly capital growth investment criterion: Theory and practice*, pages 11–24. World Scientific, 2011.

D. Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

J. Bhandari, D. Russo, and R. Singal. A finite time analysis of temporal difference learning with linear function approximation. *Oper. Res.*, 69:950–973, 2018.

S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

C. Bishop. Bishop-pattern recognition and machine learning-springer 2006. *Antimicrob. Agents Chemother*, pages 03728–14, 2014.

L. Bisi, P. Liotet, L. Sabbioni, G. Reho, N. Montali, M. Restelli, and C. Corno. Foreign exchange trading: a risk-averse batch reinforcement learning approach. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8, 2020a.

L. Bisi, L. Sabbioni, E. Vittori, M. Papini, and M. Restelli. Risk-averse trust region optimization for reward-volatility reduction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4583–4589. International Joint Conferences on Artificial Intelligence Organization, 7 2020b. doi: 10.24963/ijcai.2020/632. URL `https://doi.org/10.24963/ijcai.2020/632`. Special Track on AI in FinTech.

L. Bisi, L. Sabbioni, E. Vittori, M. Papini, and M. Restelli. Risk-Averse Trust Region Optimization for Reward-Volatility Reduction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4583–4589, 2020c.

V. S. Borkar and S. P. Meyn. Risk-sensitive optimal control for markov decision processes with monotone cost. *Mathematics of Operations Research*, 27(1):192–209, 2002.

J. A. Boyan. Least-squares temporal difference learning. In *ICML*, pages 49–56, 1999.

A. Castelletti, S. Galelli, M. Restelli, and R. Soncini-Sessa. Tree-based reinforcement learning for optimal water reservoir operation. *Water Resources Research*, 46(9), 2010.

A. Castelletti, F. Pianosi, and M. Restelli. Multi-objective fitted q-iteration: Pareto frontier approximation in one single run. In *2011 International Conference on Networking, Sensing and Control*, pages 260–265. IEEE, 2011.

Z. Chen, S. Khodadadian, and S. T. Maguluri. Finite-sample analysis of off-policy natural actor-critic with linear function approximation. *arXiv preprint arXiv:2105.12540*, 2021.

Y. Chow, A. Tamar, S. Mannor, and M. Pavone. Risk-Sensitive and Robust Decision-Making: a CVaR Optimization Approach. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *NeurIPS 28*, pages 1522–1530. Curran Associates, Inc., 2015.

Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *JMLR*, 18(1):6070–6120, 2017. Publisher: JMLR. org.

K. Chung and M. Sobel. Risk-sensitive discounted mdps. *SIAM J. Control Optim*, 25: 49–62, 1986.

W. Dabney, G. Ostrovski, D. Silver, and R. Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pages 1096–1105. PMLR, 2018a.

W. Dabney, G. Ostrovski, D. Silver, and R. Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pages 1096–1105. PMLR, 2018b.

W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018c.

W. Dabney, Z. Kurth-Nelson, N. Uchida, C. K. Starkweather, D. Hassabis, R. Munos, and M. Botvinick. A distributional code for value in dopamine-based reinforcement learning. *Nature*, 577(7792):671–675, 2020.

M. Dahleh, M. A. Dahleh, and G. Verghese. Lectures on dynamic systems and control, 2004.

J. Danielsson, J.-P. Zigrand, B. N. Jorgensen, M. Sarma, and C. de Vries. Consistent measures of risk. 2006.

J. de Lope et al. Learning autonomous helicopter flight with evolutionary reinforcement learning. In *International Conference on Computer Aided Systems Theory*, pages 75–82. Springer, 2009.

M. 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 (ICML-11)*, pages 465–472. Citeseer, 2011.

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

E. V. Denardo. Contraction mappings in the theory underlying dynamic programming. *Siam Review*, 9(2):165–177, 1967.

J. Dhaene, S. Vanduffel, Q. Tang, M. Goovaerts, R. Kaas, and D. Vyncke. Solvency capital, risk measures and comonotonicity: a review. *DTEW Research Report 0416*, pages 1–33, 2004.

Y. B. E. Coumans. PyBullet, a Python module for physics simulation for games, robotics and machine learning., 2016. Publication Title: Github repository.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005a.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005b.

E. Even-Dar, S. M. Kakade, and Y. Mansour. Online markov decision processes. *Mathematics of Operations Research*, 34(3):726–736, 2009.

A.-m. Farahmand. Regularization in reinforcement learning. 2011.

Y. Fei, Z. Yang, Y. Chen, Z. Wang, and Q. Xie. Risk-sensitive reinforcement learning: Near-optimal risk-sample tradeoff in regret. *arXiv preprint arXiv:2006.13827*, 2020.

H. Föllmer and A. Schied. *Stochastic finance: an introduction in discrete time*. Walter de Gruyter, 2011.

J. García and F. Fernandez. A Comprehensive Survey on Safe Reinforcement Learning. *JMLR*, 16:1437–1480, 2015. URL `http://jmlr.org/papers/v16/garcia15a.html`.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

G. J. Gordon. Stable function approximation in dynamic programming. In *Machine Learning Proceedings 1995*, pages 261–268. Elsevier, 1995.

V. Goyal and J. Grand-Clement. Robust markov decision process: Beyond rectangularity. *arXiv preprint arXiv:1811.00215*, 2018.

M. Grant. How to prove the 2-norm of an invertible matrix is exactly the reciprocal of its minimum singular value? Computational Science Stack Exchange, 2014. URL `https://scicomp.stackexchange.com/q/10465`. URL:https://scicomp.stackexchange.com/q/10465 (version: 2014-01-06).

S. Gu, T. Lillicrap, I. Sutskever, and S. Levine. Continuous deep q-learning with model-based acceleration. In *International conference on machine learning*, pages 2829–2838. PMLR, 2016.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *ICML*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 1856–1865. JMLR.org, 2018.

W. Härdle and L. Simar. *Applied multivariate statistical analysis*, volume 22007. Springer, 2012.

R. A. Howard and J. E. Matheson. Risk-sensitive Markov decision processes. *Management science*, 18(7):356–369, 1972. Publisher: INFORMS.

D. A. Iancu, M. Petrik, and D. Subramanian. Tight approximations of dynamic risk measures. *Mathematics of Operations Research*, 40(3):655–682, 2015.

G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

S. C. Jaquette. Markov decision processes with a new optimality criterion: Discrete time. *The Annals of Statistics*, pages 496–505, 1973.

D. R. Jiang and W. B. Powell. Risk-averse approximate dynamic programming with quantile-based risk measures. *Mathematics of Operations Research*, 43(2):554–579, 2018.

A. Kacelnik and M. Bateson. Risky theories-the effects of variance on foraging decisions. *American Zoologist*, 36(4):402–434, 1996.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *ICML*, volume 2, pages 267–274, 2002.

S. M. Kakade. A natural policy gradient. In *NeurIPS*, pages 1531–1538, 2002.

M. Kato and K. Nakagawa. Direct expected quadratic utility maximization for mean-variance controlled reinforcement learning. *Available at SSRN 3818994*, 2020.

D. L. Kaufman and A. J. Schaefer. Robust modified policy iteration. *INFORMS Journal on Computing*, 25(3):396–410, 2013.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2014.

A. H. Klopf. A neuronal model of classical conditioning. *Psychobiology*, 16(2):85–125, 1988.

H. Kumar, A. Koppel, and A. Ribeiro. On the sample complexity of actor-critic method for reinforcement learning with function approximation. *arXiv preprint arXiv:1910.08412*, 2019.

S. Lange, T. Gabel, and M. Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13:3041–3074, 2012.

Y. Le Tallec. *Robust, risk-sensitive, and data-driven control of Markov decision processes*. PhD thesis, Massachusetts Institute of Technology, 2007.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

S. H. Lim, H. Xu, and S. Mannor. Reinforcement learning in robust markov decision processes. In *Advances in Neural Information Processing Systems*, pages 701–709, 2013.

B. Liu, J. Liu, M. Ghavamzadeh, S. Mahadevan, and M. Petrik. Finite-sample analysis of proximal gradient td algorithms. *arXiv preprint arXiv:2006.14364*, 2020.

S. Mannor and J. Tsitsiklis. Mean-variance optimization in markov decision processes. *arXiv preprint arXiv:1104.5601*, 2011.

S. Mannor, O. Mebel, and H. Xu. Lightning does not strike twice: Robust MDPs with coupled uncertainty. *arXiv preprint arXiv:1206.4643*, 2012.

S. I. Marcus, E. Fernández-Gaucherand, D. Hernández-Hernandez, S. Coraluppi, and P. Fard. Risk sensitive markov decision processes. In *Systems and control in the twenty-first century*, pages 263–279. Springer, 1997.

H. Markowitz. Portfolio Selection. *The Journal of Finance*, 7(1):77–91, 1952.

F. S. Melo and M. Lopes. Fitted natural actor-critic: A new algorithm for continuous state-action mdps. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 66–81. Springer, 2008.

A. M. Metelli, M. Papini, F. Faccio, and M. Restelli. Policy optimization via importance sampling. *arXiv preprint arXiv:1809.06098*, 2018.

A. M. Metelli, M. Papini, N. Montali, and M. Restelli. Importance sampling techniques for policy optimization. *J. Mach. Learn. Res.*, 21:141–1, 2020.

O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. *Machine learning*, 49(2-3):267–290, 2002. Publisher: Springer.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and others. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. Publisher: Nature Publishing Group.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, pages 1928–1937, 2016.

T. M. Moldovan and P. Abbeel. Risk aversion in Markov decision processes via near optimal Chernoff bounds. In *Advances in neural information processing systems*, pages 3131–3139, 2012.

G. E. Monahan. State of the art-a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982.

P. R. Montague, P. Dayan, and T. J. Sejnowski. A framework for mesencephalic dopamine systems based on predictive hebbian learning. *Journal of neuroscience*, 16(5):1936–1947, 1996.

J. Moody and M. Saffell. Learning to trade via direct reinforcement. *IEEE transactions on neural Networks*, 12(4):875–889, 2001. Publisher: IEEE.

O. Morgenstern and J. Von Neumann. *Theory of games and economic behavior*. Princeton university press, 1953.

T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka. Nonparametric Return Distribution Approximation for Reinforcement Learning. In *ICML*, 2010.

P. Muliere and G. Parmigiani. Utility and means in the 1930s. *Statistical Science*, pages 421–432, 1993.

R. Munos. Error bounds for approximate value iteration. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1006. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

R. Munos and C. Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

D. Nass, B. Belousov, and J. Peters. Entropic Risk Measure in Policy Search. *arXiv preprint arXiv:1906.09090*, 2019.

G. Neu and C. Pike-Burke. A unifying view of optimism in episodic reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1392–1403, 2020.

A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

A. Nilim and L. El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005a.

A. Nilim and L. El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005b. Publisher: INFORMS.

Y. Niv, J. A. Edlund, P. Dayan, and J. P. O'Doherty. Neural prediction errors reveal a risk-sensitive reinforcement-learning process in the human brain. *Journal of Neuroscience*, 32(2):551–562, 2012.

T. Osogami. Iterated risk measures for risk-sensitive Markov decision processes with discounted cost. *arXiv preprint arXiv:1202.3755*, 2012a.

T. Osogami. Robustness and risk-sensitivity in Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 233–241, 2012b.

A. B. Owen. *Monte Carlo theory, methods and examples*. 2013.

A. Palmisano. Risk-sensitive reinforcement learning for the dva hedging. Master's thesis, Politecnico di Milano, 2019.

M. Papini, M. Pirotta, and M. Restelli. Adaptive batch size for safe policy gradients. In *NeurIPS*, pages 3591–3600, 2017.

M. Papini, M. Pirotta, and M. Restelli. Smoothing Policies and Safe Policy Gradients, 2019. _eprint: 1905.03231.

S. Parisi, M. Pirotta, N. Smacchia, L. Bascetta, and M. Restelli. Policy gradient approaches for multi-objective sequential decision making. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 2323–2330. IEEE, 2014.

I. P. Pavlov and G. V. Anrep. *Conditioned reflexes: an investigation of the physiological activity of the cerebral cortex*, volume 142. london: oxford University Press, 1927.

J. Peters and S. Schaal. Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE, 2006.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008. Publisher: Elsevier.

J. Peters, S. Vijayakumar, and S. Schaal. Natural actor-critic. In *European Conference on Machine Learning*, pages 280–291. Springer, 2005.

L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta. Robust adversarial reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2817–2826. JMLR. org, 2017.

M. Pirotta, M. Restelli, and L. Bascetta. Adaptive Step-Size for Policy Gradient Methods. In *NeurIPS 26*, pages 1394–1402. Curran Associates, Inc., 2013.

M. Pirotta, M. Restelli, and L. Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 100(2-3):255–283, 2015. Publisher: Springer.

M. L. Platt and S. A. Huettel. Risky business: the neuroeconomics of decision making under uncertainty. *Nature neuroscience*, 11(4):398–403, 2008.

L. A. Prashanth and M. Fu. Risk-sensitive reinforcement learning: A constrained optimization viewpoint. *arXiv preprint arXiv:1810.09126*, 2018.

L. A. Prashanth and M. Ghavamzadeh. Actor-critic algorithms for risk-sensitive MDPs. In *Advances in neural information processing systems*, pages 252–260, 2013.

L. A. Prashanth and M. Ghavamzadeh. Variance-Constrained Actor-Critic Algorithms for Discounted and Average Reward MDPs. *CoRR*, abs/1403.6530, 2014. URL `http://arxiv.org/abs/1403.6530`. _eprint: 1403.6530.

M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann. Stable baselines3. *GitHub repository*, 2019.

K. Regan and C. Boutilier. Regret-based reward elicitation for markov decision processes. *arXiv preprint arXiv:1205.2619*, 2012.

M. Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European conference on machine learning*, pages 317–328. Springer, 2005.

A. Riva, L. Bisi, P. Liotet, L. Sabbioni, E. Vittori, M. Pinciroli, M. Trapletti, and M. Restelli. Learning fx trading strategies with fqi and persistent actions. 2021.

R. T. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, 26(7):1443–1471, 2002. Publisher: Elsevier.

R. T. Rockafellar, S. Uryasev, and others. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

R. T. Rockafellar, S. Uryasev, and M. Zabarankin. Generalized deviations in risk analysis. *Finance and Stochastics*, 10(1):51–74, 2006.

D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

G. A. Rummery and M. Niranjan. *On-line Q-learning using connectionist systems*, volume 37. Citeseer, 1994.

A. Ruszczyński. Risk-averse dynamic programming for Markov decision processes. *Mathematical programming*, 125(2):235–261, 2010. Publisher: Springer.

B. Scherrer. Approximate policy iteration schemes: a comparison. In *International Conference on Machine Learning*, pages 1314–1322. PMLR, 2014.

E. Schuitema, L. Buşoniu, R. Babuška, and P. Jonker. Control delay in reinforcement learning for real-time dynamic systems: a memoryless approach. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3226–3231. IEEE, 2010.

J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust Region Policy Optimization. In *ICML*, pages 1889–1897, 2015a.

J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347, 2017.

R. Sharma, R. Kumar, R. Saini, and G. Kapoor. Complementary upper bounds for fourth central moment with extensions and applications, 2015.

Y. Shen, R. Huang, C. Yan, and K. Obermayer. Risk-averse reinforcement learning for algorithmic trading. In *2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, pages 391–398. IEEE, 2014.

C. Sherstan, D. R. Ashley, B. Bennett, K. Young, A. White, M. White, and R. S. Sutton. Comparing direct and indirect temporal-difference methods for estimating the variance of the return. In *UAI*, pages 63–72, 2018.

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. A. Riedmiller. Deterministic Policy Gradient Algorithms. In *ICML*, volume 32 of *JMLR and Conference Proceedings*, pages 387–395. JMLR.org, 2014.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, and others. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484, 2016. Publisher: Nature Publishing Group.

D. Silver, S. Singh, D. Precup, and R. S. Sutton. Reward is enough. *Artificial Intelligence*, page 103535, 2021.

S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3):287–308, 2000.

S. Singh, R. L. Lewis, and A. G. Barto. Where do rewards come from. In *Proceedings of the annual conference of the cognitive science society*, pages 2601–2606. Cognitive Science Society, 2009.

M. J. Sobel. The variance of discounted Markov decision processes. *Journal of Applied Probability*, 19(4):794–802, 1982.

T. Spooner and R. Savani. A natural actor-critic algorithm with downside risk constraints. *arXiv preprint arXiv:2007.04203*, 2020.

R. S. Sutton. *Temporal credit assignment in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 1984.

R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0-262-19398-1.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 2000a. URL https://proceedings.neurips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, pages 1057–1063, 2000b.

C. Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

A. Tamar and S. Mannor. Variance adjusted actor critic algorithms. *arXiv preprint arXiv:1310.3697*, 2013.

A. Tamar, D. Di Castro, and S. Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1651–1658, 2012a.

A. Tamar, D. Di Castro, and S. Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1651–1658, 2012b.

A. Tamar, S. Mannor, and H. Xu. Scaling up robust mdps using function approximation. In *International conference on machine learning*, pages 181–189. PMLR, 2014.

A. Tamar, Y. Chow, M. Ghavamzadeh, and S. Mannor. Policy Gradient for Coherent Risk Measures. *CoRR*, page 9, 2015a.

A. Tamar, Y. Glassner, and S. Mannor. Optimizing the CVaR via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015b.

A. Tamar, D. D. Castro, and S. Mannor. Learning the variance of the reward-to-go. *Journal of Machine Learning Research*, 17(13):1–36, 2016a. URL http://jmlr.org/papers/v17/14-335.html.

A. Tamar, D. Di Castro, and S. Mannor. Learning the variance of the reward-to-go. *The Journal of Machine Learning Research*, 17(1):361–396, 2016b. Publisher: JMLR. org.

C. Tessler, D. J. Mankowitz, and S. Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

P. Thomas. Bias in natural actor-critic algorithms. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 441–448, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/thomas14.html.

A. Tirinzoni, M. Petrik, X. Chen, and B. Ziebart. Policy-conditioned uncertainty sets for robust Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 8939–8949, 2018.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001. Publisher: Springer.

J. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997. doi: 10.1109/9.580874.

A. Tversky and D. Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty*, 5(4):297–323, 1992.

N. A. Urpí, S. Curi, and A. Krause. Risk-averse offline reinforcement learning. *arXiv preprint arXiv:2102.05371*, 2021.

P. Vamplew, R. Dazeley, E. Barker, and A. Kelarev. Constructing stochastic mixture policies for episodic multiobjective reinforcement learning tasks. In *Australasian joint conference on artificial intelligence*, pages 340–349. Springer, 2009.

L. Wang, Q. Cai, Z. Yang, and Z. Wang. Neural policy gradient methods: Global optimality and rates of convergence. *arXiv preprint arXiv:1909.01150*, 2019.

S. Wang. Premium calculation by transforming the layer premium density. *ASTIN Bulletin: The Journal of the IAA*, 26(1):71–92, 1996.

C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

W. Wiesemann, D. Kuhn, and B. Rustem. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013. Publisher: INFORMS.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. Publisher: Springer.

S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1): 3–34, 2015. Publisher: Springer.

G. Wu and R. Gonzalez. Curvature of the probability weighting function. *Management science*, 42(12):1676–1690, 1996.

Y. Wu, W. Zhang, P. Xu, and Q. Gu. A finite time analysis of two time-scale actor critic methods. *arXiv preprint arXiv:2005.01350*, 2020.

T. Xie, B. Liu, Y. Xu, M. Ghavamzadeh, Y. Chow, D. Lyu, and D. Yoon. A block coordinate ascent algorithm for mean-variance optimization. In *Advances in Neural Information Processing Systems*, pages 1065–1075, 2018.

H. Xu and S. Mannor. Distributionally robust Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 2505–2513, 2010.

J. Xu, Y. Tian, P. Ma, D. Rus, S. Sueda, and W. Matusik. Prediction-guided multiobjective reinforcement learning for continuous robot control. In *International Conference on Machine Learning*, pages 10607–10616. PMLR, 2020a.

T. Xu, Z. Wang, and Y. Liang. Improving sample complexity bounds for (natural) actor-critic algorithms. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4358–4369. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper/2020/file/2e1b24a664f5e9c18f407b2f9c73e821-Paper.pdf.

M. E. Yaari. The dual theory of choice under risk. *Econometrica: Journal of the Econometric Society*, pages 95–115, 1987.

Z. Yang, K. Zhang, M. Hong, and T. Başar. A finite sample analysis of the actor-critic algorithm. In *2018 IEEE conference on decision and control (CDC)*, pages 2759–2764. IEEE, 2018.

T. Zahavy, B. O'Donoghue, G. Desjardins, and S. Singh. Reward is enough for convex mdps. *arXiv preprint arXiv:2106.00661*, 2021.

J. Zhang, A. Koppel, A. S. Bedi, C. Szepesvari, and M. Wang. Variational policy gradient method for reinforcement learning with general utilities. *arXiv preprint arXiv:2007.02151*, 2020a.

S. Zhang, B. Liu, and S. Whiteson. Per-Step Reward: A New Perspective for Risk-Averse Reinforcement Learning. *arXiv preprint arXiv:2004.10888*, 2020b.

S. Zhang, B. Liu, and S. Whiteson. Mean- variance policy iteration for risk- averse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35. Association for the Advancement of Artificial Intelligence, 2021.

Z. Zheng, J. Oh, M. Hessel, Z. Xu, M. Kroiss, H. Van Hasselt, D. Silver, and S. Singh. What can learned intrinsic rewards capture? In *International Conference on Machine Learning*, pages 11436–11446. PMLR, 2020.