POLITECNICO DI MILANO
Master of Science Degree in Computer Engineering
Department of Electronics, Information and Bioengineering

A study on off–line Policy Gradient Algorithms and their application to Risk–Averse Learning

AI & R Lab
The Artificial Intelligence and Robotics Lab
of the Politecnico di Milano

Advisor:
Prof. Marcello Restelli
Co-advisor:
Dott. Matteo Pirotta

Author:
Luca Mastrangelo 770545

Academic Year 2013-2014

Alla mia famiglia,
che ancora una volta mi ha permesso di arrivare dove sono.

Al professore Restelli e al dottor Pirotta
per avermi insegnato ad insegnare.

Abstract

This thesis provides a study of risk–averse reinforcement learning in general sequential decision–making problems. It focuses on likelihood ratio methods based on variance related criteria such as the Sharpe Ratio and the mean–variance criterion, which are the most common risk measures in economics and operations research.

Many real financial problems are too complex to describe the underlying model in terms of small finite sets of states and actions. Moreover a learning agent can be mostly trained with historical data collected by other agents, because, in the general case, it is not convenient, or even possible, to explore a risky environment and extract enough data to perform an on–policy learning.

For these reasons the main contribution of this work is to present risk–aware algorithms devised to both operate in an off–line batch context and solve problems with continuous state and/or action spaces.

Estratto in Lingua Italiana

In molti settori, come ad esempio nel campo dell'automazione, della ricerca operativa o della robotica, esistono compiti complessi che non possono essere risolti tramite formule matematiche giá pronte all'uso, ma la soluzione deve essere appresa sfruttando l'esperienza che l'individuo ha del problema specifico. L'apprendimento per rinforzo ha come obiettivo quello di fornire ad un agente artificiale tecniche per scegliere in maniera automatica il comportamento migliore da utilizzare per eseguire efficacemente un certo compito di controllo. Nello specifico l'agente non ha una conoscenza completa del problema e non necessita di un modello per rappresentarne le dinamiche interne poiché é in grado di interagire e modificare l'ambiente attraverso un certo numero di azioni e di ricevere da esso segnali di rinforzo che sono indici della bontá delle sue scelte.

L'agente deve decidere quale azione sia meglio utilizzare in ogni possibile situazione, o stato, per massimizzare il valore dei rinforzi accumulati, cioé quello che é chiamato ritorno totale. Questo paradigma é molto interessante in quanto sufficientemente astratto e generale per modellare in modo efficiente l'interazione tra agenti e ambienti di svariate tipologie.

Questa tesi presenta uno studio delle tecniche di apprendimento per rinforzo nell'ambito dell'avversione al rischio per generici processi decisionali. Qui l'agente é guidato nelle sue decisioni dalla volontá di massimizzare il profitto medio, ma é anche nel suo interesse mantenere basso il rischio di incorrere in perdite inattese aumentando la stabilitá dei suoi guadagni. In molti problemi questo é possibile solo rinunciando a comportamenti troppo avventati e sacrificando una parte del profitto totale per ottenere un guadagno piú basso ma piú certo.
Tra tutte le tecniche di apprendimento per rinforzo, in questa tesi sono presi in considerazione i metodi basati sull'indice di verosimiglianza (likelihood ratio methods) che sfruttano la tecnica di ascesa dei gradienti di criteri in cui puó essere presente o meno la nozione di rischio. L'indice di Sharpe o il crite-

rio media–varianza sono le misure di rischio più diffuse in economia e ricerca operativa e sono caratterizzate da funzioni obiettivo in cui il criterio standard di massimizzazione del valore atteso del ritorno totale viene penalizzato in base alla sua stessa varianza. A differenza di altri metodi che sfruttano la nozione di funzione di utilitá, qui il processo di apprendimento consiste nel continuo aggiornamento dei soli parametri della politica dell'agente che avviene seguendo la direzione lungo la quale il guadagno stimato cresce.

Molti problemi nell'ambito finanziario o medico sono troppo complessi per poterne descrivere il sottostante modello tramite insiemi di stati e azioni che siano di piccole dimensioni. Inoltre un agente può utilizzare nel suo processo di apprendimento prevalentemente dati storici collezionati da altri agenti (apprendimento off–policy), perchè, in generale, non è conveniente, o addirittura possibile, esplorare un ambiente rischioso ed estrarre dati a sufficienza per eseguire un apprendimento in linea. Per queste ragioni il principale contributo di questo lavoro è quello di presentare algoritmi di avversione al rischio ideati sia per operare con dati cosiddetti off–line, sia per risolvere problemi con spazi di stato e azione continui.

Apprendere non potendo decidere in prima persona quanti campioni raccogliere e con quale criterio farlo é un arduo compito e, nell'ambito dell'apprendimento artificiale, questa difficoltá si concretizza in una stima molto variabile del gradiente che guida gli aggiornamenti della politica. Per ovviare a questo problema sono formalizzate, e inserite all'interno degli algoritmi, le nozioni di baseline e le formulazioni REINFORCE e GPOMDP del gradiente. Aggiungere una baseline alla stima del gradiente contribuisce a ridurne drasticamente la variabilitá senza alterare il suo valore atteso, il che permette di apprendere piú velocemente, con maggiore precisione e utilizzando meno campioni come dimostrano gli esperimenti eseguiti sia sugli algoritmi on–policy sia su quelli off–policy.

In ambito off–policy si é analizzato come le prestazioni degli algoritmi dipendano dalla distanza tra la politica target e quella behavior nello spazio dei loro parametri. In particolare i risultati evidenziano che piú le due politiche sono diverse piú é elevata la varianza del gradiente indipendentemente dalla funzione obiettivo che si sta ottimizzando.

In conclusione si analizzano le due misure di rischio che contengono un parametro di penalizzazione, cioé il criterio media–varianza e il gradiente modificato. In questi test é ben visibile che agendo sul parametro é possibile gestire il livello di rischio a piacimento ottenendo tutti i comportamenti compresi tra la neutralitá e la totale avversione.

Contents

List of Figures

Chapter 1

Introduction

Many real sequential decision–making problems can be approximately solved using Reinforcement Learning (RL) techniques. Regardless of the specific contexts in which these problems arise, there are some common aspects which must be taken into account in order to identify which learning method is more suitable for the task.

For example, when the problem is too complex to describe the underlying model in terms of small finite sets of states and actions, the algorithm must be able to exploit function approximations to represent the information in a compact way. These approaches employ an approximator to learn some mappings among state, action and reward spaces. However introducing these elements in the algorithm structure can lead to biased results or oscillation phenomena, that do not allow to converge to good solutions.

In this thesis we have restricted our attention to policy gradient algorithms, in particular to the likelihood ratio methods, which are able to learn optimal parameters for a certain family of policies without need to estimate value functions as instead happens in critic–only and actor–critic algorithms. Given that the performance of these algorithms drastically depends on the accuracy of the gradient evaluations, most of the efforts must be dedicated to devise methods capable of reducing the variability of these estimations. The baseline technique and the GPOMDP formulation of the gradient are two of the most effective tools exploitable in this context.

Another important aspect of the learning methods is the frequency at which the agent can interact with the environment in order to draw samples. In fact if the environment is not able to react quickly to the decisions taken by the agent then the sole sample collection may take a very long

time, slowing down the entire learning process. In finance, medicine or robotic applications the agent can be mostly trained with historical data collected by other agents, because, in the general case, it is not convenient, or even possible, to explore a whole environment starting with no experience and extracting enough data to perform an on–line learning. Therefore it is necessary to switch the context from on–line on–policy scenario to off–line off–policy learning, where the key idea is that the actor updates are guided by the performance of a target policy evaluated using samples drawn from another probability distribution. Unfortunately learning in this new setting with policy gradient methods may lead to very unreliable results because:

1. The variance of the gradient estimated becomes higher due to the negative effect of the importance weights introduced in off–policy learning.

2. From fixed batch data the performance of policies very different from the behavior policy, that has generated them, cannot be well estimated, hence this is a further source of uncertainty to be controlled.

A large part of this work is devoted to describe how it is possible to overcome these issues, studying and formally deriving the off–policy formulations of baselines for both REINFORCE and GPOMDP–like gradients.

After all these considerations a further analysis is performed within the risk–averse framework. This choice is due to the fact that some of the most complex problems of practical interest cannot be solved with standard RL techniques, because the concept of optimal policy is no more connected only to the pure expected total return, but it also involved a risk sensitivity. In terms of optimization problem the risk management implies to minimize the variability of returns in addition to the maximization of the standard RL criterion, i.e., the expected sum of discounted rewards. This can be done using as objective function the Sharpe ratio or the mean–variance criterion, which are the most common risk measures in economics and operations research. Off–line learning and risk–sensitive optimization are two concepts deeply related, because nobody is willing to explore by himself a risky environment in order to learn from his own mistakes. A better solution is to extract from a batch dataset useful information and develop a policy which is able to guarantee the desired risk tolerance, even if this means to sacrifice some of the expected profit in order to increase the stability of the gain.

As a conclusion, a new objective function called Modified Gradient is introduced. This measure is a concept which involves elements from both

off–line learning and risk–sensitive RL, because it penalizes the improvement towards policies for which it is not possible to verify the performance gain from the batch dataset because too different from the behavior policy. Now the agent takes into account the trade–off between remaining close to the well estimated performance of a non–optimal behavior and moving toward potentially improved policy whose effects on the system are more uncertain.

For all the motivations presented in this brief introduction the overall goal of this thesis is to present risk–aware algorithms devised to both operate in an off–line batch context and solve problems with continuous state and/or action spaces. An important contribution is given by the experiments, that empirically support the theoretical analysis showing the effects of the implemented techniques in all the several scenarios previously described.

1.1 Overview

This thesis is structured as follows.

Chapter 2 presents the fundamental concepts related to the proposed theme, such as Reinforcement Learning, policy gradient methods, off–line learning and risk–sensitive optimization. Moreover it includes a brief analysis of the previous works in the literature and the definition of the notation used in the rest of this document.

In Chapter 3 the standard on–policy formulations are redefined in order to operate in off–policy scenarios. It is also explained how different baseline techniques can be used to enhance the performance of the algorithms.

In Chapter 4 are defined three risk measures and the new elements needed in off-line risk–sensitive algorithms.

Chapter 5 describes the model used to perform the tests, the analysis of the results and empirical evaluations and comparisons of the proposed algorithms. The experiments are organized according to the structure of the theoretical part of this thesis. In fact first the risk–neutral methods are presented, secondly the notions of risk are introduced in the objective functions in order to create risk–aware algorithms. Most of the algorithms are tested both in on–policy and off–policy scenarios, and in both REINFORCE and GPOMDP–like formulations.

Finally, Chapter 6 presents the reached conclusions from the analysis, and includes some suggestions for future work.

Chapter 2

State of the Art

In this chapter we describe the theories and the concepts that constitute a necessary prerequisite to understand the problem and developed the solution proposed in this thesis.

The chapter is organized as follows. In Section 2.1 Reinforcement Learning is contextualized in the area of machine learning, presenting also its most important elements. In Section 2.2 is introduced the MDP framework and the notation that will be used in this text. In Section 2.3 is presented a general classification of RL methods, pointing out the features useful to easily understand this thesis. Special attention is reserved here to the description of policy gradient methods. Section 2.4 explains how RL is used to deal with risk–averse optimization, identifying the already developed solutions and the diversification of the several techniques adopted so far.

2.1 Reinforcement Learning

Machine learning (ML) is the field of artificial intelligence that studies how algorithms can learn from data. One class of techniques used to achieve this purpose is Reinforcement Learning (RL) [81].

In general the learning problem is a complex and difficult task because the available data (training data) represent only a subset of the whole knowledge of the problem, therefore each method must have a generalization procedure to be also able to face all the cases not covered by the data.

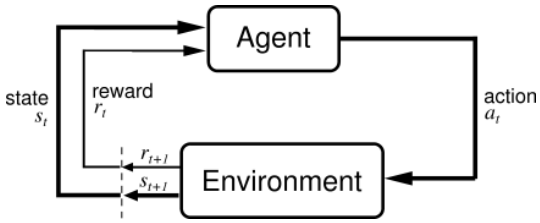


Figure 2.1: The RL framework. Figure represents the interaction between the agent and the environment.

ML can be mainly divided into three subclasses: supervised learning, unsupervised learning and, the already mentioned, RL. Each of these categories has its own competence area and it can use only data with a specific structure. For example, RL can deal with optimal sequential decision problems in which an agent has to learn how to behave in uncertain environment in order to attain a certain goal (e.g., win a game or maintain balance).

The goal is formalized in terms of numerical signals (i.e., rewards) that the agent receives from the system as result of their interactions through actions. Each reward indicates the intrinsic desirability of being in the current situation, in particular it is the way of communicating to the agent *what* is the goal to achieve, but not *how* the goal can be achieved.

If the rewards respect these requirements then reaching the goal and maximize the total amount of reward (i.e., return) are the same concept.

In RL, differently from supervised learning [58], the training data cannot be provided by an external supervisor who already knows how to label the behaviors, otherwise the learning problem would be already trivially solved and the environment could no longer be described as uncertain. For this reason each RL agent must be able to interact with the environment to collect its own experience as shown in Figure 2.1.
At each time step t the agent perceives as input the state s_t of the system and automatically she chooses and executes a control action a_t. As a result of this state-action pair the environment changes producing a transition to a new state s_{t+1} and providing a reward r_{t+1} to the agent.

RL algorithms have to learn how map states to actions, i.e., a policy, to improve the long–term future utility relying only on data samples formed by state observations, actions and immediate reward signals.

The key features of the problem previously described can be formalized using the Markov Decision Process (MDP) framework [72]. This general and abstract formulation allows RL techniques to be useful in several disciplines, such as automation control, operational research, game theory, economy and medicine.

Instead, if the agent wanted to approach the same problem from a supervised learning point of view then it would need different structured data, in particular each sample should associate a state with the return accumulated from that state on, following a certain sequence of states and actions.

RL and optimal control theory share the aim to find an optimal control law exploiting the Dynamic Programming (DP) [81] approach to solving complex problems: dividing the problem in simpler parts (subproblems) it is possible to combine easily their optimal solutions to reach an overall optimal solution. However RL and DP start from different assumptions because the former needs to know neither the environment dynamics nor the mapping function between state-action pairs and rewards, while in the latter all the characteristics of the system must be perfectly specified.

This deep knowledge of the system makes DP methods not computationally feasible for most problems of practical interest due to two distinct reasons. First, the real model of the environment may not be known even to the system experts, secondly, assuming that the model is fully defined, trying to solve exactly the problem in all the states of the model requires amount of time and space exponential in the number of state variables.

On the other hand, as direct consequence of the weaker requirements, in the general case RL can only find an approximate solution to a complex problem, but it can be a good compromise between spending a lot of time and effort to obtain the real optimal solution using DP algorithms and not being able to compute it at all.

2.2 Markov Decision Process

An MDP is described by a tuple $\langle \mathcal{S}, \mathcal{A}, T, \mathcal{P}, \mathcal{R}, D, \gamma \rangle$ where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $T \in \mathbb{N}_+$ is the learning horizon that can be either finite or infinite, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the Markovian transition model, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, D is the distribution of the initial state and $\gamma \in [0, 1)$ is the discount factor for future rewards.

State and action spaces may be arbitrary sets either discrete or continuous ($\mathcal{S} \subseteq \mathbb{R}^n$ and $\mathcal{A} \subseteq \mathbb{R}^m$). The control policy is characterized by a density distribution $\pi(\cdot|s)$ that specifies for each state s the density distribution over the action space $\mathcal{A}$.

At each time step $t \in 0, \ldots, T$, given the state s_t, the agent selects an action $a_t \sim \pi(\cdot|s_t)$ and a new state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ is reached. The function $\mathcal{P}(s_{t+1}|s_t, a_t)$ defines the transition density between state s_{t+1} and s_t under action a_t. Upon a transition the agent receives a scalar signal $r_{t+1} \sim \mathcal{R}(\boldsymbol{z}_t)$ that represents the immediate reward for the state–action pair $\boldsymbol{z}_t := (s_t, a_t)$. For each state s the utility of following a stationary policy π is defined as:

$$V^{\pi}(s) = \mathop{\mathbb{E}}_{\substack{a_j \sim \pi \\ s_j \sim \mathcal{P}}} \left[\left(\sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j) \right) \middle| s_0 = s \right]. \tag{2.1}$$

Policies can be ranked by their expected discounted return starting from the state distribution D:

$$J(\pi) = \int_{\mathcal{S}} D(s) V^{\pi}(s) \, \mathrm{d}s. \tag{2.2}$$

Solving an MDP means finding a policy π^* that maximizes the expected long-term reward: $\pi^* \in arg \max_{\pi \in \Pi} J(\pi)$. This work considers the problem of finding a policy that maximizes the expected discounted reward over a class of parametrized policies $\Pi_{\boldsymbol{\theta}} = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\}$, where $\Theta \subseteq \mathbb{R}^K$ $(K \in \mathbb{N}_+)$, is the parameter space.

There are different models of optimal behavior [39, 81] which result in different versions of the expected return (Equation 2.2). A finite-horizon model only attempts to maximize the expected reward for the horizon T

$$V^{\pi}(s) = \mathop{\mathbb{E}}_{\substack{a_j \sim \pi \\ s_j \sim \mathcal{P}}} \left[\left(\sum_{j=0}^{T} \mathcal{R}(\boldsymbol{z}_j) \right) \middle| s_0 = s \right]$$

while, in the limit, when $\gamma = 1$ the metric approaches what is known as the average–reward criterion

$$V^{\pi}(s) = \lim_{T \to \infty} \mathop{\mathbb{E}}_{\substack{a_j \sim \pi \\ s_j \sim \mathcal{P}}} \left[\left(\frac{1}{T} \sum_{j=0}^{T} \mathcal{R}(\boldsymbol{z}_j) \right) \middle| s_0 = s \right].$$

Relevant analyses about the fundamental aspects of MDPs can be found in [10, 11, 23, 72, 81].

2.3 RL algorithm classification

RL methods can be classified considering several dimensions, the ones useful to introduce this work are presented in the next sections.

2.3.1 Tabular and Function approximation

If the states $\mathcal{S}$ of the environment and the actions $\mathcal{A}$ available to the agent are small discrete sets then all the informations processed by a RL algorithm can be stored in a compact memory space as a *table* with one entry for each state or for each state-action pair.

Unfortunately many real problems are defined in continuous (or large discrete) state space and/or action space. This is a severe issue not just for the memory needed for larger tables and the time taken to read from them, but also because with continuous spaces most of the states and actions will never be sampled, so the only way to learn anything at all on them is to generalize from the most similar examples in data.

Function approximation is an instance of supervised learning which consists in representing each state and/or action as a small number of features, thus any of the methods studied in this field can also be combined with RL algorithms (e.g., parametric linear/non–linear function, neural networks, decision trees, coarse/tile coding, radial basis function).

2.3.2 Value–based, Policy–based, and Actor-critic

Value–based (or Critic–only) algorithms such as Q-learning (QL) [92] and SARSA [74] use an indirect approach: they search for an optimal value function from which derive the optimal policy. The value function represents a solution to the credit assignment problem, namely determines how the success of a system's overall performance is due to the various contributions of the system's components.

This concept is very abstract when referred to animal learning, but in RL it is implemented storing in memory entries or parameters of a critic (i.e., look-up table or a function approximator), which will be updated propagating the influence of delayed reward to all states and actions that have an effect on that reward according to a RL formulation of the Bellman equation (TD methods [80] and eligibility traces [45]).

Figure 2.2: Example of value function used in a critic–only method for a discrete model and the derived optimal policy (arrows).

Policy–based (Policy search or Actor–only) algorithms work directly in the actor (i.e., policy) space without the help of value functions [65, 71, 82]. The two approaches available are gradient-free methods and gradient-based.

Gradient-free methods is a large class of methods avoids relying on gradient information. These include simulated annealing, cross-entropy search or methods of evolutionary computation, such as genetic algorithms [18, 76, 89].

When the possible behaviors are represented by a parametrized family of policies policy search methods can be based on approximating gradient ascent in which the gradient of the performance J, w.r.t. the actor parameters $\boldsymbol{\theta}$, is directly estimated by simulation and the parameters are updated in a direction of improvement.

$$\boldsymbol{\theta}_{h+1} = \boldsymbol{\theta}_h + \alpha_h \nabla_{\boldsymbol{\theta}} J|_{\boldsymbol{\theta}=\boldsymbol{\theta}_h} \tag{2.3}$$

where $\alpha_h \in (0, 1)$ denotes a learning rate and $h \in \{0, 1, 2, \ldots\}$ the current update number.

Policy gradient methods have received a lot of attention thanks to REINFORCE [93] and GPOMDP [9, 43] algorithms, but also to the more recent POWER [46] used in the field of dynamic motor primitives. In particular, the already mentioned, GPOMDP can be seen as a more efficient implementation of the REINFORCE algorithm. In fact the latter does not perform an optimal credit assignment, since it ignores that the reward at time t does not depend on the action performed after time t. GPOMDP overcomes this issue taking into account the causality of rewards in the REINFORCE definition of policy gradient.

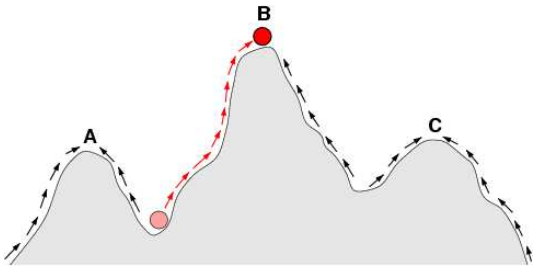


Figure 2.3: Visual representation of gradient ascent methods.

Actor–critic methods explicitly represent both the policy and the value function with two independent function approximations. The critic must learn about and critique whatever policy is currently being followed by the actor. In particular, after each action selection, the critic evaluates the new state to determine whether the results are improving or not. As consequence of this analysis the action will be recommended again in the future or discarded by the actor.

Using a critic to help the actor update leads to better results when the real value function of the problem is contained in the parametrized family of functions chosen, in this case actor–critic methods can converge to the optimal policy faster than other techniques.

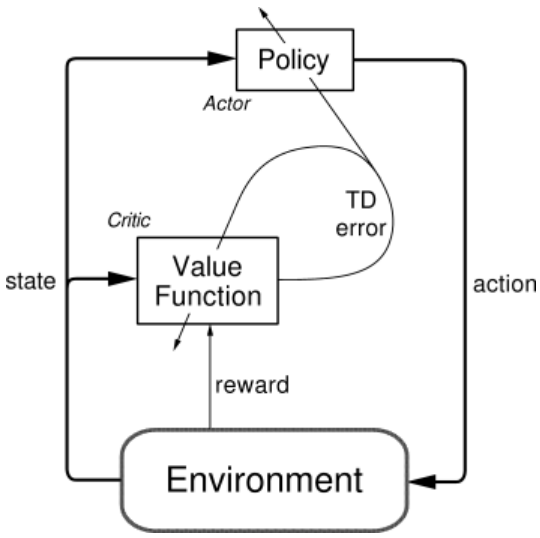


Figure 2.4: The actor-critic architecture.

On the other hand, if the parametric value function is either too general or it cannot well represent the real utility, then the critic estimations can be an obstacle to the learning process. In fact in the first case the total variance of the gradient estimation increases and in the second case the result obtained is biased. Therefore working with actor–only methods can be a smarter solution when the dynamics of the problem are very uncertain, as it is supposed in this work.

2.3.3 Policy Gradient methods

The main problem in policy–based algorithms is to obtain a good estimator of the policy gradient $\nabla_{\boldsymbol{\theta}} J(\pi)$, the most significant approaches in RL are finite–difference and likelihood ratio methods [68].

Finite–difference methods

Finite–difference techniques [79] estimate gradient direction by resolving a regression problem based on the performance evaluation of policies associated to different small perturbations of the current parametrization. These methods have some advantages: they are easy to implement, do not need assumptions on the differentiability of the policy w.r.t. the policy parameters, and are efficient in deterministic settings. On the other hand, when used on real systems, the choice of parameter perturbations may be difficult and critical for system safeness. Furthermore, the presence of uncertainties may significantly slow down the convergence rate.

Likelihood Ratio methods

The second class of policy search approaches is called *likelihood ratio* methods [3, 27, 28] and the intensive study of them is the foundation of this work. They evaluate policies using an alternative formulation of the expected return definition (see Equation 2.2), which uses the essential concept of trajectory space $\mathbb{T}$. A trajectory $\tau \in \mathbb{T}$ is the ordered set of all states s_t and actions a_t that the agent experiences in a certain simulation started at time $t = 0$ till reaching the learning horizon T. Formally $\tau = \{s_t, a_t\}_{t=0}^{T} = \{\boldsymbol{z}_t\}_{t=0}^{T} = \boldsymbol{z}_{0:T}$ is a trajectory of length $T + 1$ and its discounted return is $\mathcal{R}(\tau) = \sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j)$.

Given that τ depends on the distribution of the initial state D, the transition model $\mathcal{P}$ and the actual policy π used by the agent, the trajectory is said drawn from density distribution $p(\tau|\pi)$ simply described by the following equation:

$$p(\tau|\pi) = D(s_0)\pi(\boldsymbol{z}_0)\prod_{k=1}^{T}\mathcal{P}(s_k|\boldsymbol{z}_{k-1})\pi(\boldsymbol{z}_k).$$

These considerations allow to redefine the policies evaluation criterion as

$$J(\pi) = \int_{\mathcal{S}} D(s)V^{\pi}(s)\,\mathrm{d}s = \int_{\mathbb{T}} p(\tau|\pi)\,\mathcal{R}(\tau)\,\mathrm{d}\tau = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi)}[\mathcal{R}(\tau)]. \tag{2.4}$$

As shown in [68, 93] the equation above is used to derive $\nabla_{\boldsymbol{\theta}}J(\pi)$ for the actor update rule (Equation 2.3), in particular the gradient expression in a REINFORCE–like fashion is:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}J(\pi) &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi)}[\mathcal{R}(\tau)\nabla_{\boldsymbol{\theta}}\log p(\tau|\pi)] \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi)}\left[\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}}\log\pi(\boldsymbol{z}_i)\right].\end{aligned} \tag{2.5}$$

Now we present a quick overview on pros and cons of policy gradient methods.

The advantages of policy gradient methods for real world applications are numerous. Compared to several traditional reinforcement learning approaches, policy gradients scale well to high–dimensional continuous state and action problems, in particular they can deal with continuous actions in the exactly same way as discrete ones. If the gradient estimate is unbiased and learning rates fulfill $\sum_{h=0}^{\infty}a_h > 0$ and $\sum_{h=0}^{\infty}a_h^2 = const$ (see Equation 2.3), the learning process is guaranteed to converge at least to a local maximum. The problem of finding the global maximum instead of a local one can be relieved using multiple different starting parameters as any gradient ascent technique.

Furthermore, policy representation can be properly designed for the given task, thus allowing to:

1. Incorporate domain knowledge into the algorithm useful to speed up the learning process.
2. Manage fewer parameters than in value–based approaches.
3. Prevent the unexpected execution of dangerous policies that may harm the system or the agent.
4. Be used either model–free or model–based.

A possible drawback of policy search algorithms is that the gradient estimators may have a large variance and in addition, as the policy changes, a new gradient is computed independently of past estimates making the use of on–line sampled data not very efficient. Finally, they always have an open parameter, the learning rate, whose choice decides over the order of magnitude of the speed of convergence or it can bring to oscillations or even divergence in the policy parameters updates. To overcome this issue new approaches have been designed as the ones inspired by expectation–maximization [46, 91] or automatic selection techniques [69, 70].

Baseline

The variance of the gradient estimator is a significant practical problem in gradient ascent methods (actor–only or actor–critic).

The technique of discounting future rewards was introduced in [44], and it exploits the trade off bias–variance, i.e., variance in the gradient estimates can be reduced increasing the estimation bias [55].

A much better way to solve this problem is to add to the gradient estimation formula a new term, called baseline $b \in \mathbb{R}^K$. The baseline technique affects the estimation variance without adding bias to the final result. In particular among the all possible terms that can be added, the optimal baseline can guarantee to obtain policy gradients with minimum variance, thus the best performance improvement possible. This topic is well studied in [8, 29, 66, 87].

2.3.4 On–policy and Off–policy

As pointed out in Section 2.1 and in Section 2.2 the purpose of RL is to find the optimal policy π^*, i.e., the solution to the MDP which describes the environment of the problem. RL algorithms achieve this goal mainly in two different ways: on–policy and off–policy. This choice depends on how the agent can interact with the system.

On–Policy

On–policy algorithms are related to the most common idea of learning: evaluating and improving the same policy π that the agent uses to make decisions. The agent starts from an initial policy (e.g., randomly selected or using a default parametrization), observing the agent from an external point of view it is possible to notice, update by update, the gradual changes in its behavior which, at the end, converges to the optimal policy.

This technique wants to imitate the animal learning, in fact it is designed to learn from the mistakes made by the agent itself applying the necessary corrections while the samples are collected. Analyzing the sequences of interaction from the starting time until the conclusion of the learning process, it can be noticed a progressive increment in the accumulated total return per sequence.

Off–Policy

In *off–policy* algorithms two policies, called behavior $\pi^{\mathcal{B}}$ and target $\pi^{\mathcal{T}}$, are involved. As their names suggest the first is used to select the action for the interaction with the system and collect the samples, while the second is used to evaluate the agent performance and it is improved in each update.

This technique is very useful in several contexts, for example, while the agent is following a behavior policy to explore the environment it can apply the learned information only to the target policy without affecting its exploratory behavior, in this way the agent may end up learning strategies not necessarily exhibited during the learning phase.

Another common use of off-policy methods is in the off–line batch context, i.e., when the training samples are collected before the agent starts its learning process (e.g., historical data or experiments performed in safety conditions). In this settings the agent has no control on the behavior policy, so the data cannot be chosen, modified or extended and the learning process must reuse iteratively the same samples to determine the optimal policy.

Importance Sampling

Importance sampling is an essential component of off–policy learning [40, 48, 52, 73], this technique allows to compute an expectation value of a random variable $f(x)$ under a probability distribution $P(x)$ given samples x from a different distribution $Q(x)$. On–policy methods can be seen as particular cases of off-policy learning where $P(x)$ and $Q(x)$ are the same distribution, in fact at each step the agent wants to estimate the performance of the same policy used to collect samples.
The desired goal is to evaluate

$$\mathop{\mathbb{E}}_{x\sim P(\cdot)}[f(x)] = \int P(x)f(x)\,\mathrm{d}x$$

starting from samples drawn from the proposal distribution Q.
This can be achieved observing that

$$\mathop{\mathbb{E}}_{x\sim P(\cdot)}[f(x)] = \int P(x)f(x)\,\mathrm{d}x = \int Q(x)\frac{P(x)}{Q(x)}f(x)\,\mathrm{d}x = \mathop{\mathbb{E}}_{x\sim Q(\cdot)}\left[\frac{P(x)}{Q(x)}f(x)\right]$$

which is a simple adjustment of the original values of the samples with the importance weight $\omega(x) = \frac{P(x)}{Q(x)}$.

Importance Sampling in RL

Referring to the formalization introduced in Section 2.3.3 besides to the concept of trajectory, the corresponding off–policy RL scenario involves:

- trajectory τ as the basic sample x;
- total discounted return $\mathcal{R}(\tau)$ as the random variable $f(x)$;
- $J\left(\pi^{\mathcal{T}}\right)$ as the expectation value of the random variable (see Equation 2.4);
- the trajectory probability under the target policy $p\left(\tau|\pi^{\mathcal{T}}\right)$ as $P(x)$;
- the trajectory probability under the behavior policy $p\left(\tau|\pi^{\mathcal{B}}\right)$ as $Q(x)$.

As a consequence, the importance weight correction along the whole trajectory is

$$J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{T}})}[\mathcal{R}(\tau)] = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}[\omega(\tau)\mathcal{R}(\tau)] \tag{2.6}$$

where $\omega(\tau) = \frac{p\left(\cdot|\pi^{\mathcal{T}}\right)}{p(\cdot|\pi^{\mathcal{B}})} = \omega(\boldsymbol{z}_{0:T}) = \prod_{w=0}^{T}\omega(\boldsymbol{z}_w)$ and $\omega(\boldsymbol{z}_w) = \frac{\pi^{\mathcal{T}}(a_w|s_w)}{\pi^{\mathcal{B}}(a_w|s_w)}$.

The derivation of the off–policy gradient from Equation 2.6 can be found in [38], here is presented its final formulation:

$$\nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j)\, \omega(\tau) \sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) \right]. \tag{2.7}$$

In order to well define $\omega(\tau)$ the behavior policy can be any arbitrary policy as long as it has non zero probability of selecting any action in every state, formally $\pi^{\mathcal{B}}(a|s) > 0,\ \forall\, s \in \mathcal{S},\, a \in \mathcal{A}$.

Unfortunately the importance sampling technique can lead to unreliable estimates when the two policies, $\pi^{\mathcal{B}}$ and $\pi^{\mathcal{T}}$, are too different. This scenario can happen only in off-policy setting, because in on–policy learning the two distributions are always identical. Let us consider a batch dataset in which most of the samples have probabilities to be drawn from the target distribution $\pi^{\mathcal{T}}$ very different from the ones to be drawn from the behavior policy $\pi^{\mathcal{B}}$. The corresponding importance weights $\frac{p\left(\cdot|\pi^{\mathcal{T}}\right)}{p(\cdot|\pi^{\mathcal{B}})}$ are either too high or too small, therefore the final estimation takes into account only the contributions of the samples with the highest importance weights without even consider the presence of the other samples in the dataset.

This concept is formalized in the effective sample size (ESS) [49, 88], which is a measure of the quality of the importance sampled estimate.

$$ESS = \frac{m}{1 + Var(\omega_i)}$$

where m is the number of samples and ω_i are the importance weights. In off-policy learning the distance between the parameters of the behavior policy and the ones of the target policy may increase after each update. As a direct consequences of this the ESS decreases and the variance of the gradient estimation increases.

2.4 Risk–Sensitive Reinforcement Learning

Risk–sensitive problems have been studied in various fields besides machine learning, e.g., optimal control [33], operations research [14, 21, 34], finance [50, 75, 90], as well as neuroscience [31, 63].

In order to introduce the general risk–averse concepts within the RL framework it is useful to notice that all the definitions of policy utility V^π presented in Section 2.2 have in common the straightforward idea of a strong connection between the maximization of the expected return and utility. A rational agent must do whatever is possible to obtain more and more reward. Denying in part this key component of standard RL is one of the main features of risk–sensitive optimization.

As a matter of fact in many real sequential decision–making problems people are willing even to sacrifice part of the profit in order to decrease the probability of losses and/or increase the stability of the return. As a consequence an agent may want to manage risk by trade off the minimization of some measure of variability in rewards and the maximization of the expected reward (see Equation 2.2).

The variability in a problem can be due to two types of uncertainties:

1. Uncertainties in the model parameters, which are related to the imperfect knowledge of the problem parameters, they are the topic of robust MDPs [12, 26, 36, 64, 95].

2. The inherent uncertainty related to the stochastic nature of the system (random reward and transition function) which makes the consequences of actions unpredictable at the time when a decision is made, this is the topic of risk-sensitive RL.

The first attempts [16, 53] to incorporate a measure of risk in optimal control problem suggested the variance of the return as secondary criteria to chose among policies which offer the same expected return maximizing the standard J objective function.

In [56] is analyzed the process of selecting a portfolio, in particular the rational rules that the investor should use. One of these rules considers the expected return a desirable property while its variance is an unwanted feature. There is a rate at which the investor can gain expected return by increasing variance, or reduce variance by giving up expected return.

To show this intuitive concept [56] introduces the notion of the (E,V)–space in which each portfolio can be represented as one point by its Expected return and its Variance under certain probability beliefs. Plotting these E,V pairs in this space allows to identify the Pareto frontier, in this way an agent can choose among the efficient portfolios.

A linear programming approach is used in [22] to solve problems in which an agent with k limited consumable resources needs policies that attempt to achieve good performance while respecting these limitations. This is possible using a constrained MDP (CMDP) [4] which is a standard MDP (see Section 2.2) where the resources are not explicitly modeled, but rather are treated as constraints that are imposed on the space of feasible solutions.

Besides the reward function $\mathcal{R}(s_t, a_t)$, the agent obtains also informations from the environment through the cost function $C : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^k$, which defines a vector of consumed units per resource when the action $a_t \in \mathcal{A}$ is executed in state $s_t \in \mathcal{S}$. The risk lies in the probability of violating the resource constrains.

In worst case control [19, 32] the α–value is the main concept: it denotes a lower bound for the return that can be reached at least with probability α. The minimax criterion is a special case of α–value criterion useful to optimize the worst possible outcome of the return or to guarantee that the return will not exceed a given threshold with probability 1.

When risk sensitive control is based on the usage of exponential utility [35, 47] the objective is to maximize an exponential function of the return which depends on a parameter used to penalize or enforce policies with a high variance:

$$U(V) = -sign(c)\, e^{-cV}$$

where U is the utility function, V is the expected return and c is the risk aversion coefficient of the agent. In [59] is used the Chernoff functional as risk–aware objective in which the total return appears as an exponential utility.

In [25] the risk is represented by special states in the MDP which are considered dangerous. The objective is to find a policy that maximizes the return while maintains the probability of entering a risky state below a certain threshold. This goal is achieved first introducing a risk function $\bar{\mathcal{R}}(\boldsymbol{z})$, similar to the standard reward function, which has the purpose of

pointing out to the agent what are the undesirable states, then a "risk–utility" function is estimate as the normal utility $V^{\pi}(s)$ (see Equation 2.1):

$$\rho^{\pi}(s) = \mathop{\mathbb{E}}_{\substack{a_j \sim \pi \\ s_j \sim \mathcal{P}}} \left[\left(\sum_{j=0}^{T} \bar{\gamma}^j \bar{\mathcal{R}}(\boldsymbol{z}_j) \right) \middle| s_0 = s \right].$$

Another possible implementation of a risk–averse algorithm is proposed in [51] by penalizing each reward $\mathcal{R}(\boldsymbol{z}_t)$ with its standard deviation obtaining the adjusted reward $\mathcal{R}^c(\boldsymbol{z}_t) = \mathcal{R}(\boldsymbol{z}_t) - c\,\sigma(\mathcal{R}(\boldsymbol{z}_t))$, where c is a risk averse coefficient.

In [57] the temporal difference error δ is transformed with a parametric function that allows to choose between high and low variance policies. By tuning a scalar parameter $\kappa \in (-1, 1)$ is possible to specify the desired risk–sensitivity:

$$\delta = \begin{cases} (1-\kappa)\,\delta & \text{if } \delta \geq 0, \\ (1+\kappa)\,\delta & \text{otherwise.} \end{cases}$$

After this transformation [57] applies the standard critic–only update to the estimated value function

$$\hat{V}^{\pi}_{h+1}(s) = \hat{V}^{\pi}_{h}(s) + \alpha_h \delta.$$

2.4.1 Variance related measures

As previously described, many different risk metrics have been considered over time, but for sure the most common approach is to use, as general objective, a risk–averse criterion such as the expected exponential utility [35], a variance related measure or the percentile performance [94].

When a measure of risk is included in an optimality criterion, the corresponding optimal policy is usually no longer Markovian stationary and even when the MDP's parameters are known, many of these problems are computationally intractable, and some are not even approximable [24]. For example both mean–variance optimization and percentile optimization for MDPs have been shown to be NP–hard in general. It seems to be difficult to find an optimization objective which correctly models our intuition of risk awareness. Even though expectation, variance and percentile levels relate to risk awareness, optimizing them directly can lead to counterintuitive policies as illustrated in [20, 54, 59].

In the last years the percentile performance or the CVaR (Conditional Value at Risk) criterion has taken a prominent place in many studies [1, 7,

15, 42, 61, 62, 85]. Here the main idea is to optimize the distribution of the random variable using estimation of its percentiles.

However this thesis is based to another common measure of risk which is the variance of the return, introduced by Sobel in [37].

Considering the total return as the random variable x with cumulative distribution $F(x)$, its m–moment is defined as $V_{(m)} = \int_{-\infty}^{\infty} x^m \mathrm{d}F(x)$. In this way the standard evaluation measure V (see Equation 2.1) is the first moment $V_{(1)}$ of the return and its variance is $V_{(2)} - V_{(1)}^2$.

In order to simplify the rest of the text the following notation will be used:

$$W^{\pi}(s) = \mathop{\mathbb{E}}_{\substack{a_j \sim \pi \\ s_j \sim \mathcal{P}}} \left[\left(\sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j) \right)^2 \middle| s_0 = s \right]$$

is the second moment $V_{(2)}$ of the total return and the corresponding policy evaluation measure is

$$M(\pi) = \int_{\mathcal{S}} D(s) W^{\pi}(s) \, \mathrm{d}s = \int_{\mathbb{T}} p(\tau|\pi) \, \mathcal{R}(\tau)^2 \, \mathrm{d}\tau. \tag{2.8}$$

The same is done for the variance, where

$$\Lambda^{\pi}(s) = VAR \left[\sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j) \middle| s_0 = s \right]$$

and

$$Var(\pi) = M(\pi) - (J(\pi))^2 .$$

Note that $VAR[\cdot]$ denotes the statistical measure of the variance of a random variable, while $Var(\pi)$ indicates the estimate of the variance of the total return induced by the policy π and the distribution D.

Typical performance criteria that include the variance as risk penalization are:

1. Maximize $J(\pi)$ s.t. $Var(\pi) \leq b$ (Variance constrained return)
2. Minimize $Va(\pi)r$ s.t. $J(\pi) \geq b$
3. Maximize $\frac{J(\pi)}{\sqrt{Var(\pi)}}$ (Sharpe Ratio)
4. Maximize $J(\pi) - c\sqrt{Var(\pi)}$ or $J(\pi) - c\,Var(\pi)$ (mean–variance)

Sobel proposed a Bellman equation of Λ^π, but being a non linear function it lacks of the monotonicity property of dynamic programming and thus it cannot be optimize using standard DP methods. However, in order to use the variance in a risk averse criterion, there is no need to compute its exact value, but is sufficient to have just an accurate estimation. This is a task suitable for RL techniques, in particular Actor–only [17, 60] or Actor–Critic [2, 86, 84] methods (see Section 2.3). Critic–Only algorithms [77, 83] were proposed only for the policy evaluation phase since it is not yet clear how to develop also the policy improvement step.

In [17] is presented an on–policy actor–only algorithm which updates the policy parameters w.r.t. variance related criteria such as the Sharpe Ratio or Variance constrained return. This method estimates the expected values of $J(\pi)$ and $Var(\pi)$ using exponential moving averages of the total undiscounted returns of each trajectories $\mathcal{R}(\tau)$. It also computes the gradient $\nabla_{\boldsymbol{\theta}} J(\pi)$ with the likelihood ratio technique as illustrated in the following pseudocode.

Algorithm 1: Pseudocode of the Actor–only algorithm used in [17]

output: $\boldsymbol{\theta}^*$

initialize $\boldsymbol{\theta}_0, J_0, Var_0$;
for $h \leftarrow 0$ **to** $\boldsymbol{\theta}_h$ *converges* **do**
  perform a trial and obtain $\tau_h := \{s_t, a_t\}_{t=0}^{T}$;
  set $\mathcal{R}(\tau_h) \leftarrow 0$;
  for $t \leftarrow 0$ **to** T **do**
    $\mathcal{R}(\tau_h) \leftarrow \mathcal{R}(\tau_h) + \mathcal{R}(s_t, a_t)$;
  end
  $J_{h+1} \leftarrow J_h + \alpha_h(\mathcal{R}(\tau_h) - J_h)$;
  $Var_{h+1} \leftarrow Var_h + \alpha_h \left((\mathcal{R}(\tau_h))^2 - J_h^2 - Var_h \right)$;
  $\nabla_{\boldsymbol{\theta}} J \leftarrow$ Equation 2.5;
  $\nabla \mathcal{J}^R \leftarrow function\,(J_h, Var_h, \nabla_{\boldsymbol{\theta}} J_h, \mathcal{R}(\tau_h))$;
  $\boldsymbol{\theta}_{h+1} \leftarrow \boldsymbol{\theta}_h + \beta_h \nabla_{\boldsymbol{\theta}} \mathcal{J}^R$;
end
return $\boldsymbol{\theta}$;

Chapter 3

Off–Policy Gradients

All the measures involved in actor–only methods have been presented in Sections 2.2 and 2.3.3 in on–policy fashion. The main goals of this chapter are to redefine them in the off–policy context and to formalize the differences between REINFORCE and GPOMDP approaches.

Furthermore, three types of baselines are introduced (one for REINFORCE and two for GPOMDP method) in order to improve the performance of the algorithms. All these concepts are the starting point for the considerations presented in the next chapter.

3.1 Expected return J

In this section we investigate the formulation of the expected return. We provide the connection between the trajectory–based definition and the step–based one. In particular we show that the Markov chain induced by the policy can be expressed by the repeated application of a kernel.

The derivation is shown in the on–policy and off–policy scenario.

Expected return $J(\pi)$ in on–policy settings: from expectation over trajectories to expectation over state–action pairs.

$$
\begin{aligned}
J(\pi) &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\boldsymbol{\theta})}[\mathcal{R}(\tau)] \quad \text{(see Equation 2.4)} \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi(a_0|s_0;\boldsymbol{\theta}) \int_{s_1} \mathcal{P}\left(s_1|s_0,a_0\right) \int_{a_1} \pi(a_1|s_1;\boldsymbol{\theta}) \cdot \ldots \cdot \\
&\quad \cdot \int_{s_2} \mathcal{P}\left(s_2|s_1,a_1\right) \int_{s_T} \mathcal{P}\left(s_T|s_{T-1},a_{T-1}\right) \int_{a_T} \pi(a_T|s_T;\boldsymbol{\theta})\, \mathcal{R}(\boldsymbol{z}_{0:T}) \\
&\quad (\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}s_2, \mathrm{d}a_1, \mathrm{d}s_1, \mathrm{d}a_0, \mathrm{d}s_0) \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi(\boldsymbol{z}_0;\boldsymbol{\theta}) \left\{ \prod_{k=1}^{T} \underbrace{\int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi(\boldsymbol{z}_k;\boldsymbol{\theta})}_{\text{on–policy kernel}} \right\} \mathcal{R}(\boldsymbol{z}_{0:T}) \\
&\quad (\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0)\,.
\end{aligned}
$$

Expected return $J(\pi)$ in off–policy settings ($\pi \leftarrow \pi^{\mathcal{T}}$): from expectation over trajectories to expectation over state–action pairs.

$$
\begin{aligned}
J\left(\pi^{\mathcal{T}}\right) &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{T}})}[\mathcal{R}(\tau)] = \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\frac{p\left(\cdot|\pi^{\mathcal{T}}\right)}{p\left(\cdot|\pi^{\mathcal{B}}\right)}\mathcal{R}(\tau)\right] = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}[\omega(\tau)\mathcal{R}(\tau)] \quad \text{(see Equation 2.6)} \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(a_0|s_0) \frac{\pi^{\mathcal{T}}(a_0|s_0)}{\pi^{\mathcal{B}}(a_0|s_0)} \cdot \\
&\quad \cdot \int_{s_1} \mathcal{P}\left(s_1|s_0,a_0\right) \int_{a_1} \pi^{\mathcal{B}}(a_1|s_1) \frac{\pi^{\mathcal{T}}(a_1|s_1)}{\pi^{\mathcal{B}}(a_1|s_1)} \cdot \ldots \cdot \\
&\quad \cdot \int_{s_T} \mathcal{P}\left(s_T|s_{T-1},a_{T-1}\right) \int_{a_T} \pi^{\mathcal{B}}(a_T|s_T) \frac{\pi^{\mathcal{T}}(a_T|s_T)}{\pi^{\mathcal{B}}(a_T|s_T)} \cdot \\
&\quad \cdot \mathcal{R}(\boldsymbol{z}_{0:T})\, (\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_1, \mathrm{d}s_1, \mathrm{d}a_0, \mathrm{d}s_0) \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0) \left\{ \prod_{k=1}^{T} \underbrace{\int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)}_{\text{off–policy kernel}} \right\} \cdot \\
&\quad \cdot \mathcal{R}(\boldsymbol{z}_{0:T})\, (\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0)\,.
\end{aligned}
$$

3.2 REINFORCE

Given the formulation of the expected return in off policy settings, it easy to derive the REINFORCE formulation of the gradient (see Equation 2.7). For the sake of simplicity we report the formulation here:

Definition 1.

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\mathcal{R}(\tau)\omega(\tau)\nabla_{\boldsymbol{\theta}}\log p(\tau|\pi^{\mathcal{T}})\right] \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].\end{aligned}$$

3.2.1 Component-dependent Baseline

As mentioned in 2.3.3 the policy gradient is a K–dimensional vector and for each component k it is possible to compute the baseline b_k which minimizes the variance of the estimation. This concept can be applied both in on–policy and off–policy learning, for this reason the performance of the previous gradient estimate can be improved as follows.

$$\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j) - b_k\right)\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

Let

$$F^{(\tau)} = \mathcal{R}(\tau) = \sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)$$

$$G_k^{(\tau)} = \omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).$$

Then the variance of the k–th component of the gradient is

$$\begin{aligned}&\mathtt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(\tau)} - b_k\right)G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(F^{(\tau)} - b_k\right)G_k^{(\tau)}\right]\right\}^2 \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(\tau)} - b_k\right)G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}G_k^{(\tau)}\right]\right\}^2,\end{aligned}$$

because $\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[b_k G_k^{(\tau)}\right] = 0.$

$$\texttt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{F^{(\tau)}G_k^{(\tau)} - b_k G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}G_k^{(\tau)}\right]\right\}^2$$

$$= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(F^{(\tau)}\right)^2\left(G_k^{(\tau)}\right)^2 - 2b_k F^{(\tau)}\left(G_k^{(\tau)}\right)^2 + (b_k)^2\left(G_k^{(\tau)}\right)^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}G_k^{(\tau)}\right]\right\}^2.$$

Minimizing previous equation w.r.t. b_k we get

$$2b_k \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(G_k^{(\tau)}\right)^2\right] - 2\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}\left(G_k^{(\tau)}\right)^2\right] = 0$$

$$b_k = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}\left(G_k^{(\tau)}\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(G_k^{(\tau)}\right)^2\right]} = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\left(\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}.$$

3.3 GPOMDP

The GPOMDP algorithm (see Section 2.3.2) distributes the utility of the reward obtained at time t to only the actions performed before time t. This reward causality can be expressed modifying the REINFORCE gradient expression (see Equation 2.5).

The off–policy gradient estimate in GPOMDP–like settings is given by

Definition 2.

$$\nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\sum_{t=0}^{T} \gamma^t \mathcal{R}(\boldsymbol{z}_t) \omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) \right].$$

Let

$$\begin{aligned}
A_{0:T} &= \left(\sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) \right) \left(\sum_{j=0}^{T} \gamma^j \mathcal{R}(\boldsymbol{z}_j) \right) \\
&= \left(\sum_{i=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T) \right) \left(\sum_{j=0}^{T-1} \gamma^j \mathcal{R}(\boldsymbol{z}_j) + \gamma^T \mathcal{R}(\boldsymbol{z}_T) \right) \\
&= A_{0:T-1} + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T) \sum_{j=0}^{T-1} \gamma^j \mathcal{R}(\boldsymbol{z}_j) + \gamma^T \mathcal{R}(\boldsymbol{z}_T) \sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) \\
&= A_{0:T-1} + A_T = \sum_{t=0}^{T} A_t
\end{aligned}$$

then, with an abuse of notation, the gradient without baseline is given by

$$\nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\omega(\tau) \sum_{t=0}^{T} A_t \right] = \sum_{t=0}^{T} \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\omega(\tau) A_t \right].$$

Split the expectation and consider the single term:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)A_t\right] = \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0) \left\{\prod_{k=1}^{T} \int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad A_t\left(\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0) \left\{\prod_{k=1}^{t} \int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad \left(\nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_t) \sum_{j=0}^{t-1} \gamma^j \mathcal{R}(\boldsymbol{z}_j) + \gamma^t \mathcal{R}(\boldsymbol{z}_t) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right) \\
&\quad \left\{\prod_{k=t+1}^{T} \int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \left(\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0) \left\{\prod_{k=1}^{t} \int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad \left(\nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_t) \sum_{j=0}^{t-1} \gamma^j \mathcal{R}(\boldsymbol{z}_j) + \gamma^t \mathcal{R}(\boldsymbol{z}_t) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right) \\
&\quad \left(\mathrm{d}a_t, \mathrm{d}s_t, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right) \int_{a_0} \pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0) \left\{\prod_{k=1}^{t} \int_{s_k} \mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right) \int_{a_k} \pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad \gamma^t \mathcal{R}(\boldsymbol{z}_t) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) \left(\mathrm{d}a_t, \mathrm{d}s_t, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \mathop{\mathbb{E}}_{\tau_{0:t}\sim p(\cdot|\pi^{\mathcal{B}})} \left[\omega(\tau_{0:t})\gamma^t \mathcal{R}(\boldsymbol{z}_t) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right] \\
&= \mathop{\mathbb{E}}_{\tau_{0:t}\sim p(\cdot|\pi^{\mathcal{B}})} \left[\left(\prod_{w=0}^{t} \frac{\pi^{\mathcal{T}}(\boldsymbol{z}_w)}{\pi^{\mathcal{B}}(\boldsymbol{z}_w)}\right) \gamma^t \mathcal{R}(\boldsymbol{z}_t) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right]
\end{aligned}
$$

given that in

$$\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\sum_{j=0}^{t-1}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right]=$$
$$=\int_{s_0}D\left(s_0\right)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{t-1}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\}$$
$$\cdot\sum_{j=0}^{t-1}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\int_{s_t}\mathcal{P}\left(s_t|\boldsymbol{z}_{t-1}\right)\int_{a_t}\pi^{\mathcal{B}}(\boldsymbol{z}_t)\omega(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)$$
$$(\mathrm{d}a_t,\mathrm{d}s_t,\ldots,\mathrm{d}a_0,\mathrm{d}s_0)$$

the term

$$\int_{a_t}\pi^{\mathcal{B}}(\boldsymbol{z}_t)\omega(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t=\int_{a_t}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t$$
$$=\int_{a_t}\nabla_{\boldsymbol{\theta}}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t=\nabla_{\boldsymbol{\theta}}\int_{a_t}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t=\nabla_{\boldsymbol{\theta}}1=0.$$

The rest of the section is devoted to the derivation of the baseline terms.

3.3.1 Component–dependent Baseline (step baseline)

The first formulation we analyze is the (step,component)–based version of the baseline [68].

$$\nabla_{\boldsymbol{\theta}_k}J\left(\pi^{\mathcal{T}}\right)=\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\left(\gamma^t\mathcal{R}(\boldsymbol{z}_t)-b_k^{(t)}\right)\omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right]$$
$$=\sum_{t=0}^{T}\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\gamma^t\mathcal{R}(\boldsymbol{z}_t)-b_k^{(t)}\right)\omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

The aim is to minimize the variance ($\mathop{\mathtt{Var}}_{t}$) of each t–th term of this sum. Let

$$F^{(t)}=\gamma^t\mathcal{R}(\boldsymbol{z}_t)$$
$$G_k^{(t)}=\omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).$$

Then the variance of the k–th component of the gradient is

$$\begin{aligned}
&\underset{t}{\mathtt{Var}}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(t)} - b_k^{(t)}\right) G_k^{(t)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(F^{(t)} - b_k^{(t)}\right) G_k^{(t)}\right]\right\}^2 \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(t)} - b_k^{(t)}\right) G_k^{(t)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(t)} G_k^{(t)}\right]\right\}^2,
\end{aligned}$$

because $\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[b_k^{(t)} G_k^{(t)}\right] = 0.$

$$\begin{aligned}
\underset{t}{\mathtt{Var}}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{F^{(t)} G_k^{(t)} - b_k^{(t)} G_k^{(t)}\right\}^2\right] \\
- \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(t)} G_k^{(t)}\right]\right\}^2
\end{aligned}$$

$$\begin{aligned}
= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(F^{(t)}\right)^2 \left(G_k^{(t)}\right)^2 - 2 b_k^{(t)} F^{(t)} \left(G_k^{(t)}\right)^2 + \left(b_k^{(t)}\right)^2 \left(G_k^{(t)}\right)^2\right] \\
- \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(t)} G_k^{(t)}\right]\right\}^2.
\end{aligned}$$

Minimizing previous equation w.r.t. $b_k^{(t)}$ we get

$$2 b_k^{(t)} \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(G_k^{(t)}\right)^2\right] - 2 \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(t)} \left(G_k^{(t)}\right)^2\right] = 0$$

$$\begin{aligned}
b_k^{(t)} &= \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(t)} \left(G_k^{(t)}\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(G_k^{(t)}\right)^2\right]} \\
&= \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\gamma^t \mathcal{R}(\boldsymbol{z}_t) \left(\omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}_k} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}_k} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}.
\end{aligned}$$

3.3.2 Component–dependent Baseline (single baseline)

A simpler baseline can be obtained by considering a single baseline for each component of the gradient, as done for REINFORCE. As a consequence the gradient is given by

$$\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\left(\gamma^t \mathcal{R}(\boldsymbol{z}_t) - b_k\right)\omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

Let

$$F^{(t)} = \gamma^t \mathcal{R}(\boldsymbol{z}_t)$$
$$G_k^{(t)} = \omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).$$

Then the variance of the k–th component of the gradient is

$$\begin{aligned}
&\mathtt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\sum_{t=0}^{T}\left(F^{(t)} - b_k\right)G_k^{(t)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\left(F^{(t)} - b_k\right)G_k^{(t)}\right]\right\}^2 \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\sum_{t=0}^{T}\left(F^{(t)} - b_k\right)G_k^{(t)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}F^{(t)}G_k^{(t)}\right]\right\}^2,
\end{aligned}$$

because $\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[b_k\sum_{t=0}^{T}G_k^{(t)}\right] = 0.$

$$\begin{aligned}
&\mathtt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\
&= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\sum_{t=0}^{T}F^{(t)}G_k^{(t)} - b_k\sum_{t=0}^{T}G_k^{(t)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}F^{(t)}G_k^{(t)}\right]\right\}^2.
\end{aligned}$$

Minimizing previous equation w.r.t. b_k we get

$$2b_k \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T}G_k^{(t)}\right)^2\right] - 2\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T}F^{(t)}G_k^{(t)}\right)\left(\sum_{t=0}^{T}G_k^{(t)}\right)\right] = 0$$

$$b_k = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T}F^{(t)}G_k^{(t)}\right)\left(\sum_{t=0}^{T}G_k^{(t)}\right)\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T}G_k^{(t)}\right)^2\right]}.$$

Chapter 4

Off–line risk–sensitive RL

The previous chapter has presented all the modifications needed to adapt the standard policy–based methods to the off–policy context. Now the scenario becomes more complex: besides the off–policy component , we will show how an actor–only algorithm can optimize a risk measure, which can juxtapose, or even replace, the standard RL criterion.

In many real problems an agent cannot explore a risky environment trying all the possible behaviors in order to well–estimate their outcomes. The main reasons are two:

1. The state and action spaces are not finite sets.

2. Many policies may damage the agent itself or waste limited goods associated to some kind of utility (e.g., money, reputation, consumable resources).

At the same time in many scenarios, e.g., in finance or medical field, the agent can rely on a huge historical dataset which collects past interactions between the system and other agents.

These two aspects ask for to devise an off-line learning technique able to face risk in complex problems. Experience of both good and poor strategies helps the agent to learn its own policy, which respects the desired risk–aversion level.

Given $\mathcal{J}^R$, a criterion that is related to some risk measure R, the gradient ascent update (see Equation 2.3) is modified to take into account the new risk function: $\boldsymbol{\theta}_{h+1} = \boldsymbol{\theta}_h + \alpha_h \nabla_{\boldsymbol{\theta}} \mathcal{J}^R|_{\boldsymbol{\theta}=\boldsymbol{\theta}_h}$. In particular the structure of an off-line actor-only algorithm is reported in Algorithm 2.

Algorithm 2: General structure of off–line Actor–only algorithm

input : $nbUpdates$, $BatchData$
output: $\boldsymbol{\theta}^*$

initialize $\boldsymbol{\theta}_0$;
for $h \leftarrow 0$ **to** $nbUpdates$ **do**
  $\nabla_{\boldsymbol{\theta}} \mathcal{J}^R \leftarrow 0$;
  for $Ep \leftarrow 1$ **to** $|BatchData|$ **do**
    extract the Ep–th episode from $BatchData$;
    for $t \leftarrow 0$ **to** T **do**
      use the sample $\langle s_t, a_t, r_{t+1}, s'_{t+1} \rangle_{Ep}$ in $\nabla_{\boldsymbol{\theta}} \mathcal{J}^R$ estimation;
    end
  end
  $\boldsymbol{\theta}_{h+1} \leftarrow \boldsymbol{\theta}_h + \alpha_h \nabla_{\boldsymbol{\theta}} \mathcal{J}^R$;
end
return $\boldsymbol{\theta}$;

4.1 Variance related criteria

Here are derived two policy gradients for the most common measures used in risk–averse application, i.e. the Sharpe Ratio and the mean–variance criterion. The common idea of these two indexes is using the variance as an active penalty term in the objective function. Not only if two investments have the same expected return and different variances, the one with the lower variance is the better choice, but they compare in different ways these two variables.

An abbreviated notation is used to referring to the statistical variables presented in the previous chapters.

In order to well–define these optimization problems $\forall \boldsymbol{\theta} \in \Theta$ the values of $J(\pi_{\boldsymbol{\theta}}), M(\pi_{\boldsymbol{\theta}})$ and $Var(\pi_{\boldsymbol{\theta}})$ must be bounded.

4.1.1 Sharpe Ratio

In finance, the Sharpe ratio measures the excess return per unit of deviation in an investment asset or a trading strategy [78]. In the general case the Sharpe ratio is defined as:

$$SR = \frac{\mu - r_f}{\sigma},$$

where μ is the mean portfolio return, r_f is the risk–free rate, σ is the standard deviation of portfolio return. One intuition of this calculation is that a portfolio engaging in "zero risk" investment, such as the purchase of T–bills (for which the expected return μ is the risk-free rate), has a Sharpe ratio of exactly zero.

However in this thesis we use a simpler version of this index, where the risk–free rate is 0, $\mu = J(\pi)$ and $\sigma = \sqrt{Var(\pi)}$:

$$\mathcal{J}^{SR} = \frac{J}{\sqrt{Var}}.$$

The associated gradient w.r.t. the policy parameters is

$$\nabla_{\boldsymbol{\theta}} \mathcal{J}^{SR} = \frac{1}{\sqrt{Var}} \left(\nabla_{\boldsymbol{\theta}} J - \frac{J}{2Var} \nabla_{\boldsymbol{\theta}} Var \right) \tag{4.1}$$

$$= \frac{1}{\sqrt{M - J^2}} \left(\nabla_{\boldsymbol{\theta}} J - \frac{J}{2(M - J^2)} \left(\nabla_{\boldsymbol{\theta}} M - 2J \nabla_{\boldsymbol{\theta}} J \right) \right) \tag{4.2}$$

$$= \frac{M \nabla_{\boldsymbol{\theta}} J - \frac{1}{2} J \nabla_{\boldsymbol{\theta}} M}{(M - J^2)^{\frac{3}{2}}}. \tag{4.3}$$

Another assumption needed for the Sharpe ratio is that $\forall \boldsymbol{\theta} \in \Theta \; Var(\pi_{\boldsymbol{\theta}}) > 0$.

4.1.2 Mean Variance

The mean variance criterion uses the variance of the return $Var(\pi)$ as a penalty term against the expected return $J(\pi)$ [56]. By looking at the expected return and variance of an asset, investors attempt to make more efficient investment choices–seeking the lowest variance for a given expected return, or seeking the highest expected return for a given variance level.
The objective function is

$$\mathcal{J}^{MV} = J - c \, Var,$$

and the associated gradient w.r.t. the policy parameters is

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \mathcal{J}^{MV} &= \nabla_{\boldsymbol{\theta}} J - c \, \nabla_{\boldsymbol{\theta}} \left(M - J^2 \right) \\ &= \nabla_{\boldsymbol{\theta}} J - c \, \left(\nabla_{\boldsymbol{\theta}} M - 2J \nabla_{\boldsymbol{\theta}} J \right). \end{aligned}$$

It is important to notice that in this simple family of variance related criteria only four variables (J, $\nabla_{\boldsymbol{\theta}} J$, M and $\nabla_{\boldsymbol{\theta}} M$) are enough to define any possible gradient. The formalization of J in off–policy settings is proposed in the second part of Section 3.1, while for $\nabla_{\boldsymbol{\theta}} J$ the REINFORCE and GPOMDP alternatives are available in Sections 3.2 and 3.3, respectively.

In the next section the other two elements, M and $\nabla_{\boldsymbol{\theta}} M$, involved in the gradient expressions are derived.

Obtaining an unbiased estimate of the term $J\nabla_{\boldsymbol{\theta}} J$ from a single trajectory is impossible. To overcome this issue two alternative approaches can be exploited:

1. Use a two timescale algorithm, where estimates of J are calculated on the fast time scale and the gradient $\nabla_{\boldsymbol{\theta}} J$ and the parameters $\boldsymbol{\theta}$ are respectively computed and updated at the end of each episode on a slower time scale.

2. Divide the data in two disjoined sets and compute J independently from $\nabla_{\boldsymbol{\theta}} J$.

The algorithms used for the experiments in Chapter 5 adopt the second option. In each test the number of trajectories used for estimate the gradients $\nabla_{\boldsymbol{\theta}} J$ and $\nabla_{\boldsymbol{\theta}} M$ is indicated as *grad_trajectories/update*, while the number of trajectories used to estimate J and M is *eval_trajectories/update*.

4.2 Second moment M

In this section we provide a deep analysis of the second moment of the policy performance J. We start reporting the definition in off-policy settings and we derive the REINFORCE- and GPOMDP–like gradient formulations. For sake of clarity we restate the formulation of the second moment (see Equation 2.8):

$$M(\pi) = \int_{\mathcal{S}} D(s) W^{\pi}(s)\, \mathrm{d}s = \int_{\mathbb{T}} p(\tau|\pi)\, \mathcal{R}(\tau)^2\, \mathrm{d}\tau$$

The corresponding off–policy formulation is derived applying the importance sampling technique described in Section 2.3.4, with the only difference that now the random variable is $\mathcal{R}(\tau)^2$ and no more $\mathcal{R}(\tau)$.

$$\begin{aligned} M\left(\pi^{\mathcal{T}}\right) &= \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{T}})} \left[\mathcal{R}(\tau)^2\right] \\ &= \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\frac{p\left(\cdot|\pi^{\mathcal{T}}\right)}{p\left(\cdot|\pi^{\mathcal{B}}\right)} \mathcal{R}(\tau)^2\right] = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\pi^{\mathcal{B}})} \left[\omega(\tau) \mathcal{R}(\tau)^2\right]. \end{aligned}$$

4.2.1 REINFORCE

By following the same derivation used for the off–policy gradient, it is easy to obtain the following off–policy formulation of the second moment REINFORCE estimate:

Definition 3.

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} M\left(\pi^{\mathcal{T}}\right) &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\mathcal{R}(\tau)^2\omega(\tau)\nabla_{\boldsymbol{\theta}}\log p(\tau|\pi^{\mathcal{T}})\right] \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].\end{aligned}$$

Component–dependent Baseline

As we have seen in the previous chapter the formulation of the REINFORCE gradient estimate allows to introduce a component-dependent baseline:

$$\nabla_{\boldsymbol{\theta}_k} M\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2 - b_k\right)\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

Let

$$\begin{aligned}F^{(\tau)} &= \mathcal{R}(\tau)^2 = \left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2 \\ G_k^{(\tau)} &= \omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).\end{aligned}$$

Then the baseline is obtained as explained for $\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)$ in Section 3.2.1. The final expression differs only in the definition of $F^{(\tau)}$:

$$b_k = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\left(\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}.$$

4.2.2 GPOMDP

By considering the causality relationship between actions and rewards it is possible to provide a better estimate of the gradient also for the second moment of the policy performance J:

Definition 4.

$$\nabla_{\boldsymbol{\theta}} M\left(\pi^{\mathcal{T}}\right) = \\ = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\gamma^t \mathcal{R}(\boldsymbol{z}_t)\; H_t\; \omega(\tau_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

Let

$$B_{0:T} = \left(\sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)\left(\sum_{j=0}^{T}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)^2$$

$$= \left(\sum_{i=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T)\right) \cdot$$

$$\cdot \left(\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j) + \gamma^T \mathcal{R}(\boldsymbol{z}_T)\right)^2$$

$$= \left(\sum_{i=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i) + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T)\right) \cdot$$

$$\cdot \left(\left(\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)^2 + \left(\gamma^T \mathcal{R}(\boldsymbol{z}_T)\right)^2 + 2\gamma^T \mathcal{R}(\boldsymbol{z}_T)\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)$$

$$= B_{0:T-1} + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T)\left(\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)^2 +$$

$$+ \sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\left(\left(\gamma^T \mathcal{R}(\boldsymbol{z}_T)\right)^2 + 2\gamma^T \mathcal{R}(\boldsymbol{z}_T)\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)$$

$$= B_{0:T-1} + \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_T)\left(\sum_{j=0}^{T-1}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)^2 +$$

$$+ \gamma^T \mathcal{R}(\boldsymbol{z}_T)\sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\left(2\sum_{j=0}^{T}\gamma^j \mathcal{R}(\boldsymbol{z}_j) - \gamma^T \mathcal{R}(\boldsymbol{z}_T)\right)$$

$$= B_{0:T-1} + B_T = \sum_{t=0}^{T} B_t$$

then, with an abuse of notation, the gradient without baseline is given by

$$\nabla_{\boldsymbol{\theta}} M\left(\pi^{\mathcal{T}}\right) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)\sum_{t=0}^{T} B_t\right] = \sum_{t=0}^{T}\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)B_t\right]$$

Split the expectation and consider the single term

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)B_t\right] = \\
&= \int_{s_0} D\left(s_0\right)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{T}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad B_t\left(\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{t}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad B_t\left\{\prod_{k=t+1}^{T}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\}\left(\mathrm{d}a_T, \mathrm{d}s_T, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{t}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad B_t\left(\mathrm{d}a_t, \mathrm{d}s_t, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \int_{s_0} D\left(s_0\right)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{t}\int_{s_k}\mathcal{P}\left(s_k|\boldsymbol{z}_{k-1}\right)\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\} \\
&\quad \gamma^t\mathcal{R}(\boldsymbol{z}_t)\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\left(2\sum_{j=0}^{t}\gamma^j\mathcal{R}(\boldsymbol{z}_j) - \gamma^t\mathcal{R}(\boldsymbol{z}_t)\right) \\
&\quad \left(\mathrm{d}a_t, \mathrm{d}s_t, \ldots, \mathrm{d}a_0, \mathrm{d}s_0\right) \\
&= \mathop{\mathbb{E}}_{\tau_{0:t}\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau_{0:t})\,\gamma^t\mathcal{R}(\boldsymbol{z}_t)\sum_{i=0}^{t} H_t\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right]
\end{aligned}$$

where $H_t = \left(2\sum_{j=0}^{t}\gamma^j\mathcal{R}(\boldsymbol{z}_j) - \gamma^t\mathcal{R}(\boldsymbol{z}_t)\right)$,

given that in

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\omega(\tau)\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\left(\sum_{j=0}^{t-1}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\right]\\
&=\int_{s_0}D(s_0)\int_{a_0}\pi^{\mathcal{B}}(\boldsymbol{z}_0)\omega(\boldsymbol{z}_0)\left\{\prod_{k=1}^{t-1}\int_{s_k}\mathcal{P}(s_k|\boldsymbol{z}_{k-1})\int_{a_k}\pi^{\mathcal{B}}(\boldsymbol{z}_k)\omega(\boldsymbol{z}_k)\right\}\cdot\\
&\quad\cdot\left(\sum_{j=0}^{t-1}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\int_{s_t}\mathcal{P}(s_t|\boldsymbol{z}_{t-1})\int_{a_t}\pi^{\mathcal{B}}(\boldsymbol{z}_t)\omega(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\\
&\quad(\mathrm{d}a_t,\mathrm{d}s_t,\ldots,\mathrm{d}a_0,\mathrm{d}s_0)
\end{aligned}$$

the term

$$\begin{aligned}
\int_{a_t}\pi^{\mathcal{B}}(\boldsymbol{z}_t)\omega(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t&=\int_{a_t}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t\\
&=\int_{a_t}\nabla_{\boldsymbol{\theta}}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t=\nabla_{\boldsymbol{\theta}}\int_{a_t}\pi^{\mathcal{T}}(\boldsymbol{z}_t)\,\mathrm{d}a_t=\nabla_{\boldsymbol{\theta}}1=0
\end{aligned}$$

Component–dependent Baseline (step baseline)

As done for the first moment, the first formulation we analyze is the (step,component)–based version of the baseline:

$$\begin{aligned}
&\nabla_{\boldsymbol{\theta}_k}M\left(\pi^{\mathcal{T}}\right)=\\
&=\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\left(\gamma^t\mathcal{R}(\boldsymbol{z}_t)\,H_t\,-b_k^{(t)}\right)\omega(\tau_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right]\\
&=\sum_{t=0}^{T}\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\gamma^t\mathcal{R}(\boldsymbol{z}_t)\,H_t\,-b_k^{(t)}\right)\omega(\tau_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].
\end{aligned}$$

The aim is to minimize the variance ($\mathop{\mathtt{Var}}_{t}$) of each t–th term of this sum.
Let

$$\begin{aligned}
F^{(t)}&=\gamma^t\mathcal{R}(\boldsymbol{z}_t)\,H_t=\gamma^t\mathcal{R}(\boldsymbol{z}_t)\left(2\sum_{j=0}^{t}\gamma^j\mathcal{R}(\boldsymbol{z}_j)-\gamma^t\mathcal{R}(\boldsymbol{z}_t)\right)\\
G_k^{(t)}&=\omega(\boldsymbol{z}_{0:t})\sum_{i=0}^{t}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).
\end{aligned}$$

Then the baseline is obtained as explained for $\nabla_{\boldsymbol{\theta}_k}J\left(\pi^{\mathcal{T}}\right)$ in Section 3.3.1.

The final expression differs only in the definition of $F^{(t)}$:

$$b_k^{(t)} = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\gamma^t \mathcal{R}(\boldsymbol{z}_t)\, H_t \left(\omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}.$$

Component–dependent Baseline (single baseline)

As done for the first moment, we also consider a baseline that does not depend on the specific time step. We exploit the complete derivation reported in Section 3.3.2 by reporting here only the differences. The gradient estimate with baseline is given by:

$$\begin{aligned}&\nabla_{\boldsymbol{\theta}_k} M\left(\pi^{\mathcal{T}}\right) = \\ &= \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\sum_{t=0}^{T}\left(\gamma^t \mathcal{R}(\boldsymbol{z}_t)\, H_t \; - b_k\right) \omega(\tau_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}_k} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].\end{aligned}$$

Let

$$\begin{aligned}F^{(t)} &= \gamma^t \mathcal{R}(\boldsymbol{z}_t)\, H_t = \gamma^t \mathcal{R}(\boldsymbol{z}_t)\left(2\sum_{j=0}^{t} \gamma^j \mathcal{R}(\boldsymbol{z}_j) - \gamma^t \mathcal{R}(\boldsymbol{z}_t)\right)\\ G_k^{(t)} &= \omega(\boldsymbol{z}_{0:t}) \sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}_k} \log \pi^{\mathcal{T}}(\boldsymbol{z}_i).\end{aligned}$$

Then the baseline is obtained as explained for $\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)$ in Section 3.3.2. The final expression differs only in the definition of $F^{(t)}$:

$$b_k = \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T} F^{(t)} G_k^{(t)}\right)\left(\sum_{t=0}^{T} G_k^{(t)}\right)\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{t=0}^{T} G_k^{(t)}\right)^2\right]}.$$

4.3 Modified Gradient

To conclude this chapter we present another source of risk, originated from the use of off-line batch data as training dataset.

In on–line learning the number of available samples increases as the learning process goes on. Unfortunately only the most recently extracted trajectories are evaluated to estimate the gradient for the next update, because they are representative of the current policy performance. All the samples already used in older updates can be either memorized in the dataset or discarded, since they do not contain useful informations to improve the actual policy and no other update will take them into account.

Although this is a waste of resources and time, it is a good property exploited in on–policy algorithms for gradually improving the policy parametrization towards the optimal one.

When the experience comes from a fixed off-line dataset, the further the agent moves its target policy away from the behavior one, the less reliable information can be extracted. This is a form of risk caused by the difficult choice between staying close to the behavior policy in order to have a good estimation of the expected performance and trying to reach a new policy without knowing its real effect on the system. In this context it seems clear that not necessarily an "improved" policy performs better than the previous ones, simply because in the evaluation dataset there are not enough samples similar to the ones that are likely to be generated from the new policy.

The agent cannot blindly trust the learning process as described so far, so the risk–sensitive criterion must be redesigned to penalize policies very different from the one used to interact with the environment.

In other words the gradient $\nabla_{\boldsymbol{\theta}} J$ must be followed only if the future performance can be estimated from the batch dataset with enough precision, otherwise the wisest choice would be not moving from the current policy.

This goal can be achieved considering as measure of variability the sample variance of the off–policy return $\texttt{Var}\left(J\left(\pi^{\mathcal{T}}\right)\right)$, defined as follows.
Given a batch dataset $\mathcal{D}$ with $|\mathcal{D}| = n$ trajectories, the sample variance is

$$\begin{aligned}\texttt{Var}\left(J\left(\pi^{\mathcal{T}}\right)\right) &= \frac{1}{n} VAR\left[w(\tau)\mathcal{R}(\tau)\right] \\ &= \frac{1}{n}\left(\mathop{\mathbb{E}}_{\tau\in\mathcal{D}}\left[w^2(\tau)\mathcal{R}^2(\tau)\right] - \left(\mathop{\mathbb{E}}_{\tau\in\mathcal{D}}\left[w(\tau)\mathcal{R}(\tau)\right]\right)^2\right) \\ &= \frac{1}{n}\left(\underbrace{\mathop{\mathbb{E}}_{\tau\in\mathcal{D}}\left[w^2(\tau)\mathcal{R}^2(\tau)\right]}_{M(J(\tau))} - \left(J\left(\pi^{\mathcal{T}}\right)\right)^2\right).\end{aligned}$$

The risk–averse criterion is given by $\mathcal{J}^R(\pi^{\mathcal{T}}) = J\left(\pi^{\mathcal{T}}\right) - p\ \texttt{Var}\left(J\left(\pi^{\mathcal{T}}\right)\right)$, and the associated gradient w.r.t. the policy parameters is

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{J}^R(\pi^{\mathcal{T}}) &= \nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) - p\,\nabla_{\boldsymbol{\theta}}\,\texttt{Var}\left(J\left(\pi^{\mathcal{T}}\right)\right) \\ &= \nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right) - \frac{p}{n}\left(\nabla_{\boldsymbol{\theta}} M\left(J\left(\tau\right)\right) - 2J\left(\pi^{\mathcal{T}}\right)\nabla_{\boldsymbol{\theta}} J\left(\pi^{\mathcal{T}}\right)\right),\end{aligned}$$

where $p \in \mathbb{R}_{\geq 0}$ is the penalty coefficient.

The only elements to define are $\nabla_{\boldsymbol{\theta}} M\left(J\left(\tau\right)\right)$ and the associated baseline used to reduce the variability of the estimation.

In the following section we present the derivation of the gradient in REINFORCE–scenario. Unfortunately, the quadratic dependence on the importance sampling terms prevents the definition of a GPOMDP–like version.

4.3.1 REINFORCE

The REINFORCE–like estimation is obtained as follows:

$$\begin{aligned}&\nabla_{\boldsymbol{\theta}} M\left(J\left(\tau\right)\right) = \nabla_{\boldsymbol{\theta}} \int p\left(\tau|\pi^{\mathcal{B}}\right)\left(\frac{p\left(\tau|\pi^{\mathcal{T}}\right)}{p\left(\tau|\pi^{\mathcal{B}}\right)}\right)^2 \mathcal{R}(\tau)^2 \mathrm{d}\tau \\ &= 2\int p\left(\tau|\pi^{\mathcal{B}}\right)\frac{p\left(\tau|\pi^{\mathcal{T}}\right)}{p\left(\tau|\pi^{\mathcal{B}}\right)}\frac{\nabla_{\boldsymbol{\theta}} p\left(\tau|\pi^{\mathcal{T}}\right)}{p\left(\tau|\pi^{\mathcal{B}}\right)}\mathcal{R}(\tau)^2 \mathrm{d}\tau \\ &= 2\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\mathcal{R}(\tau)^2\omega^2(\tau)\nabla_{\boldsymbol{\theta}}\log p(\tau|\pi^{\mathcal{T}})\right] \\ &= 2\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\left(\prod_{j=0}^{T}\omega^2(\boldsymbol{z}_j)\right)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].\end{aligned}$$

Component–dependent Baseline

The formulation of the REINFORCE gradient estimate allows to introduce a component-dependent baseline:

$$\nabla_{\boldsymbol{\theta}_k} M\left(\pi^{\mathcal{T}}\right) = \\ = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\left(\sum_{j=0}^{T}\gamma^j \mathcal{R}(\boldsymbol{z}_j)\right)^2 - \frac{b_k}{\omega(\tau)}\right)\omega^2(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right].$$

Let

$$F^{(\tau)} = \mathcal{R}^2(\tau) = \left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2$$

$$G_k^{(\tau)} = \omega^2(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i).$$

Then the variance of the k–th component of the gradient is

$$\mathtt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\ = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(\tau)} - \frac{b_k}{\omega(\tau)}\right)G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(F^{(\tau)} - \frac{b_k}{\omega(\tau)}\right)G_k^{(\tau)}\right]\right\}^2 \\ = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{\left(F^{(\tau)} - \frac{b_k}{\omega(\tau)}\right)G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}G_k^{(\tau)}\right]\right\}^2,$$

because $\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\frac{b_k}{\omega(\tau)}G_k^{(\tau)}\right] = 0.$

$$\mathtt{Var}\left(\nabla_{\boldsymbol{\theta}_k} J\left(\pi^{\mathcal{T}}\right)\right) = \\ = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left\{F^{(\tau)}G_k^{(\tau)} - \frac{b_k}{\omega(\tau)}G_k^{(\tau)}\right\}^2\right] - \left\{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[F^{(\tau)}G_k^{(\tau)}\right]\right\}^2.$$

Minimizing previous equation w.r.t. b_k we get

$$2b_k \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\frac{G_k^{(\tau)}}{\omega(\tau)}\right)^2\right] - 2\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\frac{F^{(\tau)}}{\omega(\tau)}\left(G_k^{(\tau)}\right)^2\right] = 0$$

$$\begin{aligned} b_k &= \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\frac{F^{(\tau)}}{\omega(\tau)}\left(G_k^{(\tau)}\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\frac{G_k^{(\tau)}}{\omega(\tau)}\right)^2\right]} \\ &= \frac{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\frac{1}{\omega(\tau)}\left(\sum_{j=0}^{T}\gamma^j\mathcal{R}(\boldsymbol{z}_j)\right)^2\left(\omega^2(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}{\mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^{\mathcal{B}})}\left[\left(\omega(\tau)\sum_{i=0}^{T}\nabla_{\boldsymbol{\theta}_k}\log\pi^{\mathcal{T}}(\boldsymbol{z}_i)\right)^2\right]}. \end{aligned}$$

Chapter 5

Experiments

This chapter is dedicated to the analysis of the performance of several actor–only algorithms. The aim is to show the different levels of effectiveness of gradient ascent/descent methods which follow gradients composed by the four main elements ($J, \nabla_{\boldsymbol{\theta}} J$, M and $\nabla_{\boldsymbol{\theta}} M$) presented in Chapters 3 and 4.

The analysis takes into account several dimensions that characterize the algorithms:

- On/off–policy learning;
- Number of samples used;
- Objective function to optimize;
- Risk neutral/averse objective function;
- REINFORCE/GPOMDP–like gradient estimation;
- Gradient estimation with/without baseline.

Each graph is the result of 20 simulations, which are represented by their mean value and the 95% confidence interval as error bars. The simulations are performed with a fixed learning rate tuned by hand and they start from the same initial policy parameterization $\boldsymbol{\theta}(0) = [0, 0, 0, 0, 0, 0]^T$.

5.1 Portfolio Domain

The environment used to perform the tests is very similar to the one described in [17]. Although it is a rather simplistic model, it is an interesting financial problem where the agent's actions can influence the variance of the return. Moreover the state transitions are stochastic and the actions produce both immediate and delayed rewards. For these reasons it is a good test case, suitable for the evaluation of our off–line algorithms.

We use a portfolio management problem (shown in Figure 5.1) where the investment assets can be either liquid or non–liquid and in each state the available actions are:

1. Maintain unchanged the liquid asset
   (action 0: do not invest).
2. Invest a fixed fraction α of liquid asset in one non–liquid asset
   (action 1: invest).

The chosen action is determined by a stochastic policy, in particular the action "invest" is performed with probability

$$\pi(a = 1|s; \boldsymbol{\theta}) = \epsilon + (1 - 2\epsilon)e^{-\frac{(\boldsymbol{\theta}\cdot s - 10)^2}{100}}, \tag{5.1}$$

while $\pi(a = 0|s; \boldsymbol{\theta}) = 1 - \pi(a = 1|s; \boldsymbol{\theta})$.

The dot product $\boldsymbol{\theta} \cdot s$ of the two vectors $\boldsymbol{\theta}, s \in \mathbb{R}^n$ is defined as $\sum_{i=1}^{n} \boldsymbol{\theta}_i s_i$ and $\epsilon \in \mathbb{R}_+$ is a parameter that guarantees $\forall \boldsymbol{\theta} \in \Theta$ and $\forall s \in \mathcal{S}$ non zero probability of choose each the action.

Besides the behavior of the agent, the system has its own dynamics, in fact at each time step t ($0 \leq t \leq T$ where T is the time horizon):

- The liquid asset guarantees a fixed interest rate r_l to the agent.
- The non–liquid assets have a time dependent interest rate $r_{nl}(t)$, earned by the agent only after a maturity period of N steps from the time of investment.
- The interest rate $r_{nl}(t)$ has some risk probability p_{risk} of not being paid once it has reached the maturity. Furthermore it can take one of the two values r_{nl}^{low} or r_{nl}^{high} and the transition between these values occur stochastically with switching probability p_{switch}.
- The reward given to the agent is the sum of the reward from the liquid investment and the one from the non–liquid investment (the latter is 0 when there is not any mature non–liquid asset or if it is lost).

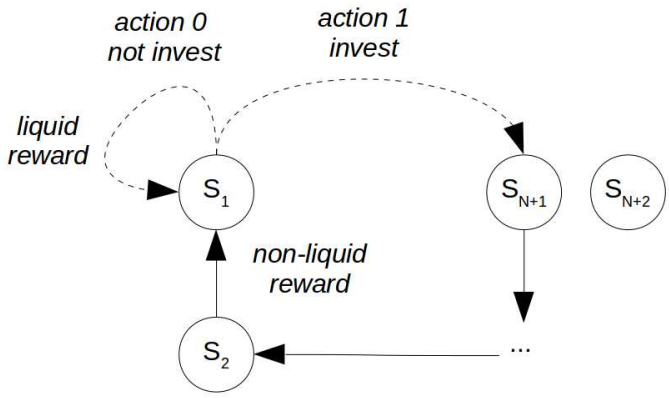


Figure 5.1: The Portfolio Model. Figure represents the environment used to test the algorithms. The circles are the $N+2$ states, the dashed lines are the two actions and the solid lines are the maturity period of the non–liquid assets. The arrows indicate the direction in which the fixed fractions of investment are moved according to the actions and the system dynamics.

The state of the model is represented by a vector $s(t) \in \mathbb{R}^{N+2}$, where the first state variable $s_1(t) \in [0,1]$ is the fraction of investment in liquid assets and the last variable $s_{N+2}(t)$ is the difference between the current non–liquid interest rate $r_{nl}(t)$ and its mean value computed from the starting time $\mathop{\mathbb{E}}_{0 \leq i \leq t} [r_{nl}(i)]$.

The other N state variables $s_2(t), \cdots, s_{N+1}(t) \in [0,1]$ represent the fraction of investments in the non–liquid assets. More precisely $s_2(t)$ is the non–liquid asset that will be ready in the next step $t+1$ and it will contribute in the reward $\mathcal{R}(s_{t+1}, a_{t+1})$, while $s_{N+1}(t)$ is the fraction moved from liquid to non–liquid in the previous step t and it needs N steps to become mature.

The initial state $s(0)$ is deterministic and assumes that all the investments are in liquid assets, thus $s(0) = \left[1, \underbrace{0, \cdots, 0}_{N+1}\right]$.

Despite the policy used in these tests differs from the one presented in [17], they have the same range of probability values for each action, in fact $\forall \boldsymbol{\theta} \in \Theta,\ \epsilon \leq \pi(a|s;\boldsymbol{\theta}) \leq 1-\epsilon$, thus for $\epsilon > 0$ the agent can never act with a deterministic behavior and the variance of the total return is always positive.

The parameters used in the experiments are:

$T = 50$	$p_{risk} = 0.05$	$r_{nl}^{high} = 2$
$N = 4$	$p_{switch} = 0.1$	$r_{nl}^{low} = 1.1$
$\alpha = 0.2$	$r_l = 1$	$\epsilon = 0.05$

With these parameters the mean values of $J(\pi)$ and $Var(\pi)$ are bounded as follows: $50.8 \leq J(\pi) \leq 65.15$ and $0.96 \leq Var(\pi) \leq 43.34$.
Moreover the optimal policy parameterization w.r.t. the standard RL criterion is $\boldsymbol{\theta}^* = [10, 10, 10, 10, 10, 0]^T$, which defines the upper bounds for $J(\pi)$ and $Var(\pi)$ just presented.

In order to simplify the analysis descriptions in the rest of the text, here are defined some relevant parameterizations and their labels:

Beh0: $\boldsymbol{\theta} = \boldsymbol{\theta}(0) = [0, 0, 0, 0, 0, 0]^T$ **Beh6:** $\boldsymbol{\theta} = [6, 6, 6, 6, 6, 0]^T$

Beh2: $\boldsymbol{\theta} = [2, 2, 2, 2, 2, 0]^T$ **Beh8:** $\boldsymbol{\theta} = [8, 8, 8, 8, 8, 0]^T$

Beh4: $\boldsymbol{\theta} = [4, 4, 4, 4, 4, 0]^T$ **Beh10:** $\boldsymbol{\theta} = \boldsymbol{\theta}^* = [10, 10, 10, 10, 10, 0]^T$

Two interesting properties of these policies are:

- The expected return $J(\pi_{\boldsymbol{\theta}})$ increases going from Beh0 to Beh10.
- Given that in all the following off–policy tests the target policy is always initialized with Beh0, the initial distance between $\pi^{\mathcal{T}}$ and $\pi^{\mathcal{B}}$ increases going from Beh0 to Beh10.

5.2 On–Policy Risk-neutral

The first tests (Figures 5.2 and 5.3) are performed on on–policy algorithms with standard objective function $max_{\boldsymbol{\theta}}\ J(\pi_{\boldsymbol{\theta}})$. This means that, at each update h, new samples are collected using the current policy $\pi_{\boldsymbol{\theta}_h}$ in order to estimate the gradient $\nabla_{\boldsymbol{\theta}} J|_{\boldsymbol{\theta}=\boldsymbol{\theta}_h}$.

Here the context is risk neutral, thus the agent wants to maximize its utility without taking care of the variability of the returns. It is recalled that all the off–policy gradients discussed in the previous chapters can be translated in the on–policy settings imposing that each importance weight $\omega(\boldsymbol{z}_t) = 1,\ 0 \leq t \leq T$.

The aim is to show that:

- GPOMDP estimations of the gradient have less variance than the ones based on the REINFORCE formulation.
- The baseline can drastically reduce the variance of the estimation in both REINFORCE and GPOMDP methods.

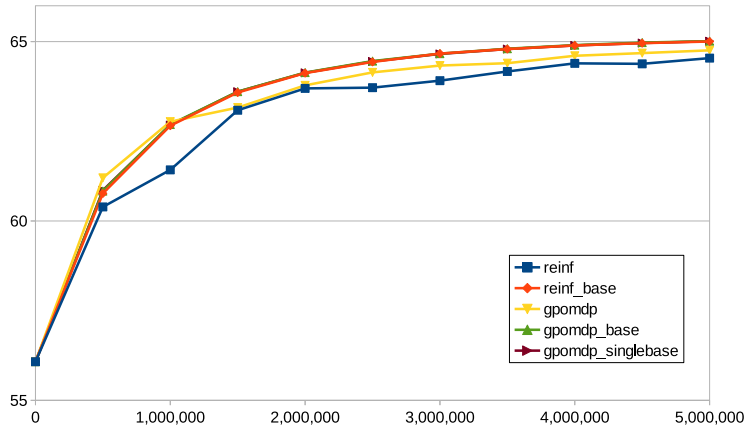
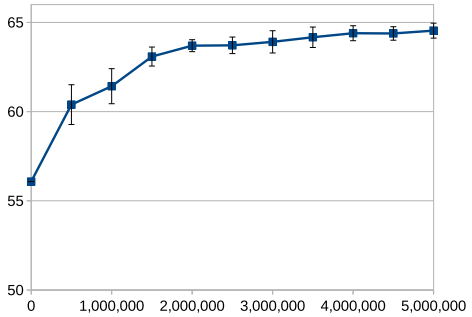
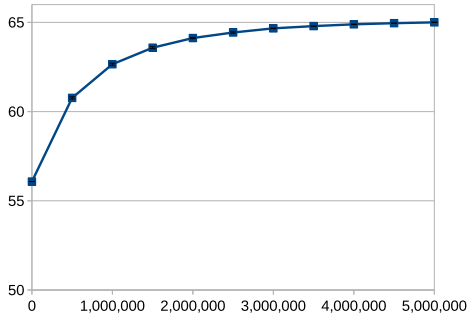


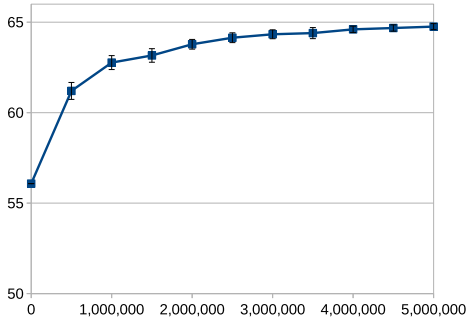
Figure 5.2: Max J On–Policy comparative graph (x axis: $samples$, y axis: J) updates: $2\,000$, *trajectories/update:* 50, *steps/trajectory:* 50, *learning rate:* $6E-2$ *total samples:* $5\,000\,000$
(The mean values of REINFORCE baseline, GPOMDP baseline and GPOMD single baseline are overlapped)



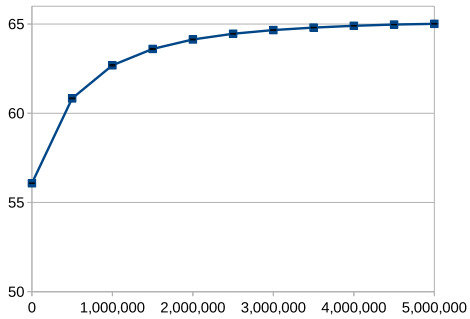
(a) *REINFORCE*



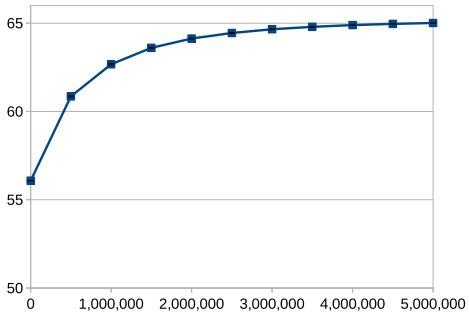
(b) *REINFORCE baseline*



(c) *GPOMDP*



(d) *GPOMDP baseline*



(e) *GPOMDP single–baseline*

Figure 5.3: Max J On–Policy (x axis: samples, y axis: J)
updates: $2\,000$, *trajectories/update:* 50, *steps/trajectory:* 50, *learning rate:* $6E-2$
total samples: $5\,000\,000$

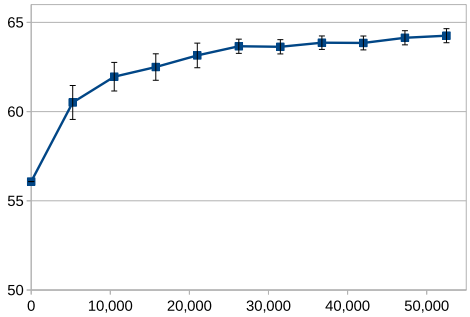
5.2.1 Number of samples

The variance reduction attributed to the introduction of the baseline technique can be exploited also in a different dimension: the total number of samples needed in the learning process.

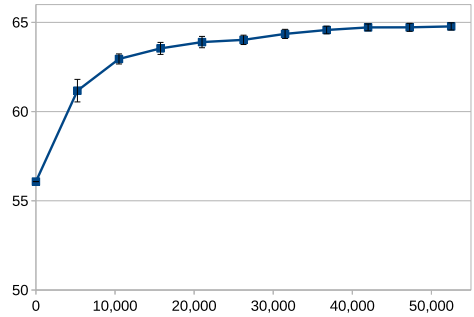
In the previous test 5 000 000 samples were used in order to restrict the variance of the two algorithms without baseline. In fact the variance increases very fast as the number of trajectories or updates is lowered.

In the next graphs (Figure 5.4) the samples drawn are only 52 500 and the learning rate increases from $6E-2$ to $8E-1$, but these modifications do not substantially alter the performance of the three algorithms with baseline, which maintain the performance very close to the previous tests. The GPOMDP–like formulations (Figures 5.4b and 5.4c) succeed to estimate the gradients more precisely than the REINFORCE–like one (Figure 5.4a), which has higher variance.

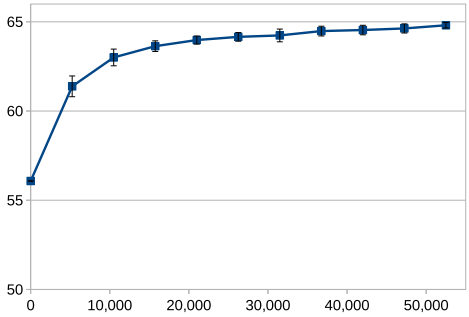
Even if this consideration seems to have a minor importance in on–line context, because the agent can, in some cases, interact with the environment as long as it wants, this is a fundamental aspect in off–line learning. Actually when the experience is limited using algorithms that can learn faster, with less variance and with few samples is a great advantage.



(a) *REINFORCE baseline*



(b) *GPOMDP baseline*



(c) *GPOMDP single–baseline*

Figure 5.4: Max J On–Policy (x axis: $samples$, y axis: J)
updates: 350, *trajectories/update:* 3, *steps/trajectory:* 50, *learning rate:* $8E-1$
total samples: $52\,500$

5.3 Off–Policy Risk–neutral

As done in the theoretical part of this thesis, the topic is now switched to off–policy off–line learning without taking into account any risk factor yet. In off–policy scenarios the data are fixed and the agent must reuse the same experience each time to update the target policy $\pi^{\mathcal{T}}$ parameters.

The importance weight technique discussed in Section 2.3.4 adds further variability in the gradient estimations, for this reason the use of baselines is even more important respect to the on–policy case. Figure 5.5 shows that the performance of the algorithms with baseline are much higher than the ones of methods without baseline, so this useful tool allows to extract more precise information from the batch dataset achieving better results.

From this first test it is already possible to identify one of the limitations of the off–line learning. In general, a batch dataset does not contain enough information to learn the optimal policy reachable with an on–line algorithm, but a good sub–optimal result can be achieved with a few samples. This may happen if the training dataset contains very few samples or when the behavior policy allows to visit only part of the environment.

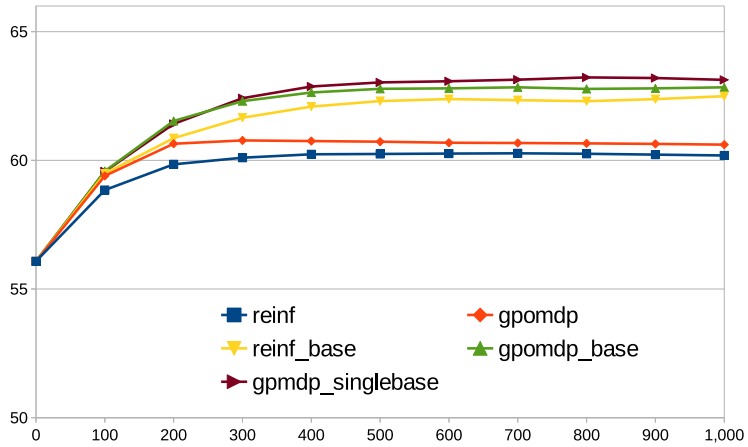


Figure 5.5: Max J Off–Policy Beh0 comparative graph (x axis: $updates$, y axis: J) updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* $1\,000\,000$

5.3.1 Baseline and Importance weights

Other important aspects of off–policy algorithms are:

- The behavior policy used to collect the batch data.
- The gradual improvement of the target policy.

In the previous graph (Figure 5.5) the behavior policy parameterization is "Beh0" (see last part of Section 5.1). This particular case is useful to explain how the baseline affects the variance of the gradients in the different phases of the learning process.

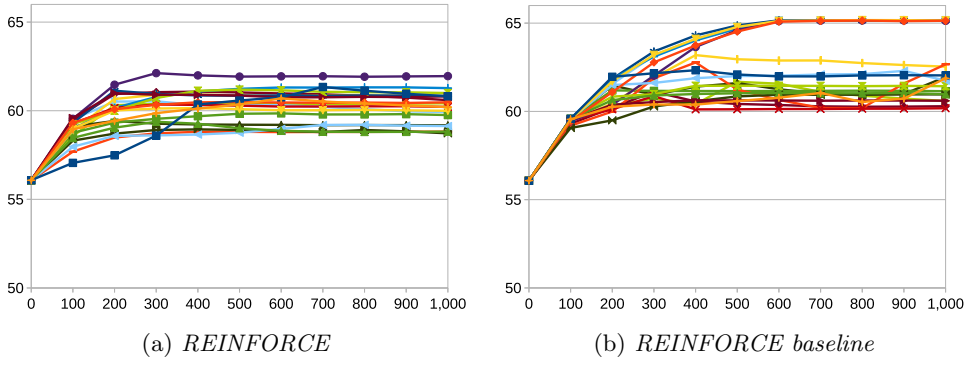
Beh0 is also used as the initial parameterization for the target policy, therefore the importance weight computed in the first updates are very closed to 1, which makes the initial phase similar to an on–policy learning. Here the variability introduced by the importance weights is minimal and the baseline can maximize its effect.

In order to better appreciate the differences between REINFORCE and GPOMDP methods with/without baseline in the off–policy scenario and provide a deep analysis of them, it is convenient to split Figure 5.5 into 5 distinct graphs (Figure 5.6), one for each algorithm showing the learning curves for each run.

As mentioned before, the initial distance between the target policy and the behavior is 0 and it remains small until the 100–th update. In this part of the graphs it can be noticed how both GPOMDP methods and the baseline techniques are very effective in reducing the variance, especially when they work at the same time in graphs 5.6d and 5.6e.

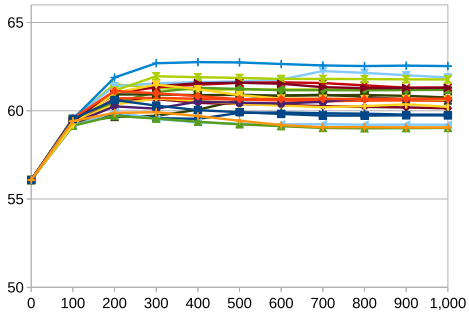
A "breaking point" can be observed in each graph around the 100-th update. After this point the 20 trajectories follow different directions, because the distance between the improving target policy and the fixed behavior policy increases the variability of the importance weights. From this point on, the beneficial effect of the baseline on the gradient estimation is inevitably dominated by the importance weight component.

Focusing on the rightmost part of the graphs, it can be noticed that the three algorithms with baseline have much more variability than the ones without it. This is not a fault of the baseline techniques, but it is due to the fact that, using the baseline, the performance of the algorithm increases, thus the target policy becomes further more different respect to the behavior one, strengthening also the negative effect due to high–variant importance weights.

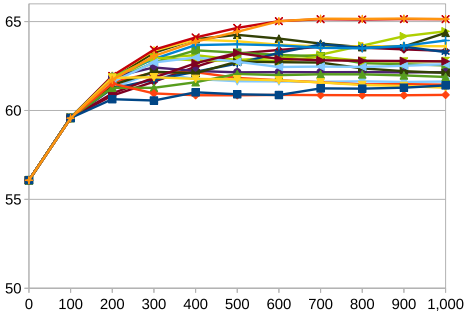


(a) *REINFORCE*

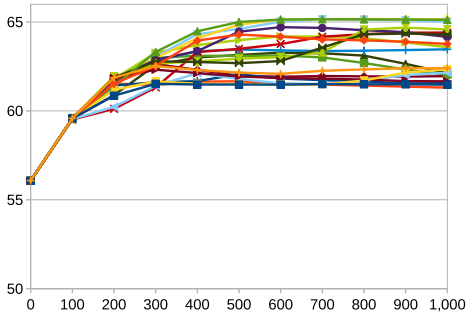
(b) *REINFORCE baseline*



(c) *GPOMDP*



(d) *GPOMDP baseline*



(e) *GPOMDP single–baseline*

Figure 5.6: Max J Off–Policy Beh0 single algorithms (x axis: $updates$, y axis: J)
each graph contains 20 learning curves in order to show the different shapes of their breaking points
updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$
total samples: $1\,000\,000$

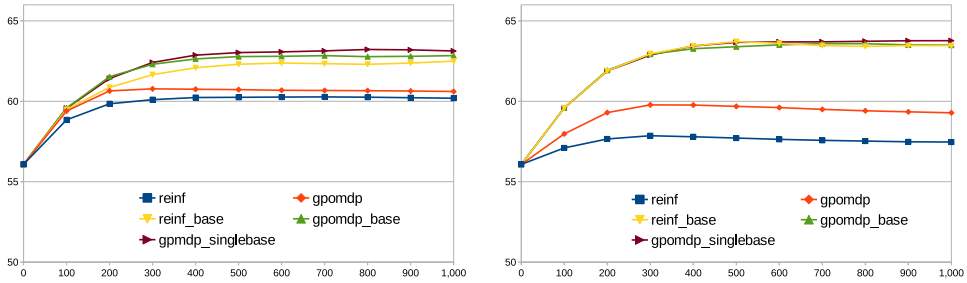
5.3.2 Behavior policies

In the following tests it is studied how the off–policy learning process reacts when the distance between the initial target parameterization and the behavior policy is increased, moving the latter from Beh0 to Beh10.

As conclusion, it will be established that using the baseline technique and starting from a target policy close to the behavior policy, which has generated the dataset, are both profitable choices.

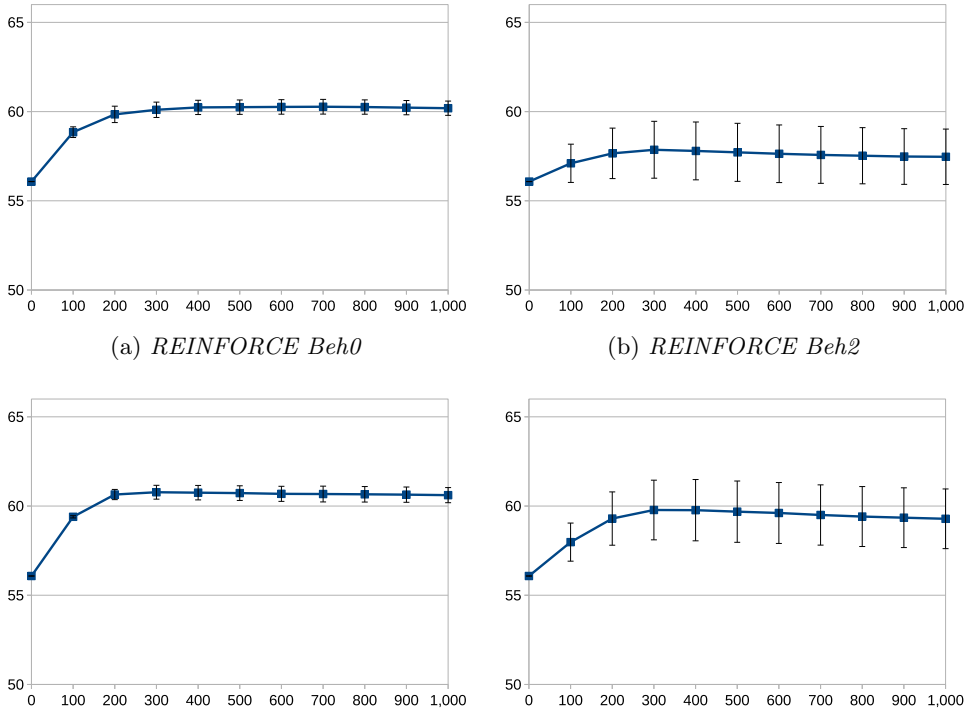
Beginning the analysis from the two algorithms without baseline it is immediately clear that moving the behavior policy away from the target of a single step is sufficient to strongly degrade their performance in terms of lower expected values and higher variances. In Figures 5.7, 5.8 and 5.9 are compared the performance when using Beh0 (leftmost graphs, already presented before) and Beh2 (rightmost graphs).

The "breaking points" visible in Figures 5.9a and 5.9c are moved to the left when the Beh2 is used (Figure 5.9b and 5.9d). This means that, for the basic algorithms without baseline, the distance between the target (Beh0) and the behavior (Beh2) produces importance weights with high variances even from the beginning of the learning process. The misleading estimations bring the gradient ascent algorithms to perform optimization steps in wrong directions obtaining poor results.



(a) *Performance with Beh0* (b) *Performance with Beh2*

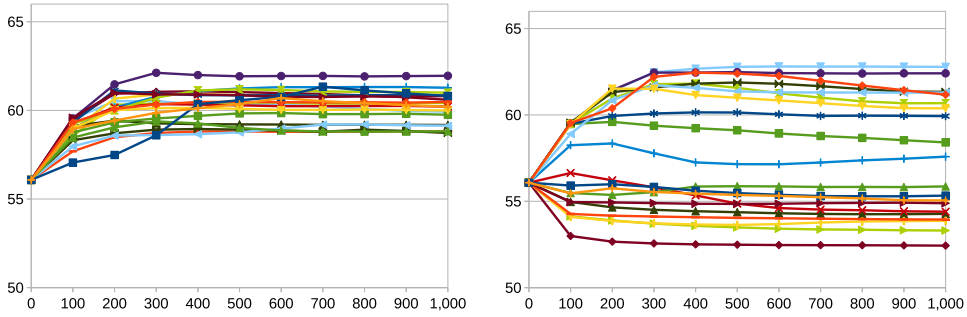
Figure 5.7: Max J Off–Policy comparative graph (x axis: updates, y axis: J) updates: 1 000, *batch trajectories:* 20 000, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* 1 000 000



(a) *REINFORCE Beh0* (b) *REINFORCE Beh2*

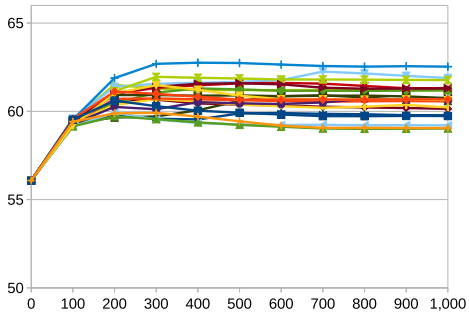
(c) *GPOMDP Beh0* (d) *GPOMDP Beh2*

Figure 5.8: Max J Off–Policy algorithms without baseline average values
(x axis: $updates$*, y axis:* J*)*
updates: $1\,000$*, batch trajectories:* $20\,000$*, steps/trajectory:* 50*, learning rate:* $5E-2$
total samples: $1\,000\,000$

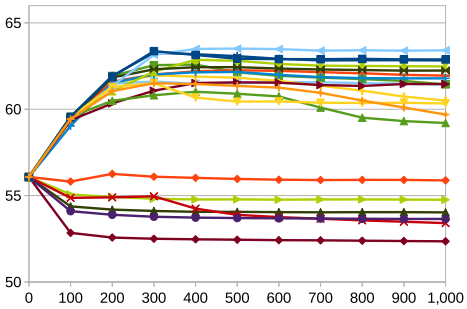


(a) *REINFORCE Beh0*

(b) *REINFORCE Beh2*



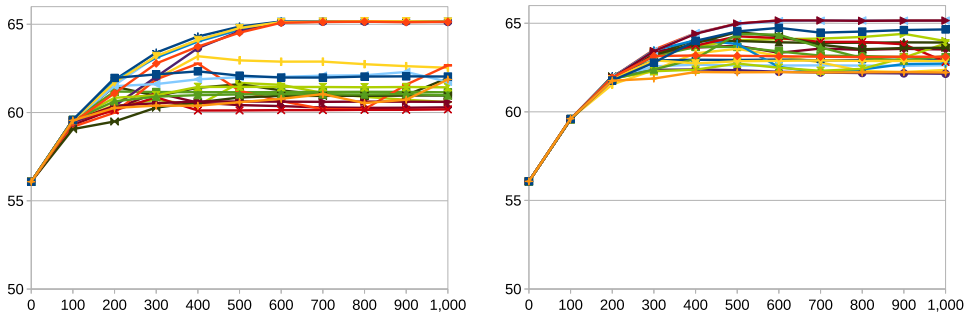
(c) *GPOMDP Beh0*



(d) *GPOMDP Beh2*

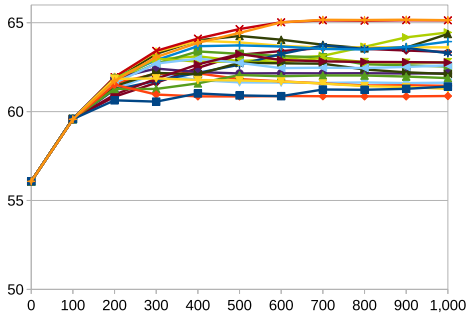
Figure 5.9: Max J Off–Policy algorithms without baseline trajectories
(x axis: $updates$*, y axis:* J*)*
updates: $1\,000$*, batch trajectories:* $20\,000$*, steps/trajectory:* 50*, learning rate:* $5E-2$
total samples: $1\,000\,000$

The baseline successfully reduces the initial importance weight penalty and shifts the "breaking points" toward the 200–th update (see Figures 5.10b, 5.10d and 5.10f). The extra updates kept under the control by the baseline allow to enhance the performance of the algorithms, compared to both the Beh0 tests and the Beh2 algorithms without baseline (see Figure 5.7).

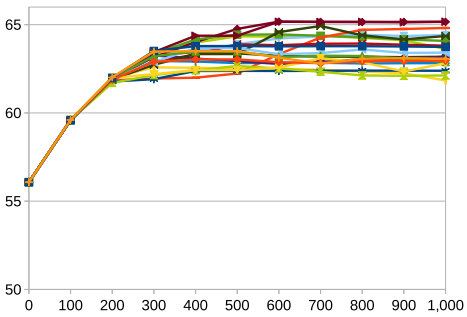


(a) *REINFORCE baseline Beh0*

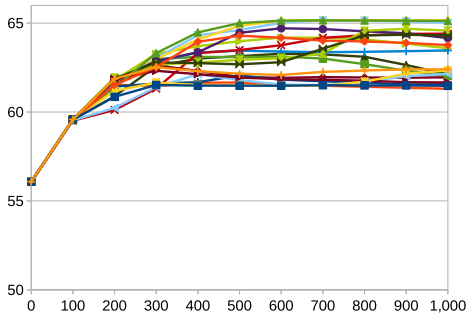
(b) *REINFORCE baseline Beh2*



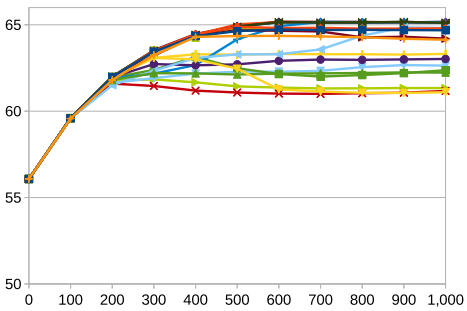
(c) *GPOMDP baseline Beh0*



(d) *GPOMDP baseline Beh2*



(e) *GPOMDP single baseline Beh0*



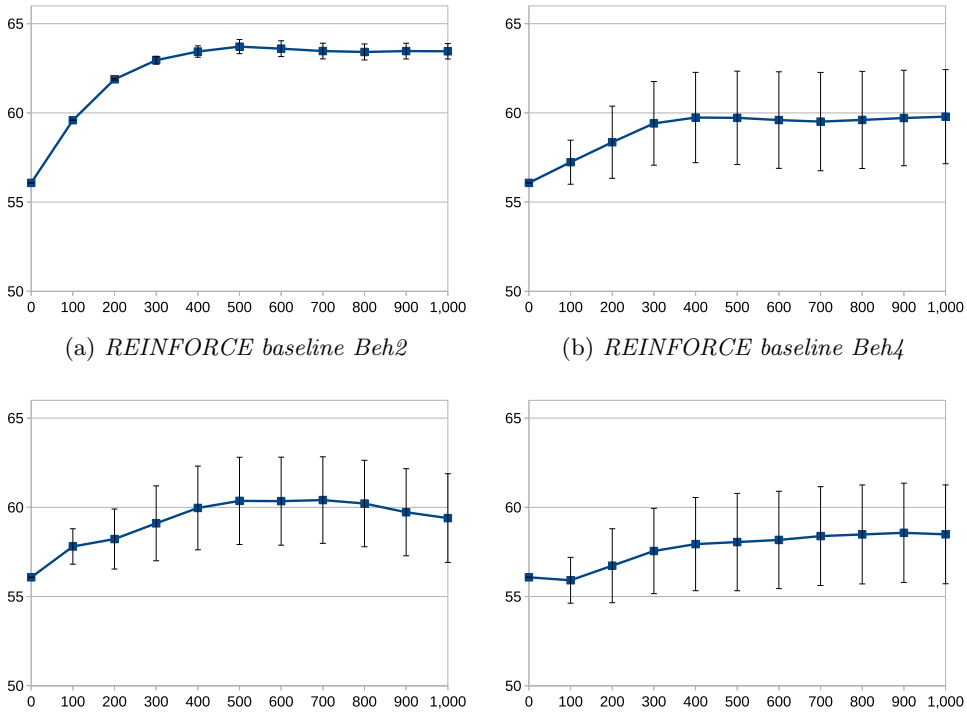
(f) *GPOMDP single baseline Beh2*

Figure 5.10: Max J Off–Policy algorithms with baseline (x axis: updates, y axis: J) updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* $1\,000\,000$

The tests presented in the last part of this section (Figures 5.11, 5.12 and 5.13) emphasize that further increasing the distance between the target and the behavior policies necessarily deteriorates also the performance of the baseline algorithms.

This is the empirical proof that confirms the advantage obtained starting with a target policy similar to the behavior which has generated the batch dataset.

In Section 5.4.4 is analyzed the Modified Gradient technique, which is a possible solution to this problem. The standard RL gradient used in these off–policy tests is revised in order to reduce the variance of the importance weights when the target and the behavior policy are very different.



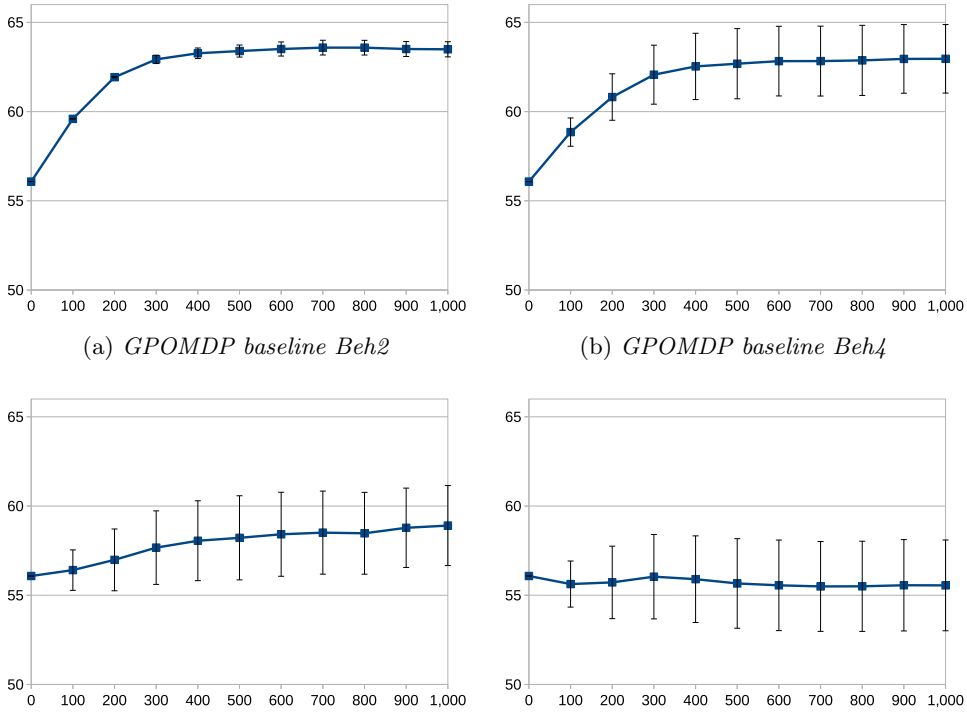
(a) *REINFORCE baseline Beh2*

(b) *REINFORCE baseline Beh4*

(c) *REINFORCE baseline Beh6*

(d) *REINFORCE baseline Beh10*

Figure 5.11: Max J Off–Policy REINFORCE baseline (x axis: $updates$, y axis: J) updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* $1\,000\,000$



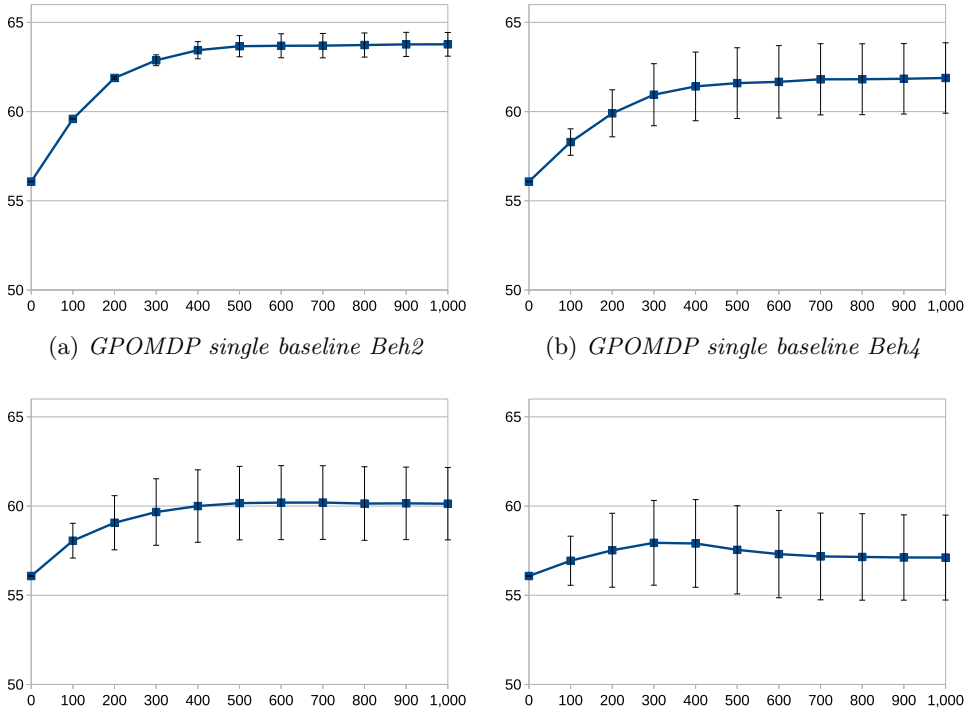
(a) *GPOMDP baseline Beh2*

(b) *GPOMDP baseline Beh4*

(c) *GPOMDP baseline Beh6*

(d) *GPOMDP baseline Beh10*

Figure 5.12: Max J Off–Policy GPOMDP baseline (x axis: $updates$, y axis: J) updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* $1\,000\,000$



(a) *GPOMDP single baseline Beh2*

(b) *GPOMDP single baseline Beh4*

(c) *GPOMDP single baseline Beh6*

(d) *GPOMDP single baseline Beh10*

Figure 5.13: Max J Off–Policy GPOMDP single baseline (x axis: $updates$, y axis: J) updates: $1\,000$, *batch trajectories:* $20\,000$, *steps/trajectory:* 50, *learning rate:* $5E-2$ *total samples:* $1\,000\,000$

5.4 Risk–related

In this section are presented both the on–policy and off–policy tests, without repeating all the conclusions already drawn in the Sections 5.2 and 5.3.

5.4.1 Variance

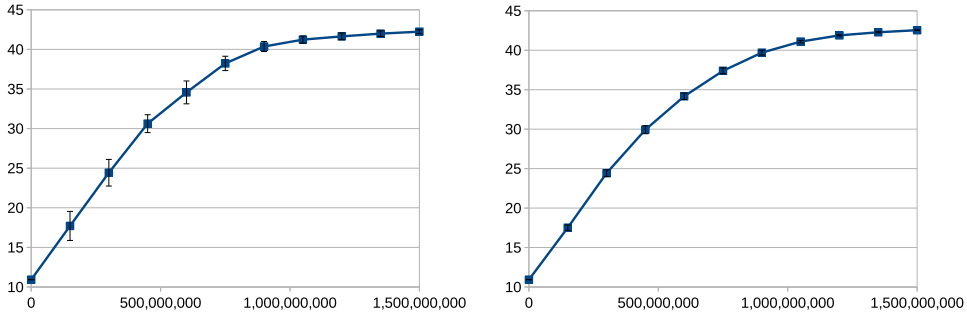
The preliminary risk–related test aims at proving that the portfolio problem admits an optimal policy π^* which leads to returns with both the maximum expected value $J(\pi^*)$ and the maximum variance $Var(\pi^*)$.

The model presents the same property of many real problems in which it is not possible to increase the expected gain without affecting also its variance. Therefore it can be stated that this simple problem is suitable for to a risk–sensitive analysis.

Following the gradient $\nabla_{\boldsymbol{\theta}} Var(\pi) = \nabla_{\boldsymbol{\theta}} M(\pi) - 2J(\pi)\nabla_{\boldsymbol{\theta}} J(\pi)$ all the algorithms converge to the same $\boldsymbol{\theta}^*$ defined in Section 5.1. Figure 5.14 shows this result only in the cases of on–policy GPOMDP and GPOMDP with single–baseline algorithms, just to avoid proposing all the same results many times. The baseline reduces also in these cases the variance of the estimated gradients.

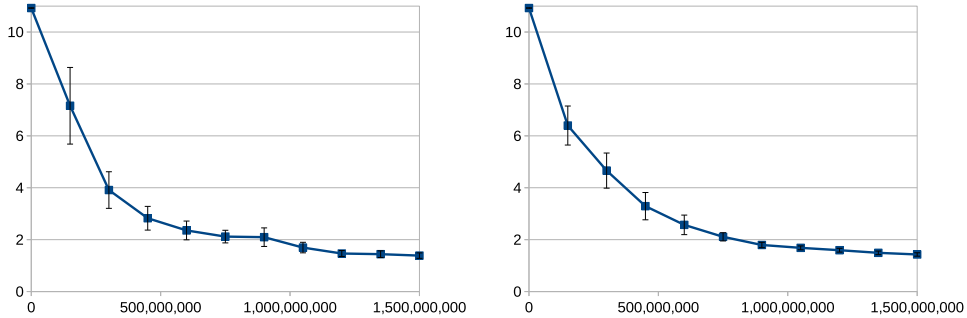
When the objective function aims to maximize or minimize the pure variance of the return the algorithms need much more samples in order to achieve good performance. This is due to the fact that the estimation of $\nabla_{\boldsymbol{\theta}} Var(\pi)$ is computed as $\nabla_{\boldsymbol{\theta}} M(\pi) - 2J(\pi)\nabla_{\boldsymbol{\theta}} J$, that is the difference of two terms with the same order of magnitude. Most of the time the gradients are small values close to 0, therefore a bigger number of samples is needed to ensure the estimation of the right direction to follow.

The next step is to show that the policy–gradient algorithms can also find a parameterization for which the variance is minimal (Figures 5.15 and 5.16 for on–policy tests, Figure 5.17 for off–policy tests). This is an extreme case of risk aversion, in which the agent is satisfied with getting any return provided that the variance is the lowest possible.



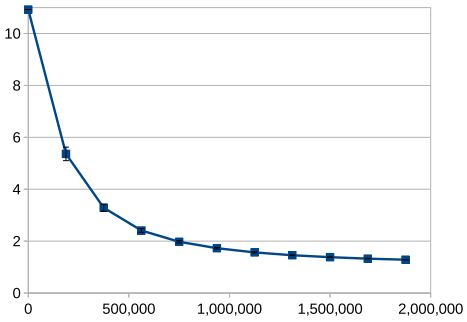
(a) *GPOMDP* (b) *GPOMDP single baseline*

Figure 5.14: Max Var On–Policy (x axis: $samples$, y axis: Var)
updates: $2\,000$, *grad_trajectories/update:* $10\,000$, *eval_trajectories/update* $5\,000$, *steps/trajectory:* 50, *learning rate:* $1E-2$
total samples: $1\,500\,000\,000$

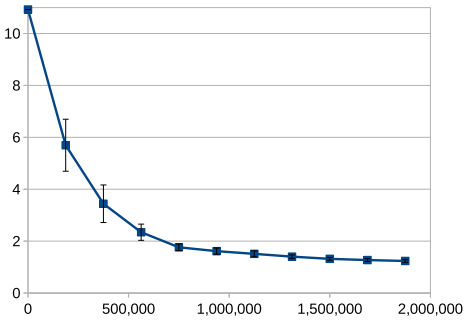


(a) *REINFORCE* (b) *GPOMDP*

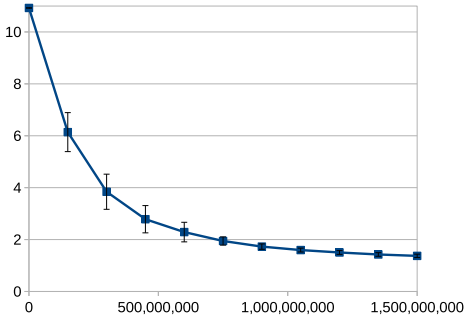
Figure 5.15: min Var On–Policy without baseline (x axis: $samples$, y axis: Var)
updates: $2\,000$, *grad_trajectories/update:* $10\,000$, *eval_trajectories/update* $5\,000$, *steps/trajectory:* 50, *learning rate:* $1E-2$
total samples: $1\,500\,000\,000$



(a) *REINFORCE baseline*

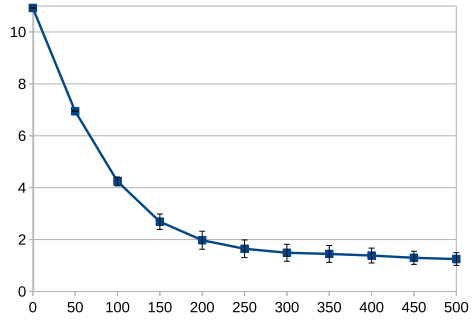


(b) *GPOMDP baseline*

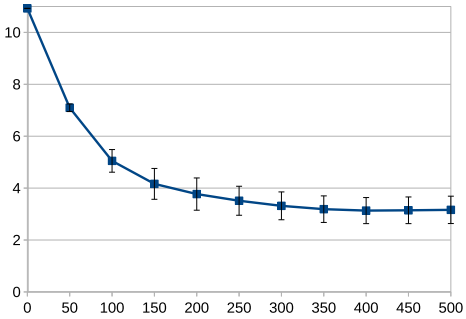


(c) *GPOMDP single baseline*

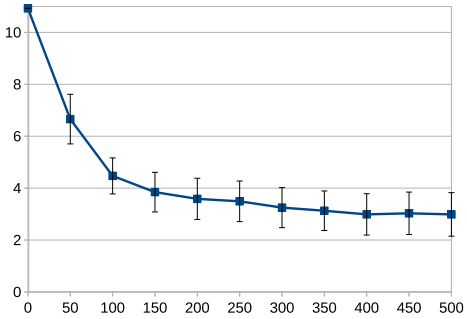
Figure 5.16: min Var On–Policy with baseline (x axis: $updates$*, y axis:* Var*)*
updates: 250*, grad_trajectories/update:* 100*, eval_trajectories/update* 50*, steps/trajectory:* 50*, learning rate:* $1E-1$
total samples: $1\,875\,000$



(a) *REINFORCE baseline*



(b) *GPOMDP baseline*



(c) *GPOMDP single baseline*

Figure 5.17: min Var Off–Policy Beh0 (x axis: $updates$, y axis: Var)
updates: 500, *grad_trajectories/update:* 10 000, *eval_trajectories/update* 2 500, *steps/trajectory:* 50, *learning rate:* $5E-2$
total samples: 625 000

Unfortunately, from our tests it is not yet clear why risk–sensitive algorithms with baseline have higher variance when the gradient has a GPOMDP–like formulation respect to the REINFORCE–like formulation. In order to avoid misunderstandings and mistakes, from now on in this document we will report only the results of the tests performed on REINFORCE–like algorithms.

5.4.2 Sharpe Ratio

As introduced in Section 4.1 the Sharpe Ratio is a measure of risk used in finance, defined as $SR(\pi) = \frac{J(\pi)}{\sqrt{Var(\pi)}}$. An important aspect of this index is the absence of control parameters, in fact it cannot be tuned using the risk aversion level of the agent.

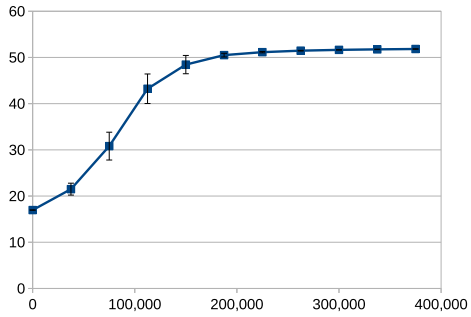
The policy chosen for this model (see Expression 5.1) always guarantees a positive variance of the return (see 5.1), thus the Sharpe Ratio is limited for each parameterization $\boldsymbol{\theta}$ and the maximization problem is well defined.

In Figures 5.18 and 5.19 are shown the performance of the REINFORCE with baseline algorithm in both on–policy and off–policy scenarios. Each test is explained using 3 graphs:

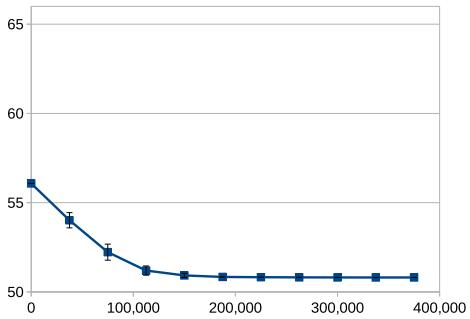
- Graphs 5.18a and 5.19a report the learning curves for the maximization of the Sharpe Ratio value.
- Graphs 5.18b and 5.19b show how the expected return changes during the learning process.
- Graphs 5.18c and 5.19c are the equivalent of the previous graphs in terms of the variance of the return instead of its expectation.

Although this risk measure has the most complex gradient expression among the ones presented in this thesis (see Expression 4.1), the results obtained are remarkably good both in on–line and off–line tests, where the performance of the algorithms are quite similar.

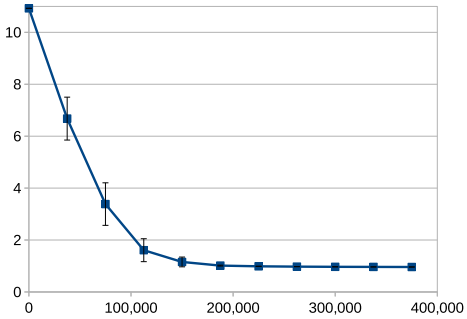
Since the policy admits some parameterizations for which the variance of the return is less than 1 and its expected value is not too low, the Sharpe ratio assumes the highest value just in correspondence of one of these policies. Therefore the optimal solution for this specific risk–averse objective function is the one already found in the experiments described in Section 5.4.1, where the aim was simply minimizing the variance.



(a) *Sharpe Ratio*

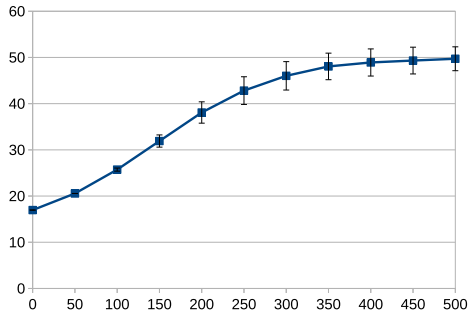


(b) J

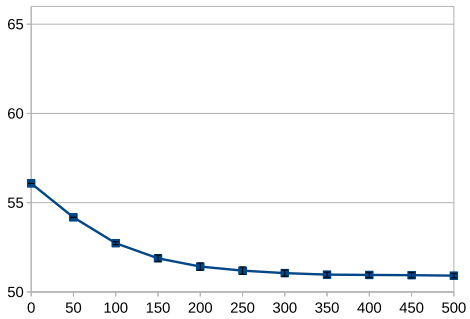


(c) Var

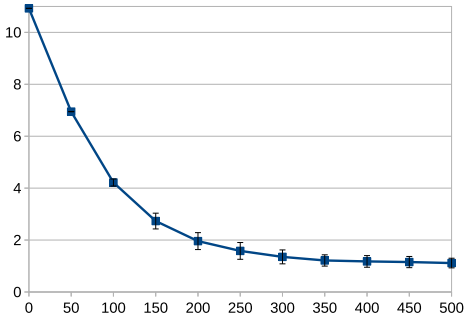
Figure 5.18: Max Sharpe Ratio On–Policy REINFORCE baseline
(x axis: $samples$, y axis: $SR\,(a), J\,(b), Var\,(c)$)
updates: 50, *grad_trajectories/update:* 100, *eval_trajectories/update* 50, *steps/trajectory:* 50, *learning rate:* $3E-1$
total samples: $375\,000$



(a) *Sharpe Ratio*



(b) J



(c) *Var*

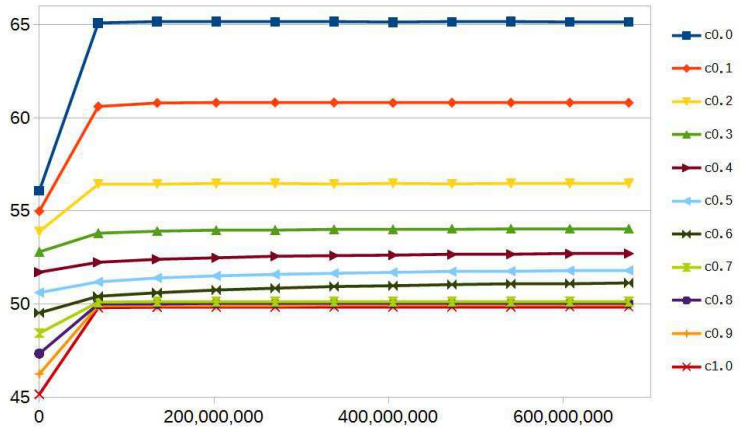
Figure 5.19: Max Sharpe Ratio Off–Policy REINFORCE baseline Beh0
(x axis: $updates$*, y axis:* $SR\,(a), J\,(b), Var\,(c)$ *)*
updates: 500*, grad_trajectories/update:* $10\,000$*, eval_trajectories/update* $2\,500$*, steps/trajectory:* 50*, learning rate:* $5E-2$
total samples: $625\,000$

5.4.3 Mean–Variance criterion

The mean–variance (MV) criterion uses the variance of the return $Var(\pi)$ as penalty to prevent the uncontrolled growth of $J(\pi)$. Differently from the Sharpe Ratio it takes advantage of a penalty coefficient c which the agent can modify in order to vary its risk–aversion.

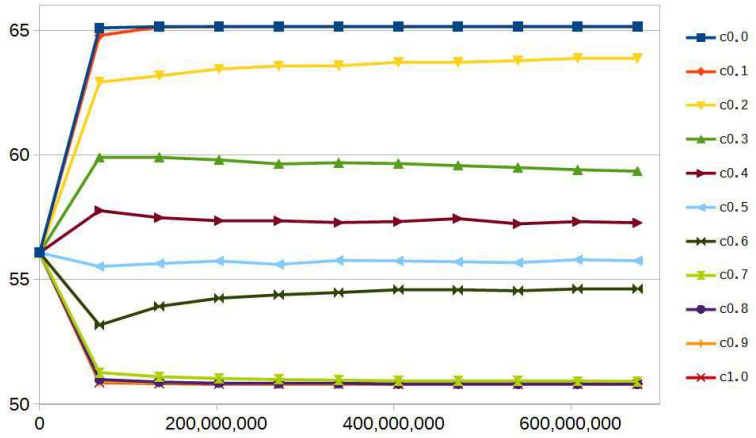
In Figure 5.20 are shown the different values that the objective function $J - cVar$ can assume in the proposed model. For $c = 0$ the optimization problem trivially degenerates into the already discussed risk–neutral maximization of the expected return. On the other hand for $c \geq 0.7$ minimizing the variance becomes much more important than obtaining any gain.

In this scenario the agent can choose the appropriate penalty coefficient to achieve the desired performance as presented in Figures 5.21 and 5.22. Both in the on–policy and off–policy learning the expected return can be sacrificed (5.21a and 5.22a) in order to also decrease the variability (5.21b and 5.22b).

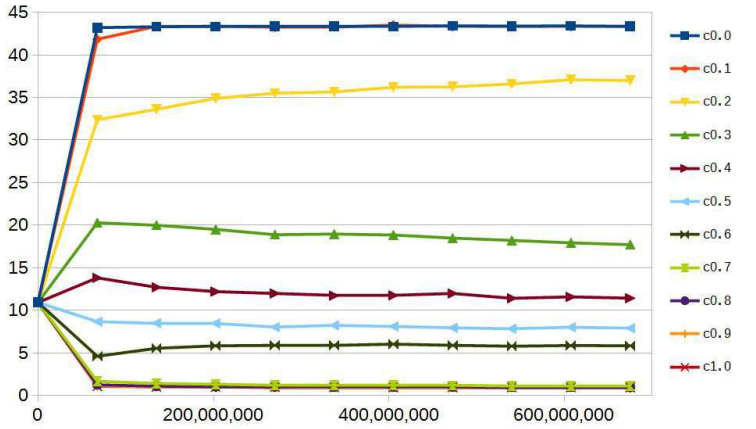


(a) *Mean Variance*

Figure 5.20: Max Mean Variance On–Policy REINFORCE baseline
(x axis: samples, y axis: $J - cVar$)
updates: 3 000, *grad_trajectories/update:* 3 000, *eval_trajectories/update* 1 500, *steps/trajectory:* 50, *learning rate:* $5E-1$
total samples: 675 000 000

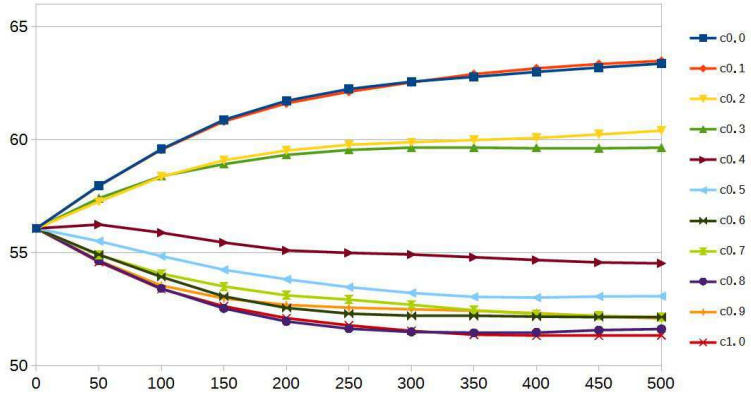


(a) *J projection of the MV criterion*

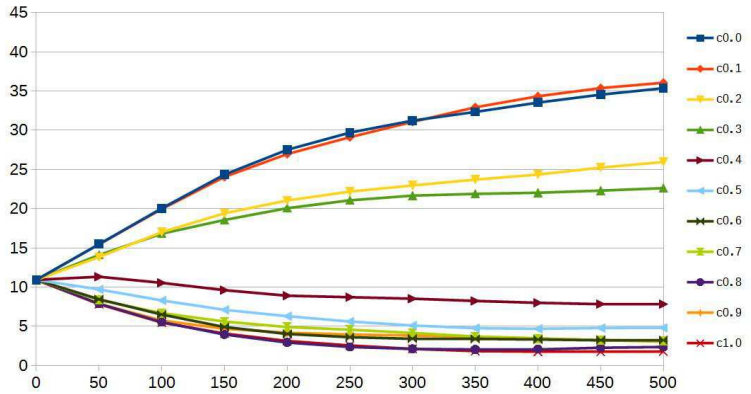


(b) *Var projection of the MV criterion*

Figure 5.21: Max Mean Variance On–Policy REINFORCE baseline
(x axis: samples, y axis: $J(a), Var(b)$ *)*
updates: $3\,000$, *grad_trajectories/update:* $3\,000$, *eval_trajectories/update* $1\,500$, *steps/trajectory:* 50, *learning rate:* $5E-1$
total samples: $675\,000\,000$



(a) *J projection of the MV criterion*



(b) *Var projection of the MV criterion*

Figure 5.22: Max Mean Variance Off–Policy REINFORCE baseline Beh2
(x axis: $samples$, y axis: $J(a), Var(b)$)
updates: 500, *grad_trajectories/update:* 10 000, *eval_trajectories/update* 2 500, *steps/trajectory:* 50, *learning rate:* $5E-2$
total samples: 625 000

The penalty coefficient allows to transform the problem in a multi–criteria decision analysis, in which the maximization of J and the minimization of Var are two conflicting criteria that need to be evaluated in making decisions.

Typically some solutions perform well in some criteria and some perform well in others. Therefore it does not exist a unique optimal solution for such problems and it is necessary to use decision maker's preferences to differentiate between solutions. In order to do this in the portfolio model it is possible to show the Pareto–optimal frontier plotting the J–Var pairs selected varying the penalty coefficient.

In Figure 5.23 the blue line represents the Pareto frontier, while the red line identifies how J and Var change during a standard risk–neutral learning process (see Section 5.2). It can be noticed that incorporating the risk measure into the objective function allows the algorithms to find policies which, in correspondence of the same expected return value, have less variance.

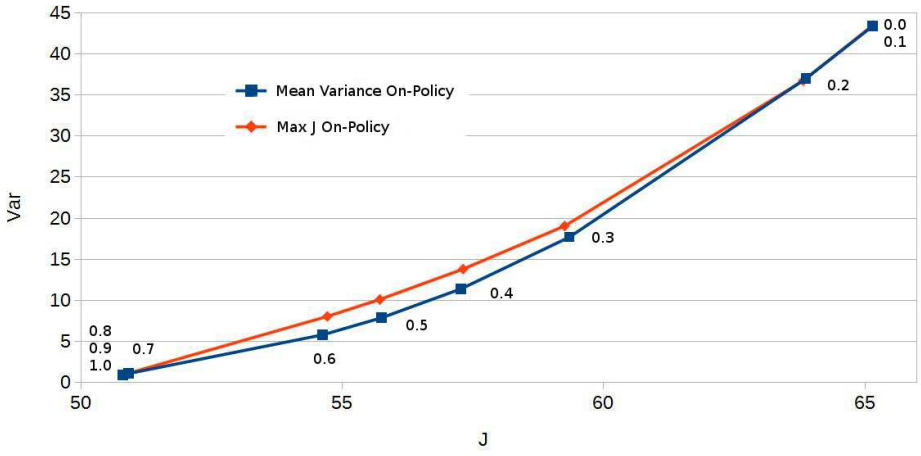


Figure 5.23: Max Mean Variance On–Policy Pareto frontier (x axis: J, y axis: Var)

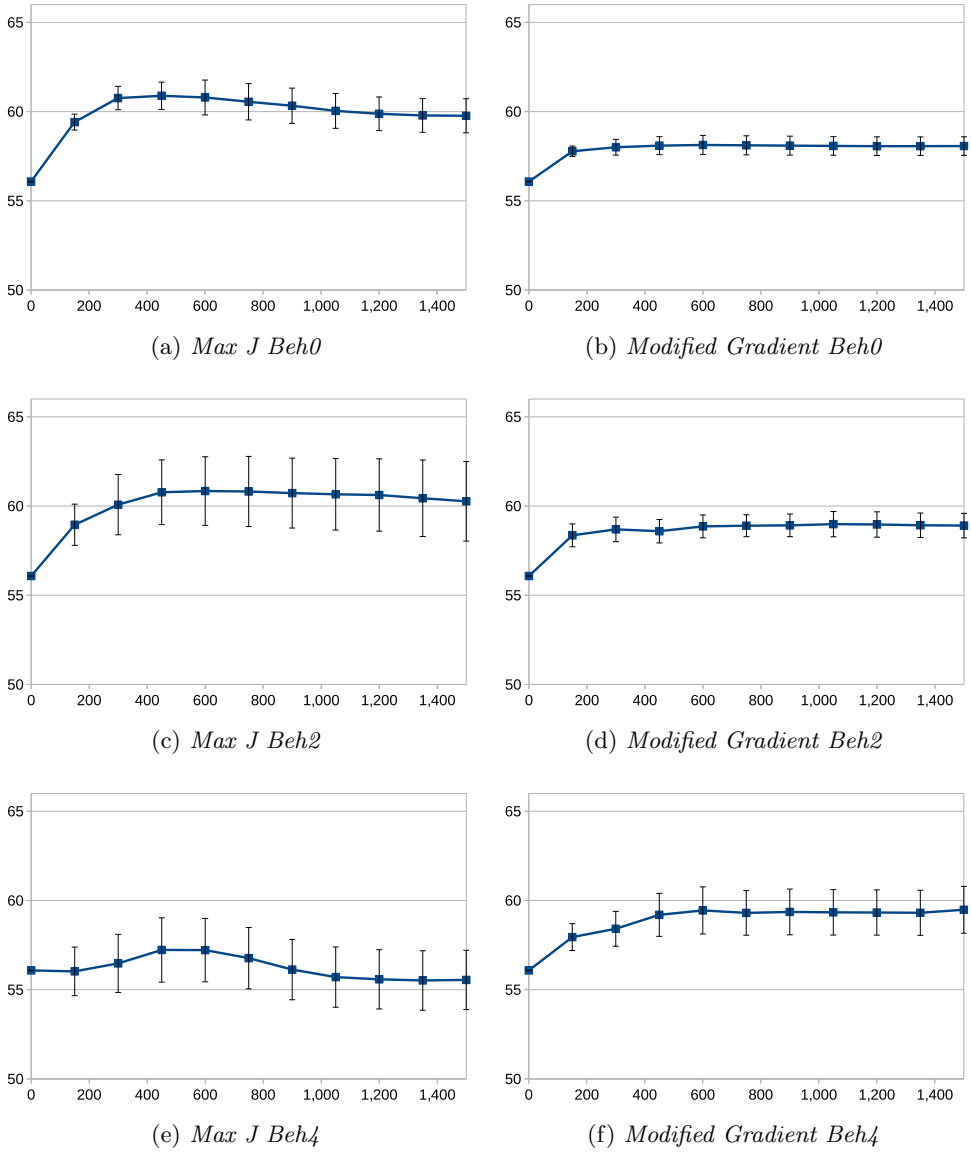
5.4.4 Modified Gradient

The last risk–sensitive analysis is about the criterion described in Section 4.3. Here the notion of risk is not the one used in the previous tests in Sections 5.4.1, 5.4.2 and 5.4.3, in fact the main difference is that it is meaningful only in off–line scenarios. The main purpose of the objection function $J - p\,Var(J)$ is to obtain the best possible gain from a batch dataset without speculate too much on policy performance which cannot be well estimated from the fixed available experience. Identifying the maximum amount of exploitable informations contained in the current dataset, avoids that the learning process becomes more similar to a random guess.

Before testing this risk–sensitive measure, we first repeat the experiments conducted in Section 5.3, where the objective function is the standard maximization of $J(\pi)$ in the off–line context. The only difference between these tests and the ones presented before is the number of samples used. Here, in order to make evident the desired effects, are used only 10 000 samples against the previous 1 000 000.

Figures 5.24 and 5.25 show very interesting aspects of the off–line learning. The 6 graphs on the left are the new risk–neutral tests in which the batch dataset is created using different behavior policies. As already mentioned in Section 5.1, moving from Beh0 to Beh10 increases the expected return of the trajectories in the training data, but also it increases the distance between the initial target parameterization and the behavior policy which influences the variance of the importance weights. When the updates exceed the breaking point the good effect of the baseline vanishes, leaving the variance of the gradient under the control of the importance weights noise.

The problem is that using few samples the breaking point moves very quickly towards the starting point, leading to some phases in which the optimization follows the right direction and other phases in which the expected return is minimized. The main advantage of the modified gradient measure, shown in Figures 5.24 and 5.25 (graphs on the right), is that the expected return increases until the effects of the importance weights becomes too variable. At this point the gain is maintained over time while the standard algorithm continues the "optimization" process, which however brings to poor performance. To emphasize the drastic change in the behavior of the algorithm are also shown the same previous graphs revealing all the single learning curves (see Figures 5.26 and 5.27).

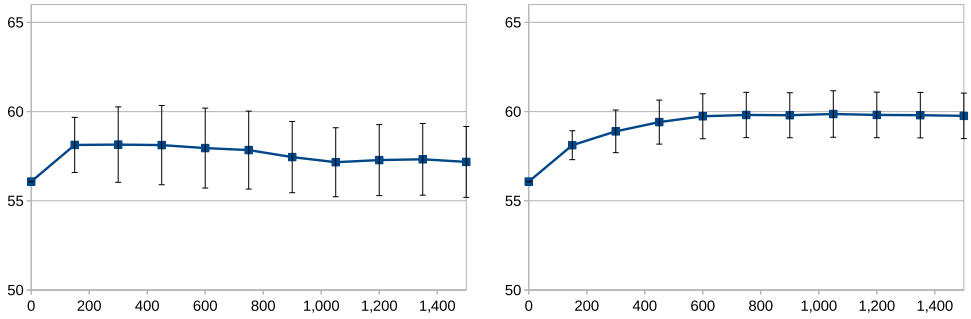


(a) *Max J Beh0* (b) *Modified Gradient Beh0*

(c) *Max J Beh2* (d) *Modified Gradient Beh2*

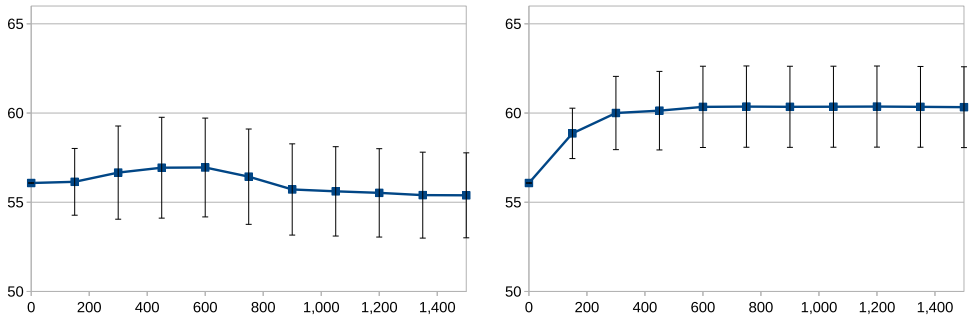
(e) *Max J Beh4* (f) *Modified Gradient Beh4*

Figure 5.24: Max J Off–Policy (left) and ModifiedGradient (right) REINFORCE baseline (x axis: $updates$, y axis: J) Behavior policies from Beh0 to Beh4 updates: $1\,500$, *total samples:* $10\,000$, *learning rate:* $5E-2$, *penalty coefficient:* 0.01



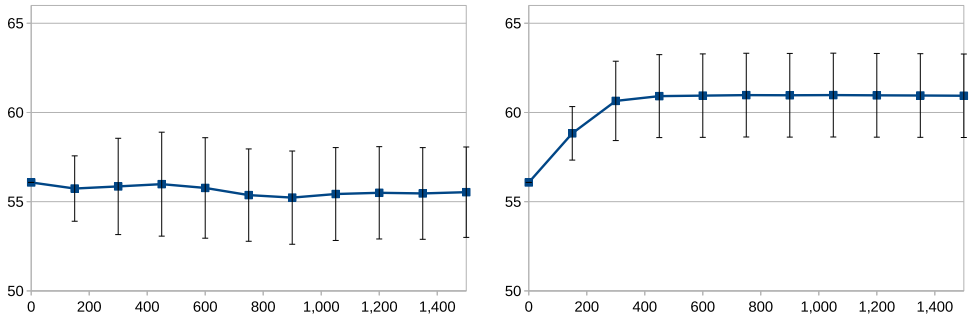
(a) *Max J Beh6*

(b) *Modified Gradient Beh6*



(c) *Max J Beh8*

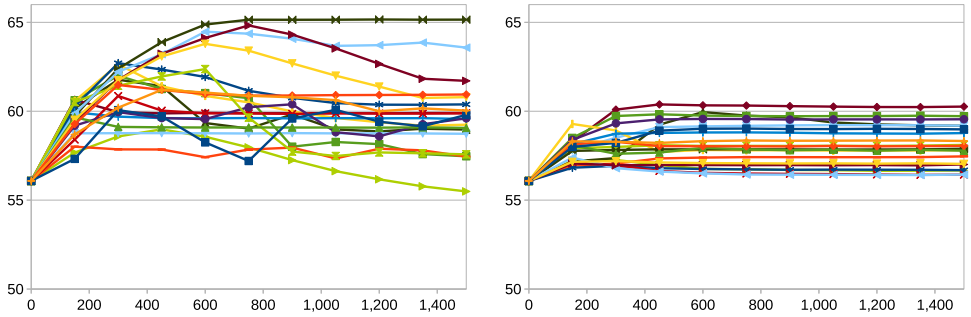
(d) *Modified Gradient Beh8*



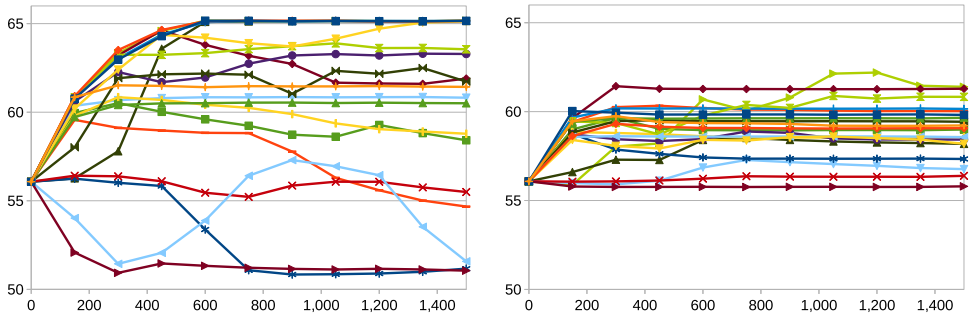
(e) *Max J Beh10*

(f) *Modified Gradient Beh10*

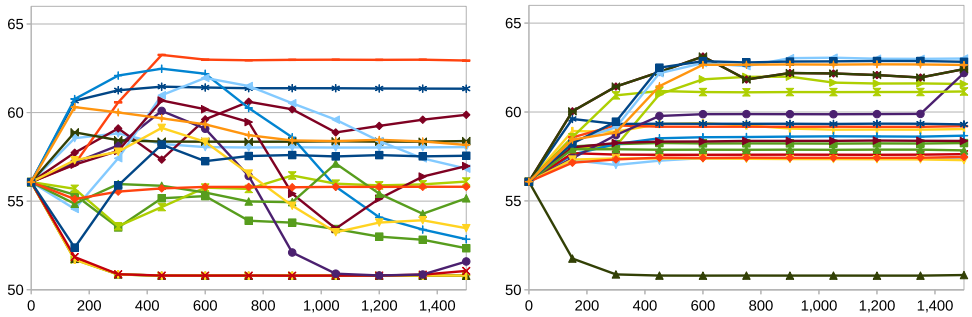
Figure 5.25: Max J Off–Policy (left) and ModifiedGradient (right) REINFORCE baseline (x axis: $updates$, y axis: J) Behavior policies from Beh6 to Beh10 updates: $1\,500$, *total samples:* $10\,000$, *learning rate:* $5E-2$, *penalty coefficient:* 0.01



(a) *Max J Beh0* (b) *Modified Gradient Beh0*

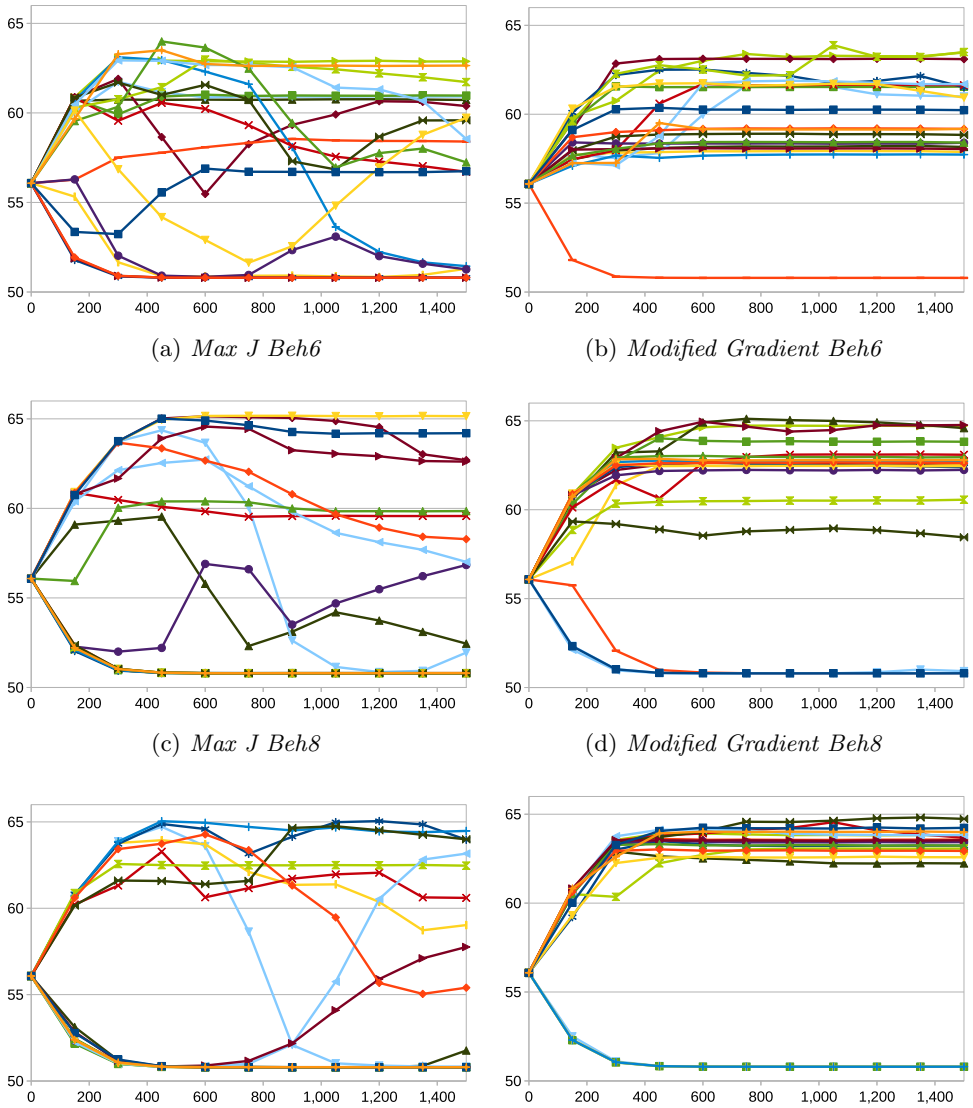


(c) *Max J Beh2* (d) *Modified Gradient Beh2*



(e) *Max J Beh4* (f) *Modified Gradient Beh4*

Figure 5.26: Max J Off–Policy (left) and ModifiedGradient (right) REINFORCE baseline (x axis: $updates$, y axis: J) Behavior policies from Beh0 to Beh4 updates: $1\,500$, total samples: $10\,000$, learning rate: $5E-2$, penalty coefficient: 0.01



(a) *Max J Beh6* (b) *Modified Gradient Beh6*

(c) *Max J Beh8* (d) *Modified Gradient Beh8*

(e) *Max J Beh10* (f) *Modified Gradient Beh10*

Figure 5.27: Max J Off–Policy (left) and ModifiedGradient (right) REINFORCE baseline (x axis: $updates$, y axis: J) Behavior policies from Beh6 to Beh10 updates: $1\,500$, *total samples:* $10\,000$, *learning rate:* $5E-2$, *penalty coefficient:* 0.01

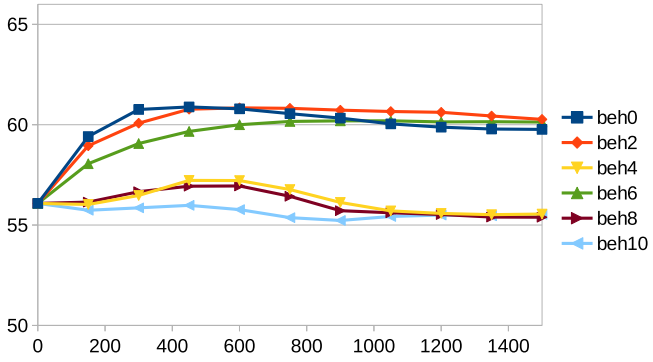
Figure 5.28 summarizes the previous graphs pointing out the main benefits of the modified gradient technique (Figure 5.28b) respect to the standard optimization of the expected return (Figure 5.28a).

The modified gradient succeeds in increasing the performance when the distance between target and behavior policies is high. This is due to the fact that those behavior policies are actually better than the others (see Section 5.1), but the importance weights variance hides this truth when the standard algorithms are used. Another important achievement is that the learning curves are now sorted according to the expected return of each behavior policy, while without the penalty term there is not any particular order.

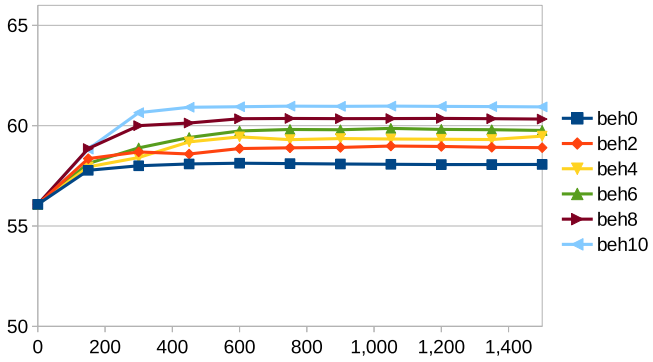
When the behavior policies are close to the initial target parameterizations (e.g., Beh0 in Figure 5.24b and Beh2 in Figure 5.24d) the modified gradient seems to perform worst then the standard maximization of the expected return. This is due to the fact that in all the previous experiments the penalty coefficient p is fixed, thus the learning process stops when the target policy is quite distant from the behavior policy and not because the training dataset does not contain further information to exploit. To solve this problem is enough to gradually decrease the penalty term as shown if Figure 5.29.

The orange lines are the learning curves of the modified gradient algorithm version in which after every 1 000 updates the coefficient p is multiplied by 0.9 (decaying factor). It can be noticed that the performance are higher respect to both the risk–neutral J maximization and the Modified Gradient algorithm with fixed penalty coefficient.

The last consideration is related to tests made using penalty coefficients p too high. Here the algorithm interrupts the learning process since the first updates and its expected performance are represented with flat lines in Figure 5.30. here even a small step toward a new policy is perceived as a significant risk.

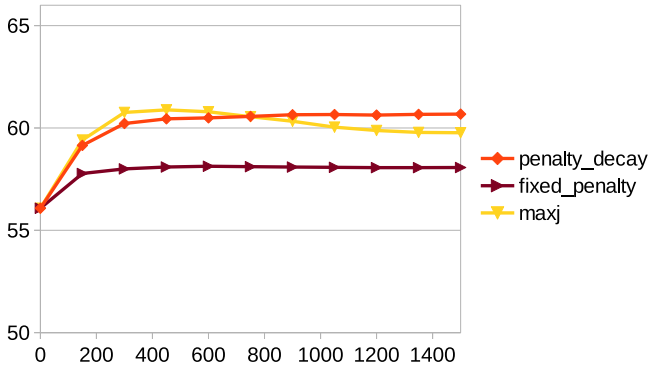


(a) *Max J Off–Policy*

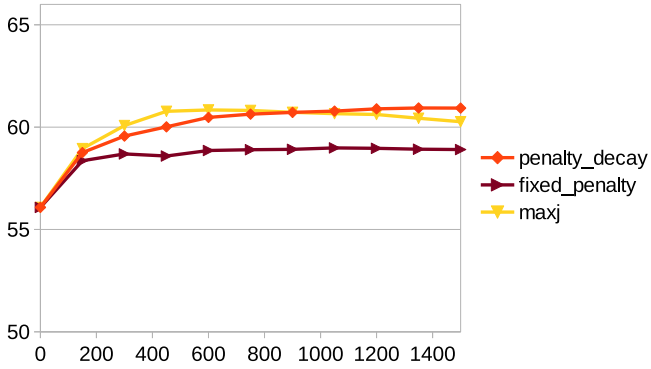


(b) *Modified Gradient* $p = 0.01$

Figure 5.28: Performance with different behavior policies
REINFORCE baseline (x axis: $updates$*, y axis:* J*)*
updates: $1\,500$*, total samples:* $10\,000$*, learning rate:* $5E-2$



(a) *Beh0*



(b) *Beh2*

Figure 5.29: Decaying penalty parameter in Modified Gradient REINFORCE baseline (x axis: $updates$, y axis: J)
Behavior policies from Beh6 to Beh10
updates: $1\,500$, *total samples:* $10\,000$, *learning rate:* $5E-2$, *penalty coefficient:* 0.01, *coefficient decaying:* 0.9 *every* $1\,000$ *updates*

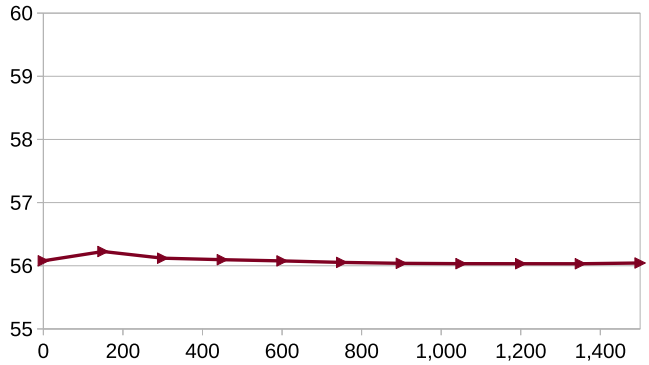


Figure 5.30: ModifiedGradient Beh0 with high penalty coefficient REINFORCE baseline (x axis: $updates$, y axis: J) updates: $1\,500$, *grad_trajectories/update:* 100, *eval_trajectories/update* 100, *steps/trajectory:* 50, *learning rate:* $5E-2$, *penalty coefficient:* 100 *total samples:* $10\,000$

Chapter 6

Conclusions

This chapter provides a broad overview of the entire work presented in this thesis. The final part suggests the direction of future research and development which could be taken on the basis of this work.

This thesis has provided theoretical and empirical contributions in the field of risk–sensitive Reinforcement Learning, moreover we have pointed out the need of operating in off–line context in order to not directly expose the agent to a risky environment. The transition to off–policy scenarios introduces new limitations in the learning process, in fact the agent may have insufficient experience to well estimate the performance of some policies due to the fixed batch dataset.

We have empirically proved that GPOMDP–like gradient formulation and the baseline technique are useful tools in decreasing the variance of gradient estimations. In off–line scenarios this is a huge advantage because with the same dataset it can happen that only the optimized algorithms can lead to a good solution while the other ones are not able to learn.

Although we have derived all the formulations in off–policy context, all the benefits of the methods presented can be exploited also in on–policy learning as we have shown in the tests in Chapter 5. Moreover the off–policy algorithms have reached a performance very similar to the one of the corresponding on–policy versions and the risk averse techniques have proved to be effective. In particular when a penalty parameter can be tuned in the objective function (e.g., the mean–variance and the modified gradient criteria) both on–policy and off–policy algorithms are able to produce different results, which vary from the risk neutral behavior to the extreme risk aversion.

Future works proposed in this topic have two main purposes:

1. enhance the performance of off–line off–policy learning;
2. use other definitions of risk as penalty term in the objective function.

Natural policy gradient techniques [5, 6, 13, 30, 38, 41, 67] use the Fisher-information matrix $F_{\boldsymbol{\theta}}$ to define the actor update step as $\Delta\theta \propto F_{\boldsymbol{\theta}}^{-1}\nabla_{\boldsymbol{\theta}}J$, where $\nabla_{\boldsymbol{\theta}}J$ is the "vanilla" policy gradient used in this thesis. A learning process based on natural policy gradients often converges significantly faster for most practical cases, thus it is a good idea to devise natural policy gradient algorithms able to operate in off–policy risk–averse scenarios.

Each algorithm implementation can be improved in order to take advantage both of on–policy and off–policy learning techniques. Hybrid versions can be developed to handle both batch data and on–line samples. In this way the training data can be slowly enlarged over time thanks to new interactions with the system, which are used in on–line learning during the current policy update while they will be treated as off–line batch data in all the future off–policy gradient estimations.

The Sharpe ratio and the mean–variance criterion are not complex risk measures, as we mention in Section 2.4 in the last years the percentile performance or the CVaR criterion has taken a prominent place in many studies. These measures allow to perform more accurate estimations of the risk, but they start from theoretical bases different from the ones presented in this thesis. Thus future work can be aimed to unify these families of approaches under a common structure or to exploit some percentile information also in variance related measures presented so far.

Bibliography

[1] Prashanth L. A. Policy gradients for cvar-constrained mdps. pages 155–169, 2014.

[2] Prashanth L. A. and Mohammad Ghavamzadeh. Actor-critic algorithms for risk-sensitive reinforcement learning. *CoRR*, abs/1403.6530, 2014.

[3] V. M. Aleksandrov, V. I. Sysoyev, and V. V. Shemeneva. Stochastic optimization. *Engineering Cybernetics*, 5:11–16, 1968.

[4] Eitan Altman and Inmanysituationsintheoptimizationofdynamicsystems Asingleutility. Constrained markov decision processes, 1999.

[5] S. Amari and S.C. Douglas. Why natural gradient? In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 2, pages 1213–1216 vol.2, 1998.

[6] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, February 1998.

[7] Olivier Bardou, Noufel Frikha, and Gilles PagÃ¨s. Computing var and cvar using stochastic approximation and adaptive unconstrained importance sampling. *Monte Carlo Meth. and Appl.*, 15(3):173–210, 2009.

[8] J. Baxter and P. Bartlett. Direct gradient-based reinforcement learning, 1999.

[9] Jonathan Baxter and Peter L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[10] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[11] Dimitri P Bertsekas. *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, third edition, 2007.

[12] Dimitris Bertsimas, David B. Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM Rev.*, 53(3):464–501, aug 2011.

[13] Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, November 2009.

[14] V. S. Borkar. Q-learning for risk-sensitive control. *Math. Oper. Res.*, 27(2):294–311, May 2002.

[15] Vivek S. Borkar and Rahul Jain. Risk-constrained markov decision processes. In *CDC*, pages 2664–2669. IEEE, 2010.

[16] Jaquette S. C. Markov decision processes with a new optimality criterion: discrete time, 1973.

[17] Dotan Di Castro, Aviv Tamar, and Shie Mannor. Policy gradients with variance related risk criteria. In *ICML*. icml.cc / Omnipress, 2012.

[18] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Number 2 in Genetic and Evolutionary Computation. Springer-Verlag New York, Inc., New York, USA, September 2007.

[19] Stefano P. Coraluppi and Steven I. Marcus. Risk-sensitive and minimax control of discrete-time, finite-state markov decision processes. *AUTOMATICA*, 35:301–309, 1999.

[20] Erick Delage and Shie Mannor. Percentile optimization in uncertain markov decision processes with application to efficient exploration. In *In Proceedings of the Twenty-fourth International Conference on Machine Learning (ICML-07*, pages 225–232, 2007.

[21] Erick Delage and Shie Mannor. Percentile optimization for markov decision processes with parameter uncertainty. *Operations Research*, 58(1):203–213, 2010.

[22] Dmitri A. Dolgov and Edmund H. Durfee. Approximating optimal policies for agents with limited execution resources. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 1107–1112. Morgan Kaufmann, 2003.

[23] E. Feinberg and A. Shwartz. Handbook of markov decision processes methods and applications. 2002.

[24] J. A. Filar, L. C. M. Kallenberg, and H. M. Lee. Variance-penalized markov decision processes. *Math. Oper. Res.*, 14(1):147–161, mar 1989.

[25] Peter Geibel and Fritz Wysotzki. Risk-sensitive reinforcement learning applied to control under constraints. *CoRR*, abs/1109.2147, 2011.

[26] Robert Givan, Sonia M. Leach, and Thomas L. Dean. Bounded-parameter markov decision processes. *Artif. Intell.*, 122(1-2):71–109, 2000.

[27] Peter W. Glynn. Likelihood ratio gradient estimation: an overview. pages 366–375, 1987.

[28] Peter W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM*, 33(10):75–84, oct 1990.

[29] Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. In *Journal of Machine Learning Research*, pages 1471–1530. MIT Press. In press, 2001.

[30] Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(6):1291–1307, 2012.

[31] Tobias U. Hauser, Reto Iannaccone, Susanne Walitza, Daniel Brandeis, and Silvia Brem. Cognitive flexibility in adolescence: Neural and behavioral mechanisms of reward prediction error processing in adaptive decision making during development. *NeuroImage*, 104(0):347 – 354, 2015.

[32] Matthias Heger. Consideration of risk in reinforcement learning, 1994.

[33] Daniel Hernandez-Hernandez and Steven I. Marcus. Risk sensitive control of markov processes in countable state space, 1996.

[34] Ronald A. Howard and James E. Matheson. Risk-sensitive markov decision processes. *Management Science*, 18(7):356–369, 1972.

[35] Ronald A. Howard and James E. Matheson. Risk-sensitive markov decision processes. *Management Science*, 18(7):356–369, 1972.

[36] G. Iyengar. Robust dynamic programming. *Math. Oper. Res*, 30:257–280, 2004.

[37] Sobel M. J. The variance of discounted markov decision processes, 1982.

[38] F. Jurcicek. Reinforcement learning for spoken dialogue systems using off-policy natural gradient method. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pages 7–12, Dec 2012.

[39] Leslie Pack Kaelbling, Michael L. Littman, and Andrew P. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[40] M. Kahn and A. W. Marshall. Methods for reducing sample size in Monte-Carlo computations. *Oper. Res.*, 1:263–278, 1953.

[41] Sham Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14:1531–1538, 2001.

[42] Hisashi Kashima. Risk-sensitive learning via minimization of empirical conditional value-at-risk. *IEICE Transactions*, 90-D(12):2043–2052, 2007.

[43] Hajime Kimura, Kazuteru Miyazaki, and Shigenobu Kobayashi. Reinforcement learning in pomdps with function approximation. In Douglas H. Fisher, editor, *ICML*, pages 152–160. Morgan Kaufmann, 1997.

[44] Hajime Kimura, Masayuki Yamamura, and Shigenobu Kobayashi. Reinforcement learning by stochastic hill climbing on discounted reward. In Armand Prieditis and Stuart J. Russell, editors, *ICML*, pages 295–303. Morgan Kaufmann, 1995.

[45] A. Klopf. Brain function and adaptive systems-a heterostatic theory. 1972.

[46] J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Advances in Neural Information Processing Systems 22 (NIPS 2008), Cambridge, MA: MIT Press*, 2009.

[47] Sven Koenig and Reid G. Simmons. Risk-sensitive planning with probabilistic decision graphs. In Jon Doyle, Erik Sandewall, and Pietro Torasso, editors, *KR*, pages 363–373. Morgan Kaufmann, 1994.

[48] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[49] Augustine Kong, Jun S. Liu, and Wing H. Wong. Sequential Imputations and Bayesian Missing Data Problems. *Journal of the American Statistical Association*, 89(425):278–288, March 1994.

[50] Pavlo Krokhmal, Jonas Palmquist, and Stanislav Uryasev. Portfolio optimization with conditional value-at-risk objective and constraints. *JOURNAL OF RISK*, 4:11–27, 2002.

[51] Jian Li and Laiwan Chan. Reward adjustment reinforcement learning for risk-averse asset allocation. In *IJCNN*, pages 534–541. IEEE, 2006.

[52] A. Rupam Mahmood, Hado P van Hasselt, and Richard S Sutton. Weighted importance sampling for off-policy learning with linear function approximation. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3014–3022. Curran Associates, Inc., 2014.

[53] Petr Mandl. On the variance in controlled markov chains. *Kybernetika*, 07(1):(1)–12, 1971.

[54] Shie Mannor and John N. Tsitsiklis. Mean-variance optimization in markov decision processes, 2011.

[55] Peter Marbach and John N. Tsitsiklis. Simulation-based optimization of markov reward processes. *IEEE Trans. Automat. Contr.*, 46(2):191–209, 2001.

[56] Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[57] Oliver Mihatsch and Ralph Neuneier. Risk-sensitive reinforcement learning. *Machine Learning*, 49(2-3):267–290, 2002.

[58] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[59] Teodor Mihai Moldovan and Pieter Abbeel. Risk aversion in markov decision processes via near-optimal chernoff bounds.

[60] John E. Moody and Matthew Saffell. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4):875–889, 2001.

[61] Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. Nonparametric return distribution approximation for reinforcement learning. In Johannes FÃ$\frac{1}{4}$rnkranz and Thorsten Joachims, editors, *ICML*, pages 799–806. Omnipress, 2010.

[62] Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. Parametric return density estimation for reinforcement learning. *CoRR*, abs/1203.3497, 2012.

[63] Arne J. Nagengast, Daniel A. Braun, and Daniel M. Wolpert. Risk-sensitivity and the mean-variance trade-off: decision making in sensorimotor control. *Proceedings of the Royal Society of London B: Biological Sciences*, 278(1716):2325–2332, 2011.

[64] Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Oper. Res.*, 53(5):780–798, September-October 2005.

[65] Leonid Peshkin. Reinforcement learning by policy search, 2000.

[66] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2219–2225. IEEE, 2006.

[67] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7):1180–1190, 2008.

[68] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[69] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Adaptive step-size for policy gradient methods. In *Advances in Neural Information Processing Systems 26*. 2013.

[70] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 2015.

[71] M. J. D. Powell. Direct search algorithms for optimization calculations, 1998.

[72] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, New York, NY, April 1994.

[73] Reuven Y. Rubinstein. *Simulation and the Monte Carlo Method*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1981.

[74] G. A. Rummery and M. Niranjan. On-line q-learning using connectionist systems. Technical report, 1994.

[75] Andrzej Ruszczynski. Risk-averse dynamic programming for markov decision processes. *Math. Program.*, 125(2):235–261, October 2010.

[76] Conor Ryan and Maarten Keijzer, editors. *Genetic and Evolutionary Computation Conference, GECCO 2008, Proceedings, Atlanta, GA, USA, July 12-16, 2008*. ACM, 2008.

[77] Makoto Sato and Shigenobu Kobayashi. Variance-penalized reinforcement learning for risk-averse asset allocation. In Kwong-Sak Leung, Lai-Wan Chan, and Helen Meng, editors, *IDEAL*, volume 1983 of *Lecture Notes in Computer Science*, pages 244–249. Springer, 2000.

[78] William F. Sharpe. Mutual Fund Performance. *The Journal of Business*, 39(1):119–138, 1966.

[79] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *Automatic Control, IEEE Transactions on*, 37(3):332–341, 1992.

[80] Richard S. Sutton. Learning to predict by the methods of temporal differences. In *MACHINE LEARNING*, pages 9–44. Kluwer Academic Publishers, 1988.

[81] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, March 1998.

[82] Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1057–1063. The MIT Press, 1999.

[83] Aviv Tamar, Dotan Di Castro, and Shie Mannor. Policy evaluation with variance related risk criteria in markov decision processes. *CoRR*, abs/1301.0104, 2013.

[84] Aviv Tamar, Dotan Di Castro, and Shie Mannor. Temporal difference methods for the variance of the reward to go. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Proceedings*, pages 495–503. JMLR.org, 2013.

[85] Aviv Tamar, Yonatan Glassner, and Shie Mannor. Policy gradients beyond expectations: Conditional value-at-risk. *CoRR*, abs/1404.3862, 2014.

[86] Aviv Tamar and Shie Mannor. Variance adjusted actor critic algorithms. *CoRR*, abs/1310.3697, 2013.

[87] Jie Tang and Pieter Abbeel. On a connection between importance sampling and the likelihood ratio policy gradient. In *NIPS*, pages 1000–1008, 2010.

[88] Jie Tang and Pieter Abbeel. On a connection between importance sampling and the likelihood ratio policy gradient. In John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, editors, *NIPS*, pages 1000–1008. Curran Associates, Inc., 2010.

[89] Back Thomas. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, Oxford, UK, 1996.

[90] Francesco Vallascas and Jens Hagendorff. The risk sensitivity of capital requirements: Evidence from an international sample of large banks*. *Review of Finance*, 17(6):1947–1988, 2013.

[91] Nikos Vlassis, Marc Toussaint, Georgios Kontes, and Savas Piperidis. Learning model-free robot control by a monte carlo em algorithm. *Autonomous Robots*, 27(2):123–130, 2009.

[92] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.

[93] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, May 1992.

[94] Evert Wipplinger. Philippe jorion: Value at risk the new benchmark for managing financial risk. *Financial Markets and Portfolio Management*, 21(3):397–398, 2007.

[95] Huan Xu and Shie Mannor. On robustness/performance tradeoffs in linear programming and markov decision processes, 2007.