UNIVERSITÉ DE LORRAINE

**UFR mathématiques et informatique** **École doctorale IAEM Lorraine**

**Département de formation doctorale en informatique**

# Des algorithmes presque optimaux pour les problèmes de décision séquentielle à des fins de collecte d'information

## THÈSE

présentée et soutenue publiquement le 4 février 2013

pour l'obtention du

**Doctorat de l'université de Lorraine**

**(spécialité informatique)**

par

Mauricio Alejandro Araya López

**Composition du jury**

| | |
|---|---|
| *Président :* | Michèle Sébag (Directrice de recherche, CNRS, Orsay) |
| *Rapporteurs :* | Joelle Pineau (Associate Professor, McGill University, Montréal)<br>Régis Sabbadin (Directeur de recherche, INRA, Toulouse) |
| *Examinateurs :* | Philippe Chassaing (Professeur, Université de Lorraine, Nancy),<br>Florent Teichteil-Königsbuch (Ingénieur de recherche, ONERA, Toulouse)<br>Olivier Buffet (Chargé de recherche, INRIA, Nancy)<br>Vincent Thomas (Maître de conférence, Université de Lorraine, Nancy)<br>François Charpillet (Directeur de recherche, INRIA, Nancy) |

**Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503**

Loria
Laboratoire lorrain de recherche en informatique et ses applications

Mis en page avec la classe thloria.

# Acknowledgements

First and foremost, I am indebted to my advisors. François, Vincent and Olivier are not only a marvelous first-class research team to work with, but also they are incredibly humble, keen and hard-working human beings. I want to acknowledge my advisor François Charpillet for taking that leap of faith and accepting my candidature to the PhD position only based on my CV. Also, he was always there for helping me with all my paperwork, both scientific and administrative, and I have to say they are numerous when you are a foreigner student. My co-advisor Vincent Thomas was always very concerned and helpful with my well-being in France, both professionally and personally, and I am very grateful of his willingness to always find a slot for me in his tight calendar. I owe to him most of my professional development during the thesis, not only by his wise guidance for this dissertation, but also by encouraging me to retake teaching activities, and to improve my presentation and scientific dissemination skills. Without any doubt, Olivier Buffet has been also my thesis co-advisor, even though formalities only allow one. He has selflessly guided this dissertation with his vast knowledge of the domain, his implacable scientific rigour, and his horizontal work-team philosophy. I owe to him most of my scientific development during the thesis, always willing to discuss any idea, always helping me with technical or philosophical entanglements, and always proposing new research directions and references. I also want to acknowledge his willingness to replace me in those conferences to which I was not able to attend, and his invaluable help with both English and French grammar, which I know it could be tedious and very time consuming.

I want to acknowledge the jury for a remarkable work, starting with Joelle Pineau and Régis Sabbadin for accepting reviewing my thesis during the holidays, and also for their insightful comments and corrections before the thesis defense. In the same line, I want to thank the rapporteurs and examiners for their scientifically rigorous but constructive questions during the thesis defense, hoping to have answered accordingly to their expectations.

These three and a half years of thesis were possible only by a strong institutional support from several entities. First, I have to acknowledge the Chilean government through CONICYT and the Embassy of France in Chile, for granting me the scholarship to pursue the PhD degree here in France. Also, I want to acknowledge the Lorraine University and the LORIA/INRIA Grand Est institute, which was not only the physical place where this dissertation was developed, but also a diverse and keen community of people that build the perfect environment for research. This includes not only other researchers, engineers and directives, but all the staff of the canteen, security, secretaries and maintenance.

In this context, I have to particularly acknowledge all the support of the MAIA Team which hosted me during my thesis, specially to Alain Dutech and Bruno Scherrer for their insightful comments and suggestions that have unblocked my research in several occasions. In the same line, I want to thank my office-mate Arnaud Glad for his help and patience during all the stages of the thesis. Also, I want to thank all the permanent researchers from whom I have received advise and recommendations, like Joerg, Nazim, Olivier, Amin and Vincent. Obviously, I have to thank my fellow PhD students and post-docs for their help and friendship all along this time, including Jano, Tomás, Olivier, Antoine, Mohamed, Cedric, Arsène, Boris, Jilles, Nikos, Emil, and so on. I want also to specially thank the secretaries of the team, Céline, Christel and Véronique, hoping they can forgive my laziness and incompetence in administrative affairs.

I want to thank also my friends from the CORTEX team, from the KIWI team, and specially to the Chilean friends that helped me during all this time here in France, including Mauricio, Victor, Daniela, Carlos, Isabel, Alvaro, Cristian, Andres and so on.

In a more personal domain, I want to thank my parents and siblings for always supporting me in all my academic and personal endeavors, and for making me the person I am through their example of excellence, correctness and kindness in all the aspects of life.

Finally, I want not only to acknowledge my wife Carolina for her infinite patience, support, and help during the thesis, but indeed to dedicate this dissertation to her. She has been the latent driving force of all this work, being strong and supportive in all those moments where the stress and tiredness threatened to overwhelming me, and been cheerful and joyous on each one of my small accomplishments. I owe her much more beyond this dissertation, so no acknowledgment nor dedication are enough to thank her for being always by my side.

*"With magic, you can turn a frog into a prince.*
*With science, you can turn a frog into a Ph.D and you still have the frog you started with."*
***Terry Pratchett, Ian Stewart and Jack Cohen*** — The Science of Discworld.

# Preface

When I was searching for a PhD position, my range of interest was wide open. Astro-engineering, robotics, multi-core compilers, evolutionary computation and swarm intelligence were just some of the topics that interested me at the moment. I did my Master thesis in real-time distributed systems, so in some sense I was already interested on decision theory, but in those days I did not even knew what an MDP was. Also, I had an enormous respect to the statistical learning and information theory fields, yet at the time I honestly believed that working in such topics was ahead of my abilities.

The turning point was the work done with my friend and colleague Nicolas Barriga about path planning, which introduced me to the amazing field of decision and planning. Very soon the Bellman's principle of optimality crossed my way, leaving me completely awestruck by its simplicity and power. In my humble opinion, artificial intelligence must aim first to optimality, and then relax the conditions to make things computable affordable, not the other way around. Consequently, the work done by the MAIA Team fit perfectly my interests, so I decided to ask for a PhD position blindly through e-mail. Fortunately, François was courageous enough to accept my application without knowing me, so I hope to have met his expectations.

The original thesis subject was about active sensing, specifically motivated by the problem of finding the policy that a predator agent must follow to find a prey. In this early research, the difference between seeing the prey and knowing were prey is became evident, raising the question about information-gathering in POMDPs. Soon enough, the thesis focus changed from the soft concept of active sensing to the more general and quantifiable concept of information-gathering. Without even noticing it, I was working directly within the formerly unattainable field of information theory.

The second point of inflection was noticing that the n-armed bandit problem can be modelled as a POMDP. This apparently technical trifle hides the very important concept that *learning is planning*. This notion lead me to explore Bayesian reinforcement learning, and within, the bewildering basics of statistical learning theory. Under this prism, an active learning process becomes an information-gathering decision process, which motivates several research directions that goes far beyond this dissertation scope.

This dissertation is the culmination of the work that has taken place over a period of more than three years, on which I have learned more things that I have never imagined, not only about decision processes, information theory and machine learning, but also about life, the world and myself.

— *Mauricio Araya, January 2013*

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Nomenclature

| | |
|---|---|
| $x$ | Latin and Greek letters are used to denote variables |
| $\theta$ | Greek letters are usually used to denote parameter variables |
| $X$ | Capital latin letters are usually used to denote random variables |
| $\Theta$ | Capital Greek letters are to denote random parameters and variable size sets |
| $\mathcal{X}$ | Calligraphic capital letters are used to denote fixed size sets |
| $\mathbf{x}$ | Bold letters (latin or greek) are used to denote vectors |
| $\mathbf{X}$ | Capital bold letters (latin or greek) are used for denoting random vectors |
| $\mathbb{P}$ | Blackboard bold letters are usually used to denote probability distributions, but sometimes they are use to denote standard number sets like $\mathbb{R}$ or $\mathbb{N}$ |
| $X_t$ | The subscript $t$ (and sometimes $i$ or $h$) are used to denote different time versions of the object |
| $X^{(t)}$ | When the subscript space is already used, the superscripts $(t)$ is used to denote different time versions of the object |

# Chapter 1

# Introduction

## Contents



*"Nothing is more difficult, and therefore more precious, than to be able to decide."*
— Napoleon Bonaparte.

In general terms, this dissertation is about making rational decisions under challenging environments. This subject has been studied extensively through history, from the ancient Greek philosophers to the modern artificial intelligence scientists, yet the fundamental entanglements remain the same: besides the size of the problem, incomplete information and sequential decisions are the two other ingredients of a challenging problem.

In order to bound this broad subject, the focus is set in this dissertation on numerical methods, usually neglecting the symbolic, cognitive, biological and philosophical implications. However, this does not mean that what is presented here is inconsistent with these other areas of knowledge. Specifically, we address these problems from the Artificial Intelligence (AI) perspective, where the objective is to produce intelligent autonomous agents that can act rationally in challenging environments.

In the early beginnings of AI, people thought that, with the rapid development of fast processors and memory (i.e., Moore's law), the problem of creating generic intelligent artificial agents would be solved in a couple of decades. Reality has proven to be tougher than expected: several theoretical questions remain unsolved, some problems are proven to be undecidable, and the computational complexity of the solvable but hard problems usually overwhelms any computational system. Probably, this expectation gap is due to the initial underestimation of complex real-world environments (e.g., uncertain, dynamic, unbounded, large, etc.), which is the main stumbling block for AI agents so far. The confluence of several disciplines around AI, such as statistics, control theory, information theory, optimization, biology and neuroscience, has been the source of success in most of the complex problems where AI is applied nowadays. However, developing autonomous agents that make correct decisions under real-world environments is still a key challenge for AI.

## 1.1 Agent Perspective

In simple terms, an agent is an entity that perceives its environment through sensors, generating data that typically feeds an internal model of the environment used to infer and make decisions. These decisions are translated into actions performed by the actuators that modify the environment and/or the agent itself (see Figure 1.1).

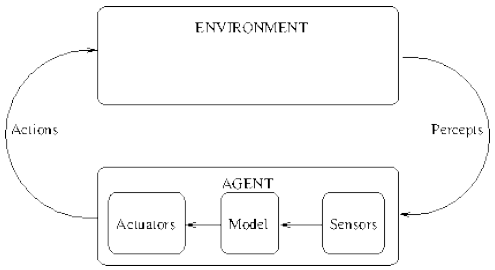



Figure 1.1: Generic representation of an agent, composed by an internal model of the environment, sensors to perceive the real environment, and actuators to act in the environment.

The difficulties that arise while building such agents are many, such as defining what is a correct decision or weighting the consequences of current decisions in the future decision phases. Moreover, uncertainty is present through all the elements of an agent, where noisy and limited sensors, imprecise actuators, narrow models and bounded resources are the rule more than the exception.

Ruling out those very simple agents that can be hand-designed to react to a small number of scenarios, there are two general types of agents: *goal-based* agents when the objective is to satisfy certain conditions, and *utility-based* agents when the objective is to optimize certain values (Russell and Norvig, 2003). In practice, goal-based problems can easily be cast as optimization problems and thus they are also compatible with the utility-based type. Consequently, the focus is set on agents that try to optimize a well-defined utility function.

Through this dissertation, the main modern tools to address the agent's sequential decision making problem with incomplete information are introduced and explained, considering the results from different research fields like computer science, statistics, control theory and information theory. From there, novel techniques that enhance previous results and solve new problems are proposed and substantiated through a proper theoretical justification and experimental studies.

## 1.2 Sequential Information Gathering

More precisely this dissertation is concerned with the complex problem of *sequentially gathering information* in the agent context. When an agent faces incomplete information, it usually has to choose between exploratory decisions to acquire more information for next decisions, and decisions that uses the current information for fulfilling the agent's objective. The analysis of the information gathering process needed to solve this twofold problem is sometimes neglected in the sequential decision literature because gathering information is usually presented not as an end, but just as a means to achieve the objective. In contrast, the main objective of this work is to analyze the information-gathering problem directly, by trying to answer the following general questions.

1. How to represent, measure and update the information held by an agent? How to optimally decide how to use this information?
2. Is (implicit) information gathering a solved issue in sequential decision problems? If not, what can be done?
3. Is (explicit) information gathering a sequential decision problem by itself, without the need of an objective? If yes, how can it be solved?

The first question is addressed using the statistical decision, inference and learning theory, by introducing the main concepts for representing knowledge and infer with it. This base ground of definitions, assumptions and representations serves as a framework for all the methods described afterwards.

The second question amounts to examining the state of the art of two classes of sequential decision problems with incomplete information: Partial Observable MDPs (POMDPs) and Reinforcement Learning (RL). Ideally, the agent's objective and the implicit information-gathering problem can be embedded

in a unique optimization criterion. However, solving such problems is so hard in practice that approximations are needed to scale up to real-world problems. In particular the focus is set on near-optimal approximations, because they offer theoretical guarantees about the closeness of the solution with respect to the optimum. Unfortunately, these approximations lead again to the same twofold problem: choosing between actions that gather information and actions that optimize the specific objective under the current information. This problem is known in RL as the exploration/exploitation dilemma.

The question of how to solve explicit information-gathering problems is again addressed for both POMDPs and RL. This dissertation defines, relates and tries to solve problems where no specific objective is defined besides explicitly gathering information. Even though this problem has been tangentially addressed by several communities, this dissertation presents a general framework to unify the modelling and formalization of such problems. Also, adaptations of the techniques described in the state of the art are presented to face sequential information-gathering problems.

## 1.3 An Example Problem

This section gives a simple example of the kind of problems that are addressed by this dissertation: the well-studied and famous $n$-armed bandit problem (Gittins, 1979).

**Example 1.3.1** ($n$-armed bandit problem)**.** *In this problem, an agent confronts a slot machine with n arms. Each arm has a different (and independent) probability of winning, yet these probabilities are unknown to the agent. The n-armed bandit problem consists in finding the best arm pulling strategy for achieving the maximum return from the slot machine. In other words, the objective is to find the (conditional) sequence of actions that optimizes the expected return. If the probabilities are given to the agent, the problem is trivially solvable because the best strategy is to always pick the arm with highest winning probability. Yet, as these probabilities are not given, the problem falls into the harder category of decision problems with incomplete information.*

This problem can be modelled through two paradigms:

- **Planning Paradigm:** At each step a "win" or "lose" *observation* of the internal state is obtained by pulling one of the arms (partial information of the winning probabilities). By developing all the possible future observations and actions, a conditional optimal *plan* can be obtained for deciding the best action depending on the current history of actions, wins and loses. The general framework used for this paradigm is POMDPs and is described in Chapter 3.
- **Learning Paradigm:** At each step a "win" or "lose" *sample* of one of the arms is obtained (a realization of a random variable). Then, a model for each arm (or sometimes for the complete system) can be *learned* using these samples, which can be used to infer the best action to perform. The general framework used for this paradigm is RL and is described in Chapter 5.

These two paradigms are usually treated as different frameworks, yet it can be shown that, for certain classes of problems, these two paradigms are equivalent. This is explicitly shown in Chapter 5 where Bayesian RL is modelled as a POMDP, showing that learning is planning.

In some specific cases an analytic solution can be found, such as the Gittins index solution for the $n$-armed bandit problem (Gittins, 1979). Unfortunately, not all POMDP and RL problems have analytic solutions. Indeed, POMDP and RL problems are proven to be generally hard to solve, so usually an extensive mathematical and algorithmic machinery is needed to (approximately) solve them. This dissertation focuses on this general case were no analytical solution can be easily derived for the problems.

**Example 1.3.2** ($n$-armed Bandit Diagnosis)**.** *In the n-armed bandit problem the agent's objective is to maximize the earnings in that slot machine. Consider now the related (but different) problem of actively learning the probabilities of the arms, no matter the money collected during this process. The objective is now to maximize the information about the bandit, and not to select the best arm as in the original problem. This can be useful, for example, to efficiently diagnose failing slot machines, or for the authorities to efficiently audit a casino. Under this setup, the agent confronts now an information-gathering problem where there is no preferred outcome between "win" and "lose", and actions must be*

*selected to increase the knowledge about the system and not to collect money. Unfortunately, properly defining this problem is not easy, because it depends on the representation of the agent's knowledge and on how is defined the preference relationship between different information states.*

Furthermore, if certain states or actions are *more important* than others, the information-gathering problem is cost-sensitive. For example, if there is a legal minimum of 1% chances of winning in any slot machine, an auditor agent should prioritize those arms that seem to be out of the law, and explore less those that are clearly legal. However, if all the arms are legal, or if the illegal ones are already disambiguated, the agent still has to explore the others to deliver an accurate report.

How to generally solve these information-gathering problems is the main concern of this dissertation, presenting how to adapt POMDP and RL algorithms to suit these problems. However, some advances in solving normal RL problems are also presented in Chapter 6, where information gathering is used as a means to approximately solve state-based reward problems.

## 1.4 Contributions

Besides the state of the art discussion of Chapters 2, 3 and 5, this dissertation concentrates its contributions in the remaining three other chapters.

In Chapter 4, about information gathering and POMDPs, the contributions are:

- A proper definition of sequential information-gathering decision problems, including examples and previous work which endorse its utility in real-world applications. Also, a sound formalization called $\rho$POMDP is presented, extending the POMDP definition to support information-gathering problems, and including a well-argued discussion of possible information-based reward functions for $\rho$POMDPs.
- Formal proofs that guarantee that the $\rho$POMDP value function is convex if the reward is convex, and that common information-based rewards can be approximated by PWLC functions within a bounded error.
- Ad-hoc adaptations of exact and point-based POMDP algorithms for $\rho$POMDPs, and the formal theorems that guarantee that these adaptations can solve these problems within an arbitrary small error.
- An experimental comparison of these information-lookahead strategies versus the random and myopic strategies proposed by other authors, and a proof of concept experiment using cost-sensitive information-based rewards.

Most of the early theoretical results of this section were published in (Araya-López et al., 2010a) and in (Araya-López et al., 2011b), yet most of the discussion and the empirical results are new contributions of this dissertation.

In Chapter 6, about exploration techniques for optimistic Bayesian RL algorithms, the contributions are:

- A novel algorithm, called BOLT, that is optimistic *about* the uncertainty and not only *in the face* of uncertainty, based on the Bayes Boost idea.
- Formal proofs that BOLT is strictly optimistic under certain parameters, and that it is PAC-BAMDP under the same condition. The bound found for BOLT is tighter than the bound for the reference PAC-BAMDP algorithm called BEB.
- An empirical study showing that BOLT is more efficient than BEB when parameters that ensure optimism are selected. Also, experiments show that BOLT is more robust to parameter tuning than BEB, meaning that a wrong parameter selection has less impact on the performance for BOLT than for BEB.

Both the theoretical and empirical results of this chapter were published in (Araya-López et al., 2012a) and in (Araya-López et al., 2012b), yet the discussion and insights of the chapter are new material of this dissertation.

Finally, in Chapter 7, about exploration techniques for MDP model learning, the contributions are:

- A sound formalism to model MDP model learning problems based on Bayesian RL and information-based rewards. This formalism relies on deriving the information-based reward from an information-

based criterion, which defines the problem objective using some information measure. This formulation optimally solves the problem in theory, but the intractability of Bayesian RL forces the use of approximate algorithms.
- The proposal of three properly justified information-based criteria, each one with the corresponding derived reward. Additionally, a simple hand-made reward is proposed to speed up computations, following the common ground elements of the theoretically derived ones.
- An adaptation of the EXPLOIT algorithm for information-based rewards, and an empirical study of all the rewards and criteria under this algorithm. The results show that this naive algorithm already outperforms the baseline random strategy, which is the most used technique for exploring MDPs.

Early versions of the results of this chapter were already published in (Araya-López et al., 2011a) and in (Araya-López et al., 2011c), yet these results were polished and extended for this dissertation.

## 1.5 Outline

In Chapter 2 the basics about statistical decisions, inference and learning theory from the agent's perspective are introduced, as they will be used in all the subsequent chapters. Then, Chapter 3 introduces the theory and algorithms to solve sequential decisions problems, with a strong emphasis on the partial observable variant and on its point-based (PB) solutions as they will be used in the following chapter. In Chapter 4 the main contributions about sequential information-gathering decision problems are presented, including (1) a novel formulation called $\rho$POMDPs, (2) theoretical results about this formulation and (3) an empirical study of how to adapt PB algorithms to information-based rewards.

Moving on from POMDPs to RL, Chapter 5 introduces the very basic reinforcement learning theory in order to quickly focus the discussion on model-based Bayesian RL. This chapter introduces the approaches used to approximately solve the Bayesian RL problem, focusing on (probably correct) near-optimal algorithms as this theory will be used afterwards. Chapter 6 proposes a simple and robust BRL algorithm called BOLT, based on the optimism in the face of uncertainty principle. This algorithm is proven to be probably approximately correct in the Bayesian sense, and more robust than previous work in terms of parameter tuning.

Finally, Chapter 7 addresses the MDP model learning problem, which can be modelled as an information-gathering problem with an unknown transition model. This chapter combines the idea of information-based rewards of Chapter 4 with the BRL formulation of Chapter 5, giving better results than simple strategies like random exploration.

# Chapter 2

# Optimal Decisions Under Uncertainty

## Contents



*"Concision in style, precision in thought, decision in life"*

— Victor Hugo.

This chapter provides an agent-based perspective of optimal statistical decisions (DeGroot, 1970), within the rich mathematical framework called statistical inference. The roots of statistical inference can be tracked up to Bernoulli, Bayes and Laplace on the sixteenth and seventeenth centuries, but Fisher (1922) was the first to present a unified framework for inference and estimation in the early years of the last century. This framework, which is a mathematical formalization for reasoning under uncertainty, allows to compute numerical estimates through a simple set of probability theory axioms and to update them when new evidence is available. Specifically, probability distributions can be used to represent subjective knowledge states, known as beliefs, which lead to several benefits for constructing agents. First, they provide a *concise* representation of complex data using a manageable set of parameters. Second, the probability theory axioms provide a *precise* method for reasoning about new evidence, in particular the Bayes rule. Third, with the use of utility functions, the *decision* problem under uncertainty can be addressed numerically as an optimization problem. Also, the information contained by a probability distribution can be measured, allowing to compare the information within different beliefs.

While the specific details of the statistical models used in this dissertation will be introduced and explained, it is assumed, on the other hand, that the reader has a good understanding of statistics foundations such as set theory, probability theory, random variables, expectations, Bayes rule, etc. If not, please refer to the first chapters of (DeGroot, 1970) or to a general statistical inference book like (Casella and Berger, 2001).

## 2.1 Uncertainty, Inference and Learning

It is widely accepted that probability theory is a proper way of representing uncertainty, and although other formalisms have been proposed [1], probabilistic models are by far the most used framework for systems dealing with uncertainty, being ubiquitous nowadays. In particular, parametric distributions let represent and generalize complex data within a bounded quantity of parameters. The parameters of a distribution are *statistics* that summarize the knowledge of that distribution. When the assumption of a *family* of distributions (e.g., Gaussian) is made for a random variable, the parameters must be fully informative in order to be useful. This is called a *sufficient statistic* (Fisher, 1922). For example, the mean and variance are together a sufficient statistics for the family of Gaussian distributions, because the complete distribution can be specified through these parameters. Through this dissertation, it is assumed that all the used parameters are sufficient statistics.

Even though parametric distributions are suitable to compactly represent uncertainty in general, it is important to distinguish the source of uncertainty to understand correctly a probabilistic model. Statisticians make a strong difference between *aleatory variability* and *epistemic uncertainty*, and in the next two sections, the differences of these two types of uncertainty are explained, including how to represent them using probabilistic models. In the rest of the section, the basics of *statistical learning* and *conjugate priors* are introduced, as they will be used through this dissertation.

### 2.1.1 Aleatory Variability

In general, a model of a real-world system always has some structural uncertainty, due to the fact that it is almost impossible to consider all the factors that are really involved. Indeed, a model is by definition a simplification of reality, implying that the neglected real-world variables can make the system to perform uncertainly under the model's view. This type of uncertainty is called *aleatory variability*, because the unknown factors make the system vary from the model in an (apparently) aleatory way. *Stochastic modelling* is a way of reasonably representing this uncertainty by using probability distributions, assuming that the real-world cannot be exactly predicted, yet can be statistically analyzed.

**Definition 2.1.1** (Stochastic Function)**.** *A function $g(x;\theta)$ is* stochastic *if, for a fixed value of variable $x$ and fixed parameters $\theta$, the outcomes vary depending on the conditional probability distribution $\mathbb{P}_{x,\theta}$. This leads to considering the outcomes as a random variable $Y$, where*

$$Y = g(x;\theta).$$

*Therefore, a stochastic function can be described by using the conditional probability notation:*

$$Pr(Y|X = x) = \mathbb{P}_{x,\theta}.$$

A typical example is when a deterministic function $g(x)$ is corrupted by a random noise $\Upsilon$ representing all the other (uninteresting or unmanageable) factors that modify the value of the function, so that

$$Y = g(x;\theta) = g(x) + \Upsilon, \qquad Pr(\Upsilon) = \mathbb{P}_\theta,$$

where $P(\Upsilon = \epsilon) = \mathbb{P}_\theta(\epsilon)$ is described by some probability density function (pdf) $f(\epsilon;\theta)$ with parameters $\theta$ (e.g., for the Gaussian pdf the parameter $\theta = \langle \mu, \sigma \rangle$). Here, as $\Upsilon$ is a random variable, the outcome $Y$ is also a random variable.

Stochastic models can be used in several ways, like for computing statistics such as the mean or variance using expectations, or to predict the probability of correlated quantities. For example, consider two correlated random variables $X$ and $Y$, with known probability distributions $Pr(X|Y = y)$ and $Pr(Y)$. Then, the probability of $Y = y$ given that $X = x$ was observed can be computed by using the Bayes rule as follows

$$Pr(Y = y|X = x) = \frac{Pr(X = x|Y = y)Pr(Y = y)}{Pr(X = x)}, \tag{2.1}$$

1. Such as Dempster-Shafer's theory of evidence, possibility measures, ranking functions, etc.

where the probability of $X = x$ can be computed using the law of total probability,

$$Pr(X = x) = \int_{y \in \mathcal{Y}} Pr(X = x|Y = y)Pr(Y = y)dy.$$

These kinds of models are popular in several fields such as communication theory and control theory, because real-world systems (e.g., noisy channels or noisy sensors) can be efficiently modelled and operated using probability distributions.

### 2.1.2 Epistemic Uncertainty

The models of the past section are very useful to represent *unknown factors* that corrupt the observations, but there is another type of uncertainty that arises from *known factors* that are unobservables. These known factors are modelled using latent (or hidden) variables, meaning that these variables cannot be observed, but their relationship to other variables (latent or observables) is known. The lack of knowledge of the values of latent variables is called *epistemic* uncertainty, because it depends on the available *information* and not on non-modelled factors like in the previous section.

**Example 2.1.1** (Non-injectivity). *Consider a deterministic system with two variables $x$ and $y$. Let $y$ be a latent variable with a fixed but unknown value. Also, let $x$ be an observable variable known to be related to $y$ by some non-necessarily injective function $x = f(y)$. Observing the value of $x$ leads to discarding some possible values of $y$, but there might still be uncertainty about $y$. As the described system is completely deterministic, there is no aleatory variability, yet there is epistemic uncertainty about $y$.*

Probability distributions can be used to deal with this type of uncertainty, yet a more subtle interpretation of probabilities is needed. Now, a probability distribution represents the knowledge available about a latent variable. A consequence is that probabilities are not universal but subject-dependent, because even under the same sequence of observations, each subject could end up with different distributions depending on the initial information at hand. This is why probabilities that represent epistemic uncertainty are called *subjective probabilities*, because they always depend on the subject's prior knowledge.

Now, epistemic uncertainty can be modelled using *latent random variables*, and when new evidence is available this distribution can be updated using *Bayesian Inference*.

**Proposition 2.1.1** (Bayesian Inference). *Let $X$ be a latent random variable, with $Pr(X) = \mathbb{P}_\theta$. If the value of an evidence random variable $E = e$ is observed, where $E$ is related to $X$ through a known conditional distribution $Pr(E|X = x)$, then the epistemic distribution of $X$ can be updated using the Bayes rule as follows:*

$$Pr(X = x|E = e) = \frac{Pr(E = e|X = x)Pr(X = x)}{Pr(E = e)},$$
$$\text{with } Pr(E = e) = \int_{x \in \mathcal{X}} Pr(E = e|X = x)\mathbb{P}_\theta(x)dx.$$

*The corollary of this proposition is that the probability of $X = x$ after observing some evidence $E = e$, is proportional to the probability of observing evidence $e$ given $x$, times the probability of $x$ before observing the evidence:*

$$Pr(X = x|E = e) \propto Pr(E = e|X = x)\mathbb{P}_\theta(x).$$

Bayesian inference is used in many domains to deal with epistemic uncertainty, for example in graphical models such as belief networks, or in classification such as the naive Bayes classifier. Please note that the key concept of *Bayesian* inference is not the Bayes rule itself (which can be used in non-Bayesian scenarios like in Equation 2.1), but that uses a *prior* distribution as a base for the inference.

**Definition 2.1.2** (Prior and Posterior Distributions). *A prior distribution $\mathbb{P}_\theta$ is an epistemic distribution that encodes the prior knowledge in the parameter $\theta$. When this distribution is updated using the evidence*

*e, the new epistemic distribution* $Pr(X|E=e) = \mathbb{P}_{\theta'}$ *is called a* posterior *distribution with new parameters* $\theta'$.

In many domains, the distribution $\mathbb{P}_\theta$ is called the *belief* of $X$, to reinforce the idea that this distribution is epistemic, and that it is updated through new evidence like human beings beliefs.

### 2.1.3 Statistical Learning

In the models presented above, it is assumed that the parameters of the distributions are given, which is usually not the case in real-world applications. Statistical learning is the field concerned with inferring the distribution parameters of random variables when only a finite set of samples of the phenomenon is available. A classical statistical learning procedure is to obtain an approximation of the parameters by *maximum-likelihood estimation*

**Proposition 2.1.2** (Maximum-Likelihood Estimation)**.** *Let* $X$ *be a random variable, and* $\mathbf{x}$ *be the vector of available samples of* $X$*. Under the independent and identically distributed assumption (i.i.d) for the samples, a* maximum-likelihood *estimation of the parameters of* $X$ *is*

$$\theta^* \in \operatorname*{argmax}_\theta \mathcal{L}(\mathbf{x}|\theta) \qquad \text{where}$$
$$\mathcal{L}(\mathbf{x}|\theta) \propto \prod_i Pr(X = x_i|\theta)$$

*is called the* likelihood *function.*

This is an example of a *frequentist* approach, because loosely speaking it relies on how frequent are the values of observations to infer the parameters. Frequentist approaches are the hobbyhorse of statistical learning because they provide not only an estimate of the parameters, but also confidence intervals and other statistics about the certainty of the estimate.

*Bayesian learning* is an alternative approach to parameter inference that has gained ground only in the recent decades, and consists in extending the concept of Bayesian inference by going beyond the idea of parameters being variables: now the parameters are also *random variables* with an associated probability distribution. The late incorporation of these methods to the mainstream research can be explained by the strong criticism displayed by Fisher on the inverse probability concept in the work of Bayes and Laplace. Indeed, the term "Bayesian" was coined by Fisher as a negative appellative for considering parameters as random variables, which he thought was fundamentally wrong. The discussion about the pertinence of Bayesian and frequentist learning is broad and controversial, escaping from the scope of this dissertation.

Consider again the problem of learning the parameters of a distribution from samples like in Proposition 2.1.2, but using Bayesian learning instead of a frequentist approach.

**Proposition 2.1.3** (Bayesian Learning)**.** *Let* $\Theta$ *be a random variable representing all the possible parameters* $\theta$ *for* $X$ *with a prior distribution* $Pr(\Theta = \theta) = \mathbb{Q}_\phi$*. Also, let* $Pr(X = x|\Theta = \theta) = \mathbb{P}_\theta$ *be the conditional distribution of* $X$ *given the parameter* $\theta$*, and* $\mathbf{x}$ *be the vector of available samples of* $X$*. By assuming that the samples* $x_i$ *are i.i.d. and using Proposition 2.1.1, the updated distribution of* $\Theta$ *is*

$$\begin{aligned}
\mathbb{Q}_{\phi'}(\theta) &= Pr(\Theta = \theta|\mathbf{X} = \mathbf{x}) \\
&= \frac{Pr(\mathbf{X} = \mathbf{x}|\Theta = \theta)Pr(\Theta = \theta)}{Pr(\mathbf{X} = \mathbf{x})} \\
&\propto Pr(\Theta = \theta)\prod_i Pr(X = x_i|\theta) \\
&\propto \mathbb{Q}_\phi(\theta)\prod_i \mathbb{P}_\theta(x_i).
\end{aligned}$$

An interesting case is when $\mathbb{Q}_\phi$ is a uniform distribution. Then, by computing the most probable value of $\theta$, both Bayesian learning and maximum-likelihood estimation are equivalent:

$$\operatorname*{argmax}_\theta Pr(\Theta = \theta|\mathbf{x}) = \operatorname*{argmax}_\theta \mathcal{L}(\mathbf{x}|\theta).$$

Bayesian learning provides a more flexible representation for learning the parameters and the opportunity of encoding initial knowledge in the prior distribution. However, inferring the posterior parameters is not always tractable, and depends mainly on the chosen family of distributions. This is why special families of distributions are often used to ensure the computability of the posterior distribution.

### 2.1.4 Conjugate Prior Distributions

The previous section has intentionally neglected the discussion about the nature of the probability distributions, because the presented learning methods do not require any further assumption about these distributions. However, in practice, families of distributions (such as Gaussian, categorical, Dirichlet, etc.) are used to be able to represent the information within bounded structures.

For the frequentist approach, the parameters can be directly estimated using maximum likelihood because the probability density function is given by the chosen family of distribution. In contrast, the Bayesian approach requires a more elaborate method because it depends on the structure of prior distribution. A common way to tackle this prior dependence problem, is to use *conjugate prior* families, allowing to compute the Bayes rule for Bayesian learning in an efficient way under some mild assumptions.

Roughly speaking, the *conjugacy* property is what allows to compute the parameters of the parameter random variable in a closed form when a specific family of distributions is chosen. In Proposition 2.1.3 it is implicitly suggested that the parameters $\phi'$ of the distribution $Pr(\Theta = \theta|\mathbf{X} = \mathbf{x})$ can be computed by having $\phi$, but for doing this the pdfs of $\mathbb{P}_\theta$ and $\mathbb{Q}_\phi$ must be known. Moreover, it would be desirable that the prior and posterior parameters ($\phi$ and $\phi'$) share the same structure, and that the posterior distribution $\mathbb{Q}_{\phi'}$ belongs to the same family as the prior in order to compute the parameter update in a closed form. If this is the case, then the pdf of $\mathbb{Q}_\phi$ is a *conjugate prior distribution*.

**Definition 2.1.3** (Conjugacy Property)**.** *Consider the likelihood-based representation of the Bayes rule* $Pr(\Theta = \theta|\mathbf{X} = \mathbf{x}) = \mathcal{L}(\theta|\mathbf{X} = \mathbf{x})Pr(\Theta = \theta)$*. A prior distribution* $Pr(\Theta = \theta)$ *in the* $\mathcal{F}$ *family is called* conjugate *to the* likelihood $\mathcal{L}(\theta|\mathbf{x})$ *if the posterior distribution* $Pr(\Theta = \theta|\mathbf{X} = \mathbf{x})$ *is in the same family* $\mathcal{F}$*.*

In simpler words, the prior distribution must be "compatible" with the likelihood function in order to produce a posterior in the same family as the prior. The general result of Diaconis and Ylvisaker (1979) states that, if the likelihood belongs to the exponential meta-family of distributions, there exists a prior distribution family that is conjugate. The exponential meta-family is a broad family of distributions that includes several continuous distributions families such as Gaussian, exponential and Dirichlet, and discrete ones such as Bernoulli, Poisson and categorical. For each likelihood family there is at least one corresponding conjugate prior: the easiest example is a Gaussian likelihood, distribution known to be closed under convolution, so a Gaussian prior will produce again a Gaussian posterior.

## 2.2 Optimal Decisions

While the previous sections have described how to represent uncertainty, infer and learn using probabilistic models, this section introduces how to optimally make decisions by using these models. First, the basic theory of utilities and rewards is introduced as an extension of probability theory, in order to choose optimal decisions with respect to some real-world objective. Then, these ideas are extended to rewards that depend on intrinsic objectives, such as learning or gathering information.

### 2.2.1 Utility and Reward Functions

A *utility function* (DeGroot, 1970; Wald, 1950) is the mathematical formalization of preferences among consequences, reflecting the preference relationship between the random variable's value of consequences. With this, a measure of the utility of probability distributions of consequences can be obtained, establishing a preference relationship between these distributions. This idea, though simple, is the core principle of *decision theory*, because different decisions lead to different probability distributions of consequences, and therefore, utility-optimal decisions can be defined.

The first axiomatic development of utility was developed by Von Neumann and Morgenstern (1944), and was presented as a complement to probability theory. In the same decade, the theory of statistical decision functions was developed by Wald (1950), including fundamental concepts such as the risk of a decision and optimal decisions. From there, a rich statistical theory was developed regarding hypothesis testing and similar subjects within the Neyman and Pearson (1933) approach to statistics. The approaches of Wald, Neyman and Pearson, were also strongly criticized by Fisher (Lenhard, 2006), who had worked on experimental design as the framework for decisions (Fedorov, 1972). Nowadays, it is recognized that both approaches have their own merits, yet here, the focus is set on the theory of optimal statistical decisions rather than on optimal experiments.

**Definition 2.2.1** (Utility Function)**.** *A utility function is an extension to the random variable abstraction, where a utility value is assigned to each possible realization of a random variable that represents the possible consequences on a system after a decision. Formally, $u(w)$ is a bounded scalar function $u : \mathcal{W} \mapsto \mathbb{R}$, where $\mathcal{W}$ is a set of possible consequences. This function measures how useful is obtaining the consequence $w$.*

The original utility function definition (Von Neumann and Morgenstern, 1944), is only constrained to comply with a set of axioms of rationality, which basically ensures that the utility expectation over the probability distribution is consistent as a measure. In practice, choosing a bounded scalar utility function for which the expectation exists (like in Definition 2.2.1), is enough to comply with these axioms.

Let $W$ be a random variable over the set $\mathcal{W}$, representing a set of consequences, and $Pr(W) = \mathbb{P}$ be a probability distribution over the consequences, then the *expected utility* can be computed:

$$U(\mathbb{P}) = \mathbb{E}_W[u|\mathbb{P}] = \int_{w \in \mathcal{W}} u(w)\mathbb{P}(w)dw.$$

Please note that the elements on the set $\mathcal{W}$ can be arbitrarily complex structures, from simple elements such as monetary values, to more complex ones such as satisfaction questionnaires or possible sensors states of an aircraft. The only requirement is that for each possible element of $\mathcal{W}$ a utility value and a probability must exist.

**Definition 2.2.2** (Decisions, Outcomes and Consequences)**.** *A* decision $d \in \mathcal{D}$ *is a controllable event that produces* outcomes $y \in \mathcal{Y}$*. In the general stochastic setting, the same decision can produce different outcomes, but for a given decision $d$ and outcome $y$ only one consequence value $w \in \mathcal{W}$ can be assigned.*

**Definition 2.2.3** (Reward function)**.** *Let $\sigma(y, d)$ be the (deterministic) function that maps outcome-decision pairs to consequence values, $\sigma : \mathcal{Y} \times \mathcal{D} \mapsto \mathcal{W}$. Then a* reward function *is defined as $r(y, d) = u(\sigma(y, d))$.*

If $Y$ is a random variable of outcomes with distribution $Pr(Y = y) = \mathbb{P}_d$ when the decision $d$ has been chosen, a probability distribution over $\mathcal{W}$ is induced by $\mathbb{P}_d = \sigma(Y, d)$, which leads to an expected reward of

$$R(d) = \mathbb{E}_Y[r|d] = \int_{y \in \mathcal{Y}} r(y, d)\mathbb{P}_d(y)dy = U(\mathbb{P}_d).$$

**Definition 2.2.4** (Optimal Decision)**.** *An* optimal decision *is any decision $d^*$ for which the expected reward is maximal, i.e.,*

$$R(d^*) = \max_d R(d).$$

**Example 2.2.1** (Roulette I). *Imagine a gambler playing roulette, where the set $\mathcal{W}$ are the possible monetary consequences of playing, $\mathcal{D}$ are the possible betting options of the table and $\mathcal{Y}$ are the possible numbers on the roulette. For each gambling decision $d \in \mathcal{D}$, a probability distribution $\mathbb{P}_d$ of monetary consequences is induced by the betting configuration and the probability of getting each number in the roulette, e.g., $\mathbb{P}_d$ is always a uniform distribution independent of $d$. The gambler may want to optimally decide which is the best betting option for obtaining the highest payload. In this context, it seems natural to define a linear utility $U(w) = w$, which gives the expectation of $\sigma$ as the optimization criteria, making unnecessary the use of utilities. However, as the St. Petersburg paradox (DeGroot (1970),p.95) suggests, utility functions are often non-linear even in the simple case of monetary rewards, so in general a utility function that weighs the set of consequences $\mathcal{W}$ is needed. Despite this, in most of the applications the reward function $r : \mathcal{Y} \times \mathcal{D} \mapsto \mathbb{R}$ can be defined directly without explicitly defining $u(w)$.*

A more general setup is when the outcomes depend not only on the decision, but also on other variables of the system. This is commonly the case in most of the decision problems, where the decision is conditioned by some observations or facts. To represent this, let $r(x, d, y)$ be a *conditional* reward function that depends on a variable $x \in \mathcal{X}$, the decision $d$ and the outcome $y$. Then, the expected conditional reward is

$$R(x, d) = \mathbb{E}_Y[r|x, d] = \int_{y \in \mathcal{Y}} r(x, d, y)\mathbb{P}_{x,d}(y)dy,$$

and for a specific value of $x$, an optimal decision is

$$d_x^* \in \operatorname*{argmax}_d R(x, d).$$

**Example 2.2.2** (Roulette II). *To exemplify conditional rewards, consider that the gambler can observe some facts $x \in \mathcal{X}$ of the system, for example the initial speed of the roulette and the time when the ball is thrown. Even under these precise observations, there is still some aleatory variability due to the non-modelled factors such as the force of the throw, the friction of the air, etc. Therefore, $Pr(Y) = \mathbb{P}_{x,d}$ gives now a non-uniform probability distribution for the possible roulette numbers depending on the observed initial conditions, and therefore an optimal decision with respect to the value of $x$ can be obtained.*

Going a little further, if it is known that a variable $X$ exists but is unobservable, a latent random variable can be used to model it. Now, $X$ is a latent random variable with a distribution $Pr(X) = \mathbb{Q}_d$, so an optimal decision can be obtained by considering the expectation over $X$,

$$d^* \in \operatorname*{argmax}_d \mathbb{E}_X[R|d] = \operatorname*{argmax}_d \int_{x \in \mathcal{X}} R(x, d)\mathbb{Q}_d(x)dx. \tag{2.2}$$

**Example 2.2.3** (Roulette III). *For the roulette example, consider that the betting is closed before the roulette begins to spin, so the initial speed and the throwing time can be observed only after the decision is made. The gambler may infer, by previous observations or by common sense, that some initial roulette speeds and some ball throwing times are more likely than others, constructing hence the probability distribution for them. Please note that in general the probability distribution $\mathbb{Q}_d(x)$ may depend on betting configuration (which would mean that the croupier is cheating). Now, an optimal betting configuration can be selected even though the facts $x$ are unobservables.*

Depending on the field, the names (and signs) of the quantities described here vary, but in principle the same utility based concepts are used by statisticians, economists, mathematicians and computer scientist to construct reward functions.

### 2.2.2 Loss, Risk and Intrinsic Rewards

An interesting application of utility and reward functions is decision making in statistical learning (see Section 2.1.3). In the statistical decisions field, it has become somewhat standard to represent the reward function as a *loss* function $l(x, d, y) = -r(x, d, y)$, but this choice does not change anything in the theory besides the sense of the optimization, where an optimal decision is now one that minimizes

the expected loss. In this context, the expected loss $R(x,d) = \mathbb{E}_Y[l|x,d]$ is conveniently called the *risk* of deciding $d$.

The objective of decision making in statistical learning is to find the decision that produces accurate parameter estimates. For example, if a scientist can choose between two different experiments to investigate the same phenomenon, he must choose the one with less risk of producing a wrong estimate.

In this context, let $y$ be the data obtained after executing the decision $d$, and $\delta(y,d)$ be a *decision function* that produces an estimate $\hat{\theta}$ for the parameters of the target phenomena. For a given parameter $\theta$, the loss function can be rewritten as $l(\theta, \delta(y,d))$, and the risk function is

$$R(\theta, d) = \int_{y \in \mathcal{Y}} l(\theta, \delta(y,d)) \mathbb{P}_{\theta,d}(y) dy.$$

Typical loss functions are for example the Quadratic Loss (QL), $l_{QL}(\theta, \hat{\theta}) = \|\theta - \hat{\theta}\|_2^2$ for continuous parameters, or the 0-1 loss, $l_{0-1}(\theta, \hat{\theta}) = 1 - \mathbb{I}(\theta, \hat{\theta})$ for discrete parameters [2].

The previous equations define the risk of performing $d$ if $\theta$ is known, but usually this is not the case. However, an epistemic probability distribution over the values of $\theta$ can be constructed.

**Definition 2.2.5** (Bayes Risk)**.** *The* Bayes risk *is the straightforward application of Equation 2.2 to statistical learning loss functions, which in long-winded terms is,*

$$\mathbb{E}_\Theta[R(\Theta, d)|d] = \int_{\theta \in \Theta} \int_{y \in \mathcal{Y}} l(\theta, \delta(y,d)) \mathbb{P}_{\theta,d}(y) \mathbb{Q}_d(\theta) dy d\theta.$$

From here, an optimal learning decision can be obtained by minimizing the Bayes risk. This method is widely used in several learning scenarios, such as regression and classification (Bishop, 2006), where several decision and loss functions have been proposed.

In rough terms, these risk functions are ways of measuring the expected *information* about the parameters that a decision may produce, which is an epistemic objective rather than a real-world objective. Indeed, the square loss leads to choose decisions that reduce the expected variance of the epistemic posterior distribution, while the 0-1 loss leads to a decision that maximizes the expected probability of this distribution. These rewards will be called *intrinsic* rewards, because they focus on the internal representation of knowledge about the system, and not on external quantities of the system (such as monetary rewards). Although the line between intrinsic and external rewards is faint in some applications (like in cost-sensitive classification (Greiner et al., 2002)), there is often an intimate relationship between learning risk functions and information theory that shows that, in the end, the learning objective is always to reduce some measure of uncertainty. A more detailed discussion on this topic can be found in Chapters 4 and 7.

## 2.3 A One-step Agent-based Decision Model

This section comes back to the agent perspective of Section 1.1, using the statistical theory of the previous sections to deal with uncertain sensors and actuators. A probabilistic model of an agent can be defined to reason (and also learn) under uncertainty by modelling both aleatory variability and epistemic uncertainty. Given a set of actions, the agent must decide at each step which is the best action to perform, depending on some performance criterion that defines the agent's objectives. This section focuses on performing a *one-step* optimal decision at some time $t$, while Chapters 3 and 4 address the more general case of performing an optimal sequence of decisions.

### 2.3.1 Fully Observable Problems

Assume for the moment that an agent is equipped with perfect sensors, meaning that the agent has complete and deterministic access to all the modelled variables of the system. This situation is called a *fully observable* problem, and is worth studying because the more complex problem with imperfect

2. $\mathbb{I}(a,b)$ is an indicator function, being 1 if $a = b$ and 0 else.

sensors can be addressed as a generalization of this one. Anyhow, there are some specific applications where full observability modelling is suitable, such as some board games or computer games, and others where this assumption works as a good approximation technique.

To formalize this problem using the probabilistic models of this chapter, please note that only the sensors are assumed perfect, while the actuators and the system model are generally imperfect. Therefore, the aleatory variability induced by the non-modelled factors of the system dynamics and actuators, can be represented using an stochastic model like in Section 2.1.1.

First, let $\mathcal{S}$ be the set of possible *states* of the system, where the system involves not only the environment, but also the agent itself. Also, let $\mathcal{A}$ be the set of possible actions that the agent can perform through its actuators. The task of the agent is to choose an action $a \in \mathcal{A}$, which moves the system from a state $s \in \mathcal{S}$ to a state $s' \in \mathcal{S}$, where $s$ may be equal to $s'$. This is called a *transition* of the system. Please note that state transitions are not only driven by the actions of the agent: the dynamics of the environment may change the state of the system even under innocuous actions.

Unfortunately, even if $s$ is fully observable, the next state $s'$ may be uncertain due to the aleatory variability of the system dynamics and actuators. Consequently, the next state can be represented by a random variable $S'$, and similarly, the action can be represented also by a random variable $A$. Then, a stochastic function can be used to represent this conditional relationship between the current and next states given an action.

**Definition 2.3.1** (Transition Function)**.** *For any arbitrary values of $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$, the (stochastic)* transition function $T(s, a, s')$ *corresponds to a conditional probability distribution $Pr(S' = s'|A_t = a, S = s)$.*

Although the dynamics of the whole system may be unknown in some problems, it is always assumed that there exists *one* model for the transitions that properly explains its dynamics. For now, assume that $T$ has known distribution parameters.

For describing the objectives of the agent, a straight-forward application of utility theory of Section 2.2 can be used.

**Definition 2.3.2** ($\mathcal{S}$-reward)**.** *A* state-based reward *function (or shortly an $\mathcal{S}$-reward) is when the reward function only depends on the system state and the chosen action.*

These rewards are the common case where the function $r(s, a, s')$ is defined as a deterministic function that rewards the agent with a scalar value for transitioning from a known state $s$ to a state $s'$ when executing action $a$. For reasons of clarity and succinctness only deterministic reward functions will be used for the theory. Yet, in practice this is not a constraint, because stochastic reward functions can be converted to deterministic ones via expectation.

Here, the outcomes are the possible states of the system, and the probability distribution of these outcomes is the transition function $T$. Then, the expected reward of executing $a$ from a state $s$ can be computed as

$$R(s, a) = \mathbb{E}_{S'}[r(s, a, S')|a, s] = \int_{s' \in \mathcal{S}} r(s, a, s')T(s, a, s')ds'.$$

Then, a one-step optimal action $a^*$ at time $t$ for the state $s_{t-1}$ is simply

$$a^* \in \operatorname*{argmax}_a R(s, a).$$

There are several applications where this agent model is used over discrete and finite spaces, i.e., when $\mathcal{S}$ and $\mathcal{A}$ are discrete and finite. A common choice is using *categorical distributions*, which is a discrete distribution family where, for each possible value of the state random variable there is an associated probability parameter [3]. Therefore, the transition function $T(s, a, s')$ for a given state-action pair $(s, a)$, can be described by the conditional pdf $Pr(S' = s'|A = a, S = s) = f(s'|a, s; \theta) = \theta_{s,a}(s')$,

3. Please note that these distributions are usually misleadingly referred as discrete distributions, but categorical distributions are only one type of discrete distributions.

with $\sum_{s' \in \mathcal{S}} \theta_{s,a}(s') = 1$, and

$$R(s,a) = \sum_{s' \in \mathcal{S}} r(s,a,s')\theta_{s,a}(s')$$

The application of utility functions to artificial agents is sometimes attributed to Horvitz et al. (1988) who specifically defined that a rational agent must maximize the expected utility in the context of expert systems. However, different types of utility-based agents have been proposed since the early days of artificial intelligence (Minsky, 1961), and especially in the control theory and optimization domain (Bellman, 1954).

### 2.3.2 Learning the Transition Function

The agent's probabilistic model of the past section assumes that the distribution parameters of $T$ are known, which might not be the case in real-world environments. To overcome this problem, an additional *learning* process from samples is required. This section is focused only on inferring the values of unknown distribution parameters of $T$ when a set of samples is available, while in Chapter 5 the concepts of this section are extended to make optimal decisions when samples are obtained sequentially.

Consider an agent model with full observability and discrete state and action spaces, where the transition function represented by a categorical distribution with unknown parameters $\theta_{s,a}$ for each state-action pair $(s,a)$. Also, assume that the agent has explored the environment by following a certain strategy, for example by choosing random actions at each step. The result of this procedure is a trajectory of states and actions $s_0, a_1, s_1, \ldots, a_n, s_n$. For each state-action pair $(s,a)$, the set of next possible states can be estimated from the trajectory, $\mathcal{G}_{s,a} = \{s_i | s_{i-1} = s, a_i = a\}$ for all $i \in [1,n]$.

The objective is to learn the parameter vector $\theta_{s,a}$ of the transition function from the sample set $\mathcal{G}_{s,a}$. Using a frequentist approach, this is an optimization problem of the form

$$\begin{aligned}\hat{\theta}_{s,a} &= \operatorname*{argmax}_{\theta} \mathcal{L}(\theta | \mathcal{G}_{s,a}) \\ &= \operatorname*{argmax}_{\theta} \prod_{s' \in \mathcal{G}_{s,a}} \theta_{s'}^{s,a}\end{aligned}$$

subject to $\|\theta\|_1 = 1$ and $\theta_s \in [0,1]$ for all $s \in \mathcal{S}$.

On the other hand, the Bayesian approach considers that $\mathbf{\Theta}_{s,a}$ is now a *random vector*, and the sample set $\mathcal{G}_{s,a}$ is the evidence that updates the prior distribution $Pr(\mathbf{\Theta}_{s,a})$. As presented in Section 2.1.4, the prior distribution must belong to a family that is conjugate to the likelihood, which, by assuming independent samples, is given by

$$\mathcal{L}(\mathcal{G}_{s,a} | \mathbf{\Theta}_{s,a}) = \prod_{s' \in \mathcal{G}_{s,a}} Pr(S' = s' | S = s, A = a, \mathbf{\Theta}_{s,a}) \qquad = \prod_{s' \in \mathcal{G}_{s,a}} \mathbf{\Theta}_{s,a}(s')$$

which is again a categorical distribution. As the categorical distribution family is an instance of the exponential family, there exists a prior family that suits this specific case, namely the *Dirichlet distribution* (Diaconis and Ylvisaker, 1979).

The Dirichlet distribution is a natural way of representing "probabilities over distributions", or more precisely representing the uncertainty of a discrete probability distribution by a multi-dimensional continuous distribution.

**Definition 2.3.3** (Dirichlet Probability Density Function)**.** *The pdf of a Dirichlet distribution of a random vector* $\mathbf{X}$*, where* $\sum_i \mathbf{X}_i = 1$ *is given by*

$$D(\mathbf{X}; \phi) = \frac{\prod_i x_i^{\phi_i - 1}}{B(\vec{\phi})}$$

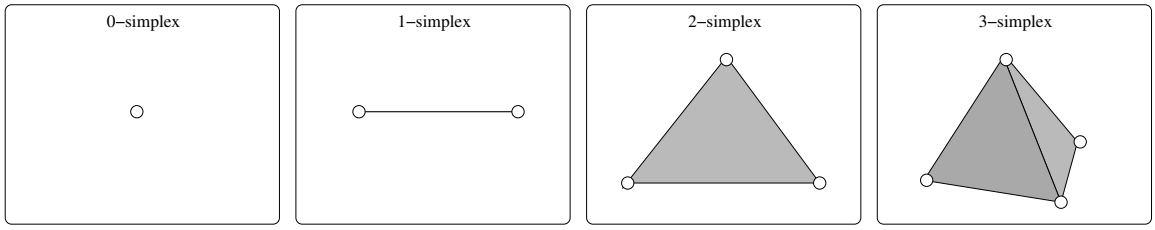



Figure 2.1: Graphical representation of simplexes of 0, 1, 2 and 3 dimensions.

*where*

$$B(\phi) = \frac{\prod_i \Gamma(\phi_i)}{\Gamma(\sum_i \phi_i)}$$

*is the generalized beta function. The gamma function is generally defined as* $\Gamma(z) = \int_0^\infty e^{-t} t^{z-1} dt$*, yet a much more simple factorial representation can be used when the arguments are integers:* $\Gamma(n) = (n-1)!$.

This distribution constrains the space of $\mathbf{X}$ to a *simplex* (see Figure 2.1) of the form

$$\Delta^{n-1} = \left\{ \mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n] \in \mathbb{R}^n \middle| \sum_{i=1}^{n} \mathbf{X}_i = 1, \mathbf{X}_i \in [0,1] \right\}, \tag{2.3}$$

which is exactly the parameter space of a categorical distribution.

In the agent context, for each state-action pair $(s, a)$, a Dirichlet prior can be defined as

$$Pr(\mathbf{\Theta}_{s,a} = \theta_{s,a}) = \mathbb{P}_{\phi_{s,a}}(\theta_{s,a}) = D(\theta_{s,a}; \phi_{s,a}) = \frac{\prod_{s' \in \mathcal{S}} \theta_{s,a}(s')^{\phi_{s,a}(s')-1}}{B(\phi_{s,a})}.$$

For example, a uniform prior distribution can be defined by $\phi_{s,a}(s') = 1, \forall s, a, s'$.

Using the Dirichlet prior distribution and the next-state sets $\mathcal{G}_{s,a}$, the parameters of the belief of the model $\phi$ can be updated using the Bayesian learning method of Section 2.1.3.

### 2.3.3 Partially Observable Problems

The transition function properly models the uncertain dynamics of the environment and the agent's actuators, but does not model the uncertainty that may be introduced by the sensors. If the state of the system cannot be directly perceived by the agent, but only partial or noisy observations can be obtained through its sensors, this is called a *partially observable* problem. Similarly to Definition 2.3.1, a stochastic function can be used to represent the sensor uncertainty.

**Definition 2.3.4** (Observation Function)**.** *If* $\mathcal{Z}$ *is the set of the possible observations, the stochastic* observation function *(or emission function)* $O(s, a, z)$ *is the conditional probability* $Pr(Z = z | S' = s', A = a)$ *with* $z \in \mathcal{Z}$*, and* $Z$ *the random variable representing the possible observations.*

Under this partial observability, Bayesian inference can be used to reason about the unaccessible state values. Let $S$ and $S'$ be latent random variables representing the current and next possible states under epistemic uncertainty, and $\phi$ be the parameters of the belief distribution $Pr(S) = \mathbb{P}_\phi$. For a given

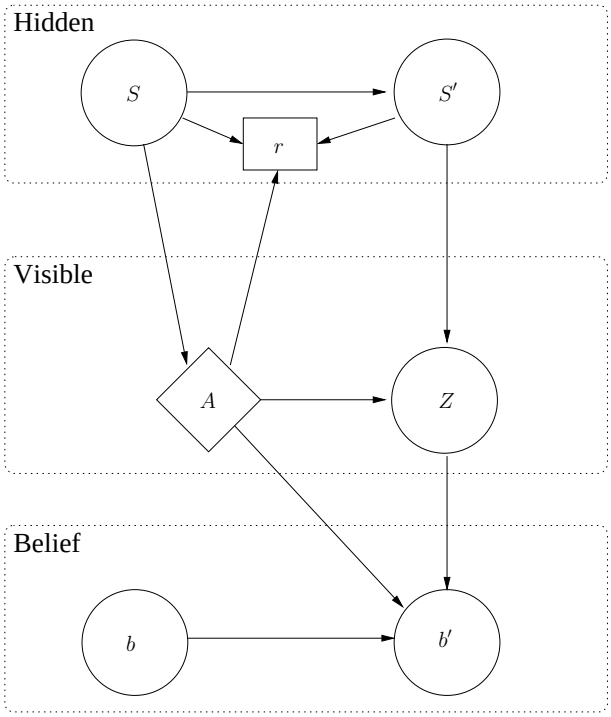



Figure 2.2: Graphical model of state-transitions, observations, rewards and belief propagation in a one-step partial observable decision problem.

*evidence* $(A = a, Z = z)$, the Bayes rule can be used to compute the belief $\phi'$:

$$\begin{aligned}
\mathbb{P}_{\phi'}(s') &= Pr(S' = s'|Z = z, A = a) \\
&= \frac{Pr(Z = z|A = a, S' = s')Pr(S' = s'|A = a)}{Pr(Z = z|A = a)} \\
&= \frac{O(s', a, z) \int_{s \in \mathcal{S}} Pr(S' = s'|A = a, S = s)Pr(S = s)ds}{Pr(Z = z|A = a)} \\
&= \frac{O(s', a, z) \int_{s \in \mathcal{S}} T(s, a, s')\mathbb{P}_\phi(s)ds}{\int_{s'' \in \mathcal{S}} O(s'', a, z) \int_{s \in \mathcal{S}} T(s, a, s'')\mathbb{P}_\phi(s)dsds''}.
\end{aligned} \tag{2.4}$$

**Definition 2.3.5** (Belief-state and Belief-propagation)**.** *The distribution $\mathbb{P}_\phi$ at any stage (priori or posteriori) is called a* belief-state*, and the process of computing the next belief-state using Equation 2.4 is called* belief propagation*.*

For convenience, the standard and shorthand notation for a belief-state will be used here, where $b = Pr(S) = \mathbb{P}_\phi$, and $b(s) = \mathbb{P}_\phi(s)$. Also, the update rule for the belief distribution described in Equation 2.4 will be denoted by the shorthand notation $b' = Bayes(b, z, a)$ for all $s'$ states. A summary of the relationship between the variables is graphically represented in Figure 2.2.

For the case of discrete and finite observations, the same categorical representation of Section 2.3.1 can be used for the observation function $O(s', a, z)$, where $Pr(Z = z|A = a, S' = s') = f(z|a, s'; \omega) = \omega_{s',a}(z)$ and $\sum_{z \in \mathcal{Z}} \omega_{s',a}(z) = 1$. Under this representation, the belief-state $b$ can be modelled also as an epistemic categorical distribution with a parameter vector $\phi = \mathbf{b}$ of dimension $|\mathcal{S}|$ such that $b(s) = f(s|\mathbf{b}) = \mathbf{b}_s$ and $\sum_{s \in \mathcal{S}} \mathbf{b}_s = 1$. It is important to stress out that, in general, $\mathbf{b}$ are the sufficient statistics of $b$, and that $b(s) = \mathbf{b}_s$ only in this specific case of categorical distributions.

Now, by applying these definitions to the inference Equation 2.4,

$$b'(s') = \frac{\omega_{s',a}(z)\sum_{s\in\mathcal{S}}\theta_{s,a}(s')b(s)}{\sum_{s''\in\mathcal{S}}\omega_{s'',a}(z)\sum_{s\in\mathcal{S}}\theta_{s,a}(s'')b(s)},$$

where $b'$ is trivially again a vector of probabilities that sums to 1. This means that if $T$, $O$ and $b_t$ are categorical distributions, then the belief propagation to $b'$ can be computed by applying this equation to all states $s' \in \mathcal{S}$.

For continuous spaces the problem is more subtle, because the choice of distributions is not straightforward like for discrete spaces. For instance, even though there are some applications where the transition and observation functions can be represented by unimodal distributions (e.g., Gaussian), the belief over the states can be a multimodal distribution. Therefore, more complex representations (e.g., Gaussian Mixtures) are needed to represent the belief. Due to this kind of complications, this dissertation will be concerned from now on only by discrete spaces. For a more detailed discussion about optimally decicion making in continuous spaces, please refer to Porta et al. (2006).

The discussion of the reward functions for partially observable problems is also more subtle, because the outcomes are in a strict sense the observations obtained after executing an action. However, as observations are only limited or imperfect queries of the system state, it is somewhat deceptive to reason directly about the utilities of observations. A forthright way to tackle this is to realize that observations inducebelief-states that can be used as outcomes.

**Definition 2.3.6** ($\Delta$-reward)**.** *A* belief-state reward function $\varrho(b,a,z)$ *is a scalar function that measures how useful is obtaining observation* $z$ *from the belief-state* $b$ *when executing action* $a$*.*

Therefore, similarly to the fully observable case, the expected belief-state reward can be computed as

$$\rho(b,a) = \mathbb{E}_Z[\varrho|a,b] = \sum_{z\in\mathcal{Z}} Pr(Z=z|A=a)\varrho(b,a,z) \tag{2.5}$$

and a one-step optimal action for a belief-state $b$ is

$$a^* \in \operatorname*{argmax}_a \rho(b,a).$$

Typically, partially observable problems are a generalization of the fully observable case, so if a state reward function $r$ is given, the belief-state reward can be computed as the expectation over the current and next states,

$$\begin{aligned}
\varrho(b,a,z) &= \mathbb{E}_{S',S}\left[r|a,b,z\right] \\
&= \sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}} Pr(S'=s', S=s|A=a, z=Z)r(s,a,s') \\
&= \sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}} \frac{Pr(Z=z|A=a,S'=s',S=s)Pr(S'=s',S=s|A=a)}{Pr(Z=z|A=a)} r(s,a,s') \\
&= \sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}} \frac{O(s',a,z)T(s,a,s')b(s)r(s,a,s')}{Pr(Z=z|A=a)}.
\end{aligned}$$

And, by replacing $\varrho$ in Equation 2.5, the expected belief-state reward ends up being

$$\begin{aligned}
\rho(b,a) &= \sum_{z\in\mathcal{Z}} Pr(Z=z|A=a)\sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}} \frac{O(s',a,z)T(s,a,s')b(s)r(s,a,s')}{Pr(Z=z|A=a)} \\
&= \sum_{s\in\mathcal{S}} b(s)\sum_{s'\in\mathcal{S}} T(s,a,s')r(s,a,s')\sum_{z\in\mathcal{Z}} O(s',a,z) \\
&= \sum_{s\in\mathcal{S}} b(s)R(s,a) = \mathbb{E}_S[R|a,b].
\end{aligned} \tag{2.6}$$

This result can be extended to problems with a reward function that also depends on the current observation (i.e., $r(s, a, s', z)$) by including the expectation over the observations in the computation of $R(s, a)$, like in the Cassandra (1998) formulation of rewards. Yet, this seems to be an unnatural modelling choice, because, for the same transition $(s, a, s')$, several observation outcomes can be obtained under pure aleatory variability, so assigning different utilities to these outcomes is hard to justify (yet not impossible).

# Chapter 3

# Sequential Decision Making

## Contents



*"Life is the sum of all your choices"*

— Albert Camus.

An autonomous agent usually has to make several decisions while executing a task. Making these decisions in isolation is obviously a bad idea, because in most of the cases the consequences of current decisions will affect later decisions. Therefore, reasoning about the one-step optimal expected reward like in the last chapter may not be the better strategy at all. Hence, one must reason about the *decision process* rather than one decision at each time step, and find a sequence of decisions that optimizes the expected sum of rewards of the whole process. This chapter focuses on introducing the theory and algorithms to solve sequential decision-making problems, with a special interest in those formalisms that combine the ideas of Chapter 2 for dealing with epistemic uncertainty.

## 3.1 Markov Decision Processes

A *decision process* is the general formulation for modelling problems that require sequential decision making, meaning that several decisions must be made one after another. As each decision must be made at some specific time, the first element of a decision process is a set of ordered time values, not necessarily finite, yet typically discrete (e.g., $\mathbb{N}$). At each of these time steps, a decision must be made, and the system moves to another state in the next time step. As the transition of the state may be uncertain, probabilistic models are commonly used.

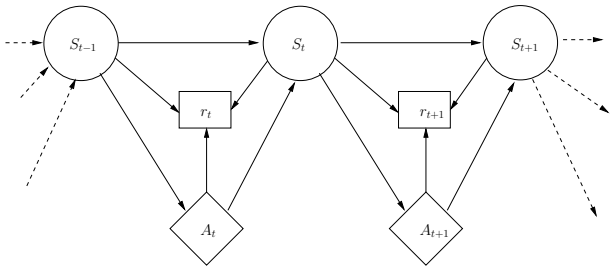



Figure 3.1: A 3-step view of a Markov Decision Process graphical model.

*Markov Decision Processes* (MDPs) (Bellman, 1954; Puterman, 1994) is the best known theoretical framework in this field, providing a sequential probabilistic model under the mild Markovian assumption. The Markov property is a common assumption in statistical modelling, where a random variable only depends on its neighboring influences, and not on all the variables of a system (Markov, 1907). In a sequential decision process, this means that the next state of the system only depends on the current state and action, and not on the past history, as is shown in Figure 3.1. This seems to be a strong assumption, but by considering the Markov property as a modelling constraint rather than as an assumption, one can argue that a wide range of sequential decision problems—if not all of them—can be modelled as MDPs.

**Example 3.1.1** (Train)**.** *For controlling a train to arrive safely and in time to the next station, at each time step a decision must be made between pushing the accelerator, the breaks, or doing nothing. A non-Markovian dynamics example is to consider that the state of the system is nothing more that the current position of the train. In fact, the next state of the system depends on the whole history of positions and actions, and not only on the last position and action, making the decision process hard to solve. In contrast, a Markovian dynamics example is to consider also the current speed of the train as part of the system state, so the next state depends now only on the last state and action, making the problem solvable through MDPs.*

**Definition 3.1.1** (Markov Decision Process)**.** *Formally, an MDP consists in a tuple* $\langle \mathcal{S}, \mathcal{A}, T, R, s_0 \rangle$*, where* $\mathcal{S}$ *and* $\mathcal{A}$ *are respectively the state and action spaces,* $T(s, a, s') = Pr(S_{t+1} = s' | S_t = s, A_{t+1} = a)$ *is a transition function,* $r(s, a, s')$ *is a scalar reward function, and* $s_0$ *is an initial state.*

This seems to be only an aggregation of the elements defined in Section 2.3.1, but as seen in Figure 3.1, the temporal and sequential aspects of the model are actually the key characteristics of MDPs. Indeed, some authors include explicitly in the tuple the time domain used. Yet, as this dissertation focuses only on discrete time problems [4] (i.e., $t \in \mathbb{N}$), there is no need to include the time domain in the definition.

In general, the transition function $T$ may depend on time, because the transition model might be different for each $t$. However, when $T$ is time-dependent the problem is hard to analyze and to solve, so usually this variability is modelled as part of the state, leading to a transition function that is invariant to time. This is called a *stationary* transition function, meaning that for all $t$ the transition function is the same. The same modelling constraint is usually applied to rewards, because stationary rewards are easier to analyze and represent. Nevertheless, the stationarity assumption is not a constraint of the model, but a property that makes problems easier to solve and represent.

### 3.1.1 Policies

In the agent context of Section 2.3, the objective was to find the optimal action with respect to a reward function, while in a sequential problem the objective is to find the optimal *sequence of actions* for the problem. However, due to the inherent uncertainty of state transitions, one single sequence of actions cannot be an optimal solution for all the possible outcomes of a problem, because different decisions may

4. For continuous-time models please refer to Chapter 11 of Puterman (1994).

be made depending on the path of states followed by the system. Consequently, rather than using a sequence of actions, a conditional decision rule is generally used.

**Definition 3.1.2** (Policy). *A* policy $\pi : \mathcal{S} \times \mathbb{N} \mapsto \mathcal{A}$ *is a function that maps a state* $s \in \mathcal{S}$ *at time* $t \in N$ *to an action* $a \in \mathcal{A}$*. This mapping from states to actions is a* solution *for an MDP.*

This definition is presented for the general case where the policy depends on time, but for some applications, formulations and algorithms, it is sufficient to use *stationary* policies, meaning that the mapping from states to actions is time invariant, simplifying the policy to $\pi : \mathcal{S} \mapsto \mathcal{A}$.

As a policy is a solution of an MDP, a performance criterion must be defined in order to compare different policies, and more specifically, to find an optimal one[5]. In Chapter 2, the performance criterion of an action was the expected reward when choosing that action. In a decision process, the performance criterion is the expected sum of rewards (also known as *return*) when following a specific policy.

### 3.1.2 Finite Horizon MDPs

When the number of time steps is fixed and known a priori, then the problem can be modelled as a *finite horizon MDP*. Here, the time domain is restricted to $[0, H] \subset \mathbb{N}$, and no rewards can be obtained after $H$. Therefore, a performance criterion for a policy $\pi$ can be defined as the expectation of the finite sum of rewards.

**Definition 3.1.3** (Expected Total Reward). *For a finite horizon MDP, the* expected total reward *performance criterion for a policy* $\pi$ *starting from state* $s_0$ *is defined as*

$$V_H^\pi(s_0) = \mathbb{E}_{S_{1:H}}\left[\sum_{t=0}^{H} r(S_t, \pi(S_t, t), S_{t+1}) \middle| S_0 = s_0, \pi\right].$$

The function $V_H^\pi$ is usually called the *value function*, and represents how valuable is a policy $\pi$ if the initial state is $s_0$.

### 3.1.3 Infinite Horizon MDPs

For those problems where the horizon is not fixed or known, one can reason over an *infinite horizon MDP*. However, the presented expected total reward might be infinite in the limit in some cases, so some additional information about the possible horizons is required. This information can be represented by a probability distribution, where $H$ is now a random variable of possible horizons. By taking the expectation over the possible horizons, and by assuming a stationary policy, the infinite horizon value function can be defined as follows:

$$\begin{aligned}
V^\pi(s_0) &= \mathbb{E}_H\left[V_H^\pi(s_0) | S_0 = s_0, \pi\right] \\
&= \lim_{n\to\infty} \sum_{h=1}^{n} Pr(H = h) \mathbb{E}_{S_{1:h}}\left[\sum_{t=0}^{h} r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi\right] \\
&= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}}\left[\sum_{h=1}^{n} Pr(H = h) \sum_{t=0}^{h} r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi\right] \\
&= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}}\left[\sum_{t=0}^{n} \left(\sum_{h=t+1}^{n} Pr(H = h)\right) r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi\right] \\
&= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}}\left[\sum_{t=0}^{n} \left(1 - \sum_{h=1}^{t} Pr(H = h)\right) r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi\right]. \qquad (3.1)
\end{aligned}$$

5. Please note that for some ill-defined problems the MDP may lack of optimal policies (e.g., if the performance criterion is not bounded). However, this dissertation considers only well-defined problems where at least one optimal policy exists.

Consider now that, at each step, the process may stop with probability $1-\gamma$. Then, as the probability of continuation is $\gamma$, the probability of the system to stop at horizon $h$ is

$$Pr(H = h) = \gamma^{h-1}(1-\gamma). \tag{3.2}$$

**Definition 3.1.4** (Discounted Expected Total Reward)**.** *For an infinite horizon MDP, the* discounted expected total reward *with a* discount factor $\gamma \in [0,1]$*, is defined as*

$$V^{\pi}(s_0) = \lim_{n\to\infty} \mathbb{E}_{S_{1:n}} \left[ \sum_{t=0}^{n} \gamma^t r(S_t, \pi(S_t), S_{t+1}) \mid S_0 = s_0, \pi \right]. \tag{3.3}$$

This discounted criterion is obtained by replacing the probability distribution described in Equation 3.2 by the general criterion of Equation 3.1:

$$\begin{aligned} V^{\pi}(s_0) &= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}} \left[ \sum_{t=0}^{n} \left( 1 - \sum_{h=1}^{t} \gamma^{h-1}(1-\gamma) \right) r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi \right] \\ &= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}} \left[ \sum_{t=0}^{n} \left( 1 - \frac{(1-\gamma^t)}{(1-\gamma)}(1-\gamma) \right) r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi \right] \\ &= \lim_{n\to\infty} \mathbb{E}_{S_{1:n}} \left[ \sum_{t=0}^{n} \gamma^t r(S_t, \pi(S_t), S_{t+1}) \middle| S_0 = s_0, \pi \right]. \end{aligned}$$

There are some specific problems where the $\gamma$ discount factor is also applied to finite horizon MDPs, where the rewards have naturally a geometric decay as part of the problem definition. In practice, this does not modify the way of solving these problems, so the $\gamma$ discount factor will be included in both finite and infinite horizon MDPs for generality. Also, there are some infinite horizon problems that do not require a discount factor to converge, such as stochastic shortest path problems (Bertsekas and Tsitsiklis, 1996). Even though the $\gamma$ discount factor is the most popular criterion for infinite horizon MDPs, other criteria may be chosen, such as the *average reward per time step* (Puterman, 1994).

### 3.1.4 Dynamic Programming

Consider now the problem of finding a policy $\pi^*$ that maximizes the performance criterion $V_h$ for a finite horizon $h$,

$$\pi^* \in \operatorname*{argmax}_{\pi} V_h^{\pi}(s_0).$$

A naive technique is to evaluate all the policies by computing the expectation of the sum of rewards, and keep the policy with the highest value. A smarter approach is to use the principle of optimality (Bellman, 1954), which states that "*an optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision*". This leads to a recursive interpretation of the *optimal value function* $V_t^*$ at time $t$, i.e., the value function of horizon $t$ by following the optimal policy $\pi^*$.

**Proposition 3.1.1** (Bellman Equation)**.** *For the presented performance criteria, the optimal value function verifies Bellman's optimality equation (Bellman, 1954) (for all $s \in \mathcal{S}$), which provides a recursive representation of $V_t^*$ for each time step $t$ under the form:*

$$V_t^*(s) = \max_{a\in\mathcal{A}} \left\{ \sum_{s'\in\mathcal{S}} T(s,a,s') \left[ r(s,a,s') + \gamma V_{t-1}^*(s') \right] \right\}.$$

This allows to compute recursively the optimal value function $V_h^*$, but not explicitly obtaining $\pi^*$. To compute the policy, an optimal value function that depends also on the next action can be defined.

**Definition 3.1.5** (State-Action Value Function)**.** *A* state-action value function $Q^{\pi}_{t+1}$ *is defined as*

$$Q^{\pi}_{t+1}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V^{\pi}_{t}(s'),$$

*and the optimal one is similarly,*

$$Q^{*}_{t+1}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V^{*}_{t}(s').$$

Now a greedy selection over the actions can be used to compute an optimal action at horizon $h$ for a given $s$,

$$\pi^{*}(s,h) \in \operatorname*{argmax}_{a \in \mathcal{A}} Q^{*}_{h}(s,a).$$

Doing this recursive computation for each horizon $t$ to compute $\pi^*$ is called *Dynamic Programming* (DP) (Bellman, 1957), and is a popular technique in several disciplines such as mathematics, economy, biology and computer science, because it allows to solve complex finite horizon problems such as (stochastic) shortest path problems or the genetic sequence alignment problem.

### 3.1.5 Value Iteration

This idea was extended by the same Bellman (1954), allowing to reason over infinite horizon decision processes using an iterative version of dynamic programming called *value iteration*. Value iteration consists in computing the values of $V^*$ by successive approximations using the Bellman equation until convergence. Formally,

$$V^{*}_{i}(s) = \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} T(s,a,s') \left[ r(s,a,s') + \gamma V^{*}_{i-1}(s') \right] \right\}. \tag{3.4}$$

Here, the left-hand side is the new value of $s$ at iteration $i$, and the $V^*_{i-1}(s')$ of the right-hand side is the old value of $s'$ at iteration $i-1$. A shorthand notation is to consider this Bellman equation as a function, where if $V_{i-1}$ is given, the function $\mathbb{B}(V_{i-1}, s)$ consists in applying Equation 3.4 for $s$. Then, the value iteration algorithm consists in computing this function for each state $s$ at each iteration $i$ (see Algorithm 1).

**Algorithm 1:** The Synchronous Value Iteration algorithm for MDPs.

**Input**: $T$, $R$, $\gamma$, $s_0$, $V_0$
**Output**: $V$
$V \leftarrow V_0$;
**while** $V$ *not converged* **do**
  $\hat{V} \leftarrow V$;
  **foreach** $s \in \mathcal{S}$ **do**
    $V(s) \leftarrow \mathbb{B}(\hat{V}, s)$;
  **end**
**end**

Notably, the same result can be obtained asynchronously by reusing the values computed for other

states (see Algorithm 2).

**Algorithm 2:** The Asynchronous variation of the Value Iteration algorithm for (infinite-horizon) MDPs

**Input**: $T$, $R$, $\gamma$, $s_0$, $V_0$
**Output**: $V^*$
$V \leftarrow V_0$;
**while** $V$ *not converged* **do**
  **foreach** $s \in \mathcal{S}$ **do**
    $V(s) \leftarrow \mathbb{B}(V, s)$;
  **end**
**end**

In the limit, these techniques always converge to a fixed point due to the Bellman operator contraction (Bertsekas and Tsitsiklis, 1996) when $\gamma < 1$, but more generally this function can be arbitrarily approximated within some accuracy $\epsilon$. The convergence criterion may vary, but the infinite norm of the difference between two successive iterations is the standard selection, where, if

$$\|V_i - V_{i-1}\|_\infty \leq \frac{\epsilon}{(1-\gamma)^2}$$

then the approximation has converged and is $\epsilon$ close to the optimal value function. If value iteration is applied to a finite horizon domain, then the convergence criterion of the algorithms is simply replaced by a finite number of iterations.

As stated before, an optimal policy is usually stationary for infinite horizon problems, and therefore it can be obtained simply by computing the optimal state-action value function after obtaining $V^*$ from the algorithm

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s')V^*(s'),$$

and choosing a best-valued action for each state,

$$\pi^*(s) \in \operatorname*{argmax}_a Q^*(s,a).$$

For the finite horizon case, all the vectors $V_h$ must be stored as the optimal policy may be non-stationary.

### 3.1.6 Policy Iteration

Policy iteration (Howard, 1960) is an alternative to value iteration, with the fundamental property that it converges to the optimal policy within a finite number of iterations, even for the infinite horizon case. Concisely, at each iteration $i$, a greedy policy $\pi_i$ is obtained from $Q_{i-1}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s')V_{i-1}(s')$, which is then evaluated to obtain $V_i(s) = V^{\pi_i}(s)$. When for two successive iterations $i-1$ and $i$ the policies are the same, then the policy iteration algorithm has converged, and $\pi_i = \pi^*$.

Several variations, improvements and theoretical guarantees have been published for policy iteration, including extensions to partially observable domains and reinforcement learning (Bertsekas and Tsitsiklis, 1996). Regardless the copious literature on this subject, here the focus is set only on value-iteration-based techniques because most of the techniques that are used through this dissertation are based on this principle. However, please be aware that policy-iteration-based techniques are technically and scientifically competitive with the ones presented in this document.

## 3.2 Partially Observable MDPs

The use of epistemic probability models on MDPs was not explicitly explored before Åström (1965). He introduces the subject as *MDPs with incomplete state information*, and naturally claims that one must

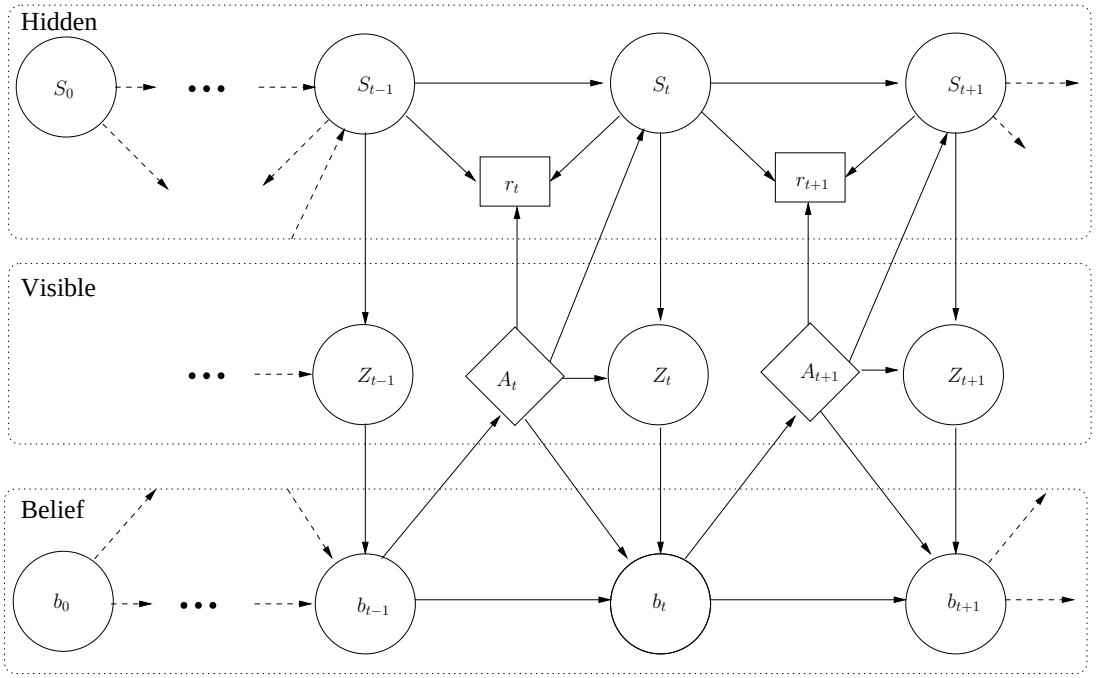



Figure 3.2: A 3-step view of a Partial Observable Markov Decision Process graphical model.

rely on probability distributions to represent the current knowledge about the state. The name Partially Observable Markov Decision Processes appears when this idea was further developed by Smallwood and Sondik (1973) for finite horizon problems and by Sondik (1978) for the infinite horizon case,

The basic idea is that the MDP framework can be extended to support partially observable sequential decision problems, by adding to the MDP tuple an observation space $\mathcal{Z}$, and a stochastic observation function $Pr(Z_t = z|S_t = s, A_t = a)$. Yet, the initial state $s_0$ is often unknown on these problems, so a belief-state is used to define $b_0$, an initial prior distribution over the states.

**Definition 3.2.1** (Partially Observable Markov Decision Process)**.** *A* Partially Observable MDP *(POMDP) (Åström, 1965; Smallwood and Sondik, 1973) consists in a tuple* $\langle \mathcal{S}, \mathcal{A}, \mathcal{Z}, T, R, O, b_0 \rangle$*, where* $\mathcal{S}$*,* $\mathcal{A}$*, and* $\mathcal{Z}$ *are the state, action and observation sets respectively,* $T(s, a, s') = Pr(S_{t+1} = s'|S_t = s, A_{t+1} = a)$ *is a transition function,* $r(s, a, s')$ *is a scalar reward function,* $O(s', a, z) = Pr(Z_t = z|S_t = s', A_t = a)$ *is an observation function, and* $\mathbf{b}_0$ *is an initial belief-state distribution.*

By following the same logic of Section 2.3.3, the system state can be represented as a latent variable $S_t$ at each time step $t$. This means that it is not possible to directly apply dynamic programming or value iteration to the underlying MDP, because the resulting policy needs a specific state to select the next action. Moreover, observations are not Markovian, meaning that an observation does not depend directly on the previous observation, but on the hidden value of the state, and therefore it is often sub-optimal to find a policy that maps observations to actions.

Fortunately, by using the belief-state abstraction of Definition 2.3.5, a proper Markovian process can be constructed over the beliefs, because the belief propagation of Equation 2.4, $\mathbf{b}' = Bayes(\mathbf{b}, z, a)$, is Markovian in the sense that the new belief-state depends directly on the previous belief and the new evidence (current observation and action), as shown in Figure 3.2. Here, the bold notation for the belief $\mathbf{b}$ will be used, because the theory will be presented for finite state spaces, where the belief-state $\mathbf{b}$ is a vector of probabilities.

Then, an MDP can be defined over the belief-state space in order to solve the underlying POMDP. Please remember that a belief-state is a probability distribution, so the belief-state space is defined as the simplex $\Delta$ of all possible probability distributions over $\mathcal{S}$ (see Equation 2.3).

**Definition 3.2.2** (Belief-state MDP). *Following the general definition of an MDP, a* belief-state MDP *is defined by the tuple* $\langle \Delta, \mathcal{A}, \tau, \rho, \mathbf{b}_0 \rangle$*, where* $\tau(\mathbf{b}, a, \mathbf{b}') = Pr(B' = \mathbf{b}' | A = a, B = \mathbf{b})$ *is the belief-state transition function,* $\rho(\mathbf{b}, a)$ *is the belief-state reward function, and* $\mathbf{b}_0$ *is the initial prior distribution over the states.*

If the transition and observation functions are stationary, the $\tau(\mathbf{b}, a, \mathbf{b}')$ function can be written in terms of $T$ and $O$ as follows

$$\begin{aligned}
\tau(\mathbf{b}, a, \mathbf{b}') &= Pr(B' = \mathbf{b}' | A = a, B = \mathbf{b}) \\
&= \sum_{z \in \mathcal{Z}} Pr(B' = \mathbf{b}' | A = a, B = \mathbf{b}, Z = z) Pr(Z = z | A = a, B = \mathbf{b}) \\
&= \sum_{z \in \mathcal{Z}} \mathbb{I}(\mathbf{b}', Bayes(\mathbf{b}, a, z)) \sum_{s' \in \mathcal{S}} \sum_{s \in \mathcal{S}} O(s', a, z) T(s, a, s') b(s). \qquad (3.5)
\end{aligned}$$

Similarly, if the reward function is also stationary, the belief-state reward function for all steps can be written as

$$\rho(\mathbf{b}, a) = \sum_{s \in \mathcal{S}} b(s) R(s, a), \qquad (3.6)$$

which is a result that has already been derived in Equation 2.6.

Now, the belief-MDP can be completely described using the elements of the POMDP tuple of Definition 3.2.1 and the solution techniques of the past section can be theoretically applied. Yet, even when the state, action and observation spaces are discrete, belief-state MDPs are multi-dimensional continuous-state MDPs, which are in general intractable to solve. Auspiciously, due to some regularities of the specific simplex space, belief-propagation and transition function, a $\epsilon$-close solution for belief-MDPs can be found in a finite number of iterations, yet typically with an overwhelming complexity Madani et al. (2003).

### 3.2.1 POMDP Policies

In the POMDP setting, it is not useful to define policies as functions of the state like in MDPs because states are hidden to the agent. A mapping of observations alone is not the solution either, because they are not Markovian and therefore they would provide sub-optimal solutions.

Åström (1965) has proved that a *control law* for POMDPs must be a function of the history of past observations and actions, showing that an optimal control law exists for any solvable POMDP. This is compatible with the belief-MDP definition of the past section, because a belief-MDP policy must depend now on belief-states, which are sufficient statistics of the history of observations and actions. In simpler words, a POMDP policy can be either a mapping from the possible histories to actions, or equivalently and more concisely, a mapping from belief-states to actions.

**Definition 3.2.3** (Belief-MDP Policy). *A belief-MDP policy is a function* $\pi : \Delta \times \mathbb{N} \mapsto \mathcal{A}$*, that maps, at each time step, a belief-state to an action. This mapping is a* solution *for an POMDP, and therefore there exists an optimal* solution *(policy) that maximizes the future return of the partially observable system.*

Please note that, similarly to MDPs, when the horizon of the problem is indefinite, infinite horizon POMDPs are used, again usually under the assumption of *stationary* belief-MDP policies.

The problem of belief-MDP policies is that they map a continuous multi-dimensional space to actions, which poses some practical problems on finitely representing them. However, by using some specific properties and assumptions, finite structures usually can be used. For example, in finite horizon problems the belief-state space is actually a finite set of points in the simplex, so a conditional plan for each possible belief-state can be constructed, represented as a tree structure of alternating levels of observations and actions. Similarly, finite state controllers can be defined for infinite horizon POMDPs when there exists cyclic optimal policies. Unfortunately, these simple types of representations suffer from combinatorial explosion when trying to find an optimal policy. For now, assume that a belief-MDP policy can be represented by a finite set of parameters, while in Section 3.2.4 an efficient representation is introduced.

### 3.2.2 Dynamic Programming for POMDPs

Section 3.1.4 presented the Bellman equation for solving MDPs using dynamic programming. This result can be extended to POMDPs by using the belief-MDP definition.

**Proposition 3.2.1** (Bellman Equation for Belief-MDPs). *The belief-MDP value function verifies Bellman's optimality equation (Bellman, 1954) (for all $b \in \Delta$), which provides a recursive representation of $V_t^*$ for each time step $t$ in the form:*

$$V_t^*(\mathbf{b}) = \max_{a \in \mathcal{A}} \left\{ \rho(\mathbf{b}, a) + \gamma \sum_{b' \in \Delta} \tau(\mathbf{b}, a, \mathbf{b}') V_{t-1}^*(\mathbf{b}') \right\}.$$

*By using Equations 3.5 and 3.6, the belief-MDP value function can be written in POMDP terms as follows:*

$$\begin{aligned} V_t^*(\mathbf{b}) &= \max_{a \in \mathcal{A}} \left\{ \sum_{s \in \mathcal{S}} b(s) R(s,a) + \gamma \sum_{b' \in \Delta} \sum_{z \in \mathcal{Z}} \mathbb{I}(\mathbf{b}', Bayes(\mathbf{b}, a, z)) \sum_{s' \in \mathcal{S}} \sum_{s \in \mathcal{S}} O(s', a, z) T(s, a, s') b(s) V_{t-1}^*(\mathbf{b}') \right\} \\ &= \max_{a \in \mathcal{A}} \left\{ \sum_{s \in \mathcal{S}} b(s) \left[ R(s,a) + \gamma \sum_{z \in \mathcal{Z}} \sum_{s' \in \mathcal{S}} O(s', a, z) T(s, a, s') V_{t-1}^*(Bayes(\mathbf{b}, a, z)) \right] \right\}. \end{aligned} \tag{3.7}$$

If there is a finite number of possible belief-states (like in the finite-horizon case), a policy can be obtained by computing the value function at each step $t$ for each possible $\mathbf{b} \in \mathcal{B}$, where $\mathcal{B} \subset \Delta$ is a finite set. However, solving POMDPs using this dynamic programming technique is only suitable for small problems like the ones presented in (Åström, 1965). To scale up, more elaborated methods are required like the ones presented in the following sections.

### 3.2.3 The PWLC Property

Probably the most remarkable advance for solving POMDPs was presented in (Smallwood and Sondik, 1973) and in (Sondik, 1978), where the authors show that the belief-MDP value function is *Piecewise-Linear and Convex* (PWLC) [6]. This mathematical property leads to a value function representation using a finite number of hyperplanes, and is the cornerstone of most of the modern POMDP solution algorithms.

A piecewise function is a function where the domain is partitioned in several subspaces, and for each subspace a different function is defined, which is only valid within that subspace. When a function is piecewise-linear, it means that the functions assigned to each partition are linear functions. For example, the absolute value function $f(x) = |x|$ can be represented as a piecewise-linear function as follows:

$$f(x) = \begin{cases} -x & \text{if } x \in (-\infty, 0], \\ x & \text{if } x \in (0, \infty). \end{cases}$$

The convexity property is a fundamental property in function analysis, and it simply means that the function has always a convex shape. Formally, a convex function must comply with,

$$f(\lambda x_1 + (1-\lambda) x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2), \tag{3.8}$$

for all $\lambda \in [0, 1]$. For example, by using the triangular inequality, it can be trivially shown that the absolute value is a convex function.

The description of piecewise-linear and convex properties are included here just as a reminder. Yet, for the rest of the dissertation it is assumed that the reader has some basic knowledge of mathematical function analysis.

**Proposition 3.2.2** ($V_t^*$ is PWLC). *The belief-MDP value function $V_t^*$ of Equation 3.7 is a PWLC function.*

6. In the limit ($t \to \infty$), this property does not hold, but it does for all other time steps

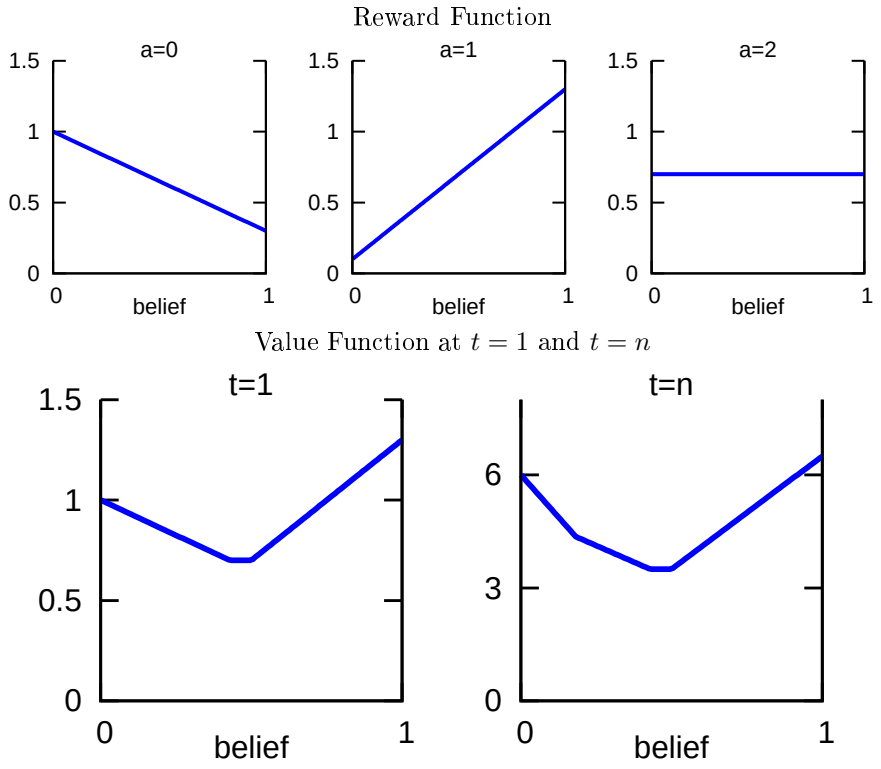



Figure 3.3: A PWLC propagation example, where the 3 upper figures are a belief-based view of an arbitrary reward function for actions 0, 1 and 2. The lower left corner represents the value function at $t = 1$ (the max of rewards), and lower right corner represents a possible value function at $t = n$.

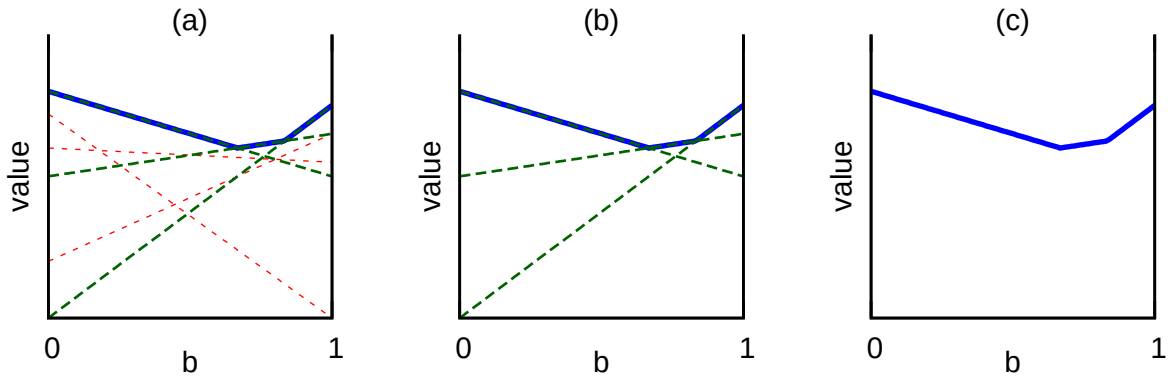



Figure 3.4: Γ-set representation of a value function in a 1-simplex space with (a) and without (b) unnecessary hyperplanes. (c) shows the actual value function. Here, (b) is a parsimonious representation of (c).

This proposition appears naturally because $\sum_{s\in\mathcal{S}} b(s)R(s,a)$ are linear functions with respect to $\mathbf{b}$ (trivially convex), because the combinations of linear functions are also linear functions. Also, the max operator preserves convexity and piecewise linearity, and moreover it naturally defines piecewise functions. This PWLC propagation is graphically represented on Figure 3.3. For a full proof of Proposition 3.2.2 please refer to (Cassandra, 1998) or (Sigaud and Buffet, 2010). Yet, in the next section this proposition becomes evident when choosing the right representation for the value function.

### 3.2.4 Exact Solution Methods

An interesting property of PWLC functions is that they can be represented by a finite set of vectors, each one representing a hyperplane in the whole space, but each one only valid in the subspace where they are maximal. For example, the absolute value function can be represented by

$$|x| = \max_{v\in\{1,-1\}} \{v \cdot x\}.$$

where $v$ can be read as a one-dimensional vector.

Similarly, and by Proposition 3.2.2, there exists a way of representing the value function at time $t$ by a set of vectors $\Gamma_t$ in the form

$$V_t^*(\mathbf{b}) = \max_{\alpha\in\Gamma_t} \{\mathbf{b}^\top \alpha\} \tag{3.9}$$

as shown in Figure 3.4(b). However, a vector representation of the value function equation is required for computing the set $\Gamma_t$ of *α-vectors*.

First, please note that when $V_0^*(\cdot) = 0$, $V_1^*(\cdot)$ can be directly written in the vector form

$$V_1^*(\mathbf{b}) = \max_a \{\mathbf{b}^\top \mathbf{r}^a\}, \tag{3.10}$$

where $\mathbf{r}_s^a = R(s,a)$. Thus, $V_1^*(\mathbf{b})$ can be represented in the form of Equation 3.9 by defining $\Gamma_1 = \{\mathbf{r}^a \mid a \in \mathcal{A}\}$ (see Figure 3.3).

The following time steps rely on the belief-propagation function $\mathbf{b}' = Bayes(\mathbf{b}, a, z)$ (see Equation 2.4) which can be conveniently written in vectorial terms as follows:

$$\mathbf{b}' = \frac{P^{a,z}\mathbf{b}}{\|P^{a,z}\mathbf{b}\|_1}, \tag{3.11}$$

where each element of the square matrix $P^{a,z}$ is defined as $P_{s,s'}^{a,z} = O(s',a,z)T(s,a,s')$.

With these elements, Equation 3.7 can be rewritten in vector terms as follows

$$V_t^*(\mathbf{b}) = \max_{a \in \mathcal{A}} \left\{ \mathbf{b}^\top \mathbf{r}^a + \gamma \sum_{z \in \mathcal{Z}} \|P^{a,z}\mathbf{b}\|_1 V_{t-1}^* \left( \frac{P^{a,z}\mathbf{b}}{\|P^{a,z}\mathbf{b}\|_1} \right) \right\}.$$

Now, by replacing $V_{t-1}$ using Equation 3.9,

$$\begin{aligned}
V_t^*(\mathbf{b}) &= \max_{a \in \mathcal{A}} \left\{ \mathbf{b}^\top \mathbf{r}^a + \gamma \sum_{z \in \mathcal{Z}} \|P^{a,z}\mathbf{b}\|_1 \max_{\alpha \in \Gamma_{t-1}} \left\{ \frac{(P^{a,z}\mathbf{b})^\top \alpha}{\|P^{a,z}\mathbf{b}\|_1} \right\} \right\} \\
&= \max_{a \in \mathcal{A}} \left\{ \mathbf{b}^\top \mathbf{r}^a + \gamma \sum_{z \in \mathcal{Z}} \max_{\alpha \in \Gamma_{t-1}} \left\{ \mathbf{b}^\top {P^{a,z}}^\top \alpha \right\} \right\} \\
&= \max_{a \in \mathcal{A}} \sum_{z \in \mathcal{Z}} \max_{\alpha \in \Gamma_{t-1}} \left\{ \mathbf{b}^\top \left( \frac{\mathbf{r}^a}{|\mathcal{Z}|} + \gamma {P^{a,z}}^\top \alpha \right) \right\} \\
&= \max_{a \in \mathcal{A}} \sum_{z \in \mathcal{Z}} \max_{\alpha \in \Gamma_t^{a,z}} \{\mathbf{b}^\top \alpha\} \\
&= \max_{a \in \mathcal{A}} \max_{\alpha \in \Gamma_t^a} \{\mathbf{b}^\top \alpha\} \\
&= \max_{\alpha \in \Gamma_t} \{\mathbf{b}^\top \alpha\} \qquad (3.12)
\end{aligned}$$

where the $\Gamma$-sets are computed as follows,

$$\Gamma_t^{a,z} \leftarrow \left\{ \frac{\mathbf{r}^a}{|\mathcal{Z}|} + \gamma {P^{a,z}}^\top \alpha \middle| \alpha \in \Gamma_{t-1} \right\}, \qquad (3.13)$$

$$\Gamma_t^a \leftarrow \bigoplus_z \Gamma_t^{a,z}, \qquad (3.14)$$

$$\Gamma_t \leftarrow \bigcup_a \Gamma_t^a. \qquad (3.15)$$

Here, the $\bigoplus$ and $\bigcup$ represent the cross-sum and union operators, where $\gamma_1 \oplus \gamma_2 = \{\alpha_1 + \alpha_2 \mid \alpha_1 \in \Gamma_1, \alpha_2 \in \Gamma_2\}$ (see Cassandra (1998) for more details) [7].

**Algorithm 3:** An exact Value Iteration algorithm for POMDPs using $\alpha$-vector propagation.

**Input**: $T$, $R$, $O$, $\gamma$, $b_0$, $\Gamma_0$
**Output**: $\Gamma$

```
Γ ← Γ_0;
while stopping criterion not met do
    foreach a ∈ A do
        foreach z ∈ Z do
            Compute Γ^{a,z} (Equation 3.13);
        end
        Compute Γ^a (Equation 3.14);
        Mark all the vectors of Γ^a with the action a;
    end
    Compute the new Γ (Equation 3.15)
end
```

An exact and finite solution (policy) can be obtained by computing each $\Gamma_t$ iteratively until convergence (or until $t = H$ in the finite horizon setting) and labeling the vectors with the action used for each computation (see Algorithm 3). This is a form of *exact value iteration* for POMDPs, which properly solves

7. Please note that with these equations, Proposition 3.2.2 can be verified by considering Equation 3.10 as the basis of an inductive proof and Equation 3.12 as the inductive step.

the problem, but with a number of vectors in $\Gamma_t$ that grows exponentially at each time step. Indeed, for a single iteration of Algorithm 3, the order of complexity is $\mathcal{O}(|\Gamma_{t-1}| \times |\mathcal{A}| \times |\mathcal{Z}| \times |\mathcal{S}|^2 + |\mathcal{A}| \times |\mathcal{S}| \times |\Gamma_{t-1}|^{|\mathcal{Z}|})$ (Littman, 1996).

A way of reducing the amount of vectors at each step, is to realize that the computation of $\Gamma_t$ considers all the combinations of vectors, so there are some vectors which do not represent any subspace of the belief-space. For example, Figures 3.4(a) and 3.4(b) represent the same value function, but (a) has superfluous vectors in red that are *dominated* by the vectors in green. The simplest representation (i.e, Figure 3.4(b)) of the value function is called *parsimonious*, and can be obtained by *pruning* the vectors that are not maximal in any part of the space. This pruning phase can be done using linear programming (LP) to find the parsimonious representation of a given $\Gamma$-set.

Monahan's *Batch Enumeration* algorithm (Monahan, 1982) uses the update rule followed by a single pruning phase at each iteration:

$$\Gamma_t \leftarrow \text{PRUNE}\left(\bigcup_a \bigoplus_z \Gamma_t^{a,z}\right).$$

In the same form, the *Incremental Pruning* algorithm (IP) (Cassandra et al., 1997)—which is computationally more efficient—can be written:

$$\Gamma_t \leftarrow \text{PRUNE}\left(\bigcup_a \text{PRUNE}\left(\bigoplus_z \text{PRUNE}\left(\Gamma_t^{a,z}\right)\right)\right).$$

The computational bottleneck of pruning-based algorithms is to solve large LPs. Each pruning of a non-parsimonious $\Gamma$-set requires to solve an LP with $|\Gamma|$ constraints. Therefore, the size of $\Gamma_{t-1}$ and the vectors' dimensionality determine the time complexity of an LP.

Even though the pruning phases are palliative steps in exact POMDP solving, the vector count usually still grows exponentially with the horizon for non-trivial problems. Moreover, the same pruning steps become expensive as the vector count increases, and therefore, the algorithms of this section do not scale well enough to address real-world problems, being useful only for problems of a very limited size. This fact raises the fundamental question of how difficult it is to solve a POMDP.

### 3.2.5 Complexity and Undecidability

To understand how hard is to solve a POMDP, first consider the complexity of solving an MDP. Solving an MDP is *P-complete* (Papadimitriou and Tsitsiklis, 1987), meaning that it is as difficult as the hardest problem in the $P$ class. Therefore, even though an MDP can be solved in polynomial time, it is very unlikely that this can be efficiently parallelized within a polynomial number of processors. The corollary of this result is that solving an MDP is an inherently sequential task, and it is very unlikely to avoid this sequentiality by using a smart algorithm.

As can be expected, the complexity of solving a POMDP is even worse, as it is not only inherently sequential, but also harder than non-deterministic polynomial time (*NP*) problems. Indeed, Papadimitriou and Tsitsiklis (1987) have also shown that finding an optimal policy for a finite horizon POMDP is *PSPACE-complete*, meaning that even though the problem can be solved using a polynomial size of memory, it is as difficult as the hardest problem in the *PSPACE* class, which is believed to be broader than the famous *NP* class. This means that unless *P=NP=PSPACE* (which is highly unlikely), solving a finite horizon POMDP is considerably more difficult than solving an MDP, and more difficult than a whole family of problems that is considered already hard by the computer science community. Moreover, for infinite-horizon POMDPs, Madani et al. (2003) have shown that, for a given POMDP and a given threshold value, it is generally impossible to know if there exist a policy with an expected value greater than the threshold, because the policy-existence problem in infinite-horizon probabilistic planning is undecidable.

These theoretical results have produced a growing interest in approximate POMDP solving techniques during the past two decades, relegating exact techniques mainly to academic purposes, because solving a *PSPACE-complete* problem is considered intractable nowadays. The value function updating process

presented in Section 3.2.4 is exact and provides value functions that can be used whatever the initial belief state $b_0$. A number of approximate POMDP solutions have been proposed to reduce the complexity of these computations, using for example heuristic estimates of the value function, or applying the value update only on selected belief points (Hauskrecht, 2000). In particular, there is a special interest in those algorithms that provide near-optimality guarantees with bounded precision, meaning that the approximation error can be arbitrarily reduced until the algorithm converges to the optimal solution in the limit. These algorithms usually need an infinite computation time to obtain an optimal solution, but near-optimal solutions can be found with much lower complexity than the exact methods. The first work in this line of research was published by Lovejoy (1991), where the belief-space is discretized using a finite-grid, approximately solving the POMDP within a bounded error (by using upper and lower bounds of the value function). This error depends mainly on the grid resolution, and therefore a POMDP solution can be arbitrarily approximated using an algorithm that falls in the $P$ class. For more details and comparisons of grid-based methods compared to other simple heuristics, please refer to (Hauskrecht, 2000).

### 3.2.6 Point-Based Methods

The idea behind *point-based* (PB) approximations is that closer points in the belief-space are usually represented by the same $\alpha$ vector in the value function. Consequently, if a suitable finite set of belief-points is selected, then the value function can be closely approximated using this finite set. This approximation is obtained by propagating the support $\alpha$-vectors that are maximal in those points, but neglecting the propagation in the rest of the belief-space. The $\alpha$-vector of a belief-point in the set works not only as an approximation of the value function for that specific point, but also as an approximation of the nearby beliefs that are not in the set.

The support $\alpha$-vector of a belief-point $\mathbf{b}$ can be obtained by reinterpreting Equation 3.12 as follows

$$\begin{aligned} V_t^*(\mathbf{b}) &= \max_{a\in\mathcal{A}} \left\{ \mathbf{b}^\top \mathbf{r}^a + \gamma \sum_{z\in\mathcal{Z}} \max_{\alpha\in\Gamma_{t-1}} \left\{ \mathbf{b}^\top P^{a,z\top} \alpha \right\} \right\} \\ &= \max_{a\in\mathcal{A}} \left\{ \mathbf{b}^\top \mathbf{r}^a + \gamma \sum_{z\in\mathcal{Z}} \mathbf{b}^\top P^{a,z\top} \operatorname*{argmax}_{\alpha\in\Gamma_{t-1}} \left\{ \mathbf{b}^\top P^{a,z\top} \alpha \right\} \right\} \\ &= \max_{\alpha\in\Gamma_t^{\mathbf{b}}} \mathbf{b}^\top \alpha, \text{ where} \\ \Gamma_t^{\mathbf{b}} &= \left\{ \mathbf{r}^a + \gamma \sum_{z\in\mathcal{Z}} P^{a,z\top} \operatorname*{argmax}_{\alpha\in\Gamma_{t-1}} \left\{ \mathbf{b}^\top P^{a,z\top} \alpha \right\} \middle| a \in \mathcal{A} \right\}. \end{aligned}$$

Using these equations, the *backup* function can be summarized as

$$backup(\Gamma_{t-1}, \mathbf{b}) = \operatorname*{argmax}_{\alpha\in\Gamma_t^{\mathbf{b}}} \mathbf{b}^\top \alpha.$$

The work of Lovejoy (1991) was the first to suggest that PB methods could provide good approximations of the value functions. However, the proposed finite-grid discretization of the belief-space suffers from several drawbacks, such as that most of the belief-points in the set are not even reachable from the initial belief-state. To overcome this problem, Hauskrecht (2000) has proposed to collect reachable points in the belief-space, and use them as support points for the $\alpha$-vector propagation. From here, several efficient and fast algorithms have been proposed, starting with PBVI (Pineau et al., 2003), PERSEUS (Spaan and Vlassis, 2005) and HSVI2 (Smith and Simmons, 2005), and more recent ones such as FSVI (Shani et al., 2007), SARSOP (Kurniawati et al., 2008) and GapMin (Poupart et al., 2011). For a recent and exhaustive study of these methods, please refer to (Shani et al., 2012).

**Point-Based Value Iteration (PBVI)**

The original Point-Based Value Iteration algorithm (Pineau et al., 2003) starts with an initial belief set $\mathcal{B}_0 = \{b_0\}$ and an initial lower bound $\alpha$-vector set $\Gamma_0 = \{\alpha_{min}\}$ where

$$\alpha_{min}(s) = \frac{\min_{s' \in \mathcal{S}, a \in \mathcal{A}} R(s', a)}{1 - \gamma}, \forall s \in \mathcal{S}.$$

**Algorithm 4:** PBVI: A point-based approximate Value Iteration algorithm using synchronous $\alpha$-vector propagation and a dynamic belief-set $\mathcal{B}$.

**Input**: $T$, $R$, $O$, $\gamma$, $\mathbf{b}_0$, $j_{max}$
**Output**: $\Gamma$

$\Gamma \leftarrow \{\alpha_{min}\}$;
$\mathcal{B} \leftarrow \{\mathbf{b}_0\}$;
**while** *stopping criterion not met* **do**
  $\mathcal{B}' \leftarrow \{\}$;
  **foreach** $\mathbf{b} \in \mathcal{B}$ **do**
    $\mathcal{B}' \leftarrow \mathcal{B}' \cup \{successor(\mathbf{b})\}$;
  **end**
  $\mathcal{B} \leftarrow \mathcal{B} \cup \mathcal{B}'$;
  **for** $j = 1$ *to* $j_{max}$ **do**
    $\hat{\Gamma} \leftarrow \Gamma$;
    $\Gamma \leftarrow \{\}$;
    **foreach** $\mathbf{b} \in \mathcal{B}$ **do**
      $\Gamma \leftarrow \Gamma \cup \{backup(\hat{\Gamma}, \mathbf{b})\}$;
    **end**
  **end**
**end**

As presented in Algorithm 4, at each iteration the algorithm performs several successive *backups* of the whole belief-set $\mathcal{B}$,

$$\Gamma_j = \{backup(\Gamma_j, \mathbf{b}) | \mathbf{b} \in \mathcal{B}\}$$

for a predefined $j_{max}$ number of iterations (or until $\Gamma_j$ converges). As the value function approximation has converged (or it is near to), a new set of belief points is needed to continue improving the value function. To do this, PBVI selects a successor for each point in the set, and adds these new belief-points to the belief set for the next iteration. The successor of a belief-point $\mathbf{b}$, is obtained by selecting the most distant forward simulated point as follows:

$$successor(\mathbf{b}) = \underset{\mathbf{b}' \in \mathcal{B}_{\mathbf{b}}^{next}}{\operatorname{argmax}} \left\{ \min_{\hat{\mathbf{b}} \in \mathcal{B}} \|\hat{\mathbf{b}} - \mathbf{b}'\|_1 \right\}, \text{ where}$$

$$\mathcal{B}_{\mathbf{b}}^{next} = \left\{ \frac{P^{a,z}\mathbf{b}}{\|P^{a,z}\mathbf{b}\|_1} \middle| a \in \mathcal{A}, z \in \mathcal{Z} \right\}.$$

Similarly to exact algorithms, the stopping criterion can vary depending on the implementation and problem requirements. For finite horizons, a fixed number of iterations must be used (usually with $j_{max} = 1$), while for infinite horizons some convergence criterion must be used.

An interesting property of this algorithm is that the error of the value function computed by PBVI

$\hat{V}_H$, with respect to the optimal value function $V^*$, is bounded as follows:

$$\begin{aligned}\epsilon_H &= \|\hat{V}_H - V^*\|_\infty \\ &\leq \|\hat{V}_H - V_H^*\|_\infty + \|V_H^* - V^*\|_\infty \\ &\leq \|\hat{V}_H - V_H^*\|_\infty + \frac{\gamma^H (R_{max} - R_{min})}{1-\gamma} \\ &\leq \frac{(R_{max} - R_{min})\delta_{\mathcal{B}}}{(1-\gamma)^2} + \frac{\gamma^H (R_{max} - R_{min})}{(1-\gamma)},\end{aligned}$$

where $R_{max}$ and $R_{min}$ are the maximum and minimum values of the reward function, and

$$\delta_{\mathcal{B}} = \max_{\mathbf{b}' \in \Delta} \min_{\mathbf{b} \in \mathcal{B}} \{\mathbf{b}' - \mathbf{b}\}_1$$

is the *density* of the belief-set $\mathcal{B}$. The first step follows from the triangular inequality, showing that the right-hand term of the equation depends only on the computing horizon, which is zero if the horizon of the problem is finite. If it is not finite, the second step shows that, by the contraction theorem in (Bertsekas and Tsitsiklis, 1996), this term is very small for large horizons, which is the case if an appropriate convergence criterion was chosen (for instance, one based on this same bound). The third step is due to Theorem 3.1 of (Pineau et al., 2003), where it is shown that the error induced at each step by the point-based approximation depends only on the density $\delta_{\mathcal{B}}$ of the belief-set. In simple words, $\delta_{\mathcal{B}}$ is the largest possible distance from the closest support point in $\mathcal{B}$, which helps to find an upper bound for the approximation error. Please be aware that calling $\delta_{\mathcal{B}}$ "density" is somewhat deceptive, because as points are added to the $\mathcal{B}$, the density shrinks, having a zero value in the limit.

### PERSEUS

The PERSEUS Algorithm (Spaan and Vlassis, 2005) is a PB algorithm that relies on collecting a large $\mathcal{B}$ set only once, and using the same belief-set for all iterations. This strategy allows to improve the

value function at each iteration by using *asynchronous* backups.

**Algorithm 5:** PERSEUS: A point-based approximate Value Iteration algorithm using asynchronous $\alpha$-vector propagation and a fixed randomized belief-set $\mathcal{B}$

**Input**: $T$, $R$, $O$, $\gamma$, $\mathbf{b}_0$, $n$
**Output**: $\Gamma$

$\mathcal{B} \leftarrow \{\mathbf{b}_0\}$;
$\mathbf{b} \leftarrow \mathbf{b}_0$;
**while** $|\mathcal{B}| \leq n$ **do**
  sample $a \in \mathcal{A}$;
  sample $z \in \mathcal{Z}$ from $Pr(Z = z|A = a, B = b)$;
  $\mathbf{b} \leftarrow Bayes(\mathbf{b}, a, z)$;
  $\mathcal{B} \leftarrow \mathcal{B} \cup \{\mathbf{b}\}$;
**end**
$\Gamma \leftarrow \{\alpha_{min}\}$;
**while** *stopping criterion not met* **do**
  $\hat{\Gamma} \leftarrow \{\}$;
  $\mathcal{B}' \leftarrow \mathcal{B}$;
  **while** $\mathcal{B}' \neq \{\}$ **do**
    choose any $\mathbf{b} \in \mathcal{B}'$;
    $\hat{\alpha} \leftarrow backup(\mathbf{b}, \Gamma)$;
    **if** $\mathbf{b}^\top \hat{\alpha} \geq V(\mathbf{b})$ **then**
      $\mathcal{B}' \leftarrow \{\hat{\mathbf{b}} \in \mathcal{B}' | \hat{\mathbf{b}}^\top \hat{\alpha} < V(\hat{\mathbf{b}})\}$;
    **else**
      $\mathcal{B}' \leftarrow \mathcal{B}' - \{\mathbf{b}\}$;
      $\hat{\alpha} \leftarrow \underset{\alpha \in \Gamma}{\text{argmax}}\{\mathbf{b}^\top \alpha\}$;
    **end**
    $\hat{\Gamma} \leftarrow \hat{\Gamma} \cup \{\hat{\alpha}\}$;
  **end**
  $\Gamma = \hat{\Gamma}$;
**end**

In Algorithm 5 it is shown that the set $\mathcal{B}$ is obtained by adding all the beliefs of a random simulation starting from $\mathbf{b}_0$ until the size of the set reaches $n$ points. This is a very fast way of collecting reachable points, which allows obtaining large $\mathcal{B}$ sets inexpensively. In a second stage, at each iteration the set $\mathcal{B}$ is copied to a working set $\mathcal{B}'$, in order to apply asynchronous *backups* of the $\alpha$-vectors in $\Gamma$. The asynchronous update is done by selecting a random point $\mathbf{b} \in \mathcal{B}'$ and checking if the new support vector $\hat{\alpha} = backup(\mathbf{b}, \Gamma)$ improves the value $V(\mathbf{b}) = \max_{\alpha \in \Gamma}\{\mathbf{b}^\top \alpha\}$. If it does improve, all the points in $\mathcal{B}'$ that are improved by this new $\hat{\alpha}$ vector are removed from the set, representing an asynchronous update for those points. In the contrary, if it does not improve $V(\mathbf{b})$, the previous support vector is selected, $\hat{\alpha} = \underset{\alpha \in \Gamma}{\text{argmax}}\{\mathbf{b}^\top \alpha\}$, and only $\mathbf{b}$ is removed from the set $\mathcal{B}'$. No matter if $\hat{\alpha}$ is an old or new vector, it is included in the next value function represented by $\hat{\Gamma}$. This procedure is repeated until there is no more points in the working set $\mathcal{B}'$.

Please note that, even though using asynchronous backups is generally very efficient, there are several constraints for its usage. First, if the belief-set changes from one iteration to the other several evaluations of the value function are needed, which may be time consuming. Also, similar to asynchronous value iteration, this value function improving technique can be used only on discounted infinite horizon POMDPs, in contrast to PBVI that can be used for finite horizon POMDPs. At last, the lower bound initialization in PERSEUS is strictly required for the algorithm to work, which is not the case in PBVI.

| **Algorithm** | COLLECT | UPDATE |
|---|---|---|
| PBVI (Pineau et al., 2003) | $L_1$-norm | full backup |
| PERSEUS (Spaan and Vlassis, 2005) | random sampling | asynchronous backup |
| HSVI* (Smith and Simmons, 2005) | bound uncertainty | newest points backup |
| GapMin (Poupart et al., 2011) | bound uncertainty | full backup |
| PEMA (Pineau et al., 2003) | approximation error | full backup |
| FSVI (Shani et al., 2007) | MDP heuristic | newest points backup |
| SARSOP* (Kurniawati et al., 2008) | bound uncertainty | newest points backup |

Table 3.1: Point-based algorithms by collect and update strategies, reproduced from (Shani et al., 2012). The algorithms marked with a * cannot be easily decoupled in COLLECT and UPDATE, yet the principles used somehow matches the respective strategies.

#### Other PB Algorithms

Even though PBVI and PERSEUS seem to be completely different algorithms, some common patterns can be found in their descriptions. Indeed, almost all the PB algorithms share the same structure, which is composed by the following two stages:

1. COLLECT is the stage that selects a new set of belief points based on the previous set and the current approximation of the value function ;
2. UPDATE is the stage where the *backups* are made, resulting in a new improved approximation of the value function.

Also, some PB algorithms consider a post-processing phase where some vectors are pruned because they are dominated or considered negligible.

Several COLLECT strategies have been proposed beside performing *random sampling* like in PERSEUS, and selecting successors faraway from the belief-set by using an *L-norm* measure like in PBVI. For example, the successors selection can be made in order to reduce the *approximation error* (Pineau et al., 2003), which leads to more accurate and informed approximations. Random sampling can also be modified to perform a more informed search, by selecting the actions not at random, but by following an *MDP heuristic* (Shani et al., 2007). Other examples are *bound uncertainty* methods, which explicitly search for those points that minimize the gap between upper and lower bounds of the value function (Smith and Simmons, 2005; Poupart et al., 2011; Kurniawati et al., 2008).

The list of different UPDATE strategies is more modest, where besides the *full backup* used by PBVI, and the *asynchronous backup* used by PERSEUS, only the *newest points backup* has been proposed. The latest performs backups at each iteration only for the new points in the set, and not for all of them like in the full version. Table 3.1 presents a summary of the methods used by the most known PB algorithms, reproduced from (Shani et al., 2012).

### 3.2.7 Other Approximate Methods

PB algorithms are not the only approximate method for solving POMDPs. Several other methods have been proposed following diverse principles. This section presents some of these alternatives for the sake of completeness.

For example, Wang et al. (2006) propose using piecewise-quadratic and convex functions rather than PWLC functions, by fitting convex quadratic upper bounds that approximate a subset of PWLC functions in a more compact form than a subset of hyperplanes. The main drawbacks of this method are (1) that non-linear programming is needed to prune quadratic functions, which can be partially mitigated by using semidefinite programming, and (2) that there is no correspondence between quadratic functions and actions like when using $\alpha$-vectors, so a one-step lookahead is necessary to obtain the next action.

In (Thrun, 2000), a Monte Carlo approach to approximately compute the value function is proposed, using particle filtering for point propagation, and nearest neighbors for value function generalization. This technique has the advantage that it can be trivially extended to continuous or very large state spaces, but attaining good performance requires a large number of samples.

A completely different approach is the RTDP-bel algorithm (Bonet and Geffner, 1998), which extends the RTDP algorithm (Barto et al., 1995) to the partially observable case. This algorithm is a combination of the Bellman equation and classical shortest path algorithms (like A*), where an admissible heuristic is used to initialize the value function approximation. Empirical results show that this method provides competitive results with respect to PB algorithms (Bonet and Geffner, 2009).

Worthy of particular mention are online planning methods like AEMS (Ross and Chaib-draa, 2007), which use an online strategy to improve the policy with each new observation. This algorithm starts from an initial value function given by a heuristic or a PB algorithm, and then uses an AND-OR tree to expand promising futures from the current belief so as to enhance the planning through tree search. A comprehensive review of online algorithms for POMDPs can be found in (Ross et al., 2008), where several other tree-search techniques are presented and compared.

More recent on-line strategies are POMCP (Silver and Veness, 2010), which uses Monte-Carlo planning to solve large POMDPs based on the UCT algorithm (Kocsis and Szepesvári, 2006), and MCVI Bai et al. (2010) which also uses Monte-Carlo planning but for dealing with continuous-state spaces. These strategies were recurrently summoned by the solvers of the last International Probabilistic Planning Competition 2011 (`http://users.cecs.anu.edu.au/~ssanner/IPPC_2011/`).

# Chapter 4

# Information-Gathering Decision Problems

## Contents



*"Information can tell us everything. It has all the answers. But there are answers to questions we have not asked, and which doubtless don't even arise."* — Jean Baudrillard.

A sequential decision problem under epistemic uncertainty is usually viewed as a *control* problem, where the objective is to reach some goal states or achieve some desired behavior. In this context, an agent must acquire information only as a means to successfully fulfill the desired objective. Yet, there is a vast family of sequential problems where the objective is to gather knowledge about the system, and where control is the means to increase the information held by the agent. Consequently, when the objective is to know more about the system, an *information-gathering* decision problem arises.

**Example 4.0.1** (Doctor House and Medical Diagnosis)**.** *Doctor House is a character of a TV-series about a medical diagnosis department in a hospital. His job is to find the correct diagnosis in those difficult cases where other doctors have failed. The particularity of this fictional medical doctor is that he is obsessed with diagnosis, so his objective is not to save the patient's life as any doctor, but just to know which is the patient's disease. Indeed, after the diagnosis is finished, the case is closed for him and other doctors must treat the patient because he does not care. Moreover, he does not care about the costs of the procedures, and any means is valid in order to acquire information. Doctor House is an example of a pure information-gathering decision problem, because the only objective for him is to make the correct decisions to rapidly* know *about the agent's disease, where the procedures' costs, moral issues and any other aspects are neglected. In real-world medical diagnosis the problem is more subtle, but in principle, the objective is the same: choosing the right decisions to disambiguate the actual disease.*

It is natural to speculate that information-gathering problems are ill-defined, because gathering information is never the final objective of a system: the information must be used for *something*. Nevertheless, there is a wide range of practical problems where that *something* is not defined, known or accessible. A brief description of some problems is given in Section 4.1 to illustrate their importance. Moreover, the same negative argument of ill-definition can be applied to classical machine learning problems such as regression or classification, and they are undoubtedly useful in real-world applications. Going even further, gathering information can lead to answering questions that were not even formulated in the first place, like in scientific experimentation, medical diagnosis or astronomy.

This kind of problems has been addressed tangentially by several research communities with different names, such as *active sensing* and *active localization* in robotics (Bajcsy, 1988; Mihaylova et al., 2006; Fox et al., 1998), *sensor management* in control theory (Hero III et al., 2007; Hager and Mintz, 1987; Reece, 2001), or *active diagnosis* and *active classification* in machine learning (Ji and Carin, 2007; Ishida, 1997; Zheng et al., 2005). Unfortunately, the proposed techniques usually rely on problem's specifics, making them hard to generalize to other problems.

## 4.1 Information-Gathering Problems

To motivate the importance of solving information-gathering problems, this section presents three classes of problems with information-gathering objectives, covering a wide range of applications: *diagnosis* problems, *surveillance* problems, and *localization* problems.

This classification of problems is far from complete. Other kinds of problems might not fit in the presented classes. Nevertheless, it helps to grasp the idea of information-gathering problems, and to show that there are numerous applications for the solution methods proposed in this chapter.

The common ground for these problems is that (part of) the objective is to explicitly optimize an information measure, in contrast with classical sequential decision making where the objective can be fully described in terms of state and transition rewards. It is self-evident that if the objective is to optimize the information, these problems are always partially observable because if not, there is nothing to optimize.

**Definition 4.1.1** (Information-Gathering Problem)**.** *An* information-gathering decision problem *is when the performance criterion incorporates an explicit measure of the information held by the agent about the system, which is based on the internal knowledge state rather than only on external system states.*

Also, an information-gathering problem usually focuses on a subset of variables of the state, and the objective is to optimize the knowledge about only these variables and not the whole belief-state. This

subset of variables will be called *target* variables, to stress the fact that the information measure is only about these variables.

### 4.1.1 Active Diagnosis Problems

Active Diagnosis problems are those problems where there is a hidden target variable with a fixed value to estimate, and information about this target variable can be obtained by performing different actions. The key characteristic of this class of problems is that the target variable does not evolve with time, yet the observation of the target could be influenced by previous decisions.

The most natural example is medical diagnosis, where there is a latent disease, and the objective is to know which disease fits better the symptoms and exams. Active medical diagnosis deals with selecting the best policy of exams or procedures to obtain the best possible diagnosis, where certain exams or procedures could conceal or affect the results of other exams (observations). An example of the decision-theoretic formulation of this problem can be found in (Pellegrini and Wainer, 2003) where the cost-sensitive medical diagnosis problem is addressed.

Other application domains include fault diagnosis (Zheng et al., 2005), mapping (Saigol et al., 2009), visual search (Vogel and Murphy, 2007), network diagnosis (Ishida, 1997), active feature acquisition (Ji and Carin, 2007), and sensor management (Williams, 2007).

### 4.1.2 Surveillance Problems

In surveillance problems the objective is also to disambiguate the state of one or more target variables, but these variables can evolve with time. As in active diagnosis, actions produce indirect observations of the target's states, and previous actions could affect the outcome of the current one. However, the key characteristic of this class of problems is that the actions do not affect the dynamics of the target, or in other words, that the behavior of the target is independent from actions.

A classic example of this kind of problems is multi-target surveillance (Kreucher et al., 2005; Spaan, 2008), where the objective is to continuously infer the position of several targets with a bounded amount of active noisy sensors. The problem consists in selecting the right sensors at each step to increase the certainty of the target positions.

Surveillance applications also include sensor networks (Zhao et al., 2002), natural resources monitoring (Singh et al., 2009), active gesture recognition (Darrell and Pentland, 1996), and military surveillance (Hintz, 1991). In the context of sensor management, the formulation of this kind of problems is extensively treated in (Hero III et al., 2007).

### 4.1.3 Active Localization Problems

Active localization problems consist in performing the best actions to disambiguate the state of the agent itself (or controller). In this setting, the target variable to disambiguate is the agent's state, and this state could evolve as the agent acts in the environment. The key characteristic of these problems is that the controller can influence the dynamics of the target variables (itself), to acquire more information.

For example, consider the active vision problem (Bajcsy, 1988): in a 3D world, a 2D image observation provides limited information about the depth of the objects in the scene. However, the depth can be inferred through the displacement of objects when moving the agent. Therefore, in this scenario, the active localization problem is to find the best policy of moves to disambiguate the position of the agent by inferring the relative position of the objects in a map.

Other examples of this type of problems are active mobile robot localization (Burgard et al., 1997; Fox et al., 1998), generic robotic active sensing (Mihaylova et al., 2006; Fox et al., 1998), and active SLAM (Sim and Roy, 2004).

## 4.2 Information Measures

A major issue, when addressing the problems of Section 4.1, is to formally characterize information. If the objective is to optimize the information held by an agent, a numerical measure must be defined to

compare different scenarios. Information theory (Cover and Thomas, 1991) provides several notions of information such as Shannon (1948), Rényi (1960), Fisher (1925) or Kolmogorov (1968), and even more elaborate definitions for relative information measures such as Kullback and Leibler (1951) or Chernoff (1952).

Fortunately, all these measures are strongly related, and almost all of them rely on the concept of random variables[8], because the assumption that knowledge can be properly represented by probability distributions helps to turn the soft-concept of *information* into numbers. The objective of this section is to present useful measures for the information held by the agent about the state of a system.

### 4.2.1 Shannon's Entropy

As presented in Chapter 2, an agent represents the latent state of the system as a random variable $S \in \mathcal{S}$ with probability distribution $Pr(S) = \mathbb{P}$. In other words, the belief-state of the agent is the information about the state. Without making any further assumptions about the nature of the set $\mathcal{S}$ or about the probability distribution family, Shannon's famous measure of the uncertainty held by a probability distribution $\mathbb{P}$ can be used to evaluate different belief-states.

**Definition 4.2.1** (Shannon Entropy)**.** *The Shannon (1948) entropy of a probability distribution over the states is defined as*

$$H(S) = H(\mathbb{P}) = \mathbb{E}_S\left[-log(Pr(S))\right].$$

Loosely speaking, the $-log(Pr(S = s))$ function measures how difficult it is to believe that the current state is $s$. If $Pr(S = s)$ is high, this quantity gives a near zero value (easy), and increases as the probability decreases. In the limit, when $Pr(S = s) = 0$, it is infinitely difficult (impossible) to believe that the current state is $s$. In this context, any base for the logarithm provides the same behavior but with different units. For example, in computer science the base 2 is used to measure [*bits*], yet more generally in physics and mathematics the base $e$ is used to measure [*nats*].

The statistical meaning of the entropy can be better understood in the parametric case when $Pr(S) = \mathbb{P}_\theta$. Here, the likelihood $\mathcal{L}(\theta|s) \propto \mathbb{P}_\theta(s)$ represents how likely the parameter $\theta$ explains the value $S = s$. Formally, Shannon's entropy consists in the expectation of the negative log-likelihood function. As the logarithm is a monotonically increasing function, the entropy can be interpreted as the mean of how much *unlikely* is that the parameter $\theta$ explains a specific state. For example, for categorical distributions like the ones used for discrete POMDP beliefs, the uncertainty can be measured as follows,

$$H(S) = -\sum_{s \in \mathcal{S}} b(s) \log(b(s)).$$

### 4.2.2 Relative Information

In general terms, the information is always a relative quantity. For example, some data that provide a lot of information to an uninformed agent, may provide only little information to an expert one. Therefore, an information-state has to be measured with respect to another information-state in order to have a proper meaning.

**Definition 4.2.2** (KL divergence)**.** *The Kullback-Leibler divergence (Kullback and Leibler, 1951) is a generalization of the entropy measure that consists in quantifying the information divergence of a probability distribution $\mathbb{P}$ from a reference distribution $\mathbb{Q}$. Let $\mathbb{P}$ and $\mathbb{Q}$ be two distributions over the same sample space, then the KL divergence is defined as*

$$D_{KL}(\mathbb{P}\|\mathbb{Q}) = \int_{x \in \mathcal{X}} \mathbb{P}(x) \log(\mathbb{P}(x)) dx - \int_{x \in \mathcal{X}} \mathbb{P}(x) \log(\mathbb{Q}(x)) dx.$$

8. Except for Kolmogorov's complexity measure, which relies on programs. Since this measure is not appropriate for the selected representation, it will not be considered for the rest of this dissertation.

The KL divergence is not a proper distance measure, because it is not symmetric nor complies with the triangular inequality, yet it is a widely accepted way to quantify the information distance between two distributions.

Under the epistemic interpretation of probabilities, if $\mathbb{P}$ is considered the current distribution of a random variable $X$ and $\mathbb{Q}$ a reference distribution for the same random variable, then the KL divergence computes the expected information needed to move from $\mathbb{Q}$ to $\mathbb{P}$, assuming that $\mathbb{P}$ is the "real" distribution. The asymmetry of this function arises from the expectation computation, because the current distribution is used in both terms, and therefore, the expected amount of information from $\mathbb{P}$ to $\mathbb{Q}$ is different, because $\mathbb{Q}$ is now the distribution used to compute the expectation. Despite this asymmetry, the KL divergence complies with some interesting properties. First, due to Gibbs' inequality, this divergence is always non-negative, no matter the distribution family or values. Also, it is additive for independent distributions, and it is convex with respect to $\mathbb{P}$ when $\mathbb{Q}$ is given.

The presented KL divergence can be related to almost all the available information notions (see Section 4.2.3), starting with Shannon's entropy, where if $\mathbb{U}$ is a uniform distribution, then

$$D_{KL}(\mathbb{P}\|\mathbb{U}) = log(|\mathcal{S}|) - H(\mathbb{P}).$$

Here it can be noted that the entropy is a measure of uncertainty, while the KL divergence quantifies relative information. The KL-divergence term will be commonly used instead of entropy, because the later one can be seen only as the special case where $\mathbb{Q} = \mathbb{U}$.

In the agent context with a discrete state space, the relative information (KL divergence) of the state random variable $S$ with respect to $\mathbb{Q}$, where $Pr(S) = \mathbb{P}$ is given by

$$D_{KL}(\mathbb{P}\|\mathbb{Q}) = \sum_{s\in\mathcal{S}} \mathbb{P}(s) \log\left(\frac{\mathbb{P}(s)}{\mathbb{Q}(s)}\right). \tag{4.1}$$

### 4.2.3 Other Information Measures

The work done by Shannon (1948) and Kullback and Leibler (1951) was extended by Rényi (1960), presenting a continuous family of entropies and divergences by relaxing some properties such as additivity. Indeed, Rényi's theory presents the entropy and the KL divergence as special cases of a more general parametric definition of information. Even though this generalization will not be explicitly studied in this dissertation, it is important to notice its existence, because some authors have used it to find specific measures for their problems, for example (Hero III et al., 2007) in sequential sensor management.

In statistical inference, the Fisher information matrix (Fisher, 1925) is a way of characterizing the information that parameters provide for a family of probability distributions. The information matrix corresponds to the curvature (second derivative) of the KL divergence (Cover and Thomas, 1991), which can be understood as an instant version of the KL divergence. When a frequentist approach is used for information-gathering problems, such as in (Mihaylova et al., 2006), some metric of the information matrix based on optimal experiment-design can be used to characterize information (Fedorov, 1972).

There are other conventional statistics that can be used as information measures, such as the $L$-norms of the belief-state. These simple measures can be seen as approximations of a log-based information measure such as the entropy or the KL-Divergence. For instance, the $L_\infty$-norm $(\max_{s\in\mathcal{S}} \mathbb{P}(s))$ is an upper linear approximation of the KL divergence when $\mathbb{Q} = \mathbb{U}$, and the squared $L_2$-norm is the most significant term of the Taylor expansion of the same KL divergence when $\mathbb{Q} = \mathbb{U}$. Therefore, a system that reduces any of these norms indirectly reduces the entropy, increasing the knowledge about the state.

## 4.3 POMDP extension for Information-Gathering

The dynamics and observability of information-gathering problems can be properly represented by probability distributions, and therefore POMDPs (see Section 3.2) may seem an adequate model to address this type of sequential problems. Unfortunately, POMDPs cannot describe properly all information-gathering problems, because the strict definition of rewards that depends only on the state and actions

(i.e., $R(s,a)$ or $r(s,a,s')$) does not allow defining objectives depending on the information, as needed by Definition 4.1.1. Coming back to the utility and reward definitions of Section 2.2, there is no such constraint for the one-step optimal decision scenario, where intrinsic rewards can be defined for non-state related tasks like in statistical learning.

**Example 4.3.1** (Hidden Object)**.** *Consider the simple problem of knowing the position of a hidden object. In some cases, this problem can be solved without even having seen the object, for instance if all the locations but one have been visited. However, the reward of a POMDP cannot model this since it is only based on the current state and action. One solution would be to include the whole history in the state, leading to a combinatorial explosion. The approach proposed here, on the other hand, is to extend POMDPs to support defining rewards based on the acquired knowledge and not only on the state-based rewards.*

This shortcoming may look trivial to solve, because almost all the POMDP solving techniques rely on (approximately) solving a belief-MDP as presented in Section 3.2, which is defined over a $\Delta$-reward function $\rho(\mathbf{b},a)$, and not directly on $R(s,a)$. Then, the expected state-dependent reward of Equation 3.6 can be replaced with an information measure to directly define $\rho$. However, the POMDP solving techniques strongly rely on the value function being PWLC (see Section 3.2.3), property that is based on the same Equation 3.6 that this extension intends to neglect.

This section explores the fact that belief-MDPs can still be solved outside the specific definition of Equation 3.6 in theory, and then discusses the feasible approximate techniques that can be used for this extension.

For correctly formalizing information-gathering problems, the idea of a $\rho$POMDPs is proposed here, where the definition of POMDPs can be extended to support arbitrary $\Delta$-rewards: rewards that depend directly on the belief-state and not only on the state of the system.

**Definition 4.3.1** ($\rho$POMDP)**.** *A $\rho$POMDP is a generalization of the POMDP framework where the reward function $\rho(\mathbf{b},a)$ is directly defined in terms of belief-states and actions, and not (only) as the expectation of an external reward $r(s,a,s')$. Then, the $\rho$POMDP tuple is defined as $\langle \mathcal{S}, \mathcal{A}, \mathcal{Z}, T, O, \rho, \mathbf{b}_0 \rangle$.*

Now the rewards are defined directly over belief-states, making the belief-MDP harder to solve. Indeed, the value function is not always PWLC because it depends on the structure of the $\rho(\mathbf{b},a)$ function.

### 4.3.1 Information-based Rewards

In Section 2.3.3 the $\rho(\mathbf{b},a)$ function was defined as the expectation of the $\Delta$-reward function $\varrho(\mathbf{b},a,z)$ over $Z$. Also, the $\varrho$ function was defined as the expected value of the reward function $r(s,a,s')$, which gives the *state-dependent reward*

$$\rho_R(\mathbf{b},a) = \sum_{s\in\mathcal{S}} \mathbf{b}(s) R(s,a),$$

(see Figure 4.1(a)).

In this chapter, this condition is relaxed to support more general $\rho(\mathbf{b},a)$ functions, specifically those that depend on the information. Even though $\Delta$-reward functions may be generally defined as $\varrho(\mathbf{b},a,z)$ functions, here the $\rho(\mathbf{b},a)$ functions will be directly defined, analogous to directly defining $R(s,a)$ rather than $r(s,a,s')$ in normal POMDPs.

The nature of $\rho(\mathbf{b},a)$ will depend on the problem, which is often related to an information measure like the ones described in Section 4.2. A common choice is to use the KL divergence between the current belief and a distribution that represents ignorance, for example the uniform distribution. Let $\mathbf{b} = \mathbb{P}$ be the current belief and $\mathbf{c} = \mathbb{U}$ a uniform distribution (the center of the simplex). Then, using Equation 4.1 the *entropy-based reward* can be defined as:

$$\rho_H(\mathbf{b},a) = D_{KL}(\mathbb{P}\|\mathbb{U}) = \sum_s \mathbf{b}(s) \log\left(\frac{\mathbf{b}(s)}{\mathbf{c}(s)}\right) = \log(|S|) + \sum_{s\in\mathcal{S}} \mathbf{b}(s) log(\mathbf{b}(s)).$$

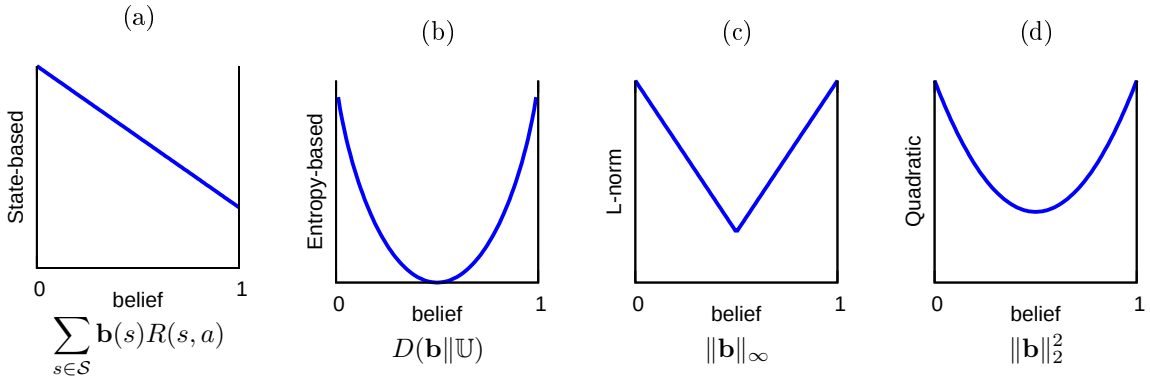



Figure 4.1: Information-based rewards in a 1-simplex: (a) state-based, (b) KL divergence, (c) $L_\infty$-norm and (d) squared $L_2$-norm.

This reward function is a convex function that represents the opposite of the entropy, where beliefs with high uncertainty produce low rewards, and beliefs with low uncertainty produces high rewards (see Figure 4.1(b)). In general, $c$ could be any distribution, where the entropy-based reward is only a special case of the *KL-based reward*.

Also, other types of rewards can be defined by using the approximations of the entropy-based reward (KL divergence when $\mathbb{Q} = \mathbb{U}$), such as the squared $L_2$-norm or the $L_\infty$-norm. These approximations are useful because they are easier to analyze and compute. In particular, the *quadratic reward* (i.e. the squared $L_2$norm of Figure 4.1(d)) is defined as

$$\rho_Q(\mathbf{b}, a) = \|\mathbf{b}\|_2^2 = \sum_{s\in\mathcal{S}} \mathbf{b}(s)^2.$$

Another interesting approximation is the *linear reward* (i.e., the $L_\infty$-norm of Figure 4.1(c)), which is a *PWLC reward* consisting in the maximum of $|\mathcal{S}|$ hyperplanes, and can be trivially defined as

$$\rho_L(\mathbf{b}, a) = \|\mathbf{b}\|_\infty = \max_s \mathbf{b}(s).$$

Please note that the $L_1$-norm has no use as a reward function because, in the simplex, all the points have an $L_1$-norm equal to 1.

These $L$-norms are not the only norm-based functions that can be used: others distances can be used such as the *distance to the simplex center* (DSC),

$$\rho_{DSC}(b, a) = \|\mathbf{b} - \mathbf{c}\|_\infty,$$

or the *distance to the corners*,

$$\rho_{COR}(b, a) = \|\mathbf{1} - \mathbf{b}\|_\infty.$$

Here, the infinity norm can be replaced by any other $p$-norm, yet this dissertation is focused on the infinity norm because it naturally defines PWLC functions [9].

In a 1-simplex, the $\rho_L$, $\rho_{DSC}$ and $\rho_{COR}$ functions look very similar, presenting the same shape but with different maximum and minimum values. Figure 4.2 presents these reward functions in a 2-simplex showing that the functions are actually different, corresponding to different possible approximations of $\rho_H$. An important question is which linear function ($\rho_L$, $\rho_{DSC}$ or $\rho_{COR}$) to use as an approximation of $\rho_H$. For instance, $\rho_{COR}$ defines an upper-bound of $\rho_H$, which can be exploited by optimistic or gap-based algorithms, yet it has a constant reward on the boundary of the simplex, which forbids the distinction of any degenerated belief-space where one dimension is rapidly discarded. Similarly, $\rho_{DSC}$ seems to resemble more the form of the $\rho_H$ function, but it also has constant values on half of the simplex boundary. For

9. The 1-norm also produces PWLC functions, but the reward form is similar yet harder to compute.

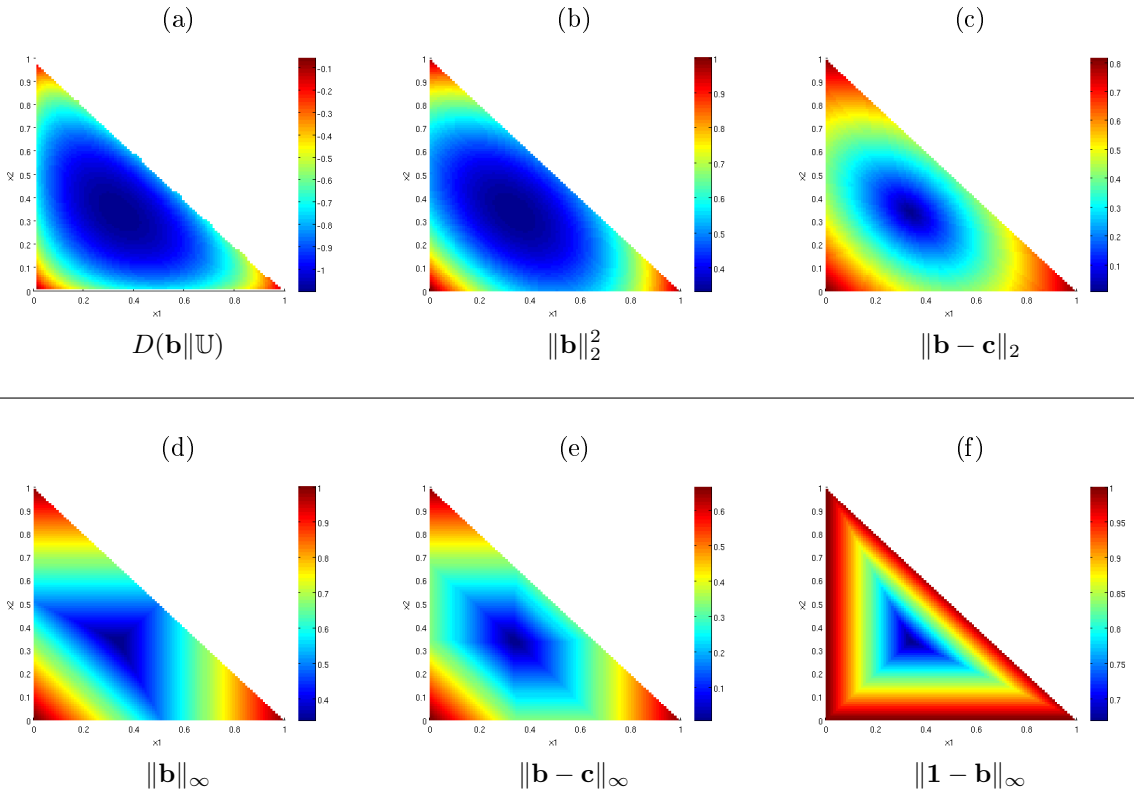



Figure 4.2: Information-based rewards in a 2-simplex: (a) KL divergence, (b) squared $L_2$-norm, (c) squared Euclidean distance to the simplex center, (d) $L_\infty$-norm, (e) linear distance to the simplex center and (f) maximum distance to to the corners.

this reason—and the ones that are discussed in Section 4.3.6—is seems that the more natural choice is $\rho_L$, not only because it is a PWLC function with not too many hyperplanes, but also because it better grasps the objective of the entropy-based reward function: more information implies higher reward.

### 4.3.2 Combined Rewards

Please note that $\rho(\mathbf{b}, a)$ is not restricted to be only an information measure. It can be a combination of the expected state-action rewards and an information measure. For example, Mihaylova et al. (2002) define the active sensing problem as optimizing a weighted sum of the entropy and costs:

$$cost(\mathbf{b}, a) = \alpha H(\mathbf{b}) + (1 - \alpha) \sum_{s \in \mathcal{S}} \mathbf{b}(s) cost(s, a),$$

with $\alpha \in [0, 1]$. As reward functions are the opposite of a cost function, and as $\alpha$ is an arbitrary number that remains constant throughout any optimization, this expression is equivalent to

$$\rho_{HR}(\mathbf{b}, a) = \rho_H(\mathbf{b}, a) + \eta \rho_R(\mathbf{b}, a) \tag{4.2}$$

with $\eta \in \mathbb{R}_0^+$, and where $R(s, a) = -cost(s, a)$.

This combined reward schema allows to define cost-sensitive information-gathering problems, where different actions may have different rewards, and where the information-gathering can be biased towards some preferred states. For example, in a more realistic scenario than Doctor House, medical diagnosis can be modelled by assigning costs to the exams and/or by considering that detecting a cancer is *more important* than detecting a flu. Please note that modelling the problem only with state-dependent rewards may lead to non-desired behaviors, like not performing a critical exam because it is too expensive, or avoiding new knowledge of low-valued diseases.

In fact, the state-based reward function $\rho_R(b, a)$ works as a bias of the information-based reward $\rho_H(b, a)$, where

$$\begin{aligned}
\rho_{HR}(s, a) &= \rho_H(b, a) + \eta \rho_R(b, a) \\
&= \log(|S|) + \sum_{s \in \mathcal{S}} b(s) log(b(s)) + \eta \sum_{s \in \mathcal{S}} b(s) R(s, a) \\
&= \log(|S|) + \sum_{s \in \mathcal{S}} b(s) (log(b(s)) + log(e^{\eta R(s,a)}))
\end{aligned}$$

and by defining $\phi_a(s) = e^{-\eta R(s,a)}$,

$$\begin{aligned}
&= \log(|S|) + \sum_{s \in \mathcal{S}} b(s) (log(b(s)) - log(\phi_a(s))) \\
&= \log(|S|) + \sum_{s \in \mathcal{S}} b(s) \log\left(\frac{b(s)}{\phi_a(s)}\right) \\
&= \log(|S|) - \log(\|\phi_a\|_1) + \sum_{s \in \mathcal{S}} b(s) \log\left(\frac{b(s)}{\frac{\phi_a(s)}{\|\phi_a\|_1}}\right) \\
&= D_{KL}\left(b \middle\| \frac{\phi_a}{\|\phi_a\|_1}\right) + \log\left(\frac{|S|}{\|\phi_a\|_1}\right).
\end{aligned}$$

Here, the first term of the last line is an information-measure biased by the reward, and the second term is constant for each action, so it does not depend on the belief-state. The same biased form can be found for approximations of $\rho_{HR}$. For example using the quadratic reward $\rho_{QR}(\mathbf{b}, a) = \rho_Q(b, a) + \eta \rho_R$ or for the linear reward $\rho_{LR}(\mathbf{b}, a) = \rho_L(b, a) + \eta \rho_R$.

**Example 4.3.2** (Cost-sensitive Cancer-Flu)**.** *Consider the medical diagnosis problem with two possible diseases:* $\mathcal{S} = \{cancer, flu\}$*. As detecting a cancer is more important that detecting a flu, let*

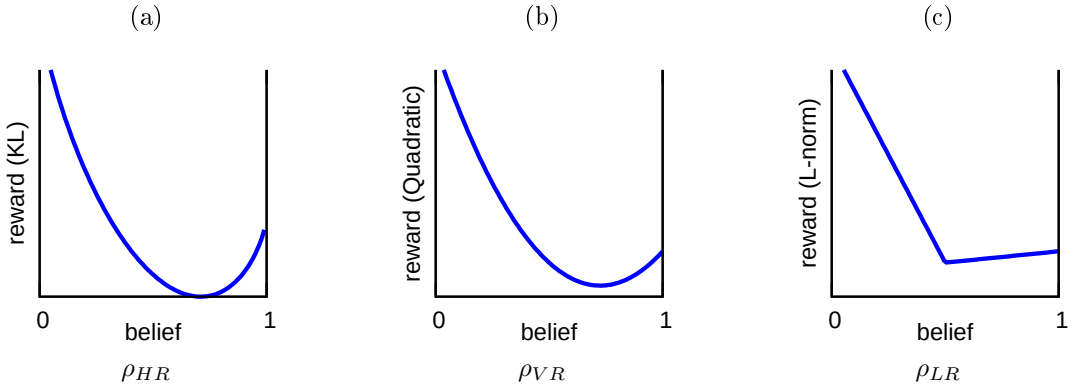



Figure 4.3: Combined rewards examples in a 1-simplex for (a) entropy + state-based, (b) quadratic + state-based, and (c) linear + state-based.

*$R(cancer, a) = 10$ while $R(flu, a) = 1$ for all $a \in \mathcal{A}$. The combined $\Delta$-rewards $\rho_{HR}, \rho_{VR}$ and $\rho_{LR}$ are plotted in Figure 4.3, for an arbitrary value of $\eta = 0.1$, where the state-based reward biases the entropy-based reward. For the cancer-flu problem, state-dependent rewards like in Figure 4.1(a), will always avoid actions that lead to diagnose a flu, while the combined reward better grasps the idea that ignorance is what should be avoided. Please note that this cost-sensitive problem make sense only because the state of the patient cannot be modified by the exams. For example, if there is a risk of acquiring cancer through a radioactive scanner, the agent may repeatedly perform this exam to maximizing the probability of causing cancer to the patient and therefore increasing the total return.*

### 4.3.3 Information-based Performance Criteria

To properly define an information-gathering problem, it is not sufficient to define an information measure as a reward, but a performance criterion must also be selected. Depending on the problem nature, the time horizon may be finite or infinite, the objective may be to optimize the total or final information, and some criterion could depend only on specific target variables rather than the complete belief.

#### Sum-of-Information Criterion

In general, the finite horizon performance criterion of a belief-MDP is

$$V_H^\pi(\mathbf{b}_0) = \mathbb{E}_{B_{1:H}}\left[\sum_{t=1}^{H} \gamma^{t-1} \rho(B_t, \pi(B_t, t)) \middle| B_0 = \mathbf{b}_0, \pi\right]. \tag{4.3}$$

In the information-gathering case, this criterion sums each step's information measure, which means that the objective is to always be in a high-valued information-state if $\gamma = 1$. On the other hand, if $\gamma < 1$ then the objective is to obtain information as quickly as possible.

Using the equations of Section 3.2.2, this performance criterion can be rewritten in terms of the elements of a $\rho$POMDP and, in theory, Bellman equations can be constructed for any $\rho$ to apply dynamic programming. In practice, analytical solutions for information-based rewards are hard to find, so an alternative is to use approximate methods.

If the horizon is not known, then the reasoning of Equation 3.3 can be applied here, evaluating the policy on an infinite-horizon $\rho$POMDP (with $\gamma < 1$).

#### Final-Information Criterion

In some problems, such as active diagnosis, the objective is to disambiguate the state of a static hidden target variable. This means that the information about the target variable increases in average

with time, so that intermediate steps are only means to arrive to a high-valued information-state. Then, the objective is to optimize the information *at the end* of the process, meaning that the information measure is a single reward that is given to the agent in the final step. Therefore, the same criterion of Equation 4.3 can be interpreted as the sum of a non-stationary $\Delta$-reward with zero reward for all steps except the final one.

This criterion can be written for the finite horizon as follows:

$$V_H^\pi(\mathbf{b}_0) = \mathbb{E}_{B_{1:H}}\left[\rho(B_H, \pi(B_H))|B_0 = \mathbf{b}_0, \pi\right].$$

If the horizon is not known, then the expectation over a stopping probability can be applied similarly to Equation 3.3.

$$\begin{aligned}
V^\pi(\mathbf{b}_0) &= \mathbb{E}_H[V_H^\pi(\mathbf{b}_0)] \\
&= \lim_{n\to\infty} \sum_{h=1}^{n} Pr(H=h)\mathbb{E}_{B_{1:h}}\left[\rho(B_h, \pi(B_h))|B_0 = \mathbf{b}_0, \pi\right] \\
&= \lim_{n\to\infty} \sum_{h=1}^{n} \gamma^{h-1}(1-\gamma)\mathbb{E}_{B_{1:h}}\left[\rho(B_h, \pi(B_h))|B_0 = \mathbf{b}_0, \pi\right] \\
&= (1-\gamma) \lim_{n\to\infty} \mathbb{E}_{B_{1:h}}\left[\sum_{h=1}^{n} \gamma^{h-1}\rho(B_h, \pi(B_h))\middle| B_0 = \mathbf{b}_0, \pi\right],
\end{aligned}$$

which is equivalent to the infinite-horizon discounted sum-of-information criterion for optimization purposes.

Information-greedy approaches commonly consider the difference of information between two information-states as the agent's reward. The sum of these rewards for the undiscounted finite horizon gives an equivalent criterion to the finite-horizon final-information criterion. This is because the initial information value is fixed, and each intermediate step is trivially canceled using the previous and next steps. Unsurprisingly, the discounted infinite horizon case is also equivalent to the infinite-horizon sum-of-information criterion.

### 4.3.4 Value Function Convexity

An important property of POMDPs is that the belief-based value function is convex (see Section 3.2.3). For $\rho$POMDPs, this property also holds if the reward function $\rho(b, a)$ is convex, as shown in Theorem 4.3.1.

**Theorem 4.3.1.** *If $\rho$ and $V_0$ are convex functions over $\Delta$, then the value function $V_i$ of the belief-MDP is convex over $\Delta$ at any iteration $i$.*

*Proof.* Assuming that $V_{i-1}(\mathbf{b})$ is a convex function, it can be shown that $V_i(\mathbf{b})$ is also convex as follows. First, the value function in Equation 3.7 can be written by parts as:

$$\begin{aligned}
V_i(\mathbf{b}) &= \max_{a\in\mathcal{A}} \{V_i^a(\mathbf{b})\}, \\
V_i^a(\mathbf{b}) &= \sum_{z\in\mathcal{Z}} V_i^{a,z}(\mathbf{b}), \\
V_i^{a,z}(\mathbf{b}) &= \frac{\rho(\mathbf{b}, a)}{|\mathcal{Z}|} + \psi^{a,z}(\mathbf{b}), \text{ and} \\
\psi^{a,z}(\mathbf{b}) &= \gamma Pr(Z = z|A = z, B = \mathbf{b}) V_{i-1}(Bayes(\mathbf{b}, a, z)).
\end{aligned}$$

Also, by using the vectorial representation of the belief-update of Equation 3.11,

$$\psi^{a,z}(\mathbf{b}) = \gamma \|P^{a,z}\mathbf{b}\|_1 V_{i-1}\left(\frac{P^{a,z}\mathbf{b}}{\|P^{a,z}\mathbf{b}\|_1}\right) = \gamma\kappa(P^{a,z}\mathbf{b})$$

with $\kappa(\mathbf{w}) = \|\mathbf{w}\|_1 V_{i-1}\left(\frac{\mathbf{w}}{\|\mathbf{w}\|_1}\right)$.

Here, $\kappa(\mathbf{w})$ is a convex function as it uses the perspective and linear-fractional convexity preserving operations (see Appendix A.1.1 for a stand-alone proof). Then, $\psi^{a,z}$ is also convex since convexity is preserved under affine maps. Consequently, $V_i^{a,z}$, $V_i^a$ and $V_i$ are convex because $\rho$ and $\psi^{a,z}$ are convex. [10] Considering this last result as the inductive step, then $V_i(b)$ is convex for any $i$, because at base step $i = 0$, $V_0$ is convex by definition. □

This last theorem is based on $\rho(\mathbf{b}, a)$ being a convex function over $\mathbf{b}$, which is a commonly found property in information measures, because the objective is to avoid belief distributions that do not give much information on which state the system is in, and to assign higher rewards to those beliefs that give higher probabilities of being in a specific state. Thus, a reward function meant to reduce the uncertainty must provide high payoff near the corners of the simplex, and low payoff near its center. However, please be aware that, for some strange or degenerate cases, information measures could be non-convex, making impossible to directly apply this and the further results of this chapter.

The initial value function $V_0$ might be any convex function for infinite-horizon problems, but by definition $V_0 = 0$ for finite-horizon problems. By starting with $V_0 = 0$, it is also easy to prove by induction that, if $\rho$ is continuous (respectively differentiable), then $V_i$ is continuous (respectively *piecewise* differentiable).

### 4.3.5 Solution Techniques for PWLC-rewards

This section addresses the problem of solving $\rho$POMDPs in the special case where $\rho$ is a convex and a PWL function (such as $\rho_L$), showing that only small adaptations of the exact and approximate POMDP algorithms are needed to compute a (near) optimal value function. The more complex case, when $\rho$ is convex but not PWL, is left for Section 4.3.7.

#### Exact Value Iteration

If $\rho(b, a)$ is a PWLC function, it can be represented by a set $\Psi^a$ for each $a$ (see Section 3.2.3). Therefore, the $\Delta$-reward can be written in the form

$$\rho(\mathbf{b}, a) = \max_{\beta \in \Psi^a} \left\{ \mathbf{b}^\top \beta \right\}.$$

This definition leads to the following changes in the value iteration expression of Equation 3.12

$$V_t(\mathbf{b}) = \max_{a \in \mathcal{A}} \left\{ \max_{\beta \in \Psi^a} \left\{ \mathbf{b}^\top \beta \right\} + \gamma \sum_{z \in \mathcal{Z}} \max_{\alpha \in \Gamma_{t-1}} \left\{ \mathbf{b}^\top {P^{a,z}}^\top \alpha \right\} \right\}.$$

With this equation, the exact algorithms of Section 3.2.4 like Batch Enumeration or Incremental Pruning can then be applied to this POMDP extension in a similar way as for POMDPs. The only difference is that Equations 3.13, 3.14 and 3.15 must be modified in order to support several reward $\alpha$-vectors in the propagation and not just one as in normal POMDPs. To do so, the reward vector $\mathbf{r}^a$ can be removed from the computation of $\Gamma_t^{a,z}$ in Equation 3.13, and the new reward-set $\Psi^a$ can be included in Equation 3.14 as an additional set to cross-sum. Concisely, the value iteration step for $\rho$POMDPs can be summarized as follows:

$$\Gamma_t^{a,z} \leftarrow \left\{ \gamma {P^{a,z}}^\top \alpha \middle| \alpha \in \Gamma_{t-1} \right\}, \text{ and}$$

$$\Gamma_t \leftarrow \bigcup_a \left( \bigoplus_o \Gamma_t^{a,z} \oplus \Psi^a \right).$$

This value function propagation also produces PWLC value functions as can be seen in Figure 4.4.

Please note that the cross-sum generates $|\Psi^a|$ times more vectors than with a classic POMDP, where $|\Psi^a|$ is the number of $\alpha$-vectors specifying the $\rho(b, a)$ function. Notably, an implementation of these algorithms for $\rho$POMDPs can be used also for normal POMDPs, because if $\Psi^a = \{\mathbf{r}^a\}$ for all $a \in \mathcal{A}$, the proposed equations are identical to the exact value iteration of Section 3.2.4.

10. Convex functions are closed under the sum and the max operators.

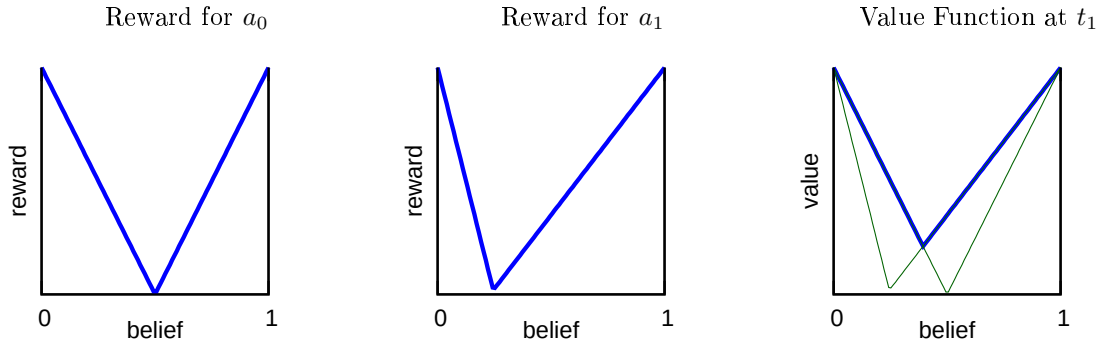



Figure 4.4: Graphical representation of the propagation of a PWLC reward into a PWLC value function. The green lines are the vectors (or parts of them) that are dominated by the blue ones.

**Point-Based Value Iteration**

PB approximations can be used in a similar way as PBVI or PERSEUS apply to POMDPs. The only difference is again the reward function representation as an envelope of hyperplanes. PB algorithms select the hyperplane that maximizes the value function at each belief-point, so the same can be applied to the set $\Psi^a$. Then, for the *backup* computations of Section 3.2.6, the $\Gamma$-set computations can be modified as follows:

$$
\begin{aligned}
V_t^*(\mathbf{b}) &= \max_{a\in\mathcal{A}} \left\{ \max_{\beta\in\Psi^a} \left\{\mathbf{b}^\top \beta\right\} + \gamma \sum_{z\in\mathcal{Z}} \max_{\alpha\in\Gamma_{t-1}} \left\{\mathbf{b}^\top P^{a,z\top}\alpha\right\} \right\} \\
&= \max_{a\in\mathcal{A}} \left\{ \mathbf{b}^\top \operatorname*{argmax}_{\beta\in\Psi^a} \left\{\mathbf{b}^\top \beta\right\} + \gamma \sum_{z\in\mathcal{Z}} \mathbf{b}^\top P^{a,z\top} \operatorname*{argmax}_{\alpha\in\Gamma_{t-1}} \left\{\mathbf{b}^\top P^{a,z\top}\alpha\right\} \right\} \\
&= \max_{\alpha\in\Gamma_t^{\mathbf{b}}} \mathbf{b}^\top \alpha, \text{ where} \\
\Gamma_t^{\mathbf{b}} &= \left\{ \operatorname*{argmax}_{\beta\in\Psi^a} \left\{\mathbf{b}^\top \beta\right\} + \gamma \sum_{z\in\mathcal{Z}} P^{a,z\top} \operatorname*{argmax}_{\alpha\in\Gamma_{t-1}} \left\{\mathbf{b}^\top P^{a,z\top}\alpha\right\} \middle| a\in\mathcal{A} \right\}.
\end{aligned}
$$

Independent of the chosen COLLECT and UPDATE techniques, all the PB methods use the same *backup* function, so any PB method can be now used for $\rho$POMDPs.

In contrast to exact value iteration, PB methods do not increase the amount of used vectors, because for each point there is only one reward $\alpha$-vector to propagate. However, a small additional computation time is needed to select the best support reward $\alpha$-vector for each belief-point. Similarly to exact value iteration, if $\Psi^a = \{\mathbf{r}^a\}$ for all $a \in \mathcal{A}$, the same implementation can be used to solve normal POMDPs.

### 4.3.6 Active Classification and PWLC-reward Functions

PWLC-reward functions may seem only a rough approximation of an information measure such as the KL divergence, but they have been implicitly used to successfully solve sequential decision problems that need information gathering. For example, active classification problems (Settles, 2009) consist in choosing the actions that minimize either the expected misclassification rate or the expected loss, which is a way of reducing the expected error of estimation. As measuring information is strongly related to the statistical notion of measuring the error, there is also a strong relationship between sequential information-gathering problems and active classification problems. This section intends to give some hints about this relationship.

Here, the focus is set on problems where the feature acquisition queries depend on the past queried features, because the independent case is not interesting from the sequential decision-theory lens. Under this dependency assumption, an active classification problem is usually modelled as a finite horizon

POMDP, where the latent target variable is the class to which the classified object belongs, the actions are the feature acquisition queries, and the observations are the actual values of the features. Also, at the final step of the process, the agent must choose from another set of actions (i.e., classification actions) to actually classify the object. The reward function can be, for example, a simple 0-1 loss function (i.e., misclassification) that gives 1 or 0 reward at the final step depending if the agent has correctly classified the object or not.

**Example 4.3.3** (Medical Classification)**.** *Consider the medical diagnosis problem (Example 4.3.2) with two diseases,* $S = \{d_1, d_2\}$*, but adding the final objective of classifying the patient into one of the two possible diseases. If the patient is incorrectly diagnosed the cost of misclassification is* 1*, while there is a cost of* 0 *if the patient is correctly classified. Also, the exams and procedures have a zero cost, but there is only a finite number of steps* $H$ *before making a classification decision.*

The relationship between this kind of problems and information-gathering problems can be established by analyzing the final step of the process. First, note that the agent only gets rewarded at the final step, meaning that the value function at iteration 1, using for example a 0-1 reward function $R(s,a) = \mathbb{I}(s,a)$, can be written as:

$$\begin{aligned} V_1(\mathbf{b}) &= \max_{a \in \mathcal{S}} \left\{ \sum_{s \in \mathcal{S}} \mathbf{b}(s) \mathbb{I}(s,a) \right\} \\ &= \max_{s \in \mathcal{S}} \{\mathbf{b}(s)\} \\ &= \rho_L(\mathbf{b}). \end{aligned}$$

Then, as all the other steps have zero reward, the active classification problem corresponds to the information-gathering problem with the linear reward under the final-information criterion. Following statistical decision theory, this same idea can be generalized to other loss functions in order to include type I and type II errors, or even consider cost-sensitive classification (Greiner et al., 2002) where combined information-based rewards can be used. Moreover, (Settles, 2009) presents similar uncertainty measures to the ones used by this chapter (like the entropy or a $L_2$-norm) to be used as loss functions.

Some classical POMDP problems, such as the *tiger* problem (Cassandra, 1998), can be modelled as active classification problems, and therefore they intrinsically optimize an information measure (PWL) before choosing the final decision.

### 4.3.7 Generalizing to non-PWL Reward Functions

Even though some interesting applications can be characterized using PWLC $\Delta$-rewards, some information measures such as the KL-divergence are not piecewise linear functions. In theory, each step of value iteration can be analytically computed using non-PWL functions, but the expressions are not closed as in the linear case, growing in complexity and making them unmanageable after a few steps. Moreover, pruning techniques cannot be applied directly to the resulting hypersurfaces, and even second order measures do not exhibit standard quadratic forms to apply quadratic programming. However, convex functions can be efficiently approximated by piecewise linear functions, making it possible to apply the techniques described in Section 3.2 with a bounded error, as long as the approximation of $\rho$ is bounded.

#### A PWLC Approximation of $\rho$

First, please note that the information-based rewards of Section 4.3.1 do not depend on the action. Therefore, and without loss of generality, $\rho(\mathbf{b})$ will be used in this section instead of $\rho(\mathbf{b}, a)$. However, when the reward function is a combination of information-based and state-based rewards, the process described in this section must be performed for each action separately.

Consider then a continuous, convex and piecewise differentiable reward function $\rho(\mathbf{b})$, and an arbitrary (and finite) set of points $\mathcal{B} \subset \Delta$ where the gradient $\nabla\rho$ is well defined (see Figure 4.5(a)). A lower PWLC approximation of $\rho(\mathbf{b})$ can be obtained by using each element $\mathbf{b}' \in \mathcal{B}$ as a base point for constructing a tangent hyperplane which is always a lower bound of $\rho(\mathbf{b})$ (see Figure 4.5(b)). Concretely, $\omega_{\mathbf{b}'}(\mathbf{b}) =$

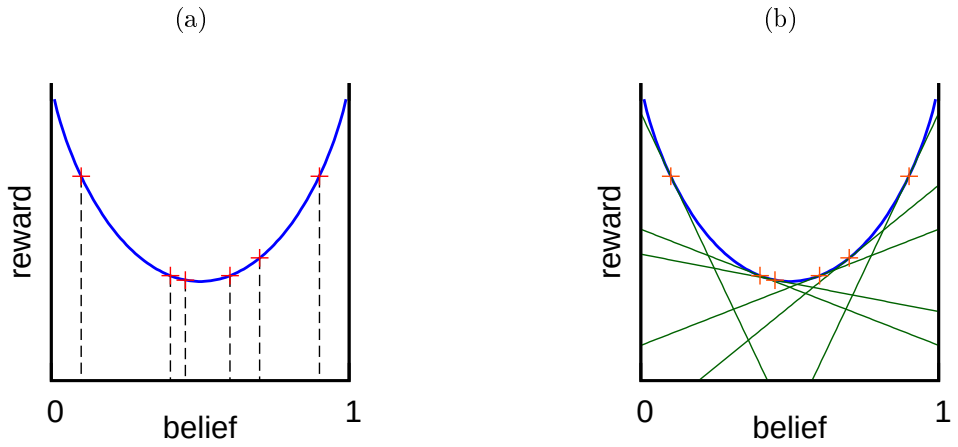



Figure 4.5: A PWLC approximation of a non-linear reward using tangent hyperplanes. In (a) a set of points in the belief-space and their projections to the reward function. In (b) the tangent hyperplanes of the reward function at each belief-point.

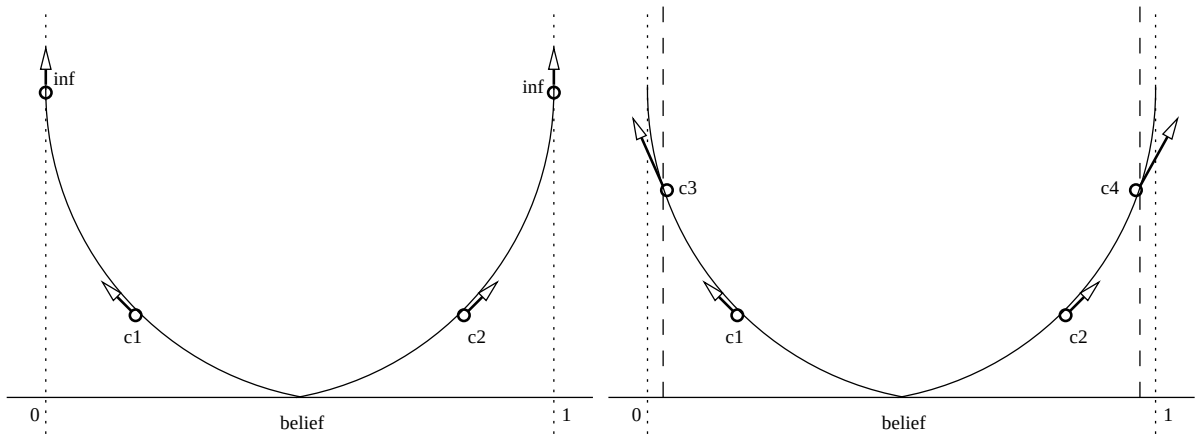



Figure 4.6: On the left a non Lipschitz but convex function. On the right the slope away from the border.

$\rho(\mathbf{b}') + (\mathbf{b} - \mathbf{b}')^\top \nabla\rho(\mathbf{b}')$ is the linear function that defines the tangent hyperplane, which leads to a lower approximation of $\rho(\mathbf{b})$ using a set $\mathcal{B}$ of

$$\rho_{\mathcal{B}}(\mathbf{b}) = \max_{\mathbf{b}' \in \mathcal{B}} \omega_{\mathbf{b}'}(\mathbf{b}).$$

At any point $\mathbf{b} \in \Delta$ the approximation error can be written as

$$\epsilon_{\mathcal{B}}(\mathbf{b}) = |\rho(\mathbf{b}) - \rho_{\mathcal{B}}(\mathbf{b})|.$$

Moreover, the worst case error can be obtained by specifically picking $\mathbf{b}$ as the point where $\epsilon_{\mathcal{B}}(\mathbf{b})$ is maximal. Next section discusses the conditions for which this worst error is bounded by a controllable parameter.

### Approximating an $\alpha$-Hölderian $\rho$

It is well known that a piecewise linear approximation of a Lipschitz function is bounded because the gradient $\nabla\rho(\mathbf{b}')$ that it is used to construct the hyperplane has bounded norm (Saigal, 1979). Unfortunately, some interesting functions are not Lipschitz, such as the KL-divergence ($f(x) = x\log(x/c)$ has

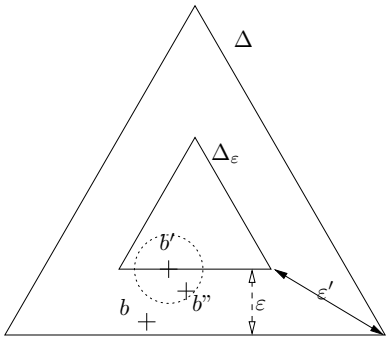



Figure 4.7: Worst distance from $b \in \Delta$ to $b'' \in \mathcal{B}$. This figure shows the simplexes $\Delta$ and $\Delta_\varepsilon$, and the points $\mathbf{b}$, $\mathbf{b}'$ and $\mathbf{b}''$.

an infinite slope when $x \to 0$), so this result is not generic enough to cover a wide range of information-gathering problems (see Figure 4.6). Yet, under certain mild assumptions, a proper error bound can still be found.

The aim of this section is to find a more generic error bound in three steps.

- First, some basic results over the simplex and the convexity of $\rho$ are presented. Informally, Lemma 4.3.2 will show that, for each $\mathbf{b}$, it is possible to find a belief point in $\mathcal{B}$ far enough from the boundary of the simplex but within a bounded distance from $\mathbf{b}$.
- In a second step, assuming that the function $\rho(b)$ verifies the $\alpha$-Hölder condition, it can be shown that the norm of the gradient is bounded for all points not in the boundary of the simplex (Lemma 4.3.3) [11].
- Finally, Theorem 4.3.4 will use both lemmas to bound the error of $\rho$'s approximation under these assumptions.

More precisely, for each point $\mathbf{b} \in \Delta$, it is possible to associate a point $\mathbf{b}^* = \underset{\mathbf{b}' \in \mathcal{B}}{\operatorname{argmax}}\{\omega_{\mathbf{b}'}(\mathbf{b})\}$ corresponding to the support point in $\mathcal{B}$ whose tangent hyperplane gives the highest (and thus best) approximation of $\rho$ at $\mathbf{b}$. Consider the point $\mathbf{b} \in \Delta$ where $\epsilon_{\mathcal{B}}(\mathbf{b})$ is maximum: this error can be easily computed using the gradient $\nabla\rho(\mathbf{b}^*)$. Unfortunately, some partial derivatives of $\rho$ may diverge to infinity (usually on the boundary of the simplex) in the non-Lipschitz case, making the error hard to analyze. Therefore, to ensure that this error can be bounded, instead of $\mathbf{b}^*$, a safe $\mathbf{b}'' \in \mathcal{B}$ is selected (far enough from the boundary) by using an intermediate point $\mathbf{b}'$ in an *inner simplex* $\Delta_\varepsilon$, where $\Delta_\varepsilon = \{\mathbf{b} \in [\varepsilon, 1]^n | \sum_{s \in \mathcal{S}} b(s) = 1\}$ with $n = |\mathcal{S}|$.

Thus, for a given $\mathbf{b} \in \Delta$ and $\varepsilon \in (0, \frac{1}{n}]$, the point $\mathbf{b}' \in \underset{\hat{\mathbf{b}} \in \Delta_\varepsilon}{\operatorname{argmin}} \|\hat{\mathbf{b}} - \mathbf{b}\|_1$ is the closest point to $\mathbf{b}$ in $\Delta_\varepsilon$ and $\mathbf{b}'' \in \underset{\hat{\mathbf{b}} \in \mathcal{B}}{\operatorname{argmin}} \|\hat{\mathbf{b}} - \mathbf{b}'\|_1$ is the closest point to $\mathbf{b}' \in \mathcal{B}$ (see Figure 4.7). These two points will be used to find an upper bound for the distance $\|\mathbf{b} - \mathbf{b}''\|_1$ based on the *density* of $\mathcal{B}$, defined as in the PBVI analysis, $\delta_{\mathcal{B}} = \min_{\mathbf{b} \in \Delta} \max_{\mathbf{b}' \in \mathcal{B}} \|\mathbf{b} - \mathbf{b}'\|_1$.

**Lemma 4.3.2.** *The distance ($L_1$-norm) between the maximum error point $\mathbf{b} \in \Delta$ and the selected $\mathbf{b}'' \in \mathcal{B}$ is bounded by $\|\mathbf{b} - \mathbf{b}''\|_1 \leq 2(n-1)\varepsilon + \delta_{\mathcal{B}}$.*

*Proof.* The largest minimum distance between the boundaries of both simplices (see Fig. 4.7) is given by the distance between their closest corners, i.e.,

$$\varepsilon' = |(1 - (n-1)\varepsilon) - 1| + (n-1)|\varepsilon| = 2(n-1)\varepsilon.$$

This is the worst case scenario for $\|\mathbf{b}' - \mathbf{b}''\|_1$. Then, using the triangular inequality:

$$\|\mathbf{b} - \mathbf{b}''\|_1 \leq \|\mathbf{b} - \mathbf{b}'\|_1 + \|\mathbf{b}' - \mathbf{b}''\|_1,$$

11. The $\alpha$-Hölder condition is a generalization of the Lipchitzian condition.

and by picking the largest possible values for both distances, the desired bound holds,

$$\|\mathbf{b} - \mathbf{b}''\|_1 \leq 2(n-1)\varepsilon + \delta_{\mathcal{B}}.$$

□

It can be ensured that $\mathbf{b}''$ is not on the boundary of the simplex $\Delta$ by picking $\varepsilon > \delta_{\mathcal{B}}$, with a minimum distance from the boundary of $\eta = \varepsilon - \delta_{\mathcal{B}}$. This will allow finding bounds for the PWLC approximation of convex $\alpha$-Hölder functions, which is a broader family of functions including the KL divergence, convex Lipschitz functions and others. The $\alpha$-Hölder condition is a generalization of the Lipschitz condition [12]. Formally, an $\alpha$-Hölderian function $f : \mathcal{D} \mapsto \mathbb{R}$, with $\mathcal{D} \subset \mathbb{R}^n$, complies with

$$\exists \alpha \in (0, 1],\ \exists K_\alpha > 0,\ \text{s.t. } |f(\mathbf{x}) - f(\mathbf{y})| \leq K_\alpha \|\mathbf{x} - \mathbf{y}\|_1^\alpha.$$

The limit case, where a convex $\alpha$-Hölder function has infinite-valued norm for the gradient, is always on the boundary of the simplex $\Delta$ due to the convexity property, and therefore the point $\mathbf{b}''$ will be free of this predicament because of $\eta$. More precisely, an $\alpha$-Hölder function in $\Delta$ with constant $K_\alpha$ in 1-norm complies with the Lipschitz condition on $\Delta_\eta$ with a constant $K_\alpha \eta^\alpha$ (see Lemma A.1.2). Moreover, the norm of the gradient $\|\nabla f(\mathbf{b}'')\|_1$ is also bounded as stated by Lemma 4.3.3.

**Lemma 4.3.3.** *Let $\eta > 0$ and $f$ be an $\alpha$-Hölder (with constant $K_\alpha$), bounded and convex function from $\Delta$ to $\mathbb{R}$, $f$ being differentiable everywhere in $\Delta^o$ (the interior of $\Delta$). Then, for all $\mathbf{b} \in \Delta_\eta$, $\|\nabla f(\mathbf{b})\|_1 \leq K_\alpha \eta^{\alpha-1}$.*

*Proof.* For readability purposes, the proof of this lemma is in Appendix A.1.2. □

**Theorem 4.3.4.** *Let $\rho$ be a continuous and convex function over $\Delta$, differentiable everywhere in $\Delta^o$ (the interior of $\Delta$), and satisfying the $\alpha$-Hölder condition with constant $K_\alpha$. The approximation error $\epsilon_{\mathcal{B}}$ can be bounded by $C{\delta_{\mathcal{B}}}^\alpha$, where $C$ is a scalar constant that depends on $\alpha$.*

*Proof.* The point with the worst approximation is chosen for bounding the approximation error, $\mathbf{b} = \operatorname{argmax}_{\hat{\mathbf{b}} \in \Delta} \epsilon_{\mathcal{B}}(\hat{\mathbf{b}})$, so every other point has a smaller error. Now, the following chain of inequalities shows that this worst error is bounded:

$$\begin{aligned}
\epsilon_{\mathcal{B}}(\mathbf{b}) &\leq \rho(\mathbf{b}) - \rho_{\mathcal{B}}(\mathbf{b}) && \text{(definition of } \epsilon_{\mathcal{B}}(\mathbf{b})) \\
&\leq \rho(\mathbf{b}) - \omega_{\mathbf{b}^*}(\mathbf{b}) && \text{(by definition of } \rho_{\mathcal{B}}) \\
&\leq \rho(\mathbf{b}) - \omega_{\mathbf{b}''}(\mathbf{b}) && (\mathbf{b}'' \in \mathcal{B} \text{ makes a worse error}) \\
&= \rho(\mathbf{b}) - \rho(\mathbf{b}'') + (\mathbf{b}'' - \mathbf{b})^\top \nabla\rho(\mathbf{b}'') && \text{(by definition of } \omega) \\
&\leq |\rho(\mathbf{b}) - \rho(\mathbf{b}'')| + |(\mathbf{b}'' - \mathbf{b})^\top \nabla\rho(\mathbf{b}'')| && \text{(by triangular inequality)} \\
&\leq K_\alpha \|\mathbf{b} - \mathbf{b}''\|_1^\alpha + |(\mathbf{b}'' - \mathbf{b})^\top \nabla\rho(\mathbf{b}'')| && \text{(by } \alpha\text{-Hölder condition)} \\
&\leq K_\alpha \|\mathbf{b} - \mathbf{b}''\|_1^\alpha + \|\nabla\rho(\mathbf{b}'')\|_\infty \|\mathbf{b}'' - \mathbf{b}\|_1 && \text{(by Hölder inequality)} \\
&\leq K_\alpha \|\mathbf{b} - \mathbf{b}''\|_1^\alpha + \|\nabla\rho(\mathbf{b}'')\|_1 \|\mathbf{b}'' - \mathbf{b}\|_1 && \text{(by norm equivalence)} \\
&\leq K_\alpha \|\mathbf{b} - \mathbf{b}''\|_1^\alpha + K_\alpha \eta^{\alpha-1} \|\mathbf{b}'' - \mathbf{b}\|_1 && \text{(by Lemma 4.3.3)} \\
&\leq K_\alpha (2(n-1)\varepsilon + \delta_{\mathcal{B}})^\alpha + K_\alpha \eta^{\alpha-1} (2(n-1)\varepsilon + \delta_{\mathcal{B}}) && \text{(by Lemma 4.3.2)} \\
&= K_\alpha ((2(n-1)\varepsilon + \delta_{\mathcal{B}})^\alpha + (\varepsilon - \delta_{\mathcal{B}})^{\alpha-1} (2(n-1)\varepsilon + \delta_{\mathcal{B}})) && \text{(by definition of } \eta)
\end{aligned}$$

This last result is a generic bound that depends on the choice of $\varepsilon$, with $\varepsilon \in (\delta_{\mathcal{B}}, \frac{1}{n}]$. By defining $\varepsilon$ as a linear function of $\delta_{\mathcal{B}}$, $\varepsilon = \lambda \delta_{\mathcal{B}}$, then the generic bound can be written as

$$\epsilon_{\mathcal{B}}(\mathbf{b}) \leq K_\alpha \left[(2(n-1)\lambda + 1)^\alpha + (\lambda - 1)^{\alpha-1}(2(n-1)\lambda + 1)\right] {\delta_{\mathcal{B}}}^\alpha = C{\delta_{\mathcal{B}}}^\alpha$$

with $\lambda \in (1, \frac{1}{\delta_{\mathcal{B}} n}]$. □

12. Please note that $\alpha$ does not correspond to an $\alpha$-vector (which are denoted by $\alpha$ in bold), but to a scalar used for the $\alpha$-Hölder condition. This (almost) overlapping notation was kept to be consistent with the literature, and because the $\alpha$-Hölder condition is used only in this section.

### Exact Value Iteration

Knowing now that the approximation of $\rho$ is bounded for a wide family of functions, the techniques described in Section 4.3.5 can be directly applied using $\rho_{\mathcal{B}}(\mathbf{b})$ as the PWLC reward function. These algorithms can be safely used because the propagation of the error due to exact updates is bounded. A hyperplane can always be represented by a single vector $\mathbf{w}_{\mathbf{b}'}$, where $\omega_{\mathbf{b}'}(\mathbf{b}) = \mathbf{b}^\top \mathbf{w}_{\mathbf{b}'}$, because $\rho(\mathbf{b}') - \mathbf{b}'^\top \nabla\rho(\mathbf{b}')$ is a constant value that can be added to each element of the gradient due to $\|\mathbf{b}'\| = 1$.

**Theorem 4.3.5.** *Let $\bar{V}_t$ be the value function obtained using the PWLC approximation $\rho_{\mathcal{B}}(\mathbf{b})$, and $V_t^*$ the optimal value function, both at time $t$. The approximation error is bounded by*

$$\|\bar{V}_t - V_t^*\|_\infty \leq \frac{C{\delta_{\mathcal{B}}}^\alpha}{1-\gamma}.$$

*Proof.* This theorem can be proven using a similar methodology as in (Pineau et al., 2006; Lovejoy, 1991). Let $\mathbb{B}$ be the exact Bellman update operator and $\bar{\mathbb{B}}$ the exact Bellman operator but using the PWLC approximation of $\rho$. Then, the error from the real value can be bounded as follows:

$$\begin{aligned}
\|\bar{V}_t - V_t^*\|_\infty &= \|\bar{\mathbb{B}}\bar{V}_{t-1} - \mathbb{B}V_{t-1}^*\|_\infty && \text{(by definition)}\\
&\leq \|\bar{\mathbb{B}}\bar{V}_{t-1} - \mathbb{B}\bar{V}_{t-1}\|_\infty + \|\mathbb{B}\bar{V}_{t-1} - \mathbb{B}V_{t-1}^*\|_\infty && \text{(by triangular inequality)}\\
&\leq |\mathbf{b}^\top \mathbf{w}_{\mathbf{b}^*} + \mathbf{b}^\top \alpha_{\mathbf{b}^*} - \rho(\mathbf{b}) - \mathbf{b}^\top \alpha_{\mathbf{b}^*}| + \|\mathbb{B}\bar{V}_{t-1} - \mathbb{B}V_{t-1}^*\|_\infty && \text{(maximum error at } \mathbf{b})\\
&\leq C{\delta_{\mathcal{B}}}^\alpha + \|\mathbb{B}\bar{V}_{t-1} - \mathbb{B}V_{t-1}^*\|_\infty && \text{(by Theorem 4.3.4)}\\
&\leq C{\delta_{\mathcal{B}}}^\alpha + \gamma\|\bar{V}_{t-1} - V_{t-1}^*\| && \text{(by contraction)}\\
&\leq \frac{C{\delta_{\mathcal{B}}}^\alpha}{1-\gamma} && \text{(sum of a geometric series)}
\end{aligned}$$

□

For these algorithms, the selection of the set $\mathcal{B}$ remains open, raising similar issues as the selection of belief points in PB algorithms. However, as more points are included in the set, the error decreases, being zero on the limit when $|\mathcal{B}| \to \infty$. Unfortunately, points from $\Delta$'s boundary, where the gradient is not defined, have to be removed or slightly moved as the proofs only rely on interior points.

### Point-Based Value Iteration

In the case of PB algorithms, the extension is also straightforward, and the modifications to the *backup* method described in Section 4.3.5 can be used also with a bounded error. The selection of $\mathcal{B}$, the set of points for the PWLC approximation, and the set of points for the algorithm, can be shared. This simplifies the study of the bound when using both approximation techniques at the same time. However, please be aware that if the gradient is not defined in the boundary of $\Delta$, the collected points in this zone must be modified or removed.

**Theorem 4.3.6.** *Let $\hat{V}_t$ be the value function computed using the PWLC approximation of $\rho$ and the PBVI algorithm, and $V_t^*$ the optimal value function, both at time $t$. The approximation error is bounded by*

$$\|\hat{V}_t - V_t^*\|_\infty \leq \frac{(R_{max} - R_{min} + C{\delta_{\mathcal{B}}}^\alpha)\delta_{\mathcal{B}} + C{\delta_{\mathcal{B}}}^\alpha}{1-\gamma}.$$

*Proof.* The error between $\hat{V}_t$ and $V_t^*$ can be bounded as follows:

$$\begin{aligned}
\|\hat{V}_t - V_t^*\|_\infty &\leq \|\hat{V}_t - \bar{V}_t\|_\infty + \|\bar{V}_t - V_t^*\|_\infty && \text{(by triangular inequality)}\\
&\leq \|\hat{V}_t - \bar{V}_t\|_\infty + \frac{C{\delta_{\mathcal{B}}}^\alpha}{1-\gamma} && \text{(by Theorem 4.3.5)}\\
&\leq \frac{(R_{max} - R_{min} + C{\delta_{\mathcal{B}}}^\alpha)\delta_{\mathcal{B}}}{1-\gamma} + \frac{C{\delta_{\mathcal{B}}}^\alpha}{1-\gamma} && \text{(see below)}
\end{aligned}$$

At the last step, the first term of the triangular inequality can be bounded by a similar reasoning as in (Pineau et al., 2003), where the worst case for an $\alpha$-vector is $\frac{R_{min} - C{\delta_{\mathcal{B}}}^{\alpha}}{1-\gamma}$, while the best case is only $\frac{R_{max}}{1-\gamma}$, because the approximation is always a lower bound. □

Again, this error goes to zero in the limit when $\delta_{\mathcal{B}} = 0$, following the intuitive idea that a more dense $\mathcal{B}$-set produces less significant errors for both the PWLC approximation and the PBVI algorithm.

## 4.4 Related Work

The state of the art of sequential information-gathering decision problems is spread among several disciplines. Control theory, information theory, robotics, signal processing and machine learning have addressed this problem through the years under distinct names such as active perception, active vision, active sensing, sensor management, active feature acquisition, etc. This section presents a non exhaustive review of the approaches found in the literature and contrast them with this chapter's proposal.

### 4.4.1 Early Work

Bajcsy (1985) was one of the first to explicitly address the problem of optimizing information as a goal for a sequence of decisions in the active vision and control theory context. Hager and Mintz have developed this idea further by proposing a stochastic control model where the objective is to reduce the uncertainty by choosing the optimal control parameter vector (Hager, 1988). They have proposed simultaneously frequentist (Hager, 1987) and Bayesian approaches (Hager and Mintz, 1987): the first one is based on the covariance matrix due to the close relationship with the Fisher information matrix (given by the Cramér-Rao bound), and the second one is based on the Bayes risk using the 0-1 loss function (see Section 2.2.2 and Section 4.3.6). Even though these methods are inspired by information-theoretic approaches, they do not explicitly optimize a justified information measure, but only a related statistic such as covariance or quadratic loss. In (Bajcsy, 1988), a review of these ideas is presented under the name of *active perception*.

In the information theory domain, Hintz (1991) has used the difference of entropy between two information states to find a solution for the *cueing* problem. This problem consists in controlling a steering sensor (like the human eye) in order to track several targets, which is very similar to the already described multi-target surveillance problem.

One issue with all these early proposals is that they are myopic: they do not consider the decision dependencies, producing a sequence of actions that will be likely suboptimal for most of the problems described in Section 4.1. On the contrary, modelling the problem as a $\rho$POMDP allows reasoning using conditional plans (policies) and therefore adapting to different decision outcomes.

### 4.4.2 Robotics and Decision Theory

Robotics is one of the most prolific source of sequential information-based techniques. The idea of combining the sequential decision theory of Chapter 3 with information theory was first explored by (Kaelbling et al., 1996a). In this paper, the mobile-robot localization problem is modelled as a POMDP, yet the objective is a state-dependent reward based on some goal state. However, as solving POMDPs is generally intractable, an information-greedy heuristic is used to guide the robot to high-confidence information-states. This heuristic is applied when the entropy goes above some predefined threshold. Even though the information-gathering is also myopic and is only a means to reach the goal-state, this work has paved the way to very interesting research in the domain. For example, Burgard et al. (1997) have proposed to merge both the myopic (one-step) expected entropy and the state-dependent value function in a single criterion as a weighted sum, similarly to the combined reward presented in Section 4.3.1. Then, the action that maximizes the value of this multi-criteria objective is selected.

Roy and Thrun (1999) were the firsts to include some type of explicit information-lookahead strategy in the robotics domain with the *coastal navigation* algorithm. The basic idea is that the information-based

statistic (i.e., the entropy) can be included in the state of the system by imposing some strong model assumptions (unimodal gaussianity, single goal state, etc.). Then, some form of dynamic programming can be applied using only state-dependent rewards because the entropy is included in the state. Besides these strong assumptions, this technique assumes that this augmented state is Markovian, although it is not, so no near-optimality guarantees can be ensured like the ones proposed in this chapter. However, the qualitative performance of this algorithm is very good, showing that optimizing the future information states is a promising idea.

Fox et al. (1998) have explicitly proposed the *active localization* problem based on an information measure (entropy) as a stand-alone problem, and not only as a means to reach some goal. However, the solution techniques were also myopic in terms of information as most of the previous work. These ideas where combined by Mihaylova et al. (2002) with the early control theory work in information gathering (Bajcsy, 1988; Hager, 1988), proposing an optimal lookahead solution for the so called *active sensing* problem. Even though the formalism is correct, and resembles the $\rho$POMDP framework with the combined information-based reward of Equation 4.2, no hints about how to solve these problems are given. In a later publication (Mihaylova et al., 2006), some experiments were made, yet the solution techniques rely on myopic approaches like in (Bajcsy, 1988) or on strong assumptions like in (Roy and Thrun, 1999).

Some non POMDP-based techniques have been proposed for sequential information-gathering problems (Sim and Roy, 2004; Singh et al., 2009). Yet, they are mainly focused on finding non-conditional plans for informative path planning or similar problems, which escapes the focus of this dissertation.

### 4.4.3 Signal-processing and Sensor Management

The signal processing community has also largely studied information-gathering problems, because noisy sensors and imperfect information are ubiquitous in this domain. Also, as the Hidden Markov Model (HMM) is a popular representation for partially observable signals, POMDPs are a natural formalism for sequential decisions in signal processing. In particular, the *sensor management* problem (Hero III et al., 2007) consists in deciding the sensor configuration that shall be used at each step to disambiguate the latent signal. This is clearly an active surveillance problem, because the sensor selection does not modify the hidden signal. Even though myopic techniques are also commonly used in this domain (e.g., (Zhao et al., 2002)), some researchers have explicitly addressed the problem of non-myopic sensor management.

In particular, Krishnamurthy (2002) has proposed a method for solving the sensor management problem modelled as a POMDP when an *error-based measure* is given, specifically when a norm distance from the simplex corners is used as a loss function. This is equivalent to solving a $\rho$POMDP with the already introduced approximations of $\rho_H$, like $\rho_{COR}$ or $\rho_Q$, depending on the norm degree. However, this work does not establish an information-based criterion, but just considers that minimizing an expected loss function (or expected error) is enough for optimally solving the problem. The solution technique proposed in this paper is based on the early grid-based approximate algorithms (Lovejoy, 1991), and the possible non-linearity of the loss functions is also tackled using a PWLC approximation, which is trivially bounded because the norms are Lipschitzian functions. Even though the proposed technique resembles the results of Section 4.3, the work is framed specifically for the sensor management problem, relying on the specific structure of the problem and proposing arbitrary loss functions rather than a formal performance criterion. Starting from the work of Krishnamurthy (2002), the same theory has been extended to support entropy-based loss functions (Rezaeian, 2007) and covariance-based loss functions (Vitus et al., 2010).

An interesting theoretical result is that for certain sensor management problems a myopic heuristic has a worst-case performance bound of one half of the optimal performance value. This result was first presented in (Krause and Guestrin, 2005) and then generalized in (Williams, 2007), showing that for certain problems a simple myopic heuristic can be adequate if only a fair performance is needed. Moreover, this theoretical bound is much closer to optimality in some specific cases, and in practice, the performances of myopic heuristics are usually significantly better than the theoretical worst case for sensor management problems.

Unfortunately, all this body of work relies on the specific properties of the sensor scheduling problem, making most of the solving techniques inapplicable to the general case of sequential information-gathering

problems addressed by this section.

On another track of sensor management, Kreucher et al. (2005) have explicitly defined the multi-target tracking problem as a POMDP with an entropy-based $\Delta$-reward. Indeed, they propose an information-based reward based on Rényi's generalization of Shannon's entropy (Rényi, 1960), which is a more flexible definition than the one used in this dissertation. To solve the non-PWL belief-MDP, they rely on Monte Carlo methods, avoiding the non-linearity problems found in Krishnamurthy (2002), and providing at the same time a solution to large or continuous state and action spaces.

### 4.4.4 Summary

The methods found in the literature for solving sequential information-gathering problems can be divided in three different groups:

- **POMDP approaches with information-theoretic heuristics**. These methods do not address the same problem as the one proposed in this chapter, but they use myopic or information-lookahead techniques to guide the approximate planning by explicitly gathering information.
- **Myopic approaches for information gathering**. Most of the related work, that explicitly defines the performance criteria based on an information-theoretic measure, falls in the category of myopic approaches, meaning that the decision is based only on a one-step information lookahead (information greedy). Unfortunately, the performance guarantees and efficiency of these methods are always restricted to specific classes problems.
- **Ad-hoc information-lookahead approaches**. There is a small number of methods that use information-lookahead to solve a proper sequential information-gathering problem. However, these methods rely on ad-hoc simplifications due to the properties of the specific problem addressed, confining their applicability only to the problems that share the same structure.

The novelty of the $\rho$POMDP framework is that it offers a generic [13] and sound way for modelling any sequential information-gathering problem, allowing to adapt classical belief-lookahead POMDP algorithms to this framework without relying on the problem's specifics.

## 4.5 Experiments

This section describes some experiments to illustrate the difference between myopic approaches and information-lookahead approaches using the $\rho$POMDP formalism. Even though there are several problems that can be fairly solved using myopic approaches (Williams, 2007), here the focus is specifically set on those problems where a myopic approach is not enough. Also, different information-based reward functions are compared to spot their difference, while the performance criterion remains fixed.

### 4.5.1 Experimental Setup

Before introducing the selected problems and showing results, this section describes the experimental setup.

#### Hardware and Software

For all the experiments, a 2.4GHz Intel Core 2 Duo CPU with 4GB of RAM was used, typically running one experiment per core. For the off-line computations and on-line simulations, a software package called *Sequential Decision Making Library* (`libSDM`) was used. This package is written in Java and is compatible with MATLAB, including support for MDPs, POMDPs and Bayesian-RL in an integrated framework that also currently supports information-based rewards. `libSDM` has been developed by the same author of this dissertation, based on the `libPOMDP` project led by Diego Maniloff. Even though `libSDM` performs the simulations and provides low-level statistics, most of the data post-processing and the experiment sequencing was made using MATLAB scripts.

13. Under the mild assumptions of convex $\alpha$-Hölderian information-based reward functions.

### Performance Criteria

The information measure used for all problems is the KL divergence, based on the wide acceptance of log-based functions as information measures, and on the axiomatic derivation of these log-based measures by Shannon (1948) and Kullback and Leibler (1951). Therefore, the performance criterion for each problem will be based on this measure, where the performance units corresponds to [*nats*] (as presented in Section 4.2). This dissertation does not claim that this is the only or the correct measure for all the applications, but for simplicity only this information measure will be used to measure performance.

In the experiments, the performance criterion corresponds either to (1) the final information or (2) the sum of information, depending on whether the horizon is fixed or not respectively. For all the infinite horizon problems the discount factor was set to $\gamma = 0.95$.

### Information-based Rewards

Regardless the fixed function for measuring performance (the KL divergence), the information-lookahead algorithms can use other information-based rewards as approximations of the KL divergence for computing the policy, as described in Sections 4.2 and 4.3.1. In this experiments section, three information-based rewards will be used:

- **Entropy-based reward**. This reward is exactly the KL divergence when the reference distribution is the uniform distribution, and is thus the most natural choice for the given performance criteria.
- **Quadratic reward**. This approximation of the entropy-based reward corresponds to the squared $L_2$-norm of the belief-state. This approximation was selected because its form resembles the entropy-based reward and because it was used in one of the few information-lookahead approaches found in the literature (Krishnamurthy, 2002).
- **Linear reward**. This upper PWLC approximation of the entropy-based reward corresponds to the $L_\infty$-norm of the belief-state. This approximation was selected due to its simplicity and because it appears naturally in the related problem of active classification (see Section 4.3.6).

### Policies and Algorithms

For each instance of a problem, the following algorithms were used:

- **Random policy**. At each step, the algorithm chooses a random action to execute. Any smart algorithm should perform better than (or at least equals to) this policy.
- **Myopic strategy**. At each step, the algorithm selects the action with the greatest expected next step reward. In simpler words, it is an *information-greedy* approach that chooses the immediate most informative action. Even though this approach may seem too simple, it is the most common method used in the literature, as presented in Section 4.4.
- **Information-lookahead approach**. Using the $\rho$POMDP definition and PWLC approximations if needed, a policy is obtained using point-based methods.

Two point-based algorithms (see Section 3.2.6) were used for the information-lookahead approach depending on the performance criterion. For infinite (and indefinite) horizons, the PERSEUS algorithm was used due to its time efficiency, performing asynchronous backups and using a constant set of belief-points gathered using random simulations. The asynchronous value iterations are stopped when the infinite norm of the difference between two successive iterations is below some threshold. Concretely,

$$\|V_i - V_{i-1}\|_\infty \leq \frac{(R_{max} - R_{min})\epsilon}{1-\gamma},$$

where $\epsilon$ is an accuracy parameter given to the algorithm. On the other hand, a modified version of PBVI was used for finite horizons, because the backup needs to be synchronous for generating non stationary policies. This PBVI version uses a constant set of belief-points as PERSEUS, rather than collecting points at each iteration as the original PBVI.

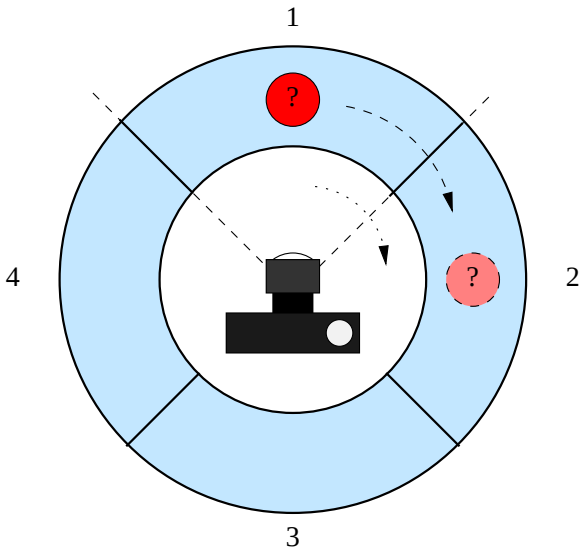



Figure 4.8: Graphical representation of the Camera-clean problem with 4 zones. The target can eventually move clockwise and the camera can be controlled to also move clockwise.

### 4.5.2 The Camera-clean Problems

The *Camera-clean* (CC) problem consists in a robotic camera that can be reoriented to shoot in $n$ different zones as shown in Figure 4.8, where $\mathcal{N} = \{1, 2, ..., n\}$ is the set of zones. The reorientation works only clockwise ($i_{move} = (i + 1)\%n + 1$), and during this move the camera cannot shoot. Also, the camera can perform an automatic maintenance procedure to *clean* the lens of the camera, because it gets dirty each time a photo is taken due to extreme environmental conditions, such as in underwater environments. Obviously, during this maintenance procedure the camera cannot shoot either. When the camera shoots, a photo is obtained and its quality depends on the state of the lens (clean or dirty). This photo is processed to determine whether the target object is in the zone or not. The success probability of this identification procedure depends on the photo quality, so clean photos are more informative than dirty ones, but an uninformative clean step is needed before shooting a clean photo.

In this section, three variations of the problem are used to exemplify the three different information-gathering problem types presented at the beginning of the chapter. All three are information-gathering problems for the described CC setup, but the objectives are different:

- **Diagnosis**: the objective is to know where the target object is after a given finite number of steps, where the camera orientation and lens's state are always known. Also, the camera movements and the clean procedure are deterministic actions.
- **Surveillance**: the objective is to "always" know where the target object is, but the object stays still or moves clockwise from a zone with some probability. Here, the camera orientation and the lens's state are also known, and again the movements and the clean procedure are deterministic.
- **Localization**: in contrast to the two previous cases, here the unknown variable is the position of the camera, while the static position of the object and the lens's state are known. The objective is to disambiguate the camera position, also with deterministic transitions for the camera.

To formalize these problems as $\rho$POMDPs, it is necessary to define first the state, action and observation spaces. A state $s \in \mathcal{S}$ of the system is factored into three variables $\langle cam, obj, len \rangle$, where $cam \in \mathcal{N}$ is the zone to which the camera is aiming, $obj \in \mathcal{N}$ is the zone where the object is, and $len$ is the state of the lens (clean or dirty). Therefore the complete state space is $\mathcal{S} = \mathcal{N} \times \mathcal{N} \times 2$. The camera actions are simply $\mathcal{A} = \{move, clean, shoot\}$, all three having a deterministic effect on the state: $move$ changes the $cam$ variable, while $clean$ and $shoot$ change the $len$ variable. The observation space is $\mathcal{Z} = \{true, false, nophoto\}$, where $true$ is obtained when the identification procedure has positively

| algorithm | zones | $\|\mathcal{B}\|$ | total return $[nats]$ | on-line time $[ms]$ | off-line time $[s]$ |
|---|---|---|---|---|---|
| **Random** | 3 | — | $0.49 \pm 0.28$ | $0.02 \pm 0.05$ | — |
| **Myopic** | 3 | — | $0.23 \pm 0.17$ | $7.03 \pm 9.84$ | — |
| **PB-Entropy** | 3 | 100 | $0.88 \pm 0.26$ | $10.95 \pm 0.94$ | $3.44 \pm 0.05$ |
| **PB-Quadratic** | 3 | 100 | $0.85 \pm 0.24$ | $11.06 \pm 0.92$ | $3.40 \pm 0.02$ |
| **PB-Linear** | 3 | 100 | $0.85 \pm 0.25$ | $\mathbf{3.70 \pm 0.92}$ | $\mathbf{1.28 \pm 0.11}$ |
| **PB-Entropy** | 3 | 2000 | $\mathbf{0.92 \pm 0.27}$ | $213.69 \pm 2.69$ | $1274.60 \pm 1.03$ |
| **PB-Quadratic** | 3 | 2000 | $\mathbf{0.92 \pm 0.27}$ | $213.87 \pm 2.66$ | $1274.16 \pm 0.67$ |
| **PB-Linear** | 3 | 2000 | $\mathbf{0.90 \pm 0.23}$ | $67.98 \pm 2.24$ | $407.22 \pm 2.50$ |
| **Random** | 4 | — | $0.45 \pm 0.28$ | $0.12 \pm 0.34$ | — |
| **Myopic** | 4 | — | $0.19 \pm 0.18$ | $9.76 \pm 10.38$ | — |
| **PB-Entropy** | 4 | 100 | $0.90 \pm 0.39$ | $19.16 \pm 2.06$ | $5.66 \pm 0.06$ |
| **PB-Quadratic** | 4 | 100 | $0.88 \pm 0.41$ | $18.64 \pm 1.20$ | $5.61 \pm 0.02$ |
| **PB-Linear** | 4 | 100 | $0.86 \pm 0.35$ | $\mathbf{4.93 \pm 0.97}$ | $\mathbf{1.78 \pm 0.14}$ |
| **PB-Entropy** | 4 | 2000 | $\mathbf{1.03 \pm 0.42}$ | $361.58 \pm 3.54$ | $2123.56 \pm 1.51$ |
| **PB-Quadratic** | 4 | 2000 | $\mathbf{1.04 \pm 0.43}$ | $361.42 \pm 3.35$ | $2123.41 \pm 0.92$ |
| **PB-Linear** | 4 | 2000 | $0.98 \pm 0.37$ | $96.33 \pm 3.26$ | $580.34 \pm 7.23$ |
| **Random** | 5 | — | $0.40 \pm 0.27$ | $0.02 \pm 0.07$ | — |
| **Myopic** | 5 | — | $0.18 \pm 0.20$ | $14.01 \pm 12.22$ | — |
| **PB-Entropy** | 5 | 100 | $0.84 \pm 0.50$ | $29.63 \pm 1.39$ | $9.16 \pm 0.07$ |
| **PB-Quadratic** | 5 | 100 | $0.75 \pm 0.47$ | $30.15 \pm 1.50$ | $9.15 \pm 0.07$ |
| **PB-Linear** | 5 | 100 | $0.80 \pm 0.43$ | $\mathbf{5.09 \pm 1.26}$ | $\mathbf{1.92 \pm 0.19}$ |
| **PB-Entropy** | 5 | 2000 | $\mathbf{1.04 \pm 0.57}$ | $577.27 \pm 5.13$ | $3365.78 \pm 3.96$ |
| **PB-Quadratic** | 5 | 2000 | $\mathbf{1.03 \pm 0.58}$ | $578.51 \pm 7.24$ | $3366.25 \pm 8.71$ |
| **PB-Linear** | 5 | 2000 | $0.96 \pm 0.48$ | $104.24 \pm 4.30$ | $622.98 \pm 15.91$ |

Table 4.1: Camera-clean Diagnosis Results. Mean total return, mean on-line time and mean off-line time over 10 repetitions of 500 trajectories of 20 steps. The results in bold are the best results for each group, and the red ones are the worst. For this and the following tables, the on-line time of the random strategy is not considered for selecting the best time, as it almost does not consume time at all.

detected the object, *false* when the object was not found, and *nophoto* when there is no photo to analyze (the action *clean* or *move* has been chosen). However, the identification procedure is not perfect, since false-positives and false-negatives can occur. Moreover, if the lens is dirty the identification procedure is even more inaccurate. Specifically, the probability of a correct identification [14] when the lens is clean is 0.8, while if the lens is dirty the probability falls to 0.55. With these values, an observation function can be defined that is common to the three variants of the problem. In contrast, the transition and reward functions largely depend on the type of the problem.

### Diagnosis

For the camera-clean diagnosis problem, the transition function $T(s, a, s')$ modifies deterministically the *cam* and *len* variables depending on $a$, but the *obj* variable never changes. The reward function is a non-stationary entropy-based function $\rho_H$ that depends on time and on the belief over *obj*, which is a probability vector of size $|\mathcal{N}|$. It is non-stationary because the rewards are zero for all but the final step of the process, inducing the final-information criterion of Section 4.3.3 ($\gamma = 1.0$).

The horizon of the problem was fixed to $H = 20$, because finite-horizon problems need to store large non-stationary policies, so larger horizons exhaust memory. Experiments were made for 3, 4 and 5 zones as shown in Table 4.1. Each experiment was repeated 10 times, because point-based methods may output different results depending on the initial (randomized) belief-point collection. Each generated policy was evaluated on-line using 500 trajectories, or in other words, 500 simulations were made for each experiment repetition. For the PB algorithms three different reward functions were used: the entropy-based reward

14. False positives and false negatives have the same probabilities.

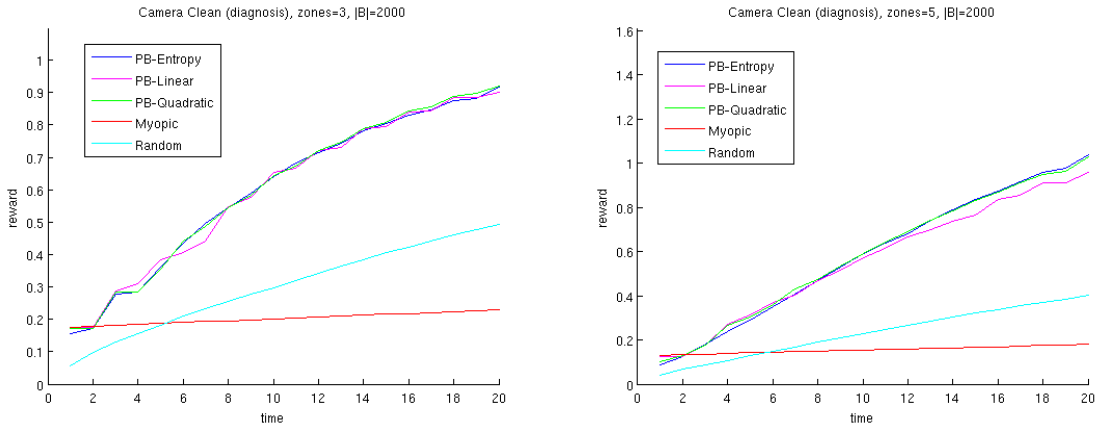



Figure 4.9: Camera-clean diagnosis reward evolution over the first 20 steps with $|\mathcal{B}| = 2000$. The left-hand figure shows the results for 3 zones and the right-hand one shows the results for 5 zones.

(with truncated gradient near the simplex boundary) $\rho_H$, the quadratic reward $\rho_Q$ and the linear reward $\rho_L$. For each of them, several sizes for the belief-point set were tested, yet in Table 4.1 the results are shown only for 100 and 2000 points [15]. The table shows the mean total return, the mean on-line time and the mean off-line time, each one with the corresponding standard deviation. However, please note that, for the total return and the on-line time, this value actually corresponds to the mean over the 10 different standard deviations.

The performance results are conclusive: the total returns of the information-lookahead approaches are always significantly larger than the myopic and random strategies, even with only a few belief-points. Indeed, with only 10 points (not in the table) the results are already slightly better than the random strategy. In particular, the myopic strategy performs very poorly in this problem because both *move* and *clean* actions do not produce any immediate information gain, degenerating into the inefficient strategy of always shooting for more information. In contrast, the random strategy is a better strategy for this problem, yet the information-gathering is slow, and many steps are needed to obtain a good performance. For example, in Figure 4.9 it can be seen that the performance achieved by the random strategy in 20 steps can be achieved with an information-lookahead approach in less than 8 steps.

The three variations of information lookahead behave similarly, the linear reward being only slightly worst for almost all the zones and belief-set sizes. However, this difference is not enough to draw a conclusion due to the high standard deviations. Therefore, the results show that using the alternative rewards ($\rho_Q$ and $\rho_L$) offers the same quality of solutions than $\rho_H$ in this specific problem.

The quantity of belief-points used to approximate the value function plays a significant role in both the value and time performances as in any PB method. The results show that with 20 times more points (100 to 2000), the total return increases significantly for all the problems and rewards, but this improvement requires approximately 20 times more on-line computational effort, and approximately 400 times more off-line computational effort. In particular, using a linear approximation offers a competitive value performance to the other rewards, with the advantage that the computational effort is approximately 4 times less than with the quadratic and entropy-based rewards. The on-line time needed for the random strategy is obviously almost zero, and the myopic approach exhibits a relatively light on-line computational-effort (with a very important standard deviation). However, these time-complexity advantages are meaningless if they cannot compete in value performance. Indeed, the highlighted rows in Table 4.1 shows that PB-Linear largely outperforms myopic and random strategies using approximately half of the on-line time needed by the myopic strategy, and only a couple of seconds in off-line time.

The upper bounds for the total returns of this problem are 1.10, 1.38 and 1.61 for 3, 4 and 5 zones respectively (minimum entropy). It is clear that the number of steps are not enough for completely

15. Values arbitrarily chosen only for illustration purposes.

| algorithm | zones | $\|\mathcal{B}\|$ | total return $[nats]$ | on-line time $[ms]$ | off-line time $[s]$ |
|---|---|---|---|---|---|
| **Random** | 3 | — | $17.04 \pm 4.18$ | $0.14 \pm 0.31$ | — |
| **Myopic** | 3 | — | $3.92 \pm 1.46$ | $26.69 \pm 12.65$ | — |
| **PB-Entropy** | 3 | 100 | $26.12 \pm 10.07$ | $\mathbf{20.89 \pm 5.49}$ | $\mathbf{2.49 \pm 0.66}$ |
| **PB-Quadratic** | 3 | 100 | $36.40 \pm 9.84$ | $25.66 \pm 6.24$ | $2.95 \pm 0.74$ |
| **PB-Linear** | 3 | 100 | $38.98 \pm 8.73$ | $28.31 \pm 6.16$ | $4.81 \pm 0.96$ |
| **PB-Entropy** | 3 | 2000 | $\mathbf{44.15 \pm 8.35}$ | $92.04 \pm 19.69$ | $21.90 \pm 4.46$ |
| **PB-Quadratic** | 3 | 2000 | $\mathbf{44.37 \pm 8.34}$ | $155.22 \pm 27.14$ | $49.11 \pm 8.96$ |
| **PB-Linear** | 3 | 2000 | $\mathbf{43.71 \pm 8.23}$ | $75.12 \pm 13.23$ | $22.56 \pm 2.87$ |
| **Random** | 4 | — | $18.37 \pm 5.53$ | $0.14 \pm 0.33$ | — |
| **Myopic** | 4 | — | $4.11 \pm 1.97$ | $41.76 \pm 12.44$ | — |
| **PB-Entropy** | 4 | 100 | $30.05 \pm 18.96$ | $\mathbf{36.59 \pm 10.22}$ | $\mathbf{3.77 \pm 0.84}$ |
| **PB-Quadratic** | 4 | 100 | $38.88 \pm 15.52$ | $41.94 \pm 8.29$ | $\mathbf{3.80 \pm 0.73}$ |
| **PB-Linear** | 4 | 100 | $42.13 \pm 13.89$ | $51.76 \pm 10.63$ | $7.10 \pm 2.25$ |
| **PB-Entropy** | 4 | 2000 | $\mathbf{55.64 \pm 13.91}$ | $220.77 \pm 45.85$ | $53.54 \pm 16.86$ |
| **PB-Quadratic** | 4 | 2000 | $\mathbf{57.12 \pm 13.70}$ | $316.11 \pm 62.47$ | $116.82 \pm 35.94$ |
| **PB-Linear** | 4 | 2000 | $\mathbf{57.27 \pm 13.26}$ | $283.64 \pm 47.07$ | $161.55 \pm 37.33$ |
| **Random** | 5 | — | $18.91 \pm 6.68$ | $0.11 \pm 0.26$ | — |
| **Myopic** | 5 | — | $4.19 \pm 2.43$ | $62.91 \pm 15.13$ | — |
| **PB-Entropy** | 5 | 100 | $23.29 \pm 19.34$ | $\mathbf{61.66 \pm 16.44}$ | $5.22 \pm 1.53$ |
| **PB-Quadratic** | 5 | 100 | $37.25 \pm 21.57$ | $68.34 \pm 25.18$ | $\mathbf{4.94 \pm 2.59}$ |
| **PB-Linear** | 5 | 100 | $35.85 \pm 23.42$ | $70.59 \pm 23.54$ | $7.43 \pm 5.74$ |
| **PB-Entropy** | 5 | 2000 | $\mathbf{66.27 \pm 18.74}$ | $395.89 \pm 95.63$ | $101.98 \pm 30.03$ |
| **PB-Quadratic** | 5 | 2000 | $\mathbf{65.95 \pm 18.91}$ | $607.50 \pm 143.66$ | $243.05 \pm 92.67$ |
| **PB-Linear** | 5 | 2000 | $\mathbf{64.96 \pm 18.57}$ | $594.32 \pm 124.00$ | $410.55 \pm 134.19$ |

Table 4.2: Camera-clean Surveillance Results. Mean total return, mean on-line time and mean off-line time over 10 repetitions of 500 trajectories of 100 steps. Again, results in bold are the best and results in red are the worst.

disambiguating the target position, yet information-lookahead approaches succeeded to do more than 82% of the task in only 20 steps for 3 zones, more than 74% for 4 zones, and more than 64% for 5 zones. On Figure 4.9 the mean behaviors of the reward for 3 and 5 zones are shown for each of the first 20 steps. For example, for the 3 zone case, the slopes of the information-lookahead curves are already decreasing as the value gets closer to the maximum, but in the 5 zone case the slopes remain almost the same. Also, the slightly different behavior of PB-Linear can be identified near the 20th step when using 5 zones. As 2000 belief points is a huge quantity of points for this small problem, this difference is probably caused by the fact that the linear reward is a harsh approximation of the entropy, but as stated before, the performance loss is not significant.

### Surveillance

For the camera-clean surveillance problem, the transition function also modifies deterministically the *cam* and *len* variables, but it is a stochastic function for the *obj* variable. The object can move to the next zone with a fixed probability of 0.05, which does not depend on the actions taken by the agent. Therefore, at each time there are 5% chances that the target silently moves to the next zone. In contrast to diagnosis, the reward function is now a stationary entropy-based function $\rho_H$ that depends only on the belief over *obj*. Therefore, the criterion used here is the sum-of-information criterion because there is no fixed horizon.

All experiments were run for an horizon of $H = 100$ steps, which is a sufficient horizon to observe converged behaviors. As for the diagnosis problem, experiments were made for 3, 4 and 5 zones, as shown in Table 4.2. All the other parameters (trials, repetitions, etc.) were set as for the diagnosis problem.

The global results are similar to the diagnosis problem, where information-lookahead techniques with

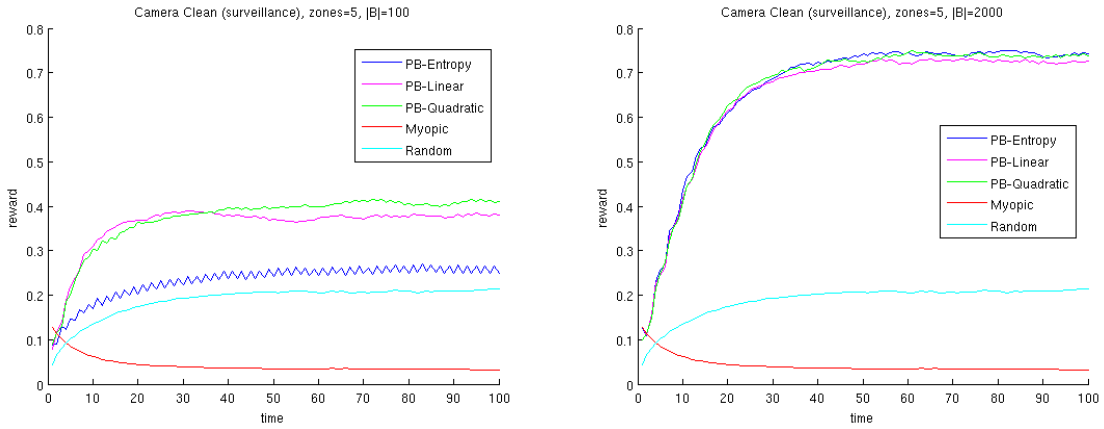



Figure 4.10: Camera-clean surveillance reward evolution for 5 zones over the first 100 steps. The left-hand figure shows the results for $|\mathcal{B}| = 100$ and the right-hand one shows the results for $|\mathcal{B}| = 2000$.

a moderate amount of points already outperformed myopic and random strategies. Unsurprisingly, the total return of information-lookahead policies improves with more points, and with it the on-line and off-line computational effort increases.

An interesting difference here is that the total return when using the entropy-based function with 100 belief-points is considerably lower than when using a linear or quadratic approximation. However, this phenomena is not observed for 2000 points, so further analysis is needed. Figure 4.10 shows the mean reward evolution for 5 zones with 100 and 2000 belief-points. The results suggest that approximating the entropy-based reward is harder than approximating the other two, i.e., that more points are needed to correctly represent the reward shape. The sawtooth curve for the entropy-based reward when using 100 points is formed due to the suboptimal policy of interleaving clean and shoot actions without any move. In contrast, with 2000 points, all rewards behave similarly. The increase of total return with more points can be easily compared in this figure by considering the area below each curve.

Another interesting result highlighted in Table 4.2 is that the on-line computational effort needed for the myopic strategy is comparable with the information-lookahead methods with 100 belief-points. Moreover, the standard deviations are also comparable. This means that a myopic strategy is not necessarily faster than an information-lookahead strategy with the right parameters. However, please remember that information-lookahead strategies need off-line computations.

Comparing the results for 100 and 2000 points, one can see that the additional computational effort of increasing the number of belief-points is less significant than for diagnosis, and there is no clearly dominant reward type in terms of total return or computational effort. This raises the question of how the total return and the computational effort change with different belief-set sizes. Figure 4.11 shows both the total return (left) and on-line computational effort (right) using 3 and 5 zones with different belief-set sizes. The left side of the figure shows that the lookahead policies outperform the random strategy just with a small number of points (between 10 and 50). Also, the entropy-based reward produces poorer results than the linear or quadratic approximations with few points, but as more points are included, all of them converge to the same mean total return. The dimensionality of the problem plays an important role in this convergence: when more zones are added, the convergence is slower as the upper and lower figures show. The increase of the on-line time is plotted on the right side part of the figure, suggesting that, for 3 zones, increasing the number of points is only worth up to 1000 points, because afterwards the computational effort continues to grow (with a milder slope) without improving the total return. This is due to the fact that using more points generates more $\alpha$-vectors, needing more time to select the best one for a belief-state. However, the figure does not show the standard deviation, which decreases (very slowly) when more points are added, as can be seen in Table 4.2, meaning that the computed policy is more adaptable to anomalous behaviors. Therefore, adding more belief points is always a good idea if computation time is available.

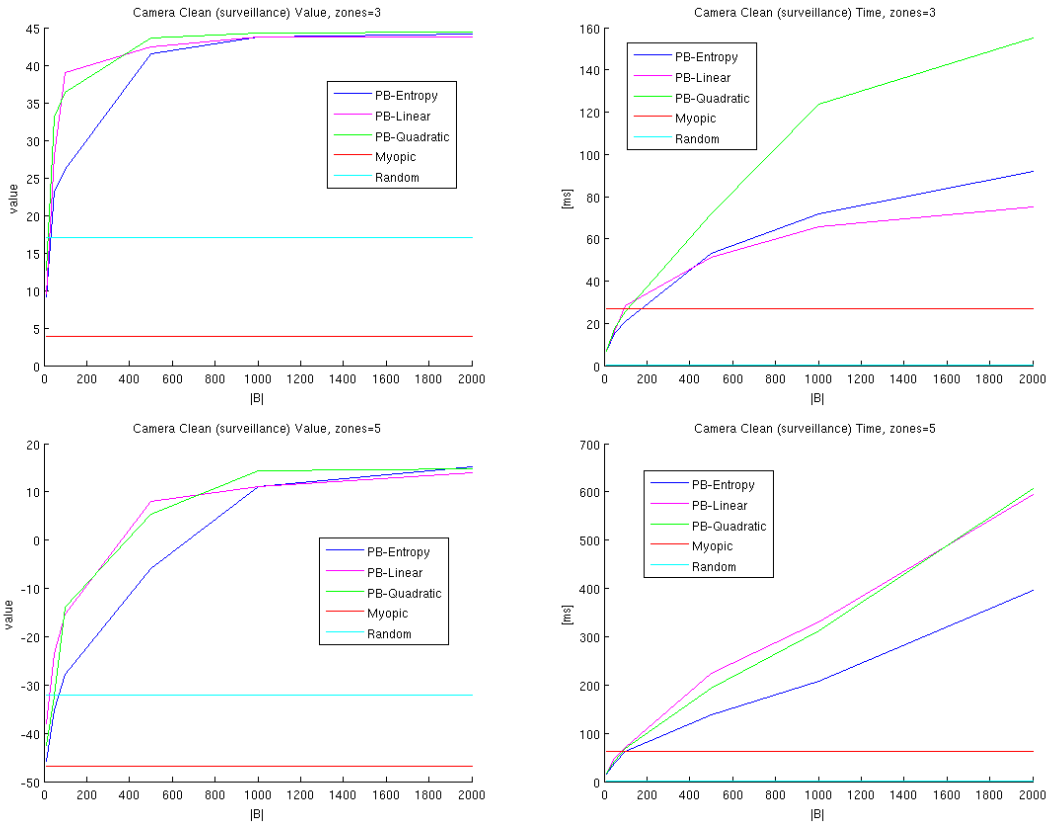



Figure 4.11: Camera-clean surveillance value and time performance. The left-hand results show the value performance depending on the number of belief-points (between 10 and 2000), and the right-hand ones show on-line time performance for the same belief-point scale. The figures at the top present the results for 3 zones and in the bottom the results for 5 zones.

| algorithm | zones | $\|\mathcal{B}\|$ | total return $[nats]$ | on-line time $[ms]$ | off-line time $[s]$ |
|---|---|---|---|---|---|
| **Random** | 3 | — | $1.03 \pm 0.17$ | $0.11 \pm 0.27$ | — |
| **Myopic** | 3 | — | $0.38 \pm 0.28$ | $26.10 \pm 12.46$ | — |
| **PB-Entropy** | 3 | 100 | $1.04 \pm 0.07$ | $\mathbf{24.93 \pm 5.60}$ | $\mathbf{2.93 \pm 0.47}$ |
| **PB-Quadratic** | 3 | 100 | $1.04 \pm 0.07$ | $27.66 \pm 9.31$ | $3.27 \pm 0.90$ |
| **PB-Linear** | 3 | 100 | $\mathbf{1.08 \pm 0.02}$ | $28.81 \pm 6.95$ | $4.22 \pm 1.00$ |
| **PB-Entropy** | 3 | 2000 | $\mathbf{1.10 \pm 0.01}$ | $130.30 \pm 37.19$ | $36.04 \pm 11.31$ |
| **PB-Quadratic** | 3 | 2000 | $\mathbf{1.08 \pm 0.01}$ | $188.47 \pm 57.59$ | $52.48 \pm 16.50$ |
| **PB-Linear** | 3 | 2000 | $\mathbf{1.09 \pm 0.01}$ | $94.12 \pm 22.10$ | $27.51 \pm 4.68$ |
| **Random** | 4 | — | $1.18 \pm 0.28$ | $0.16 \pm 0.40$ | — |
| **Myopic** | 4 | — | $0.34 \pm 0.35$ | $43.19 \pm 18.53$ | — |
| **PB-Entropy** | 4 | 100 | $1.18 \pm 0.07$ | $\mathbf{40.54 \pm 12.16}$ | $\mathbf{4.01 \pm 1.37}$ |
| **PB-Quadratic** | 4 | 100 | $\mathbf{1.36 \pm 0.04}$ | $44.03 \pm 16.01$ | $4.84 \pm 2.13$ |
| **PB-Linear** | 4 | 100 | $\mathbf{1.36 \pm 0.04}$ | $49.95 \pm 9.34$ | $6.33 \pm 1.29$ |
| **PB-Entropy** | 4 | 2000 | $\mathbf{1.39 \pm 0.01}$ | $298.89 \pm 69.72$ | $105.32 \pm 35.28$ |
| **PB-Quadratic** | 4 | 2000 | $\mathbf{1.37 \pm 0.04}$ | $386.34 \pm 61.62$ | $152.15 \pm 43.05$ |
| **PB-Linear** | 4 | 2000 | $\mathbf{1.35 \pm 0.04}$ | $491.44 \pm 104.28$ | $263.69 \pm 99.92$ |
| **Random** | 5 | — | $1.22 \pm 0.37$ | $0.21 \pm 0.45$ | — |
| **Myopic** | 5 | — | $0.31 \pm 0.39$ | $66.05 \pm 17.31$ | — |
| **PB-Entropy** | 5 | 100 | $\mathbf{1.58 \pm 0.09}$ | $\mathbf{56.02 \pm 10.33}$ | $\mathbf{3.81 \pm 1.19}$ |
| **PB-Quadratic** | 5 | 100 | $1.52 \pm 0.14$ | $75.65 \pm 28.08$ | $5.55 \pm 3.00$ |
| **PB-Linear** | 5 | 100 | $1.48 \pm 0.09$ | $76.18 \pm 11.71$ | $7.68 \pm 2.73$ |
| **PB-Entropy** | 5 | 2000 | $\mathbf{1.61 \pm 0.05}$ | $692.41 \pm 137.69$ | $235.42 \pm 79.75$ |
| **PB-Quadratic** | 5 | 2000 | $\mathbf{1.59 \pm 0.06}$ | $837.67 \pm 232.13$ | $400.98 \pm 223.64$ |
| **PB-Linear** | 5 | 2000 | $\mathbf{1.59 \pm 0.05}$ | $884.91 \pm 186.69$ | $654.04 \pm 282.80$ |

Table 4.3: Camera-clean Localization Results. Mean total return, mean on-line time and mean off-line time over 10 repetitions of 500 trajectories of 100 steps. The bold and red values stand for the bests and worst results respectively.

Unfortunately, the problem of off-line computational effort is that it cannot be predicted, because it depends on the random belief-point collection. Indeed, Figure 4.11 shows that for 5 zones the PB-Entropy is the fastest reward type, but for 3 zones it is not the case anymore. Also, Table 4.2 shows that for 2000 points, the PB-linear strategy has the worst off-line time for 4 and 5 zones in contrast to the results for 3 zones. Moreover, Table 4.2 shows that the standard deviations of all the PB on-line and off-line times are very high, implying that the computation time varies significantly from one repetition to the other.

### Localization

For the camera-clean localization problem, again the transition function modifies deterministically the *cam* and *len* variables, and the *obj* variable is visible and remains fixed through all the problem. Here, the hidden variable is the *cam* variable, so the agent can only infer its position through the indirect and stochastic observations obtained by shooting photos in the current unknown position. Please note that even though this is a variation of the camera-clean setup, the problem is completely different. Now the agent must modify the target variable to obtain more information depending on the observations.

Please recall that the CC-diagnosis problem was a finite horizon problem, and the CC-surveillance an infinite-horizon one. For completeness, consider now that for this problem an indefinite horizon is chosen, meaning that the objective is to obtain the maximum information at the end of the process (final information), but this horizon is not pre-defined. Finding a policy for this problem is equivalent to using a sum-of-information criterion as presented in Section 4.3.3. The probability of continuing was set to $\gamma = 0.95$, which in the end corresponds to the same criterion used for surveillance. However, for evaluating the policy, only the final information is considered because this is the real objective of the problem.

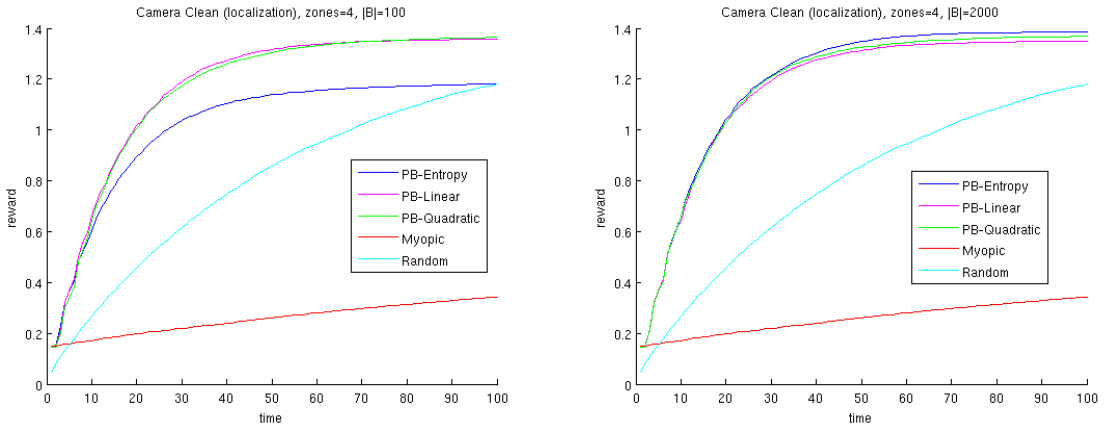



Figure 4.12: Camera-clean localization total return evolution for 4 zones over the first 100 steps. The left-hand figure shows the results for $|\mathcal{B}| = 100$ and the right-hand one shows the results for $|\mathcal{B}| = 2000$.

All experiments were run for an horizon of $H = 100$ steps and for 3, 4 and 5 zones as shown in Table 4.3. All the other parameters (trials, repetitions, etc.) were set as for the previous problems.

The myopic strategy is again outperformed by all the other methods, but now the random strategy shows results competitive with the information-lookahead approaches with a moderate quantity of points (100) as highlighted (light blue) in the table. However, the standard deviation shows that the variability of the information-lookahead policies is much lower than random, and this competitiveness decreases as the number of zones increases.

The relatively small improvement when increasing from 100 to 2000 points is due to the fact that the results for 100 points are already near the perfect information, leading to competitive values as shown in Table 4.3. Indeed, all the results in bold print are between 97.3% and 99.7% of the perfect information, meaning that not much more information can be acquired. However, the entropy reward constantly delivers the best and less variable results as highlighted (yellow) in the table.

In terms of computational effort, the myopic strategy also compares to the lookahead approaches with few points, and even though incrementing the size of the belief-set does not improve too much the total return, the computational effort does increase at similar rates as in the previous problems.

Table 4.3 only offers the results for an horizon of 100, but in an indefinite horizon problem it is interesting to analyze several horizons so as to understand how the algorithms behave. Figure 4.12 shows the total returns at different horizons (from 1 to 100) using 4 zones and belief sets of both 100 and 2000 points. The figure shows that information-lookahead techniques rapidly gain information compared to the random strategy, so the total return is higher for all horizons. With 100 points, the entropy-based reward equals the random strategy when $H = 100$, but for shorter horizons the lookahead techniques outperforms random.

Figure 4.12 also shows a relatively poor behavior of the entropy-based reward compared to the linear and the quadratic rewards when using small belief-sets, similarly to the results for the diagnosis problem. This is counter-intuitive result, because it is natural to expect always the best performance for the entropy-based reward as the other two are approximations of this same function. Yet, this phenomenon can be explained due to the highly non-linear shape of the entropy-based reward, which cannot be correctly approximated with only few points. This is confirmed by the right figure where the best result is obtained for this same reward function with 2000 points.

### Discussion

The camera-clean problem is one case where the myopic strategy fails to provide good results, yet for other problems it can be a good heuristic. Considering that the myopic strategy is a one-step information-lookahead strategy (for instance with $\gamma = 0$), the only reason to use this kind of heuristic is because of

computation time constraints. Yet, this problem shows that if only few points are used for approximating the value function, the on-line computational effort is comparable, and the total return is significantly higher for lookahead approaches.

The reasonably fair performance of the random strategy in some variations—such as in CC-localization—can be explained by the low dimensionality of the belief-space and the simple structure of the problem. The intuition behind the success of a random strategy under weakly structured problems is that randomly sampling actions generates different short-horizon policies, and some of them are efficient for information-gathering. In problems where information is easily lost (such as in CC-surveillance), random policies are not suitable at all. Moreover, the results show that, as the dimensionality increases, the gap between random and lookahead approaches also increases, showing that this strategy does not scale at all in terms of total return.

For all the three variations of the problem, the behavior of information-gathering techniques is similar but not identical. The "best" reward-type is not the same not only from variation to variation, but also when different belief-set sizes are selected. In general, the results suggest that for this problem choosing any of the information-based rewards is suitable.

Despite this, there is a tendency of the entropy-based reward to be less efficient if only 100 belief-points are available, yet it delivers the best results with 2000 points because the agent can achieve near-perfect information with those points. Also, a small tendency of the linear reward to be inefficient when using 2000 points can be identified. These results vaguely suggest that the shape of the reward function begins to play a significant role only when the solution is a near perfect information gathering policy.

In summary, these results show that the information-lookahead techniques can be successfully applied to the three classes of information-gathering problems, and that they outperform naive methods such as myopic and random strategies for this specific kind of problems. Unfortunately, only weak arguments can be wielded in favor of or against the selected reward-types. Hence, further experimentation on other problems is needed.

### 4.5.3 The Rock Diagnosis Problem

The rock diagnosis problem is an information-gathering variation of the *rock sampling* problem proposed in (Smith and Simmons, 2004). In the original problem a grid-map of rock positions is given to a rover, whose objective is to perform sampling procedures (that destroy the rocks) to those rocks that have a "good" scientific value while traversing the map. If the rover samples a "good" rock it receives a positive reward. In the contrary, it receives a negative reward for sampling "bad" rocks because the sampling procedure is expensive. Obviously, the types of the rocks are not given to the rover, but the rover is equipped with a noisy long-range sensor to query them. The efficiency of the long-range sensor (i.e., the probability of correctly identifying the rock type) decreases exponentially with the Euclidean distance to the rock.

The rock diagnosis variation (see Figure 4.13) consists in removing the sampling action from the rover, meaning that the rover can only move through the grid and use its long-range sensor to query rocks. The information-gathering objective is to reduce the uncertainty about the rocks nature, whatever their type. This information may be used later by a human to analyze the distribution of rocks, or to perform any other procedure that needs a highly-confident knowledge about the nature of the rocks. This same kind of setup can be used for industrial applications such as searching for oil wells, detecting plagues in crop fields or analyzing geologic veins for mining.

For the rock diagnosis problem, this section considers only square grids of side $l$ and a number of rocks $n$. Then, the state of the system can be described by the pair $s = \langle loc, rtype \rangle$, where $loc$ is the localization of the rover in the grid, and $rtype = \langle r_1, r_2, \ldots, r_n \rangle$ contains the type of each rock in the map, where $r_i \in \{good, bad\}$, forming a state space of size $|\mathcal{S}| = l^2 \times 2^n$. The action space corresponds to $\mathcal{A} = \{north, east, south, west, check_1, check_2, \ldots, check_n\}$, where the first 4 actions are the cardinal moves, and a $check_i$ action uses the long-range sensor to query the rock $i$.

The transition function is completely deterministic: under the cardinal moves the rover deterministically changes position according to the selected direction and the grid boundaries, and $check_i$ actions do not modify the state in any way. On the contrary, the observation function is stochastic, because observations depend on the noisy long-range sensor. The observation space is $\mathcal{Z} = \{none, good, bad\}$,

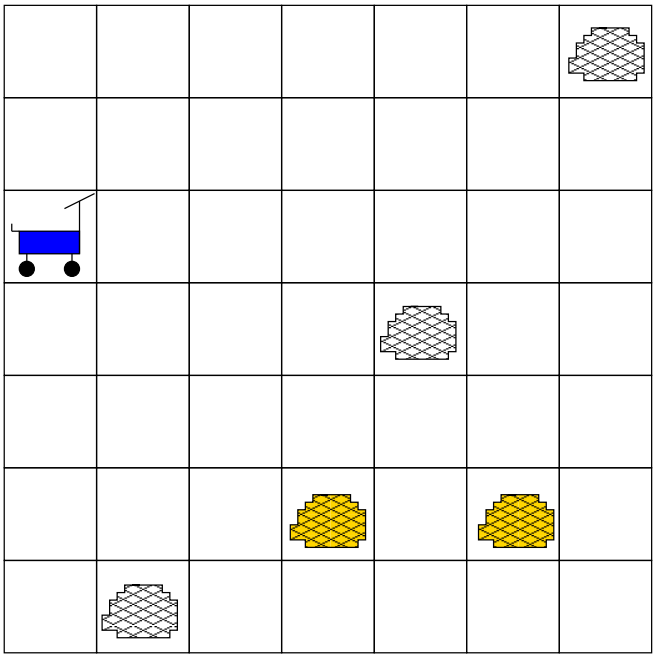


Figure 4.13: Graphical representation of the Rock Diagnosis problem with 5 rocks and a grid side of 7. The yellow rocks are the "good" rocks (with scientific value) and the others are the "bad" rocks (no scientific value), yet each rock type is unknown to the agent. Please note that, for diagnosis, a "good" rock is not more important than a "bad" one, because the objective is to disambiguate all of them, and not sampling only "good" ones like in the original Rock Sampling problem.

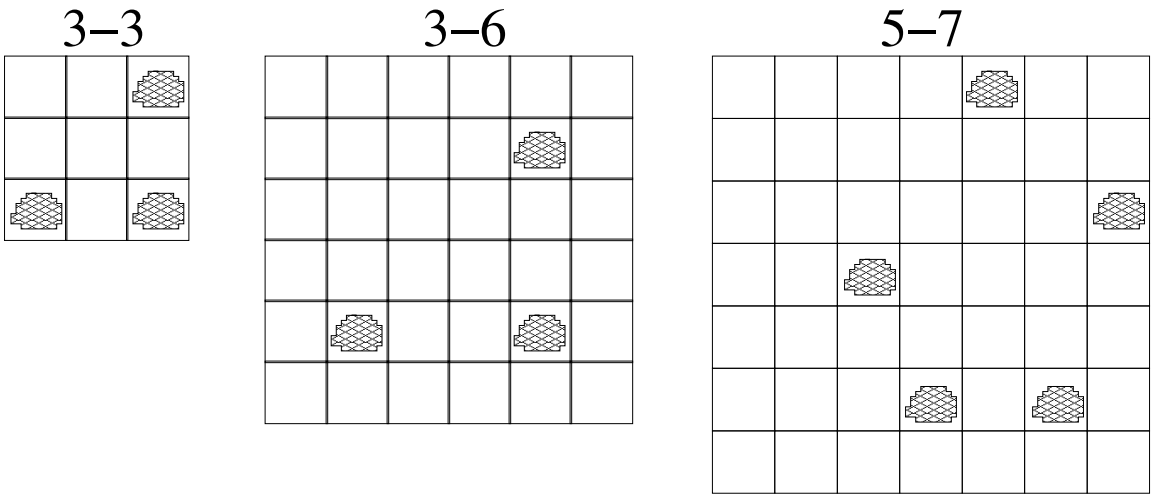



Figure 4.14: The three specific Rock Diagnosis maps used in this section. The numbers above the figures are in the form $n$-$l$, where $n$ is the number of rocks and $l$ is the side of the grid.

where *none* is obtained if and only if the agent executes a cardinal move action, and *good* or *bad* are obtained depending on the queried rock type and the distance to the target rock.

Let $r_i$ be the type and $p_i$ be the position of rock $i$ in the grid, then the efficiency of the long-range sensor is

$$efficiency = Pr(Z = r_i | A = check_i, S = \langle loc, rtype \rangle) = \frac{1 + e^{-\|loc - p_i\|_2}}{2}.$$

As the *none* observation can only be obtained by cardinal move actions, the probability of obtaining an incorrect observation by the long-range sensor is $1 - efficiency$. Please note that the $p_i$ positions are given to the agent, so they can be implicitly encoded in the observation function.

Figure 4.14 shows the specific map configurations used in this section. These configurations will be identified by the numbers above each map, which are in the form $n$-$l$ ($n$ is the number of rocks and $l$ is the side of the grid). For each trajectory, the type of each rock is sampled from the initial belief, which corresponds to the uniform distribution for this section's experiments.

**Results**

The performance criterion used for this problem is the final-information criterion under an indefinite horizon. Therefore, the information-lookahead strategies use the equivalent sum-of-information criterion with $\gamma = 0.95$ and $\epsilon = 0.1$. The target variable is *rtype*, and its maximum final reward is $n \log(2)$. The horizon of the trajectories was fixed to $H = 100$, yet results for shorter horizons can be obtained by truncating the simulation output like in the CC-localization problem. Experiments were made for all maps (i.e., 3-3, 3-6 and 5-7) as shown in Table 4.4. Each experiment was repeated 10 times, each generated policy was evaluated on-line using 100 trajectories, and several sizes for the belief-point set were tested, yet in Table 4.4 the results are shown only for 1000 and 5000 points.

The rock diagnosis problem was selected because it also needs smart exploration, where a suitable path of uninformative actions (i.e., cardinal moves) leads to highly-informative observations in the future. It can be observed in Table 4.4 that again the myopic strategy fails to provide competitive results for all maps. Unsurprisingly, information-lookahead strategies with 5000 points provide in general the best results. However, the random strategy is competitive with information-lookahead ones, usually outperforming the results with 1000 points and sometimes even with 5000 points as it is highlighted in the table. For example, for the small map (i.e., 3-3) the random strategy performs as well as the best PB result with 1000 points, and is rather near to the two worst results with 5000 points. The map 3-6 has the same number of rocks as the small one, but as the grid is larger, a much longer-term planning

| algorithm | rocks-side | $\|\mathcal{B}\|$ | total return $[nats]$ | on-line time $[ms]$ | off-line time $[s]$ |
|---|---|---|---|---|---|
| **Random** | 3-3 | — | $1.55 \pm 0.40$ | $0.82 \pm 2.24$ | — |
| **Myopic** | 3-3 | — | $0.69 \pm 0.30$ | $\mathbf{129.16 \pm 23.60}$ | — |
| **PB-Entropy** | 3-3 | 1000 | $1.17 \pm 0.27$ | $157.80 \pm 53.28$ | $\mathbf{20.69 \pm 8.06}$ |
| **PB-Quadratic** | 3-3 | 1000 | $1.39 \pm 0.29$ | $295.80 \pm 147.07$ | $65.75 \pm 61.27$ |
| **PB-Linear** | 3-3 | 1000 | $1.55 \pm 0.43$ | $450.38 \pm 70.21$ | $579.53 \pm 192.38$ |
| **PB-Entropy** | 3-3 | 5000 | $1.58 \pm 0.25$ | $675.44 \pm 154.21$ | $315.34 \pm 180.10$ |
| **PB-Quadratic** | 3-3 | 5000 | $1.58 \pm 0.24$ | $1453.35 \pm 503.57$ | $1105.48 \pm 521.32$ |
| **PB-Linear** | 3-3 | 5000 | $\mathbf{2.06 \pm 0.03}$ | $957.26 \pm 221.96$ | $10076.82 \pm 4445.40$ |
| **Random** | 3-6 | — | $0.59 \pm 0.41$ | $0.14 \pm 0.32$ | — |
| **Myopic** | 3-6 | — | $0.01 \pm 0.01$ | $434.38 \pm 30.68$ | — |
| **PB-Entropy** | 3-6 | 1000 | $0.30 \pm 0.03$ | $\mathbf{116.29 \pm 26.86}$ | $19.79 \pm 5.49$ |
| **PB-Quadratic** | 3-6 | 1000 | $0.50 \pm 0.01$ | $\mathbf{131.69 \pm 34.44}$ | $\mathbf{12.34 \pm 5.61}$ |
| **PB-Linear** | 3-6 | 1000 | $0.69 \pm 0.02$ | $403.48 \pm 167.61$ | $56.68 \pm 42.71$ |
| **PB-Entropy** | 3-6 | 5000 | $\mathbf{0.76 \pm 0.09}$ | $211.70 \pm 40.73$ | $81.83 \pm 13.39$ |
| **PB-Quadratic** | 3-6 | 5000 | $\mathbf{0.74 \pm 0.06}$ | $279.04 \pm 77.80$ | $62.06 \pm 14.66$ |
| **PB-Linear** | 3-6 | 5000 | $\mathbf{0.79 \pm 0.08}$ | $2749.77 \pm 1259.17$ | $5178.52 \pm 4463.55$ |
| **Random** | 5-7 | — | $\mathbf{0.50 \pm 0.42}$ | $0.16 \pm 0.36$ | — |
| **Myopic** | 5-7 | — | $0.03 \pm 0.04$ | $2869.11 \pm 206.44$ | — |
| **PB-Entropy** | 5-7 | 1000 | $0.08 \pm 0.02$ | $\mathbf{424.99 \pm 80.04}$ | $59.40 \pm 13.16$ |
| **PB-Quadratic** | 5-7 | 1000 | $0.11 \pm 0.03$ | $\mathbf{438.97 \pm 75.57}$ | $\mathbf{15.08 \pm 2.96}$ |
| **PB-Linear** | 5-7 | 1000 | $0.23 \pm 0.06$ | $\mathbf{495.03 \pm 104.83}$ | $25.01 \pm 8.53$ |
| **PB-Entropy** | 5-7 | 5000 | $0.37 \pm 0.09$ | $800.96 \pm 247.22$ | $281.08 \pm 75.29$ |
| **PB-Quadratic** | 5-7 | 5000 | $0.12 \pm 0.03$ | $625.34 \pm 117.43$ | $35.61 \pm 3.39$ |
| **PB-Linear** | 5-7 | 5000 | $\mathbf{0.53 \pm 0.03}$ | $846.30 \pm 175.76$ | $67.60 \pm 17.41$ |

Table 4.4: Rock Diagnosis Results. Mean total return, mean on-line time and mean off-line time over 10 repetitions of 100 trajectories of 100 steps. The bold and red values stand for the best and worst results respectively.

is needed to obtain highly-informative observations. For this map, information-lookahead strategies with 5000 points outperforms the random one because it is much more difficult for a random exploration to find the correct paths. But, for the same reason, 1000 points are not enough for surpassing random. In the 5-7 problem, the random strategy outperforms all the information-lookahead ones except for the PB-Linear strategy with 5000 points. Please note that even though random seems to be near this last one, the standard deviation shows that the variability of the random strategy is very high with respect to the value, so very poor performance can be sometimes expected.

Discerning which reward function is better for this problem is much more clear than for the Camera Clean problems. Here, the quadratic reward gives usually lower results than the other two, and the linear reward provides very good results. Probably, the entropy-based reward might exceed the linear approximation if a considerable number of points is used, but the overwhelming amount of time needed for such experiment is prohibitive.

The poor results in the last map can be explained by the curse of dimensionality: with the same amounts of points (1000 or 5000) a higher dimensional belief-space is sampled, so a much worse approximation is obtained [16]. Indeed, the linear reward is an easier function to approximate than the other two, and therefore it provides the best results. This can be seen not only in the 5-7 map, but also in the two smaller maps. In the 3-6 map the linear reward provides a slightly better result than the others, and in the 3-3 map it achieves to gather 99% of the perfect information.

The problem with increasing the number of points to achieve better performance is always related to the computational effort. In Table 4.4 and in all the previous problems, increasing the amount of points always increases the mean value but it also always increases the computational effort required. As stated before, the off-line time cannot be predicted, and its variability can be enormous. For example, consider the worst off-line time of Table 4.4, which corresponds to PB-Linear with 1000 points for the 3-3 map. The mean is near 3 hours, but its standard deviation exceeds 1 hour, meaning that some repetitions can take several hours or only a few minutes. Even in the best off-line time scenario, which is the PB-Quadratic strategy with 1000 points for the 3-6 problem, the standard deviation is more than a quarter of the mean value. However, as this depends on the belief-point collection, the off-line time might be more stable if a smarter selection of points is used.

It is important to notice that, for information-lookahead strategies, good performances are correlated with high computational effort in Table 4.4. This can be explained due to the several informative actions available at each step, generating several vectors that are not easily dominated in the whole belief-space. If a dense approximation is used, these vectors will be propagated to produce check and move actions when suitable. If only a sparse approximation is available, these vectors will be easily dominated by a small set of vectors, generating useless policies like checking always the same rock, or moving towards a rock but not checking it. These poor policies can be obtained very fast, explaining the counter intuitive results that the computational effort of higher-dimensional maps is less important than for lower-dimensional ones.

Fortunately, for this same reason it is much more affordable to increment the numbers of points for the 5-7 map than for the 3-3. This is shown in Figure 4.15 where the reward evolution for 5000 and 20000 points is plotted. Notice that even though the final reward ($H = 100$) of the random strategy is near the values of information-lookahead strategies for 5000 points, this is not true for shorter horizons. Indeed, information-lookahead strategies gather almost all the information in the first 10 steps, yet later they acquire information no faster than the myopic strategy. As expected, there is an improvement by going from 5000 to 20000 points. However, 20000 points are sill not enough to densely sample the belief-space for this random collection strategy, meaning that the best corresponding policies are still far away from the optimal one.

Figure 4.16 shows at which rate the total return and the computational effort increase with more points for the 5-7 map. The time results show that the myopic strategy is slower than the information-lookahead ones for all but the linear reward with 20000 points, but this phenomenon occurs only due to the sparse approximation of the value function as the performance results show. These results confirm that myopic strategies may not only produce very poor results, but also may be significantly slower too [17].

16. An ad-hoc amount of points for each map will imply an exponential growth of them with the dimensionality
17. This is only true for the worst case scenario where no analytical expression can be used to speed up the computations.

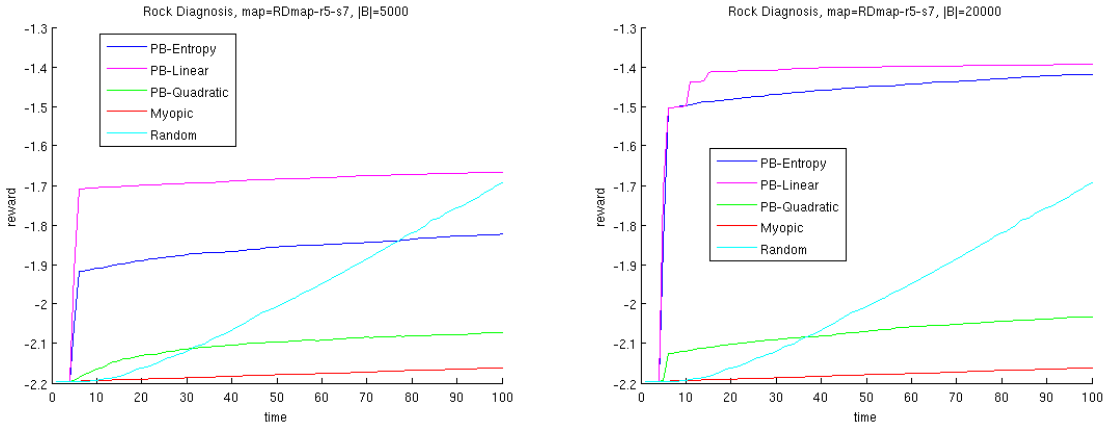



Figure 4.15: Rock Diagnosis reward evolution for the 5-7 map over the first 100 steps. The left-hand figure shows the results for $|\mathcal{B}| = 5000$ and the right-hand one shows the results for $|\mathcal{B}| = 20000$.

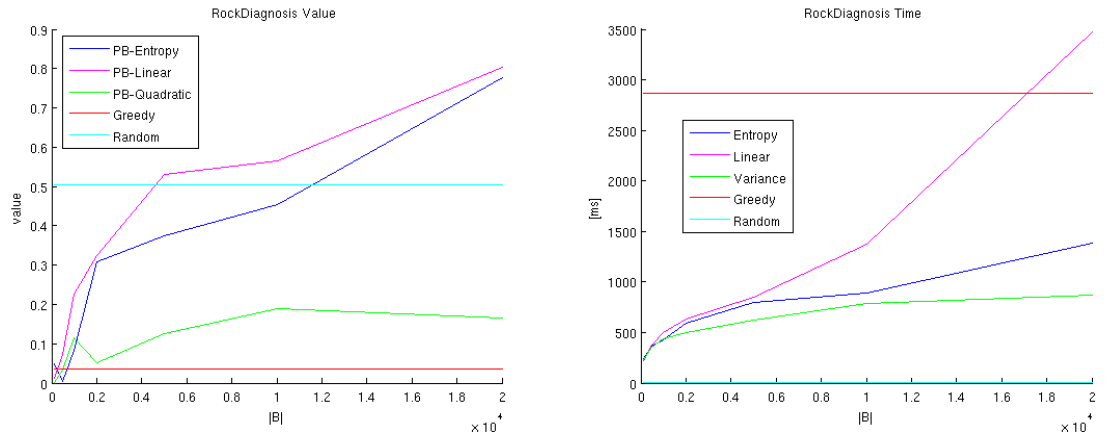



Figure 4.16: Rock Diagnosis value and time performance for the 5-7 map. The left-hand figure shows the value performance depending on the number of belief-points (between 100 and 20000), and the right-hand one shows on-line time performance for the same belief-point scale.

| algorithm | zones | $\|\mathcal{B}\|$ | total return $[nats]$ | on-line time $[ms]$ | off-line time $[s]$ |
|---|---|---|---|---|---|
| **Random** | 3-3 | — | $2.52 \pm 0.48$ | $0.07 \pm 0.18$ | — |
| **Myopic** | 3-3 | — | $1.90 \pm 0.27$ | $\mathbf{63.91 \pm 11.50}$ | — |
| **PB-Entropy** | 3-3 | 5000 | $3.09 \pm 0.36$ | $172.82 \pm 32.43$ | $105.18 \pm 34.41$ |
| **PB-Quadratic** | 3-3 | 5000 | $3.00 \pm 0.31$ | $147.93 \pm 37.54$ | $\mathbf{79.04 \pm 45.02}$ |
| **PB-Linear** | 3-3 | 5000 | $\mathbf{3.52 \pm 0.31}$ | $441.05 \pm 138.16$ | $3531.32 \pm 1794.75$ |
| **Random** | 3-6 | — | $1.75 \pm 0.30$ | $0.44 \pm 1.28$ | — |
| **Myopic** | 3-6 | — | $1.51 \pm 0.02$ | $222.87 \pm 11.75$ | — |
| **PB-Entropy** | 3-6 | 5000 | $2.14 \pm 0.16$ | $99.36 \pm 17.57$ | $74.27 \pm 11.31$ |
| **PB-Quadratic** | 3-6 | 5000 | $1.87 \pm 0.11$ | $\mathbf{79.08 \pm 47.60}$ | $\mathbf{28.71 \pm 18.69}$ |
| **PB-Linear** | 3-6 | 5000 | $\mathbf{2.22 \pm 0.26}$ | $355.12 \pm 429.71$ | $413.95 \pm 901.21$ |

Table 4.5: Cost-Sensitive Rock Diagnosis Results. Mean total return, mean on-line time and mean off-line time over 10 repetitions of 100 trajectories of 50 steps. The bold and red values stand for the bests and worst results respectively.

As expected, Figure 4.16 shows that the apparent good performance of the random strategy for the 5-7 map in Table 4.4 was only due to the relatively poor lookahead policies obtained with 5000 points. Indeed, a fast inspection of the trajectories generated by these policies confirms that they efficiently gather information for a first rock, but then they fail to examine the rest. Increasing the number of points obviously needs more computational effort, but for example the entropy reward with 20000 points outperforms random, and is faster than myopic. On the other hand, the quadratic reward fails to increase its performance with more points, and indeed this performance decreases with 20000 points. This result is unlikely explained by the variability of a random collection of points, because through all the set sizes the quadratic reward fails to provide a competitive results.

### Cost-sensitive Rock Diagnosis

In Section 4.3.2, the notion of combined rewards was introduced, where the idea was to bias the information-based rewards to some preferred values. The difference with state-based rewards is that ignorance is still the worst valued point in the simplex, so even though some pieces of information are more valuable than others, the objective is still to gather information.

Here, the early results of a cost-sensitive version of rock diagnosis are presented as a *proof of concept*, showing that these problems can also be solved using the results of this chapter. In the original Rock Sample problem a "good" rock means that the rock is scientifically valuable, and a "bad" rock means that it is not interesting. The cost-sensitive rock diagnosis problem consists in using this notion to assign weights to "good" and "bad" rocks, biasing the information-gathering towards the detection of scientifically valuable rocks.

Formally, a state-based reward function $R(good, a) = 2$ and $R(bad, a) = 1$ for all $a \in \mathcal{A}$ is combined with an information-based reward. An arbitrary $\eta = 1.0$ was selected to combine the state-based reward (see Equation 4.2) with the three information-based rewards $\rho_H$, $\rho_Q$ and $\rho_L$. The Myopic and Random strategies were also executed to compare them. Table 4.5 shows the results for the 3-3 and 3-6 maps for 100 trajectories with $H = 50$ under 10 repetitions. The results show that the linear reward overcomes all the others, but it is also the most time-consuming for both on-line and off-line time. The myopic strategy is again outperformed by all others, the quadratic reward corresponds to the worst information-lookahead strategy for all experiments, and the random strategy is always in between these two. These results show that the proposed technique adapts successfully to cost-sensitive problems, but a more detailed experimentation is needed to understand the impact of the $\eta$ parameter and the state-based cost/reward function.

In practice, ad-hoc myopic strategies will be significantly faster than lookahead ones.

## 4.6 Conclusions

This chapter paves the way for solving information-gathering problems using algorithms for approximating the value function (e.g., point-based ones) in a theoretically sound manner. Empirical results are also provided, showing the benefits and drawbacks of the proposal.

### 4.6.1 Summary of Contributions

An extension of POMDP called $\rho$POMDP was introduced, allowing to express information-gathering decision-making problems using information-based rewards. In this model, the reward $\rho$ is typically a convex function of the belief state.

Using the convexity of $\rho$, a first important result is that the Bellman backup $V_t^* = \mathbb{B}V_{t-1}^*$ preserves convexity. In particular, if $\rho$ is PWLC and the value function $V_0$ is equal to 0, then $V_t^*$ is also PWLC and it is straightforward to adapt many state-of-the-art POMDP algorithms. Yet, if $\rho$ is not PWLC, performing exact updates is much more complex. It is therefore proposed to employ PWLC approximations of the convex information-based reward function at hand to come back to a simple case, showing that the resulting algorithms converge to the optimal value function in the limit.

An important point is that the time complexity of the new algorithms only changes due to the size of the approximation of $\rho$ in the exact value iteration case. PB algorithms on the other side, remain almost the same in time complexity if the gradient of the reward function is known.

In terms of empirical results, three different information-based rewards were tested, all of them outperforming the myopic and random approaches in a number of domains. Even though the myopic approach seems a very naive approach, it is the most used technique in several communities that have addressed this problem. Regarding the performance comparison between the three information-based rewards, it seems that the entropy-based one is suitable if dense approximations can be achieved, and the linear one is more efficient if only sparse approximations are available. In general, the quadratic reward, like the one presented in (Krishnamurthy, 2002), does not fail to provide competitive results, but is almost always surpassed by one of the other two. Moreover, for the largest problem tested, it completely fails to find good policies.

Even though this dissertation focuses on *pure* information-gathering problems to stress out the importance of information-based rewards, real-world problems are often cost-sensitive. Consequently a *proof of concept* experiment was conducted for cost-sensitive information-gathering problems showing promising results.

Most of the theoretical results were already published in (Araya-López et al., 2010a) and (Araya-López et al., 2011b), yet some of them were extended and polished for this dissertation. Moreover, the empirical results and analysis are new contributions of this chapter. Other authors are beginning to use $\rho$POMDPs (such as Eck and Soh (2012)), which confirms the interest of the results of this chapter.

### 4.6.2 Future Work

The performance of PB algorithms is highly dependent on the belief-point collection. Indeed, this section showed that the belief-set size plays an important role on the performance of information-lookahead strategies. Therefore, an obvious extension to this work is to use more elaborated collection methods than the static randomized belief-set, for example dynamic set sizes and an error-based collection (see Section 3.2.6). This may lead to better approximations with less points, exploring more relevant parts of the belief-space and avoiding unrealistic or insignificant points in the set. An interesting follow up is also to consider an information-driven collection strategy, which collects belief-points depending on the information-gain rather than the error. However, such a strategy could lead to an information-greedy behavior, which is not suitable for the selected problems. Nonetheless, a smart collection strategy specifically designed for information-gathering problems may be wanted to improve either the performance or the computational time.

In the same vein, further research is needed on the counter-intuitive result that a harsh linear approximation offers a better performance than the entropy-based reward when only a small number of belief-points is available. Understanding why this phenomenon occurs may help to build faster and simpler

algorithms to approximately solve sequential information-gathering problems, and to better understand the relationship between these problems and active classification problems (see Section 4.3.6).

A major issue in POMDP solvers is scalability, because the computational time usually grows exponentially with the dimensionality. This is even worse for information-gathering problems, because most of the computational effort is consumed on the $\alpha$-vector propagation, and information-gathering problems have reward functions composed by several $\alpha$-vectors. Fortunately, real-world problems are usually highly structured, meaning that the transition and observation functions can often be factored to exploit this structure. This allows solving problems with less computational effort than when using a plain representation (Poupart, 2005). In particular, for most of the interesting information-gathering problems, the state can be factored in a *visible* and a *hidden* part, which leads to use the Mixed Observable MDP formalism (Araya-López et al., 2010bc; Ong et al., 2009). This simple distinction between visible and hidden variables may dramatically reduce the computational effort for solving information-gathering problems.

Regarding the cost-sensitive results, it is clear that more experiments are needed. In particular, it might be insightful to analyze the impact of the $\eta$ parameter, because it forms a continuous spectrum between information-gathering (low $\eta$ values) and classic state-based POMDPs (high $\eta$ values). Also, a deeper study of the theoretical and practical properties of the combination with different state-based reward functions is needed.

# Chapter 5

# Reinforcement Learning

## Contents



*"It is what we think we know already that often prevents us from learning"*
— Claude Bernard.

The problem of sequential information gathering is not exclusively bound to the POMDP framework (see Section 1.2), because any sequential problem with some sort of uncertainty will require actively exploring the unknown. Therefore, this dissertation is also concerned with another type of sequential decision problem under uncertainty that has been extensively studied in the past three decades, namely *Reinforcement Learning* (RL) problems. This chapter introduces the basic RL concepts, and a few simple algorithms that use different information gathering strategies (which in the RL domain are called *exploration strategies*), to put into perspective the contributions of Chapters 6 and 7.

While the POMDP framework deals with problems where the state of the system is unknown, RL deals with problems where the *model* of the system is unknown. RL can be broadly summarized as "a computational approach to learning from interaction" (Sutton and Barto, 1998), where the agent must sequentially interact within a (partially) unknown environment in order to maximize the sum of (potentially unknown) scalar rewards.

Under the Markov assumption, this problem can be modelled using the MDP framework of Section 3.1, yet weakening the assumption that the transition and/or reward functions are known. Even though the model is unknown to the agent, it is assumed that the state and action spaces are known, and that there is only one "true" MDP model that fits the environment. Moreover, it is usually assumed that this model is stationary for simplicity in infinite-horizon settings.

Let $M = \langle \mathcal{S}, \mathcal{A}, T, r, s_0 \rangle$ be an MDP, where $\mathcal{S}$ and $\mathcal{A}$ are known state and action spaces, $s_0$ is a known initial state, but where the transition function $T$, and sometimes the reward function $r$, are unknown.

However, after each performed action, the agent can observe the current state (and sometimes the current reward) of the system. The *reinforcement learning* problem consists in finding an algorithm (or policy) that maximizes the expected return, knowing only the history so far of actions, states and rewards (partial trajectory).

Please note that in the above definition, a solution of a RL problem is a policy that is not necessarily stationary. Moreover, for considering the information gathered so far, a policy may be a mapping of the whole history of past states, actions and rewards to actions.

## 5.1 Classical Approaches

Most of the classical RL techniques are based on the idea that an optimal solution for an RL problem is an optimal policy of the real underlying MDP when the model is given (Szepesvári, 2010).

**Definition 5.1.1.** *RL Optimal Policy Let* $M = \langle \mathcal{S}, \mathcal{A}, T, r, s_0 \rangle$ *be the true underlying MDP, and*

$$V_M^\pi = \lim_{n\to\infty} E_{S_1:S_n}\left[\sum_{t=1}^{n} \gamma^{t-1} r(S_{t-1}, \pi(A_t), S_t) \middle| S_0 = s_0, \pi, T, r\right]$$

*be the value function of a policy* $\pi$ *using that model, then an optimal policy verifies*

$$\pi^*(s) \in \operatorname*{argmax}_\pi V_M^\pi(s), \forall s \in \mathcal{S}.$$

As the underlying MDP model is unknown, RL algorithms aim to learn a policy from the partial trajectories so far, and be as close as possible to the optimal policy. There are many approaches to learn this policy (Sutton and Barto, 1998), each with its own advantages and biases, but in general the idea is to converge to a near-optimal policy as fast as possible, and to generalize well to several problems with hopefully few free parameters.

Regardless the success and efficiency of some RL algorithms, under this optimality definition it is theoretically impossible to construct an always optimal algorithm, because information about the model is only available during the same decision process, producing non-optimal steps due to the unavoidable inaccurate estimates. Therefore, the formal objective of these algorithms is often to minimize the *regret* of following suboptimal policies with respect to an optimal one (Jaksch et al., 2010), or to minimize the amount of non near-optimal steps (see Section 5.3).

### 5.1.1 Exploration/Exploitation Dilemma

Most of the classical algorithms rely on the fact that, if a good estimate of the optimal policy is obtained after some steps, a near-optimal behavior can be achieved afterwards. Moreover, if this good estimate can be obtained soon enough through a smart exploration of the dynamics of the system, the algorithm may achieve a relatively high performance (Strehl et al., 2009). Unfortunately, actions that produce these accurate estimates may not be the best actions to optimize the desired performance criterion. On the other side, accurate estimation clearly helps to achieve a better performance in the future.

Combining these two contradictory principles has proven to be difficult, because each problem needs a different compromise between estimation and return optimization. This dichotomy is called the *exploration/exploitation dilemma* (Sutton and Barto, 1998; Thrun, 1992), because the algorithm must choose between exploration—performing actions that favour learning—and exploitation—performing actions that optimize the expected return using what has been learned. In simpler words, if the estimates are poor, exploitation actions are likely inefficient, yet if the estimates are good, exploration actions are now the ones that are likely inefficient. In general, it is impossible to know a priori how much exploration and exploitation is needed, as this depends on the unknown model. Therefore, most of reinforcement learning techniques have a *learning or exploration parameter* to hand-tune this compromise between exploration and exploitation.

The exploration/exploitation dilemma has been extensively studied in the context of $n$-armed bandits (see Section 1.3), because the exploration and exploitation are explicitly represented by choosing different arms (Auer et al., 2002). The problem is tackled by defining the *regret* (opportunity loss) of choosing an arm, and the arm with the smallest expected regret is chosen. This can be easily translated to an expected return maximization problem, and by relaxing some assumptions of the bandits such as arm independence, the results for this specific problem can be extended to many other RL problems (Kocsis and Szepesvári, 2006). However, computing the exact expected return is computationally expensive, so analytical upper confidence bounds are used as a way of including in the evaluation the opportunity value of an arm. The study of the exploration/exploitation dilemma and upper confidence bounds for the bandit problem (and its variations) is still an active research domain nowadays (Audibert et al., 2008; Avner et al., 2012).

### 5.1.2 Model-based and Model-free RL

RL algorithms are commonly divided into *model-based* and *model-free* (Kaelbling et al., 1996b), depending on if the algorithm explicitly builds internal MDP models or not. Model-based algorithms are focused on inferring the transition and reward functions from the agent's partial trajectory, and use these functions to obtain an approximate value function and, therefore, a policy. On the other hand, in model-free techniques the agent's partial trajectory is used to directly infer the state-action value function or policy, without the need for inferring the underlying MDP model.

#### A Simple Model-Based Approach

Let $p_t = \{s_0, a_1, s_1, r_1, \cdots, s_t, a_t, r_t\}$ be a partial trajectory of states, actions and rewards, then a model-based algorithm uses some estimators $f$ and $g$ to compute estimates of the transition and reward functions at time $t$:

$$\hat{T}_t = f(p_t),$$
$$\hat{R}_t = g(p_t).$$

If the reward function is known, the partial trajectory is composed only of state-action pairs, and obviously there is no need for the estimate $\hat{R}$. These functions can be used to construct and solve an MDP, in order to obtain a policy $\hat{\pi}_t$ by one of the methods described in Section 3.1.

$$\hat{\pi}_t \in \underset{\pi}{\operatorname{argmax}}\{E\left[\sum_{i=0}^{\infty} \gamma^i \hat{R}_t(s_i, \pi(s_i)) \middle| \hat{T}_t, \hat{R}_t, s_0\right]\}$$

The obtained policy can be used for a few number of steps, and then new estimates are computed to repeat the process. If this procedure is made at each time step, this is called a *certainty equivalence* method (Kaelbling et al., 1996b). This method *exploits* very efficiently the available data, yet this same strength is its main weakness, because it only reinforces the promising actions already explored, doing no *exploration* at all. Moreover, short partial trajectories in the first steps might be non representative due to aleatory variability, biasing the policies towards suboptimal local maxima from which the algorithm will usually not escape by itself.

#### Adding Exploration Actions

A solution to this problem is to perform, every so often, a random action to explore those state-action pairs that are not well reinforced in the data. The *$\epsilon$-greedy approach* (Watkins, 1989) consists in performing these exploration actions with an $\epsilon$ probability, while the exploitation actions from the computed policy are executed with probability $1 - \epsilon$.

A more elaborate method is to use action-selection probabilities, by ranking the state-value function obtained from the estimates with a Boltzmann distribution, and using a temperature parameter to control their relative importance. A high temperature produces nearly equiprobable actions, while a low

temperature tends to a greedy action selection. This method is called a *softmax* approach (Watkins, 1989).

The heuristic of choosing random actions to explore the model is very efficient in highly stochastic and highly connected MDPs. However, in heavily structured MDPs the exploration must be planned ahead in order to be efficient (Chapter 7 specifically focuses on this issue).

### Optimism and R-MAX

An alternative to random exploration is to use the *optimism in the face of uncertainty* principle (Sutton and Barto, 1998; Kaelbling et al., 1996b), which states that a sound exploring strategy is to suppose that what has not been tried enough can produce high payoffs. This optimism is maintained until it is proved to be false (or true) by trying the unknown parts of the model in search for high payoffs. This can be achieved by fictitiously modifying the model estimates depending on the confidence of these estimates. Specifically, the estimates can be replaced by upper bounds that eventually shrink due to the arrival of new samples.

Even though very efficient ad-hoc heuristics can be found for specific problems, optimistic algorithms are applicable to a wide range of highly structured problems. Also, they usually exhibit theoretical guarantees such as one that will be described in Section 5.3.

R-MAX is a simple yet efficient optimistic algorithm (Brafman and Tennenholtz, 2003), that divides the set of state-action pairs into *known* and *unknown* pairs. The *known* pairs are those that have been visited at least $m$ times, while *unknown* pairs have less than $m$ visits. The R-MAX idea is to assign a maximal reward to those pairs that are *unknown*, and to use the estimated model for those that are *known*. The division between the known and unknown pairs is controlled by the free parameter $m$ that allows adapting the algorithm to different exploration requirements.

Formally, let $n(s,a)$ be the number of visits of a pair, $n(s,a,s')$ be the number of observed state transitions $(s,a) \longrightarrow s'$, and $R_i(s,a), i \in \{1,2,\ldots,n(s,a)\}$ be the observed rewards for a state-action pair, then the estimated reward and transition functions using a frequentist approach are given by

$$\hat{R}(s,a) = \frac{1}{n(s,a)} \sum_{i=1}^{n(s,a)} R_i(s,a), \text{ and}$$
$$\hat{T}(s,a,s') = \frac{n(s,a,s')}{n(s,a)}.$$

Now, instead of computing the value function directly for this empirical model (like in the certainty equivalence method), the following equations are solved—using for instance value iteration—to find the fixed point solution:

$$Q(s,a) = \hat{R}(s,a) + \gamma \sum_{s' \in \mathcal{S}} \hat{T}(s,a,s') \max_{a'} Q(s',a') \qquad \text{if } n(s,a) \geq m,$$
$$Q(s,a) = \frac{R_{max}}{1-\gamma} \qquad \text{otherwise,}$$

where $R_{max}$ is the maximum possible value for the rewards. Then, the next action is selected greedily from $Q(s,\cdot)$ and executed to obtain a new sample. At the beginning, all the $Q$-values are initialized with $R_{max}/(1-\gamma)$, so the behavior resembles a random exploration strategy. After some pairs are marked as known, the algorithm begins to explore the less attainable pairs, forcing a lookahead planning for exploration. At the end, when all pairs are marked as known, the algorithm converge to an exploitation policy, which is likely near-optimal depending on $m$.

A more sophisticated algorithm in the same direction is MBIE-EB (Strehl and Littman, 2005), which maintains upper bounds (based on confidence intervals) for the reward and transition functions. Then, the optimistic Q-values can be found using these bounds, which allows for a smoother incorporation of samples to the value function than a fixed $m$ sample threshold.

Another interesting optimistic algorithm is the generalization of the upper confidence bound used for bandits to MDP planning. This algorithm is called UCT (Kocsis and Szepesvári, 2006), which stands for

Upper Confidence for Tree search. The idea is to guide a Monte-Carlo planning over the tree expansion using analytical upper confidence bounds. Several variations of this algorithm have been proposed for model-based reinforcement learning, such as using variance-based bounds in UCT-V (Audibert et al., 2008), or combining this idea with sparse sampling in FSSS (Walsh et al., 2010). These algorithms are optimistic in the sense that they are based on upper bounds which can be understood as optimistic estimates. When the confidence of an estimate increases, it becomes less optimistic, leading in the limit to a near-optimal approximation if the bounds are tight.

#### A Simple Model-Free Approach

Model-based techniques are naturally divided in two alternating phases: (1) estimating the model and (2) solving the estimated MDP. The best known model-free algorithm is Q-learning (Watkins, 1989), which is based on the simple idea of updating the state-action value function only when a transition is observed, and selecting the next action greedily from this Q-function.

Formally, at each step the algorithm observes a reward $r$ and a transition $(s, a, s')$, and updates the Q-value as follows:

$$Q(s,a) = Q(s,a) + \alpha \left( r + \max_{a' \in \mathcal{A}} \{Q(s',a')\} - Q(s,a) \right),$$

where $\alpha$ is called the learning rate and controls the impact of one transition's observation on the value function, or in other words the "speed" of the learning. If the learning speed is too fast, it will learn sub-optimal policies due to local maxima, and if it is too slow it will need a lot of samples to learn a decent policy. Then, a natural approach is to use a dynamic $\alpha_t$, which notably makes the algorithm converge to the optimal $Q^*$ in the limit with probability one (Jaakkola et al., 1994) under reasonable conditions (i.e., finite state and action spaces, bounded reward variance, a divergent $\sum_t \alpha_t$ and a convergent $\sum_t \alpha_t^2$).

Please note that the random exploration methods $\epsilon$-greedy and softmax are not exclusive to model-based techniques. Indeed, they were initially introduced for model-free techniques (Watkins, 1989). Also, optimistic variations of model-free techniques are available, such as the *Delayed Q-Learning* algorithm, where the Q-values updates are delayed until the state-action pair becomes known as in R-MAX (Strehl et al., 2009).

## 5.2 Bayesian Reinforcement Learning

The methods of the previous section often rely on a frequentist learning approach where the uncertainty is tackled by estimating the unknown parameters from the gathered samples, hoping to be close to the real values. Following Section 2.1.3, a natural alternative is to combine MDPs with Bayesian learning, leading to the Bayesian RL framework (Strens, 2000). Now, the unknown parameters to be learned can be represented by random variables under epistemic uncertainty (see Section 2.1.3).

In *model-based Bayesian RL* (Dearden et al., 1999; Strens, 2000) the knowledge about the MDP model $M \in \mathcal{M}$ is represented by a random variable and its epistemic probability distribution over all possible models. This section is specifically focused on model-based Bayesian RL, because most of the work in Bayesian RL falls in this category. However, there are some works in model-free approaches such as the relatively early work of Dearden et al. (1998) or the more recent approach using Gaussian Processes of Ghavamzadeh and Engel (2007).

Using the Bayes rule, the distribution of $M$ can be updated when a new sample is obtained, and as more samples are acquired, some MDPs become more probable than others. The different MDPs can be integrated out, as in any Bayesian method, by using an initial prior distribution, allowing to compute the *expected value function*[18].

**Definition 5.2.1.** *Let $\phi$ be the prior parameters of a distribution over the possible MDPs, such that $Pr(M) = \mathbb{P}_\phi$. Then, the Bayesian evaluation $\mathbb{V}$ of a policy $\pi$ is the expected value of the value function*

18. Please note that this expectation is over the possible MDPs, so it is the expectation over the sum of expected rewards.

*given the parameters $\phi$:*

$$\mathbb{V}^{\pi}(s,\phi) = E_M[V_M^{\pi}(s)|\phi] = \int_{\mu\in\mathcal{M}} V_{\mu}^{\pi}(s)\mathbb{P}_{\phi}(\mu)d\mu.$$

*Under this criterion, a Bayesian optimal policy verifies*

$$\pi^*(s,\phi) \in \operatorname*{argmax}_{\pi} \mathbb{V}^{\pi}(s,\phi), \forall s \in \mathcal{S}.$$

This definition was first presented implicitly by Duff (2002) and explicitly by Dimitrakakis (2008), and is the cornerstone of Bayesian RL. A special notation for the Bayesian value function $\mathbb{V}$ is used here, to stress out the difference between a Bayesian evaluation $\mathbb{V}$, and the normal evaluation $V$. However, usually they are easy to recognize as the Bayesian one depends on the distribution parameters of the MDP random variable.

Remarkably, Bayesian RL provides a sound way of dealing with the exploration/exploitation dilemma, because both aspects are naturally embedded in a unique optimization criterion. Moreover, and in contrast to conventional RL, it is theoretically possible to compute a Bayesian optimal policy in a number of cases. Another important characteristic of Bayesian RL is the existence of a prior distribution, where prior knowledge about the problem and its structure can be encoded to speed up or improve the quality of the solutions.

Unfortunately, finding the exact Bayesian optimal policy is generally intractable, so approximate Bayesian RL algorithms also have to deal with similar compromises such as the exploration/exploitation dilemma. Additionally, it is not clear that *all* the structure of the problem can be encoded in the prior, and, in some domains, defining the prior distribution can be as hard as solving the problem.

For this and the following chapters, it will be assumed that the reward function $R$ is known and normalized to the $[0,1]$ interval [19]. This assumption can be made without loss of generality, because the reward can be considered as a part of the system state, and therefore, the uncertainty of the reward function is automatically encoded in the transition function (Poupart et al., 2006; Kolter and Ng, 2009). Under this assumption, the random variable $M$ is completely characterized by the possible parameters of the *transition function* $T$, where $T_{\mu}(s,a,s') = Pr(S'=s'|A=a,S=s,M=\mu)$. An initial prior distribution $Pr(M) = \mathbb{P}_0$ has to be specified, which is then updated using Bayes rule similarly to what was presented in Section 2.3.2.

At time $t$, the posterior $\mathbb{P}_t$ depends on the initial distribution $\mathbb{P}_0$ and the state-action history so far $h_t = s_0, a_0, \cdots, s_{t-1}, a_{t-1}, s_t$. Similarly to POMDPs, this update can be applied sequentially due to the Markov property, i.e., at time $t+1$ it is only necessary to have $\mathbb{P}_t$ and the triplet $(s_t, a_t, s_{t+1})$ to compute the new distribution:

$$\begin{aligned}
\mathbb{P}_{t+1}(\mu) &= Pr(M=\mu|h_{1:t+1},\mathbb{P}_0) \\
&= Pr(M=\mu|S_t=s_t,A_t=a_t,S_{t+1}=s_{t+1},\mathbb{P}_t) \\
&\propto Pr(S_{t+1}=s_{t+1}|A_t=a_t,S_t=s_t,M=\mu)Pr(M=\mu|\mathbb{P}_t) \\
&= T_{\mu}(s_t,a_t,s_{t+1})\mathbb{P}_t(\mu). \qquad (5.1)
\end{aligned}$$

### 5.2.1 Representation and Priors

Following the same notation as used in Section 2.3.2, the probability distribution over the transition functions can be parametrized by a vector $\phi$, where $Pr(M=\mu) = \mathbb{P}_{\phi}(\mu)$. This distribution is the one that will be updated using the Bayes rule, and summarizes the information that has been gathered about the model at the current time step. Using the POMDP terminology, this distribution is known as the *belief* over the model, so here the abuse of notation $b = \mathbb{P}_{\phi}$ is used, following the state of the art notation (Duff, 2002).

The selection of a suitable prior is an important issue in BRL algorithms, because it has a direct impact on the solution quality and computational time. This dissertation focuses only on solving RL

19. This means that always $R_{max} = 1$ and $R_{min} = 0$.

problems with discrete state and action spaces, which leads to a transition function representation using categorical distributions. For this transition function representation, a naive approach is to consider one independent Dirichlet distribution for each state-action transition as it has been presented in Section 2.3.2, known as the Flat-Dirichlet-Multinomial prior (FDM), whose pdf is defined as

$$b(\mu) = \mathbb{P}_\phi(\mu) = \prod_{s,a} D(\mu_{s,a}; \phi_{s,a}),$$

where $D(\cdot;\cdot)$ are independent Dirichlet distributions. FDMs can be applied to any MDP with categorical transition distribution, but it is only optimal under the strong condition of independence between the state-action pairs in the transition function. However, this prior has been broadly used because of its simplicity for computing the Bayesian update and the expected value.

Consider that the parameter vector $\phi$ is composed by the counters of observed transitions, where $\phi_{s,a}(s')$ corresponds to how many times the transition $(s,a,s')$ has been observed. Then the expected value of a transition probability (without considering the initial prior) is

$$\mathbb{E}[T_M(s,a,s')|B=b] = \mathbb{E}[\mu_{s,a}(s')|B=b] = \frac{\phi_{s,a}(s')}{\|\phi_{s,a}\|_1},$$

and the Bayesian update from time $t$ to $t+1$, under the evidence of a transition $(s,a,s')$, is reduced only to updating one parameter:

$$\phi_{s,a}(s')^{(t+1)} = \phi_{s,a}(s')^{(t)} + 1.$$

An initial FDM prior is composed by the initial values of these counters. For example, an initial uniform distribution can be represented by initializing all counters to one.

Even though FDMs are useful to analyze and solve benchmark algorithms, in practice they are inefficient because they do not exploit structured information about the problem. One can for example encode the fact that multiple actions share the same model by factoring multiple Dirichlet distributions, or allow the algorithm to identify such structures using Dirichlet distributions combined with Chinese Restaurant Processes or Indian Buffet Processes (Asmuth et al., 2009). Also, specific problems could lead to other families of distributions different from Dirichlet, for example the Bayesian distribution over deterministic MDPs presented by Sorg et al. (2010).

Usually, factoring Dirichlet distributions helps to grasp the structured information of the problem. However, there are some problems where factoring leads to ambiguous Bayes updates, such as in the Paint/Polish problem (Walsh et al., 2009). This means that when a next state is observed after performing an action, it is impossible to know which factored Dirichlet distribution must be updated, because two or more of them could be the cause of the observed outcome. This problem can be tackled by neglecting some of the structure of the problem through adding more independent Dirichlet distributions to the prior representation, or by using a generalization of the Dirichlet family called the Hyperdirichlet family (Hankin, 2010).

### 5.2.2 Optimal Bayesian-RL

Consider a POMDP model, like in Chapter 3, where the state $w$ is composed by both the state of the system $s$ and the transition function parameters $\mu$. If $b$ represents the belief over $\mu$, this defines a Bayes-Adaptive MDP (BAMDP)[20] (Duff, 2002), a special kind of belief-MDP where the belief-state is factored into the (visible) system states and the belief $b$ over the (hidden) model $\mu$. Moreover, due to the integration over all possible models in the value function of the BAMDP, the transition function $\tau(w,a,w')$ is given by

$$\begin{aligned}
Pr(W'=w'|W=w,A=a) &= Pr(B'=b',S'=s'|B=b,S=s,A=a) \\
&= Pr(B'=b'|B=b,S=s,A=a,S'=s')Pr(S'=s'|S=s,A=a,B=b) \\
&= \mathbb{I}(b',Bayes(b,s,a,s'))\int_{\mu\in M} Pr(S'=s'|S=s,A=a,M=\mu)Pr(M=\mu|B=b)d\mu \\
&= \mathbb{I}(b',Bayes(b,s,a,s'))E_M[T_M(s,a,s')|B=b]
\end{aligned}$$

20. BAMDP also stands for Belief-Augmented MDP (Dimitrakakis, 2008).

where $\mathbb{I}(b', Bayes(b, s, a, s'))$ is an indicator function since the model is assumed to be stationary, and $Bayes(b, s, a, s')$ is the shorthand notation of the Bayes update of Equation 5.1. The optimal Bayesian policy can then be obtained by computing the optimal Bayesian value function (Duff, 2002; Poupart et al., 2006):

$$\mathbb{V}^*(s, b) = \max_a \left[ \sum_{s'} E_M[T_M(s, a, s')|B = b](R(s, a, s') + \gamma \mathbb{V}^*(s', b')) \right] \tag{5.2}$$

with $b' = Bayes(b, s, a, s')$ the posterior after the Bayes update.

For the finite horizon case the same reasoning can be used, so that the optimal value can be computed in theory for a finite or infinite horizon by performing Bayes updates and computing expectations. However, in practice, computing this value function exactly is very expensive due to the large branching factor of the tree expansion. Therefore, it is common to use approximate algorithms to find near-optimal policies in a reasonable computing time.

Approximate Bayesian RL algorithms typically fall within one of the three following classes (Asmuth et al., 2009) that will be detailed in the following sections.

- **Belief-lookahead** approaches reformulate RL as the problem of solving a POMDP where the state is the pair $w = (s, \mu)$. Even though approximate POMDP solvers cannot be used directly, some researchers have adapted them to approximately solve the Bayesian RL formulation.
- **Undirected Exploration** approaches perform actions according to the current belief distribution, and every so often they perform exploration actions independently of the current knowledge about the environment.
- **Myopic Exploration** approaches propose exploration mechanisms that explicitly attempt to reduce the model uncertainty. In particular, here the focus is set on those algorithms relying on the principle of optimism in the face of uncertainty.

### 5.2.3 Belief-lookahead Approaches

At first glance, it is obvious to look at POMDP algorithms, because they also deal with belief-MDPs. Unfortunately, one cannot directly benefit from classical POMDP algorithms because of the infinite dimensional nature of the state space, which includes the space of MDP models. Despite that, only few generalizations are needed to theoretically solve the Bayesian RL formulation due to the special properties of the transition function, such as stationarity.

For example, Duff (2002) proposes using policy gradient techniques based on function approximators and Monte-Carlo sampling. The idea is to sample both belief and state trajectories, and learn stochastic finite-state controllers (policies) as solutions of the problem. This is a very general strategy, that can be applied to any state space and to any prior family, but unfortunately the amount of samples required to obtain a good policy is usually huge. Even though this method is using the "standard" way for dealing with continuous high-dimensional spaces in machine learning, the properties of the POMDP and its belief-space are not exploited at all.

In contrast, a method that forces the use of POMDP properties is MEDUSA (Jaulmes et al., 2005), where the belief space is discretized in a grid to apply any finite-state POMDP solver. The problem of this method are the same of any algorithm that samples in a high-dimensional continuous space: the number of points needed to produce a fair approximation grows exponentially with the dimensionality.

Another technique for Bayesian RL is introduced in (Poupart et al., 2006), where an analytical solution for the value function under the FDM prior is presented, by generalizing the concept of $\alpha$-vectors (see Section 3.2.4) to $\alpha$-functions. These functions are proved to be multivariate polynomials under the Bellman operator, and therefore they can be propagated to find the value function's fixed point. This result allows to find (in theory) an optimal policy for Bayesian RL off-line. Yet, in practice it is also intractable due to the exponential growth of terms of the polynomials, which rapidly exhausts memory and computational resources. The solution proposed by Poupart et al. (2006) is to project $\alpha$-functions to smaller space of polynomials with a bounded quantity of monomials, by using basis functions and minimizing the squared error. Then, a modified version of the PB algorithm PERSEUS (Spaan and

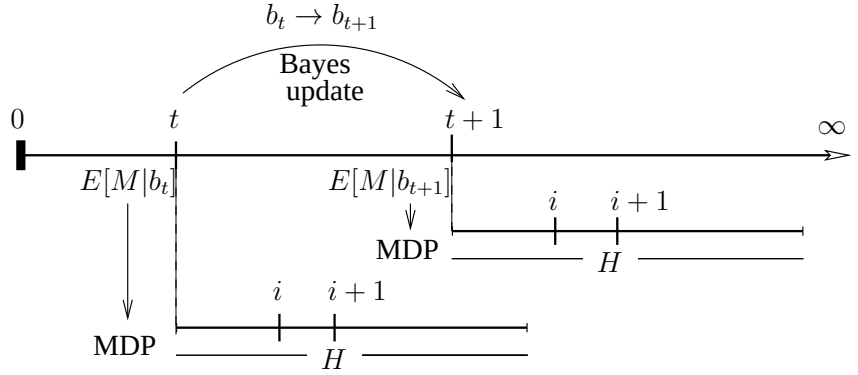



Figure 5.1: EXPLOIT algorithm schema. At each time step $t$, the algorithm performs a Bayes update of the prior, and solves the MDP derived from the expected model of the belief.

Vlassis, 2005), called BEETLE, is used to asynchronously backup the $\alpha$-functions and then re-project them to the reduced space.

Another approach is to actually develop the tree of the possible belief-states and use smart branch and bound techniques to control the tree expansion (Dimitrakakis, 2008). Even though such approach may seem efficient for heavily structured priors, it does not scale neither in space nor time due to the overwhelming branching factor.

All these computationally expensive approximate solutions have the advantages of being off-line and theoretically sound, but in practice they are usually outperformed by on-line approaches.

A natural on-line lookahead approach is to use Monte-Carlo sampling (Ross and Pineau, 2008). Moreover, Ross and Pineau (2008) embed the problem of defining the prior structure in the same optimization process, by learning Bayesian networks, which allows constraining the possible beliefs to lookahead.

Another on-line belief-lookahead strategy is Bayesian Sparse Sampling. Wang et al. (2005) propose maintaining a sparse belief tree where the root is the current belief. At each on-line step, the tree sparsely grows using Thompson sampling up to a fixed quantity of elements and then the tree is evaluated by using the Q-values of the expected model as the values for the leaves (see the EXPLOIT algorithm of next section). Recently, this idea has been combined with the FSSS algorithm (Walsh et al., 2010) to produce the BFS3 algorithm (Asmuth and Littman, 2011), a Bayesian algorithm that offers theoretical guarantees of near-optimality (PAC bounds), prior independence (tree search) and scalability (sparse Monte-Carlo sampling). Also, a similar technique called BAMCP has been recently introduced in (Guez et al., 2012), where the UCT and Monte-Carlo planning ideas are applied to the Bayes-Adaptive adaptive scenario.

### 5.2.4 Undirected Exploration Approaches

A straightforward technique for BRL is to select one representative MDP based on the posterior and act according to its optimal value function. The baseline algorithm in this family is called EXPLOIT (Dearden et al., 1999; Duff, 2002; Poupart et al., 2006), where the expected model $\bar{\mu} = \mathbb{E}[M|B = b]$ (i.e., the most likely model) is selected at each time step. Therefore, the algorithm has to solve a different MDP of horizon $H$—an algorithm parameter, not the problem horizon— at each time step $t$ as can be seen in Figure 5.1.

The value function equation for EXPLOIT is,

$$V_i^{\text{EXPLOIT}}(s, \mathbf{b}_t) = \max_a \sum_{s' \in \mathcal{S}} T(s, a, s', \mathbf{b}_t) \left[ R(s, a, s') + \gamma V_{i-1}^{\text{EXPLOIT}}(s', \mathbf{b}_t) \right]$$
$$\text{with } T(s, a, s', \mathbf{b}_t) = T_{\bar{\mu}}(s, a, s') = \mathbb{E}[T_M(s, a, s)|B = \mathbf{b}_t].$$

For example, under the FDM prior, the transition function $T(s, a, s', \mathbf{b}) = \dfrac{\phi_{s,a}(s')}{\|\phi_{s,a}\|}$. The main difference with the optimal Bayesian value function (Equation 5.2) is that the evolution of the belief, and thus the

gain of information, is neglected by EXPLOIT, so the belief $\mathbf{b}_t$ is constant for all the value iteration steps $i \in [0, H]$. This means that at each time step $t$ the MDP $\langle \mathcal{S}, \mathcal{A}, T = \bar{\mu}, R, s_0 \rangle$ must be solved in order to select the next action.

This simple algorithm lacks exploration actions, because the policy computed at each step is the optimal policy with respect to the knowledge so far, so in a sense it is the Bayesian version of the certainty equivalence method of Section 5.1.2. Following the same idea as frequentist RL algorithms, some random exploration actions can be performed every so often, producing an undirected exploration that can be manually tuned for each problem.

A different approach is to use sampling-based methods like in (Strens, 2000), where a random MDP is sampled from the posterior rather than selecting the expected model. The agent acts according the optimal solution of the sampled MDP, and therefore, an exploration heuristic naturally arises from the sampling. Ideally, the algorithm performs less and less exploration as the distribution concentrates over the real-model, similarly to a Boltzmann strategy, yet there are no guarantees for this to happen.

Even though these random exploration techniques might help escaping from local optima, they are only suitable for highly stochastic scenarios with weakly structured MDPs. Relying on randomness when smart exploration is needed will lead to poor performance or at least to slow convergence speed.

### 5.2.5 Myopic and Optimistic Exploration Approaches

The algorithms of the past section propose adding random actions to a greedy algorithm to force exploration. In this section, more directed exploration strategies are introduced. The first directed strategy for Bayesian RL was proposed by Dearden et al. (1999), where the action selection is biased by a one-step information lookahead technique called *myopic value of information*.

Optimistic Bayesian RL algorithms (Kolter and Ng, 2009; Asmuth et al., 2009; Sorg et al., 2010; Asmuth and Littman, 2011; Araya-López et al., 2012a), have recently gained attention due their simplicity, theoretical guarantees and performance. The general idea is to use the current posterior to find an optimistic approximation of the Bayesian value function and act greedily with respect to it. The optimistic approximation usually depends on the uncertainty held by the posterior, so it gets more and more narrow towards the optimal Bayesian value function as more samples are acquired.

The Bayesian Exploration Bonus algorithm (BEB) consists in solving at each time step $t$ an MDP generated by the current expected model (like EXPLOIT), but using a modified reward function depending on the posterior uncertainty (Kolter and Ng, 2009). Let $R(s, a, s')$ be the given reward function and $\mathbf{b}_t$ the belief state at time $t$, then BEB performs value iteration as follows:

$$V_i^{\text{BEB}}(s, \mathbf{b}_t) = \max_a \sum_{s'} T(s, a, s', \mathbf{b}_t) \left[ \hat{R}(s, a, s', \mathbf{b}_t) + \gamma V_{i-1}^{\text{BEB}}(s', \mathbf{b}_t) \right]$$

$$\text{with} \qquad \hat{R}(s, a, s', \mathbf{b}_t) = R(s, a, s') + \frac{\beta}{1 + \|\phi_{s,a}\|},$$

with $\beta$ a parameter that controls the optimism of the algorithm. Similarly to EXPLOIT, BEB's value iteration neglects the evolution of $\mathbf{b}_t$, so at each time step $t$ the algorithm solves an MDP with fixed transition and reward functions. This algorithm relies on the FDM prior representation, and even though it is not difficult to extend the bonus idea to tied parameters, it is unclear if this method can be generalized to more complex priors.

Sorg et al. (2010) go beyond the linear-based bonus of BEB, and propose an alternative bonus based on the variance of the posterior. This method has proved to be very successful in high-regret scenarios, where the initial actions are critical for the final return.

A completely different method is used by the BOSS (Asmuth et al., 2009) algorithm. Instead of following an EXPLOIT-like approach, BOSS samples different models from the prior and merges them in a single MDP. This MDP has an augmented action space, that considers each action of each sampled model as a different possible action. The optimal value function of this action-augmented MDP is (probably) optimistic, and becomes less optimistic as the distribution of models narrows. However, computing the optimal solution for this MDP is expensive due to the large action-space, so BOSS updates its policy only when any state-action pair gets visited more than $m$ times (i.e., the state-action pair passes to be *known*).

BEB has the advantage of being an almost deterministic algorithm [21] and does not rely on expensive sampling procedures as BOSS. On the other hand, BOSS is optimistic about the transitions, which is where the uncertainty lies, while BEB is optimistic about the reward function, even though this function is known. Also, BOSS is not restricted to FDM priors, because MDPs are sampled from an arbitrary posterior and not integrated out as in EXPLOIT-like algorithms.

## 5.3 Probably Approximately Correct Algorithms

One of the advantages of the optimistic algorithms presented in Section 5.2.5 is that they usually verify theoretical guarantees about how well the algorithm will behave in most of the cases. These guarantees are based on the statistical learning theory, which analyzes the learning complexity of problems and algorithms. This section presents only a quick introduction to two statistical learning properties used in RL: the PAC-MDP and the PAC-BAMDP properties.

*Probably Approximately Correct Learning* (PAC) provides a flexible way of analyzing the quality of learning algorithms (Valiant, 1984). This framework is based on the notion of *sample complexity*, which is the amount of samples needed to learn (or be close to learn) certain phenomena. Similarly to time or space complexity, the learning complexity can be expressed in terms of the parameters of the algorithm and the problem, and the interest is to find algorithms with low order sample complexity. However, if the sampling is stochastic, it is impossible to bound the amount of samples due to the worst case scenario (non representative samples). Therefore, a statistical analysis is often needed to ensure low order sample complexity within certain probability, which leads to the PAC analysis.

The general idea of PAC is that with high probability $1-\delta$ (probably), a machine with a low training error produces a low generalization error bounded by $\epsilon$ (approximately correct). If the number of steps needed to satisfy this condition is bounded by a polynomial function, then the algorithm is PAC-efficient.

### 5.3.1 PAC-MDP

In RL, the PAC-MDP property (Strehl et al., 2009) guarantees that an algorithm generates an $\epsilon$-close policy with probability $1-\delta$ in all but a polynomial number of steps. The major difference with classical PAC Learning is that there is no guarantee on when the non-$\epsilon$-close steps will occur, but the number of non-$\epsilon$-close steps is bounded by a polynomial. An important result is the general PAC-MDP Theorem 10 in Strehl et al. (2009), where three sufficient conditions are presented to comply with the PAC-MDP property. First, the algorithm must use at least near *optimistic* values with high probability. Also, the algorithm must guarantee with high probability that it is *accurate*, meaning that, for the known parts of the model, its actual evaluation will be $\epsilon$-close to the optimal value function. Finally, the number of non-$\epsilon$-close steps (also called *sample complexity*) must be bounded by a *polynomial* function.

**Definition 5.3.1** (PAC-MDP)**.** *In mathematical terms, PAC-MDP algorithms are those for which, with probability* $1-\delta$*, the evaluation of a policy* $\mathbf{A}_t$*, generated by algorithm* $\mathbf{A}$ *at time* $t$ *over the real underlying model* $\mu_0$*, is* $\epsilon$*-close to the optimal policy over the same model in all but a polynomial number of steps:*

$$V_{\mu_0}^{\mathbf{A}_t}(s) \geq V_{\mu_0}^{*}(s) - \epsilon.$$

Several RL algorithms comply with the PAC-MDP property, differing from one another mainly on the tightness of the sample complexity bound. For example, R-MAX (Brafman and Tennenholtz, 2003), MBIE-EB (Strehl and Littman, 2005) and Delayed Q-Learning (Strehl et al., 2009) are some non-Bayesian RL algorithms for which this property has been proved, whereas BOSS (Asmuth et al., 2009) is a Bayesian RL algorithm which is also PAC-MDP.

In PAC-MDP analysis the policy produced by an algorithm should be close to the optimal policy derived from the real underlying MDP model. This *utopic* policy (Poupart et al., 2006) cannot be computed, because it is impossible to learn exactly the model with a finite number of samples, but it is possible to reason on the probabilistic error bounds of an approximation to this policy.

21. In case of equal values, actions are sampled uniformly.

### 5.3.2 PAC-BAMDP

An alternative to the PAC-MDP approach is to be PAC with respect to the optimal *Bayesian* policy, rather than using the optimal *utopic* policy. We call this *PAC-BAMDP analysis*, because its aim is to guarantee closeness to the optimal solution of the Bayes-Adaptive MDP.

**Definition 5.3.2** (PAC-BAMDP)**.** *An algorithm is PAC-BAMDP if, with probability* $1-\delta$*, the Bayesian evaluation of a policy* $\mathbf{A}_t$ *generated by algorithm* $\mathbf{A}$ *at time* $t$ *is* $\epsilon$*-close to the optimal Bayesian value function in all but a polynomial number of steps, where the Bayesian evaluation is parametrized by the belief* $b$*:*

$$\mathbb{V}^{\mathbf{A}_t}(s,b) \geq \mathbb{V}^*(s,b) - \epsilon,$$

*with* $\delta \in [0,1)$ *and* $\epsilon > 0$*.*

This type of analysis was first introduced in Kolter and Ng (2009), under the name of *near-Bayesian* property, where it is shown that a *modified* version of BEB is PAC-BAMDP for the undiscounted finite horizon case[22]. Also, BFS3 (Asmuth and Littman, 2011) claims to be PAC-BAMDP.

### 5.3.3 Discussion

PAC-MDP and PAC-BAMDP properties might seem almost the same property with a different flavor, but in fact they represent different objectives. The objective of the PAC-MDP property is to guarantee closeness to the policy that optimizes the return over the real underlying model, while the PAC-BAMDP property guarantees closeness to the policy that best uses the current knowledge over all possible models. In other words, PAC-MDP tries to be close to the best that exists, while PAC-BAMDP tries to be close to the best that can be attained. In PAC-MDP analysis, the approximate correctness guarantee is needed because the optimal utopic policy is impossible to obtain[23]. In contrast, PAC-BAMDP approximate correctness is needed because the optimal Bayesian policy is hard to compute. Moreover, this difference has been theoretically proved by (Kolter and Ng, 2009), where they show a counterexample problem where the Bayesian optimal policy is not PAC-MDP.

Please note that there is no "correct" way to analyze RL algorithms: both guarantees have their proper justification and properties. Despite that, Bayesian RL algorithms verify more naturally the PAC-BAMDP property than the PAC-MDP one. Conversely, non-Bayesian RL algorithms verify more naturally the PAC-MDP property.

22. However, some—rectifiable—errors have been spotted in the proof of near-Bayesianness of BEB in (Kolter and Ng, 2009), as discussed with the authors.

23. An algorithm **can** find the optimal utopic policy, yet theoretically it cannot be **sure** that this policy is optimal.

# Chapter 6

# Bayesian Optimistic Local Transitions

## Contents



*"When it is not in our power to determine what is true, we ought to follow what is most probable"*

— Descartes.

The previous chapter has introduced different approaches to solve the RL problem of earning rewards in an MDP where the transition model is unknown [24]. This chapter focuses specifically on optimistic strategies for Bayesian RL, by proposing that the optimism should be localized in the transition function, where the uncertainty lies, and not directly in the value or in the reward as in some previous approaches. Following this simple idea, a novel algorithm for Bayesian-RL called BOLT (*Bayesian Optimistic Local Transitions*) is introduced, opening the second series of contributions of this dissertation. First, it is shown that by using the appropriate parameters, BOLT complies with the PAC-BAMDP property (see Section 5.3), providing theoretical guarantees through sample complexity bounds. Also, it is empirically shown that BOLT has a more robust behavior under parameter tuning than previous approaches like BEB (Kolter and Ng, 2009).

## 6.1 Optimism in RL

The optimism in the face of uncertainty is a widely accepted concept in RL nowadays [25] due to the theoretical and empirical results of the algorithms based on this principle. In a nutshell, this principle

24. Please note that the assumption of a known reward function made in Chapter 5 also holds for this chapter.
25. However, optimism is not free from criticism (Ortner, 2008).

states that a good exploration strategy consists in artificially encouraging exploration actions that lead to highly uncertain parts of the model, as they could potentially return higher payoffs. Then, when these uncertain parts of the model are visited, the artificial bias towards them decreases, because they naturally become less uncertain. Despite this, the exploration/exploitation dilemma still holds: too much optimism will over-explore, and too little optimism will not overcome the local maxima.

The optimistic model-based algorithms presented in the Chapter 5, both Bayesian and not, apply optimism in different ways.

- In **R-MAX**(Brafman and Tennenholtz, 2003) the Q-values are directly overrated by a constant upper bound to find an optimistic fixed point solution. Here, the optimism is controlled by the parameter $m$ which is the number of visits needed before declaring a state-action pair as known.
- In **MBIE-EB**(Strehl and Littman, 2005) the Q-values are directly overrated by considering a dynamic upper bound based on confidence intervals. Here, the optimism is controlled by the confidence interval's probability $1 - \delta$, which is the probability of sampling a model in the interval.
- In **BEB**(Kolter and Ng, 2009) the reward function is increased by an exploration bonus that depends on the uncertainty of the state-action pair. Here, the optimism is controlled by the parameter $\beta$ which is the maximum possible bonus when a state-action pair has never been visited.
- In **BOSS**(Asmuth et al., 2009) the algorithm merges several sampled transition models and uses the best transitions of all the sampled models to compute an optimistic value function. Here, the optimism is controlled by the parameter $K$, which is the number of models to sample for each known state-action pair, and the parameter $m$ which is the number of visits needed for a pair to be known as in R-MAX.

Next section presents a new method for applying optimism based on optimistic transitions.

### 6.1.1 Bayesian Optimistic Transitions

Framing the discussion to Bayesian RL, at each step there is a model posterior that represents the knowledge about the transitions. For using this posterior one can rely on sampling several models and trying to merge them (like in BOSS), or work directly with the expected model as an approximation (like EXPLOIT). However, this last approximation is not optimistic, so BEB proposes to modify the reward function enough to produce an optimistic value function. This dissertation proposes that constructing an optimistic transition function is a more natural approach than modifying the reward function, because the uncertainty about the reward can be transfered to the transition function, and therefore optimism is assigned where the uncertainty actually lies. Moreover, the transition functions are constrained by the laws of probability, making this type of optimism more realistic than unbounded reward bonuses. However, upper or lower transition probability bounds are meaningless by themselves in terms of optimism, because an optimistic transition might be any value depending on the reward function and the other transitions.

**Example 6.1.1** (3-state MDP)**.** *For example, consider the 3-state MDP on the top left corner of Figure 6.1, with an action space $\mathcal{A} = \{a, b\}$. Under an initial uniform prior, all the possible transitions have the same probabilities and the same certainty as the top right corner shows. At the bottom left corner the posterior after observing several transitions is represented by using colors for certainty and the thickness of the connections as the expected probabilities. The* EXPLOIT *algorithm uses a "black and white" version of this posterior, where only the expected probabilities are considered. The idea of optimistic transitions is to artificially modify the expected model to encourage transitions that can lead to high values, such as the self connections of $S_3$ (reward of 1). The figure in the bottom right corner is a possible modification of the probabilities that is optimistic in terms of the transitions.*

The main questions about optimistic transitions are how to modify the expected probabilities without breaking the probability laws, and how to ensure that this modification is actually an optimistic approximation in terms of the value function. Rather than directly modifying the expected model as in MBIE-EB, the next section describes how to modify the *posterior* in order to find an optimistically biased expected model.

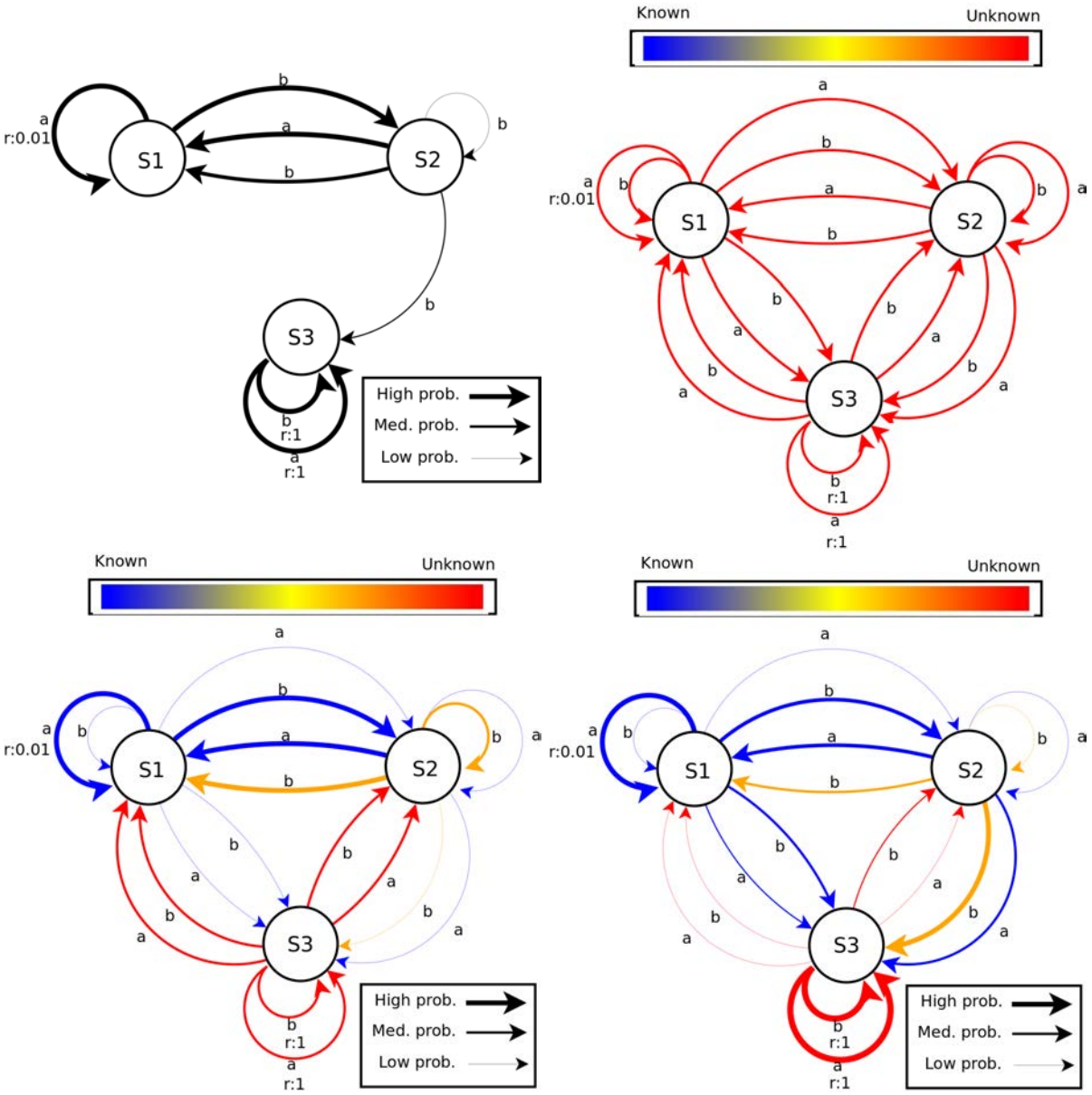



Figure 6.1: 3-state MDP example with two possible actions $\mathcal{A} = \{a, b\}$. In the top left corner, the real underlying MDP. To the right, the initial belief under a uniform prior. In the bottom left corner the figure shows the posterior belief and its expected transitions after observing some sample transitions. The bottom right corner shows a possible optimistic variant of the expected transitions.

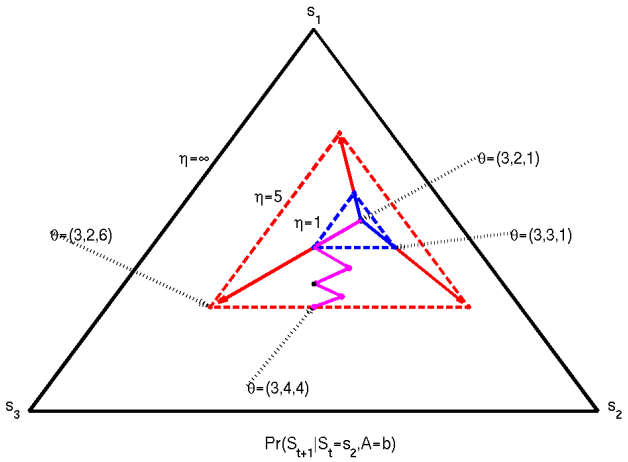



Figure 6.2: An example of all the possible Bayes Boosts of sizes $\eta = 1$ (blue) and $\eta = 5$ (red). For both boosts, the simplexes containing all the possible updates of each size are displayed. A random update of size 5 is shown in Magenta.

### 6.1.2 Bayes Boost

The natural solution is to search in the belief-space for optimistic beliefs where the expected model returns a higher value than the optimal Bayesian value function. However, the belief-space is a continuous multi-dimensional space, and obviously the optimal Bayesian value function cannot be computed. Fortunately, the *future beliefs* are a discrete set (countable), and the *convexity of the value function*[26] can be used to upper bound the optimal Bayesian value function.

To do so, the same Bayes rule can be used to find future belief-states by artificially adding local transition occurrences in a lookahead fashion. For example, in Figure 6.2 the belief-space of the state-action pair $(s_2, a)$ (using an FDM prior) is graphically represented by a 2-simplex. The point $\theta = (3, 2, 1)$ represents the current belief $\mathbf{b}_t$ that defines a distribution over transition probabilities, and from there only three possible Bayes updates are possible: $(s_2, a, s_1)$,$(s_2, a, s_2)$ and $(s_2, a, s_3)$. This forms a small simplex (in blue) containing the initial point, and the 3 possible resulting beliefs after one update ($\eta = 1$). If more updates are performed, for example five of them ($\eta = 5$), a larger simplex (in red) can be constructed by performing 5 updates in the same direction (e.g., $\theta = (3, 2, 6)$) for each possible state. All the possible 5-update combinations are inside the simplex, as the shown by the magenta trajectory. The key insight of this figure is that, as the value function is a convex function in the belief-space, the maximum value in the red simplex will be one of its corners. Therefore, no matter which will be the real updates later, they are upper-bounded by one of the corners. In other words, distributing the boosts in several transitions is always less optimistic than concentrating all the boosts in (the correct) transition.

**Definition 6.1.1.** *Bayes Boost* A Bayes Boost *corresponds to applying the Bayes update for a specific pair* $(s, a)$ *of a prior* $\mathbf{b}_t$ *using an artificial set of transitions* $\lambda^{\eta}_{s,a,\sigma} = \{(s, a, \sigma), \ldots, (s, a, \sigma)\}$ *of size* $\eta$*. The function* $\mathbf{b}' = Boost(\mathbf{b}_t, s, a, \sigma, \eta)$ *computes the artificial posterior* $\mathbf{b}' = Pr(B' = \mathbf{b}'|B = \mathbf{b}, \lambda^{\eta}_{s,a,\sigma})$ *that corresponds to applying* $\eta$ *Bayes updates in the form* $Bayes(\mathbf{b}, s, a, \sigma)$*. The* $\eta$ *parameter will be called the* size *of the boost, and* $\sigma$ *is called the* direction *of the boost.*

26. As presented in the previous section, BRL is a special case of POMDPs, so the convexity property automatically extends to this framework (at least for a finite-dimensional simplex).

Please note that a Bayes Boost is not optimistic by itself, because not all the possible directions are optimistic. As the value function is unknown, the correct $\sigma$ must be selected in an optimization process, as the algorithm of the next section proposes.

## 6.2 BOLT: A Simple and Robust BRL Algorithm

BOLT is an EXPLOIT-like algorithm (see Section 5.2.4) in the sense that, at each step, a posterior model is updated and an MDP is solved to obtain the next action. Unlike EXPLOIT, the transition model used for computing the policy is not the expected model, but an optimistic variant. The optimistic MDP variant is obtained by using Bayes Boosts for each transition while solving an action-augmented MDP (AAMDP) for selecting the boosting directions.

**Algorithm 6:** BOLT: Bayesian Optimistic Local Transitions Algorithm.

**Input**: $R$, $\gamma$, $\mathbf{b}_0$, $s_0$,$\eta$,$\gamma$
$\mathbf{b}_t \leftarrow \mathbf{b}_0$;
$s_t \leftarrow s_0$;
**while** *episode not ended* **do**
  **foreach** $(s,a) \in \mathcal{S} \times A$ **do**
    **foreach** $\sigma \in \mathcal{S}$ **do**
      $\alpha \leftarrow (a, \sigma)$;
      $\hat{T}(s,\alpha,\cdot) \leftarrow T(s,a,\cdot,Boost(\mathbf{b}_t,s,a,\sigma,\eta))$;
    **end**
  **end**
  $aamdp \leftarrow \langle \mathcal{S}, \mathcal{A}\times\mathcal{S}, \hat{T}, R, s_t\rangle$;
  $Q^{\text{BOLT}} \leftarrow ValueIteration(aamdp,\gamma)$;
  $a_t \in \underset{a}{\text{argmax}}\{Q^{\text{BOLT}}(s_t,a)\}$;
  $s' \leftarrow execute(a_t)$;
  $\mathbf{b}_t \leftarrow Bayes(s_t,a_t,s')$;
  $s_t \leftarrow s'$;
**end**

The AAMDP uses an augmented action space $\mathfrak{A} = \mathcal{A} \times \mathcal{S}$, where the transition model for action $\alpha = (a,\sigma)$ in state $s$ is the *local* expected model derived from $\mathbf{b}_t$ under a Boost of size $\eta$ and direction $\sigma$ as shown in Algorithm 6. In other words, the current belief is biased with an artificial evidence of transitions $\lambda^{\eta}_{s,a,\sigma} = \{(s,a,\sigma),\ldots,(s,a,\sigma)\}$ of size $\eta$. Therefore, the value iteration picks for each state both an action $a$ and the state $\sigma$ whose probability it is convenient to reinforce.

Using a concise formal representation, the action-augmented value iteration of BOLT can be summarized in the recursive form

$$V_i^{\text{BOLT}}(s,\mathbf{b}_t) = \max_{\alpha\in\mathcal{A},\mathcal{S}} \sum_{s'} \hat{T}(s,\alpha,s',\mathbf{b}_t)\left[R(s,a,s') + \gamma V_{i-1}^{\text{BOLT}}(s',\mathbf{b}_t)\right]$$

$$\text{with} \qquad \hat{T}(s,\alpha,s') = \mathbb{E}[Pr(S'=s'|S=s,A=a)|\mathbf{b}_t,\lambda^{\eta}_{s,a,\sigma}].$$

As in EXPLOIT or BEB, BOLT's value iteration neglects the evolution of $\mathbf{b}_t$, but the modified transition function works as an optimistic approximation of the neglected Bayesian evolution. An other view is to consider the Bayes Boost as an optimistic belief-lookahead strategy, where the boosting directions are selected to arrive to an optimistic belief.

BOLT is similar to BEB in the sense that it is an optimistic algorithm that is almost deterministic. But it can be considered also similar to BOSS because it puts the optimism in the transition function. However, BOSS relies on sampling several MDPs, which is much slower than computing an expected model, and BEB relies on modifying the reward function which is not where the uncertainty lies. BOLT intends to use the best of both worlds to construct a fast BRL algorithm with optimistic transitions.

From a computational point of view, each update in BOLT requires $|\mathcal{S}|$ times more computations than each update in BEB due to the augmented action space. This implies computation times multiplied by

$|\mathcal{S}|$ when solving finite horizon problems using dynamic programming, and probably a similar increase for value iteration. However, under structured priors, not all the next states $\sigma$ must be explored, but only those which are possible transitions. Also, for a moderate number of states BOLT should require less computational effort than sample-based algorithms like BOSS, yet this strongly depends on the parameters of the sample-based algorithm.

### 6.2.1 Optimism

In BOLT the optimism is controlled by the positive parameter $\eta$ [27]. The optimistic behavior of BOLT under different $\eta$ values will strongly depend on the choice of family of distributions. However, for common priors like FDMs, it can be proved that BOLT is always optimistic with respect to the optimal Bayesian value function.

**Lemma 6.2.1** (BOLT's Optimism)**.** *Let $(s_t, \mathbf{b}_t)$ be the current belief-state from which BOLT's value iteration is applied with an horizon of $H$ and $\eta \geq H$. Let also $\mathbf{b}_t$ be a prior in the FDM family, and let $\mathbb{V}_H(s_t, \mathbf{b}_t)$ be the optimal Bayesian value function. Then, it holds that*

$$V_H^{\text{BOLT}}(s_t, \mathbf{b}_t) \geq \mathbb{V}_H(s_t, \mathbf{b}_t).$$

*Proof.* This lemma can be proved by finding an upper bound for the optimal Bayesian value function at iteration $i$. Then, by induction, the accumulated difference between BOLT's value function and this upper bound is lower bounded by a positive quantity.

Let $\mathbb{Q}_H^*(s_t, \mathbf{b}_t, a)$ be the optimal Bayesian state-action value function. This optimal Bayesian state-action value function at iteration $i$ is upper bounded as follows:

$$\begin{aligned}\mathbb{Q}_i^*(s, \mathbf{b}, a) &= \sum_{s'} \frac{\phi_{s,a}(s')}{\|\phi_{s,a}\|}(R(s,a,s') + \gamma \mathbb{V}_{i-1}^*(s', \mathbf{b}')) \\ &\leq \max_{\theta} \sum_{s'} \frac{\phi_{s,a}^{(t)}(s') + \theta(s')}{\|\phi_{s,a}^{(t)}\| + \|\theta\|}(R(s,a,s') + \gamma \mathbb{V}_{i-1}^*(s', \mathbf{b}')),\end{aligned}$$

where $\mathbf{b}$ is a *descendent* of $\mathbf{b}_t$, meaning that $\mathbf{b}$ is a belief obtained by applying $H - i$ Bayesian updates from $\mathbf{b}_t$. Analogously, $\phi$ is the distribution parameters of the current belief after applying $H - i$ Bayesian updates from $\phi^{(t)}$. Also, the $\theta$ variable is constrained to $\|\theta\| = \|\phi_{s,a}\| - \|\phi_{s,a}^{(t)}\|$, which forms the space of all the possible updates of size $H - i$. This inequality holds since there exists a vector $\theta'$ such that $\phi_{s,a} = \phi_{s,a}^{(t)} + \theta'$. Now, let

$$\begin{aligned} f(\theta) &= \sum_{s'} \frac{\phi_{s,a}^{(t)}(s') + \theta(s')}{\|\phi_{s,a}^{(t)}\| + \|\theta\|} g(s', s, a, \mathbf{b}), \text{ with} \\ g(s', s, a, \mathbf{b}) &= R(s,a,s') + \gamma \mathbb{V}_{i-1}^*(s', \mathbf{b}'). \end{aligned}$$

Please note that $f(\theta)$ is a linear function because $\|\theta\| = C_{s,a}$ is constant for a given state-action pair. Therefore, maximizing $f(\cdot)$ can be turned into maximizing $g(\cdot, s, a, \mathbf{b})$ as follows:

$$\begin{aligned}\mathbb{Q}_i^*(s, \mathbf{b}, a) &\leq \max_{\theta} f(\theta) \\ &= \frac{1}{\|\phi_{s,a}^{(t)}\| + C_{s,a}} \left( \left[ \sum_{s'} \phi_{s,a}^{(t)}(s') g(s', s, a, \mathbf{b}) \right] + \max_{\theta} \left[ \sum_{s'} \theta(s') g(s', s, a, \mathbf{b}) \right] \right) \\ &\leq \frac{1}{\|\phi_{s,a}^{(t)}\| + C_{s,a}} \left( \left[ \sum_{s'} \phi_{s,a}^{(t)}(s') g(s', s, a, \mathbf{b}) \right] + C_{s,a} \max_{\sigma} (g(\sigma, s, a, \mathbf{b})) \right). \end{aligned} \tag{6.1}$$

27. An integer or real-valued parameter depending on the family of distributions.

Consider now the difference between the state-action value function of BOLT,

$$Q_H^{\text{BOLT}}(s_t, \mathbf{b}_t, a) = \max_\sigma Q_H^{\text{BOLT}}(s_t, \mathbf{b}_t, (a, \sigma)),$$

and the optimal Bayesian value function $\mathbb{V}^*_H(s_t, \mathbf{b}_t)$. To prove the inductive step $i+1$, one must assume that the difference between the value functions at iteration $i$ is lower bounded by a positive quantity: $0 \leq \Delta_i \leq V_i^{\text{BOLT}}(s, \mathbf{b}_t) - \mathbb{V}^*_i(s, \mathbf{b})$. Then, BOLT's value at iteration $i+1$ with $\eta = H$ can be lower bounded:

$$\begin{aligned}
Q_{i+1}^{\text{BOLT}}(s, \mathbf{b}_t, a) &= \max_\sigma \sum_{s'} \frac{\phi^{(t)}_{s,a}(s') + H\delta(\sigma, s')}{\|\phi^{(t)}_{s,a}\| + H}(R(s,a,s') + \gamma V_i^{\text{BOLT}}(s', \mathbf{b}_t)) \\
&= \max_\sigma \sum_{s'} \frac{\phi^{(t)}_{s,a}(s') + H\delta(\sigma, s')}{\|\phi^{(t)}_{s,a}\| + H}(R(s,a,s') + \gamma(\mathbb{V}^*_i(s, \mathbf{b}) + \Delta_i)) \\
&\geq \gamma\Delta_i + \frac{H \max_\sigma(g(\sigma, s, a, \mathbf{b})) + \sum_{s'} \phi^{(t)}_{s,a}(s') g(s', s, a, \mathbf{b})}{\|\phi^{(t)}_{s,a}\| + H}. \qquad (6.2)
\end{aligned}$$

Using Equations 6.1 and 6.2, the difference at iteration $i+1$ is bounded by

$$\begin{aligned}
Q_{i+1}^{\text{BOLT}}(s, \mathbf{b}_t, a) - \mathbb{Q}_{i+1}(s, \mathbf{b}, a) &\geq \gamma\Delta_i + \frac{H \max_\sigma(g(\sigma, s, a, \mathbf{b})) + \sum_{s'} \phi^{(t)}_{s,a}(s') g(s', s, a, \mathbf{b})}{\|\phi^{(t)}_{s,a}\| + H} \\
&\quad - \frac{C_{s,a} \max_\sigma(g(\sigma, s, a, \mathbf{b})) + \sum_{s'} \phi^{(t)}_{s,a}(s') g(s', s, a, \mathbf{b})}{\|\phi^{(t)}_{s,a}\| + C_{s,a}} \\
&\geq \gamma\Delta_i + \frac{(H - C_{s,a}) \sum_{s'} \phi^{(t)}_{s,a}(\max_\sigma(g(\sigma, s, a, \mathbf{b})) - g(s', s, a, \mathbf{b}))}{\|\phi^{(t)}_{s,a}\| + H} \\
&\geq \gamma\Delta_i,
\end{aligned}$$

where the last step is due to all the elements in the fraction having positive values (i.e., $H > C_{s,a}$). This implies that $V_{i+1}^{\text{BOLT}}(s, \mathbf{b}_t) - \mathbb{V}^*_{i+1}(s, \mathbf{b}) \geq \gamma\Delta_i$. So, by noticing that the base step is $\Delta_0 = 0$—because $\mathbb{V}_0(s, \mathbf{b}) = V^{\text{BOLT}}(s, \mathbf{b}_t) = 0$—and applying this last equation repeatedly, the desired result is obtained by induction. □

## 6.3 Analysis of BOLT

This section provides the proofs that BOLT is PAC-BAMDP in the discounted infinite horizon case when using an FDM prior. To follow the proofs, please keep in mind that $H$ is not the horizon of the problem (which is infinite in this analysis), but the *computing horizon of the MDPs*.

Please remember that by Definition 5.3.2, the policy $\mathbf{A}_t$ generated by a PAC-BAMDP algorithm at time $t$, e.g., $\mathbf{A}_t = \operatorname{argmax}_\pi V_H^{\text{BOLT},\pi}(s_t)$, must be $\epsilon$-close to the optimal Bayesian policy with high probability and for all but a polynomial number of steps. The following theorem guarantees this condition for BOLT.

**Theorem 6.3.1** (BOLT is PAC-BAMDP)**.** *Let $\mathbf{A}_t$ denote the policy followed by* BOLT *at time $t$ with $\eta = H$. Let also $s_t$ and $\mathbf{b}_t$ be the corresponding state and belief at that time. Then, with probability at least $1-\delta$,* BOLT *is $\epsilon$-close to the optimal Bayesian policy*

$$\mathbb{V}^{\mathbf{A}_t}(s_t, \mathbf{b}_t) \geq \mathbb{V}^*(s_t, \mathbf{b}_t) - \epsilon$$

*for all but $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|\eta^2}{\epsilon^2(1-\gamma)^2}\right) = \tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^2}{\epsilon^2(1-\gamma)^2}\right)$ time steps.*

The sample complexity is written in terms of $H$ for readability, yet in the proof it will be shown that $H$ also depends on $\epsilon$ and $\gamma$. Therefore, the sample complexity bound and the optimism parameter $\eta$ depend in the end only on the desired correctness ($\epsilon$,$\delta$) and the problem characteristics ($\gamma$,$|\mathcal{S}|$,$|\mathcal{A}|$).

### 6.3.1 Mixed Value Function

To prove that BOLT is PAC-BAMDP, some preliminary concepts and results must be introduced. First, consider for the analysis that a vector of transition counters $\theta$ is maintained, even though the priors may be different from FDMs for the lemma presented in this section. Therefore, as the belief is monitored, at each step there is a set $K = \{(s,a) | \|\phi_{s,a}\| \geq m\}$ of *known state-action pairs* (Kearns and Singh, 1998), i.e., state-action pairs with "enough" evidence.

Also, to analyze an EXPLOIT-like algorithm $\mathbf{A}$ in general (like EXPLOIT, BOLT or BEB) a *mixed* value function $\tilde{\mathbb{V}}$ will be used. This $\tilde{\mathbb{V}}$ is obtained by performing an exact Bayesian update when a state-action pair is in $K$, and $\mathbf{A}$'s update when not in $K$.

Using these concepts, Lemma 5 of Kolter and Ng (2009) can be revisited for the discounted case.

**Lemma 6.3.2** (Induced Inequality Revisited)**.** *Let $\mathbb{V}^{\pi}_{H}(s_t, \mathbf{b}_t)$ be the Bayesian evaluation of a policy $\pi$, and $a = \pi(s, \mathbf{b})$ be an action from the policy. Then, the* mixed *value function can be defined as*

$$\tilde{\mathbb{V}}^{\pi}_{i}(s, \mathbf{b}) = \begin{cases} \sum_{s'} T(s,a,s',\mathbf{b})(R(s,a,s') + \gamma \tilde{\mathbb{V}}^{\pi}_{i-1}(s', \mathbf{b}')) & \text{if } (s,a) \in K, \\ \sum_{s'} \tilde{T}(s,a,s')(\tilde{R}(s,a,s') + \gamma \tilde{\mathbb{V}}^{\pi}_{i-1}(s', \mathbf{b}')) & \text{if } (s,a) \notin K, \end{cases} \tag{6.3}$$

*where $\tilde{T}$ and $\tilde{R}$ can be different from $T$ and $R$ respectively. Here, $\mathbf{b}'$ is the posterior parameter vector after the Bayes update with $(s,a,s')$. Let also $A_K$ be the event that a pair $(s,a) \notin K$ is generated for the first time when starting from state $s_t$ and following the policy $\pi$ for $H$ steps. Assuming normalized rewards for $R$ and a maximum reward $\tilde{R}_{max}$ for $\tilde{R}$, then*

$$\mathbb{V}^{\pi}_{H}(s_t, \mathbf{b}_t) \geq \tilde{\mathbb{V}}^{\pi}_{H}(s_t, \mathbf{b}_t) - \frac{(1-\gamma^H)}{(1-\gamma)} \tilde{R}_{max} Pr(A_K),$$

*where $Pr(A_K)$ is the probability of event $A_K$.*

*Proof.* This proof follows the same reasoning as Lemma 5 of (Kolter and Ng, 2009), but generalized to be applied to a *mixed* value function, and also to the discounted reward case. Let $p_i$ be a partial path generated by policy $\pi$, $p_i = \langle s_0, a_0, s_1, \ldots, a_{i-1}, s_i \rangle$, $Pr(p_i)$ the probability of obtaining that path given the prior belief $\mathbf{b}$, and $r(p_i)$ the reward for the last step of the partial path $p_i$, i.e., $(s_{i-1}, a_{i-1}, s_i)$. Furthermore, let $\tilde{Pr}(p_i)$ and $\tilde{r}(p_i)$ be respectively the product of transition probabilities along $p_i$ and the reward for the last step of the mixed value function $\tilde{\mathbb{V}}$ for a given path $p_i$. Then,

$$\begin{aligned} \tilde{\mathbb{V}}^{\pi}_{H}(s_t, \mathbf{b}_t) - \mathbb{V}^{\pi}_{H}(s_t, \mathbf{b}_t) &= \sum_{i=1}^{H-1} \gamma^i \sum_{p_i} \tilde{Pr}(p_i)\tilde{r}(p_i) - \sum_{i=1}^{H-1} \gamma^i \sum_{p_i} Pr(p_i) r(p_i) \\ &= \sum_{i=1}^{H-1} \gamma^i \left[ \sum_{p_i \in K} \Big( \tilde{Pr}(p_i)\tilde{r}(p_i) - Pr(p_i) r(p_i) \Big) + \sum_{p_i \notin K} \Big( \tilde{Pr}(p_i)\tilde{r}(p_i) - Pr(p_i) r(p_i) \Big) \right] \\ &= \sum_{i=1}^{H-1} \gamma^i \sum_{p_i \notin K} \Big( \tilde{Pr}(p_i)\tilde{r}(p_i) - Pr(p_t) r(p_i) \Big) \\ &\leq \sum_{i=1}^{H-1} \gamma^i \sum_{p_i \notin K} \tilde{Pr}(p_i)\tilde{r}(p_i) \leq \frac{(1-\gamma^H)}{(1-\gamma)} \tilde{R}_{max} Pr(A_K), \end{aligned}$$

where the sum of paths is divided in those that are always in $K$ and those that escape from $K$. If a path is in $K$, then the rewards and probabilities are equal, so they cancel out (third step), and in the last step the difference is upper bounded by the maximum possible reward value, and the definition of $Pr(A_K)$. □

### 6.3.2 BOLT is PAC-BAMDP

Let $\tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t)$ be the evaluation of BOLT's policy $\mathbf{A}_t$ using a *mixed* value function where $\tilde{R}(s,a,s') = R(s,a,s')$ the reward function, and $\tilde{T}(s,a,s') = \hat{T}(s,\alpha,s',\mathbf{b}_t) = \mathbb{E}[Pr(s'|s,a)|\mathbf{b}_t, \lambda^{\eta}_{s,a,\sigma}]$ the BOLT transition model, where $a$ and $\sigma$ are obtained from the policy $\mathbf{A}_t$. Note that, even though BOLT's update is applied, the belief is still monitored at each step as presented in Equation 6.3. Yet, the $\hat{T}$ function uses the initial belief at time $t$, and not the monitored belief $\mathbf{b}$ as in the Bayesian update.

**Lemma 6.3.3** (BOLT Mixed Bound)**.** *The difference between the optimistic value obtained by* BOLT *and the Bayesian value obtained by the* mixed *value function under the policy $\mathbf{A}_t$ generated by* BOLT *with $\eta = H$ is bounded by*

$$V_H^{\text{BOLT}}(s_t, \mathbf{b}_t) - \tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t) \leq \frac{(1-\gamma^H)\eta^2}{(1-\gamma)m}.$$

*Proof.* This lemma is proved by induction, where the inductive step $i+1$ is based on the assumption that the difference between the two evaluations at iteration $i$ is bounded by a quantity $\Delta_i$:

$$V_i^{\text{BOLT}}(s, \mathbf{b}_t) - \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s, \mathbf{b}) \leq \Delta_i.$$

Now, the maximum delta difference for $i+1$ is computed distinguishing between two cases. The first case is when $(s,a) \notin K$ (with $\alpha = \mathbf{A}_t(s,\mathbf{b})$), meaning that the probabilities and rewards for each term are the same. Formally,

$$\begin{aligned}
\Delta_{i+1}^{(\notin K)} &= V_{i+1}^{\text{BOLT}}(s, \mathbf{b}_t) - \tilde{\mathbb{V}}_{i+1}^{\mathbf{A}_t}(s, \mathbf{b}) \\
&= \sum_{s'} \hat{T}(s,\alpha,s',\mathbf{b}_t) \left( R(s,a,s') + \gamma V_i^{\text{BOLT}}(s,\mathbf{b}_t) - R(s,a,s') - \gamma \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}') \right) \\
&\leq \gamma \Delta_i \sum_{s'} \hat{T}(s,\alpha,s',\mathbf{b}_t) \\
&= \gamma \Delta_i.
\end{aligned}$$

The second case is when $(s,a) \in K$, where the probabilities differ,

$$\begin{aligned}
\Delta_{i+1}^{(\in K)} &= V_{i+1}^{\text{BOLT}}(s, \mathbf{b}_t) - \tilde{\mathbb{V}}_{i+1}^{\mathbf{A}_t}(s, \mathbf{b}) \\
&= \sum_{s'} \hat{T}(s,\alpha,s',\mathbf{b}_t)(R(s,a,s') + \gamma V_i^{\text{BOLT}}(s,\mathbf{b}_t)) - \sum_{s'} T(s,a,s',\mathbf{b})(R(s,a,s') + \gamma \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')) \\
&\leq \gamma \Delta_i + \sum_{s'} \left( \hat{T}(s,\alpha,s',\mathbf{b}_t) - T(s,a,s',\mathbf{b}) \right) \left( R(s,a,s') + \gamma \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}') \right) \\
&= \gamma \Delta_i + \sum_{s'} \left( \frac{\phi_{s,a}^{(t)}(s') + \eta\delta(\sigma,s')}{\|\phi_{s,a}^{(t)}\| + \eta} - \frac{\phi_{s,a}(s')}{\|\phi_{s,a}\|} \right) \left( R(s,a,s') + \gamma \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}') \right) \\
&\leq \gamma \Delta_i + \sum_{s'} \frac{\eta\delta(\sigma,s')}{\|\phi_{s,a}^{(t)}\| + \eta} (R(s,a,s') + \gamma \tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')) \\
&\leq \gamma \Delta_i + \frac{\eta^2}{m}.
\end{aligned}$$

Here, the third step is due to the definition of $\Delta_i$. The fifth step holds because $\phi_{s,a}^{(t)}(s') \leq \phi_{s,a}(s')$ and $\|\phi_{s,a}^{(t)}\| + H \geq \|\phi_{s,a}\|$. The last step is obtained because the maximum value of $\tilde{\mathbb{V}}$ is always less than $H = \eta$, and $\|\phi_{s,a}^{(t)}\| + H \geq m$ because the state-action pairs are in $K$.

The maximum difference at iteration $i+1$ is $\Delta_{i+1} = \max(\Delta_{i+1}^{(\notin K)}, \Delta_{i+1}^{(\in K)}) = \frac{\eta^2}{m} + \gamma\Delta_i$, and $\Delta_0 = 0$ because $V_0^{\text{BOLT}}(s,\mathbf{b}_t) = \tilde{\mathbb{V}}_0^{\mathbf{A}_t}(s,\mathbf{b}) = 0$. By applying these equations repeatedly for each iteration step, the desired result is obtained. □

*Proof of Theorem 6.3.1.* Consider the induced inequality (Lemma 6.3.2), with $\mathbf{A}_t$ the policy generated by BOLT at time $t$, and $\tilde{\mathbb{V}}$ a *mixed* value function using BOLT's update[28] when $(s,a) \notin K$ . From here, the following chain of inequalities holds:

$$\begin{aligned}
\mathbb{V}^{\mathbf{A}_t}(s_t, \mathbf{b}_t) &\geq \mathbb{V}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t) \\
&\geq \tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t) - \frac{1-\gamma^H}{1-\gamma} Pr(A_K) \\
&\geq V_H^{\text{BOLT}}(s_t, \mathbf{b}_t) - \frac{\eta^2(1-\gamma^H)}{m(1-\gamma)} - \frac{1-\gamma^H}{1-\gamma} Pr(A_K) \\
&\geq \mathbb{V}_H^{*}(s_t, \mathbf{b}_t) - \frac{\eta^2(1-\gamma^H)}{m(1-\gamma)} - \frac{1-\gamma^H}{1-\gamma} Pr(A_K) \\
&\geq \mathbb{V}^{*}(s_t, \mathbf{b}_t) - \frac{\eta^2(1-\gamma^H)}{m(1-\gamma)} - \frac{1-\gamma^H}{1-\gamma} Pr(A_K) - \frac{\gamma^H}{(1-\gamma)},
\end{aligned}$$

where the third step is due to Lemma 6.3.3 (accuracy) and the fourth step to Lemma 6.2.1 (optimism). To simplify the analysis, consider that $\frac{\gamma^H}{(1-\gamma)} = \frac{\epsilon}{2}$ and fix $m = \frac{4\eta^2}{\epsilon(1-\gamma)}$.

If $Pr(A_K) > \frac{\eta^2}{m} = \frac{\epsilon(1-\gamma)}{4}$, by the Hoeffding[29] and union bounds it can be guaranteed that $A_K$ occurs no more than

$$O\left(\frac{|\mathcal{S}||\mathcal{A}|m}{Pr(A_K)} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta}\right) = O\left(\frac{|\mathcal{S}||\mathcal{A}|\eta^2}{\epsilon^2(1-\gamma)^2} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta}\right)$$

time steps with probability $1-\delta$. By using the complexity notation that neglects logarithms, the desired sample complexity of the theorem is obtained. This bound is derived from the fact that, if $A_K$ occurs more than $|\mathcal{S}||\mathcal{A}|m$ times, then all the state-action pairs are known and the policy will never escape from $K$ anymore. Moreover, the information already encoded in the prior is not considered here, so this bound will be much tighter in practice.

For $Pr(A_K) \leq \frac{\eta^2}{m}$,

$$\begin{aligned}
\mathbb{V}^{\mathbf{A}_t}(s_t, \mathbf{b}_t) &\geq \mathbb{V}^{*}(s_t, \mathbf{b}_t) - \frac{\epsilon(1-\gamma^H)}{4} - \frac{\epsilon(1-\gamma^H)}{4} - \frac{\epsilon}{2} \\
&\geq \mathbb{V}^{*}(s_t, \mathbf{b}_t) - \frac{\epsilon}{4} - \frac{\epsilon}{4} - \frac{\epsilon}{2} \\
&= \mathbb{V}^{*}(s_t, \mathbf{b}_t) - \epsilon
\end{aligned}$$

which verifies the proposed theorem. □

A similar PAC-BAMDP proof for BEB can be found in Appendix A.2, revisiting the analysis provided in Kolter and Ng (2009) which was constrained to finite horizon domains with an artificial stopping condition for the Bayes update. BOLT's sample complexity bound $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^2}{\epsilon^2(1-\gamma)^2}\right)$ can be compared with the theoretical results for BEB. Following Kolter and Ng (2009) (or Appendix A.2), the bound for BEB holds for $\beta \geq 2H^2$ . The sample complexity of BEB is bounded by $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^4}{\epsilon^2(1-\gamma)^2}\right)$ non $\epsilon$-close steps, which is a much looser bound than for BOLT. This does not forcedly mean that BOLT will behave better than BEB in practice, because the bounds used in the proofs are loose enough to expect the optimism property to hold for much smaller values of $\beta$ and $\eta$. However, there is no near-optimality guarantees for these small parameter values, and therefore nothing can be said about the behavior of BEB and BOLT in terms of sample complexity.

28. The $\tilde{R}_{max}$ constant is 1 for BOLT because $\tilde{R} = R$.

29. Even though the Hoeffding bound assumes that samples are independent, which is trivially not the case in MDPs, it upper bounds the case where samples are dependent. Recent results show that tighter bounds can be achieved with a more elaborate analysis like in (Szita and Szepesvári, 2010).

## 6.4 Experiments

To illustrate the characteristics of BOLT, an experimental study was performed over a number of domains. The problems of this section were specifically chosen for their exploration complexity, meaning that they are strongly structured problems where intelligent exploration is needed to solve them. In each domain, different parameters were tried for BOLT and BEB, as well as an $\varepsilon$-greedy variant of EXPLOIT, where there is an $\varepsilon$ probability of choosing a random action rather than following the policy of the expected model. However, random exploration is only suitable for weakly structured problems (see Section 5.2.4), so unsurprisingly, for all the presented problems, plain EXPLOIT ($\varepsilon = 0.0$) outperforms the $\varepsilon$-greedy variant.

Please recall that the theoretical values for parameters $\beta$ and $\eta$—that ensure optimism—depend on the horizon $H$ of the MDPs solved at each time step. In these experiments, asynchronous VI was used instead of using synchronous VI up to an horizon $H$, stopping when $\|V_{i+1} - V_i\|_\infty < \epsilon$. This corresponds to an upper bound horizon $H_{max} \approx \log(\epsilon(1-\gamma))/\log(\gamma)$ (Kearns and Singh, 1998). Moreover, the final value function at time $t$ was used as the initial value function at time $t+1$ to speed up computations. For solving these infinite MDPs, $\gamma = 0.95$ and $\epsilon = 0.01$ were used, but be aware that the performance criterion used here is the empirical estimate of the *undiscounted* total reward, following (Poupart et al., 2006; Asmuth et al., 2009).

### 6.4.1 The Chain Problem

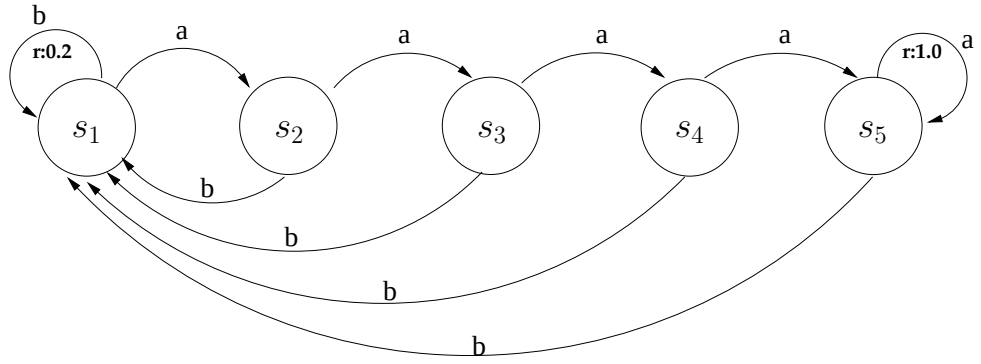



Figure 6.3: The 5-state Chain Problem (without probabilities). By default, the action $a$ moves forward in the chain while the action $b$ returns to the starting state $s_1$. However, the agent may "slip" with some probability, and perform the opposite action as intended.

In the 5-state chain problem (Strens, 2000; Poupart et al., 2006), the default behavior is that every state leads to state $s_1$ by taking action $b$ and every state $s_i$ leads to the next state $s_{i+1}$ with action $a$, except $s_5$ that leads to itself, as shown in Figure 6.3. At each step, the agent may "slip" with probability $p$, performing the opposite action as intended. Staying in $s_5$ gives a reward of 1.0 while coming back to $s_1$ gives a reward of 0.2. All other rewards are 0. The priors used for these problems are **Full** (FDM), **Tied**, where the probability $p$ is factored for all transitions, and **Semi**, where each action has an independent factored probability.

Table 6.1 shows that BEB outperforms other algorithms with a tuned up $\beta$ value for the FDM prior, as already shown by Kolter and Ng (2009). However, for a large value of $\beta$, this performance decreases dramatically. BOLT on the other hand produces results comparable with BOSS for a tuned parameter, but does not decrease too much for a large value of $\eta$. This value $\eta = H_{max} \approx \log(\epsilon(1-\gamma))/\log(\gamma) \approx 150$ corresponds to the theoretical upper bound that ensures optimism. Unsurprisingly, the results of BEB and BOLT with informative priors are not much different from other techniques, because the problem degenerates into an easily solvable problem. Nevertheless, BOLT achieves good results for a large $\eta$, in

| **Algorithm** | Tied | Semi | Full |
|---|---|---|---|
| EXPLOIT ($\varepsilon = 0$) | **366.1** | 354.9 | 230.2 |
| BEB ($\beta = 1$) | **365.9** | **362.5** | **343.0** |
| BEB ($\beta = 150$) | **366.5** | 297.5 | 165.2 |
| BOLT ($\eta = 7$) | **367.9** | **367.0** | 289.6 |
| BOLT ($\eta = 150$) | **366.6** | 358.3 | 278.7 |
| BEETLE * | **365.0** | **364.8** | 175.4 |
| BOSS * | **365.7** | **365.1** | 300.3 |

Table 6.1: Chain problem results for different priors. Average total reward over 500 trials for an horizon of 1000 with $p = 0.2$. The results with * come directly from (Poupart et al., 2006) and (Asmuth et al., 2009).

contrast to BEB that fails to provide a competitive result for the Semi prior with large $\beta$.

This variability in the results depending on the parameters raises the question of the sensitivity to parameter tuning. In an RL domain, one usually cannot tune the algorithm parameters for each problem, because the whole model of the problem is unknown. Therefore, a good RL algorithm should perform well for different problems without modifying its parameters.

Figure 6.4 shows how BEB and BOLT behave for different parameters using a **Full** prior. The ranges of values for $\beta$ and $\eta$ were initially between 0 and the theoretical parameter value that ensures optimism for each algorithm. However, for $\beta$ the maximum value corresponds to $2H^2 \approx 2 \cdot 150^2 = 45000$, yet the performance is almost constant after $\beta = 50$ as shown in Figure 6.4. Therefore, a range between $[0, 100]$ is sufficient to analyze the parameter evolution in BEB. For illustrative purposes the results of BOLT were also constrained to the range $[0, 100]$.

In the low resolution analysis BEB's performance decays very fast, while BOLT also tends to decrease, but its results are still competitive. In the high resolution results, BEB has a global maximum near 1 and then falls very quickly, while BOLT maintains a similar behavior as in the low resolution experiment.

Experiments for other values of the slip probability $p$ were also conducted. When $p$ is near 0.5, BEB and BOLT behave almost identically. Yet, when $p$ is near 0, BEB presents an even worse decay than for $p = 0.2$, while BOLT presents an almost constant good performance through all parameters.

### 6.4.2 The Paint/Polish Problem

Another illustrative example is the Paint/Polish problem (see Figure 6.5) where the objective is to deliver several polished and painted objects without a scratch, using several stochastic actions with unknown probabilities. The full description of the problem can be found in (Walsh et al., 2009). Here, the possible outcomes of each action are given to the agent, but the probabilities of each outcome are not. A structured prior that encodes part of this information was tested. Instead of using one Dirichlet distribution for each state-action pair, the structured prior factors the state in the three properties of an object (painted, polished and scratched). Each action-property pair can be represented by a Dirichlet distribution over the same property[30]. Under this representation, some transitions are structurally deterministic, such as painting an already painted object, so there is no need to learn them. This representation does not use all the information given by the STRIPS description, but significantly reduces the learning space compared to plain FDMs.

The results are summarized in Figure 6.6, using both high and low resolution analyses. Again, the decay of BEB is much stronger than for BOLT, but in contrast to the Chain problem, the best parameter of BOLT beats the best parameter of BEB. However, BOLT exhibits a less robust performance than in the Chain problem, and the best result for BOLT is only marginally better than the result for EXPLOIT. This means that this problem requires almost no optimism, so over-optimistic parameters lead to performing unnecessary exploration. Unfortunately, it is difficult to know a priori how much optimism is necessary

30. In fact they are Beta distributions because the property is a binary variable.

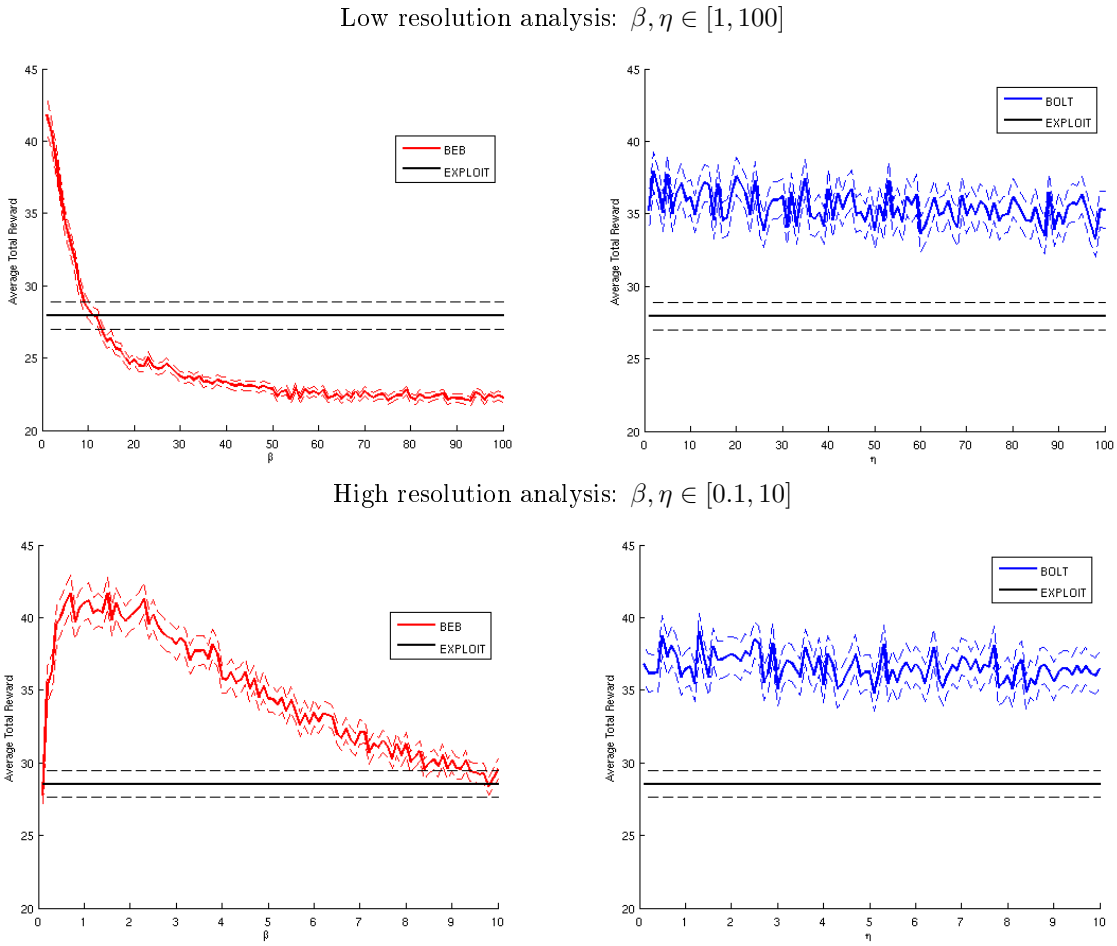



Figure 6.4: Chain problem parameter tuning results. Average total reward over 300 trials for an horizon of 150, and for $\beta$ and $\eta$ parameters between 1 and 100 (top), and between 0.1 and 10 (bottom). As a reference, the value obtained by EXPLOIT is also plotted. All results are shown with a 95% confidence interval.

*paint*(X): reward = −1
**PRE:** none
**ADD**: *Painted*(X) **DEL:** none **0.6**
**ADD**: *Painted*(X), *Scratched*(X) **DEL:** none **0.3**
**ADD**: none **DEL:** none **0.1**

*polish*(X): reward = −1
**PRE:** none
**ADD**: none **DEL:** *Painted*(X) **0.2**
**ADD**: none **DEL:** *Scratched*(X) **0.2**
**ADD**: *Polished*(X) **DEL:** *Painted*(X), *Scratched*(X) **0.3**
**ADD**: *Polished*(X) **DEL:** *Painted*(X) **0.2**
**ADD**: none **DEL:** none **0.1**

*shortcut*(X): reward = −1
**PRE:** none
**ADD**: *Painted*(X), *Polished*(X) **DEL:** none **0.05**
**ADD**: none **DEL:** none **0.95**

*done*(X): reward = 10
**PRE:** *Painted*(X), *Polished*(X),¬*Scratched*(X)
**ADD**: *Finished*(X) **DEL:** none **1.0**

Figure 6.5: Stochastic STRIPS definition of the Paint/Polish Problem. For a Stochastic STRIPS description please refer to (Younes et al., 2005).

for a given problem, which reinforces the idea of using a more conservative algorithm in the parameter space like BOLT rather than BEB.

This experiment was also conducted with a plain FDM prior, obtaining similar results as for the Chain problem. Unsurprisingly, using a structured prior provides better results than using plain FDMs. However, the high impact of being over-optimistic does not apply to FDMs, mainly because the learning phase is much shorter using a structured prior.

### 6.4.3 The Marble Maze Problem

The last example is the Marble Maze problem shown in Figure 6.7 (Asmuth et al., 2009). This problem consists in a discrete version of the puzzle game of guiding a marble to a goal by spinning the board in the correct directions, without falling into traps. For this problem it is assumed that the map is known, but the transition probabilities of each action are unknown. The idea is to arrive to the goal while learning these probabilities. In (Asmuth et al., 2009) the (hidden) probability of acting as intended is 0.8, while otherwise the marble randomly moves to one of the two perpendicular directions. Therefore, depending on the wall configuration, the transition probabilities from a cell vary, but they are the same for the group of cells with the same wall configuration. This defines clusters of cells that can be automatically learned as in (Asmuth et al., 2009), or explicitly encoded in a structured prior as the experiments of this section do.

Due to the explicit encoding of the 16 possible clusters in the prior, the exploration requirements of this problem are low. That is why EXPLOIT provides very good solutions for this problem, and BOLT provides similar results with several different parameters as shown in Figure 6.8. In contrast, for all the tested $\beta$ parameters, BEB behaves much worse than EXPLOIT. For example, for the best $\eta(= 2.0)$ BOLT scores $-0.445$, for the best $\beta(= 0.9)$ BEB scores $-2.127$, while EXPLOIT scores $-0.590$.

The increasing confidence interval for BOLT as $\eta$ increases, shown in Figure 6.8, confirms that more optimism produces more exploration and, with it, more variability in the results. However, the confidence interval remains fairly constant when $\eta \geq 7$, suggesting that the variability of the results is bounded (and confirmed by the results of all the previous problems). On the other hand, BEB seems constant from

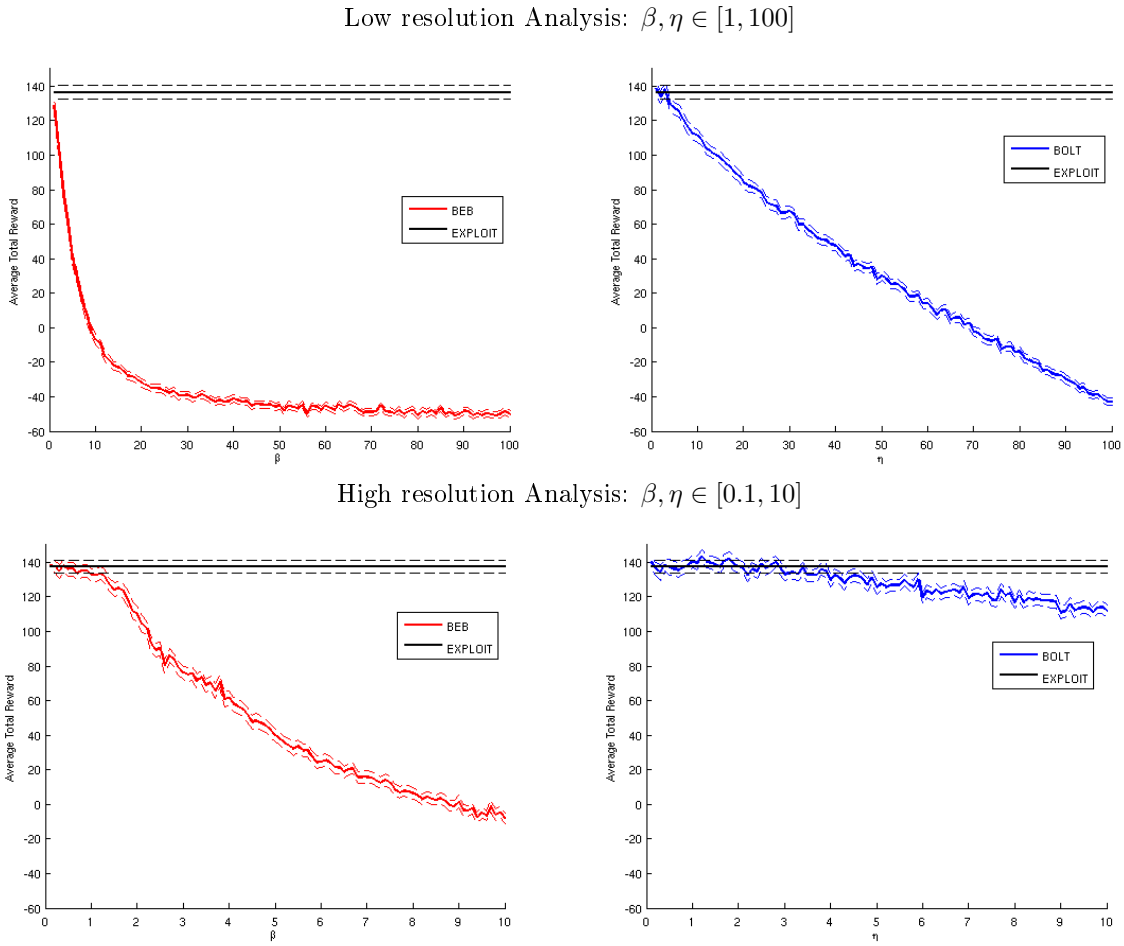



Figure 6.6: Paint/Polish Problem Parameter Tuning Results. Average total reward over 300 trials for an horizon of 150, for $\beta$ and $\eta$ parameters between 1 and 100 (top) and between 0.1 and 10 (bottom) using a structured prior. As a reference, the value obtained by EXPLOIT is also plotted. All results are shown with a 95% confidence interval.

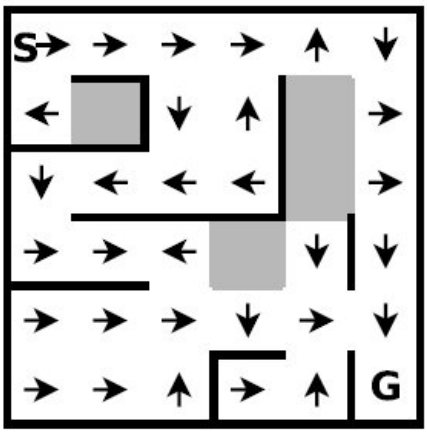



Figure 6.7: The Marble Maze Problem. The arrows correspond to an optimal policy when the transition probabilities are known.

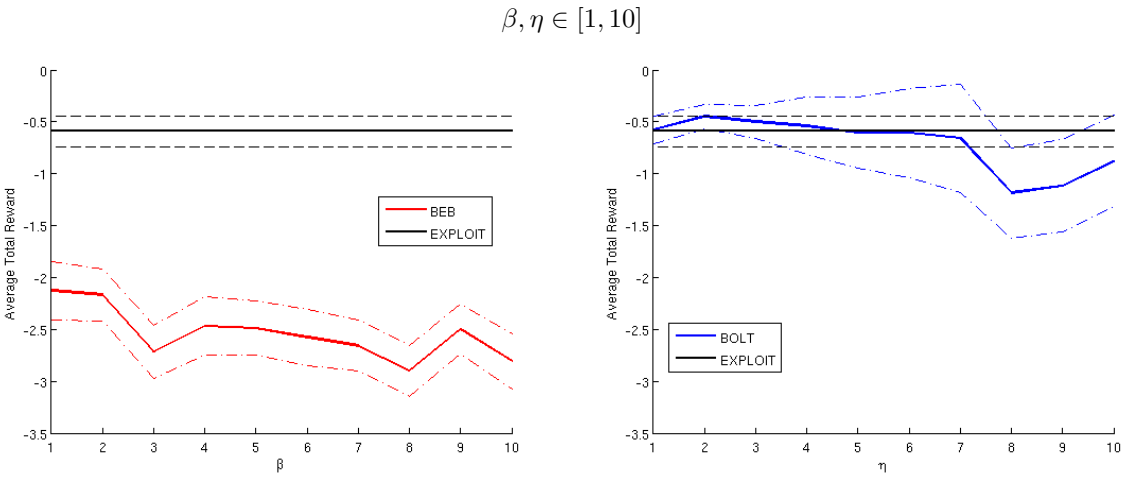



Figure 6.8: Marble Maze Problem Results. Average total reward over 100 trials for an horizon of 100, for $\beta$ and $\eta$ parameters between 1 and 10 using a structured cluster prior. As a reference, the value obtained by EXPLOIT is also plotted. All results are shown with a 95% confidence interval.

the start, but a more detailed analysis for $\beta \in [0, 1]$ should draw a continuous curve between EXPLOIT's variability ($\beta = 0$) and BEB's variability for the first parameter value ($\beta = 1$). Nevertheless, more experiments must be conducted to properly analyze and conclude about this matter.

## 6.5 Conclusion

This section has proposed a novel, simple and robust algorithm called BOLT that uses an optimistic boost of the Bayes update, which is thus optimistic *about* the uncertainty rather than just *in the face* of uncertainty.

### 6.5.1 Summary of Contributions

The theoretical results are that BOLT is strictly optimistic for certain $\eta$ parameters, and is PAC-BAMDP under this same condition for FDM priors. This chapter has also proven that tighter sample complexity bounds can be found for BOLT than for BEB.

Experiments show that BOLT is more efficient than BEB when using the theoretically derived parameters in all benchmarks. Even though it is hard to know a priori which algorithm will perform better for a specific problem with a specific prior and given certain parameters, BOLT generalizes well for a larger set of parameters, mainly because the optimism is bounded by the probability laws and not by a free parameter as in BEB.

Most of these results were already published in (Araya-López et al., 2012a) and (Araya-López et al., 2012b), yet this chapter includes a more detailed explanation of the BOLT algorithm and its motivation.

### 6.5.2 Future Work

Future work includes using a dynamic $\eta$ bonus for BOLT, by decreasing the Boost size as the iteration goes deeper, which should be particularly appropriate with finite horizons.

Also, exploring general proofs to guarantee the PAC-BAMDP property for a broader family of priors than FDMs is a very important extension to ensure BOLT's properties in real-world domains, where structured priors are the rule rather than the exception.

In the Paint/Polish problem, a prior that uses only a part of the information available in the STRIP description was selected, because modelling each outcome as Dirichlet distribution leads to ambiguity. For example, the $paint(X)$ action may paint, paint and scratch or do nothing to an object. If this action is applied to an already painted and unscratched object, and the result is again a painted and unscratched object, the agent cannot know if nothing was done to the object, or if the object was repainted. In order to exploit all the information contained by the STRIPS description, a particularly interesting research direction is to extend this work to Hyperdirichlet distributions (see Section 5.2.1) that support this type of ambiguous updates. Another option is to model the Paint/Polish problem as a partially observable RL problem (PORL), meaning that the model includes an observation function similarly to POMDPs. Under this modelling choice, the state would include not only the properties of the object, but also the outcome of the action. However, the agent only observes the properties of the object through a known observation function. Whether the observation function is given or not for a PORL problem, RL algorithms need to be modified to support this kind of problems. Therefore, combining BOLT with recent advances in partially observable BRL (Ross et al., 2011) will allow efficiently exploiting the structure of some problems (like Paint/Polish) and solving new challenging problems with unknown observation functions.

Another very interesting question is how to extend BOLT's idea to continuous state and action spaces, which will require not only solving approximate MDPs, but also defining how to perform the Bayes Boost in these domains. Under continuous spaces the possible Boost directions are infinite, so testing each direction (as BOLT suggest) is not possible. However, local search optimization techniques, such as gradient-based ones, can be used to find the best optimization direction, and therefore find optimistic local transitions for the AAMDP.

Under discrete but large state spaces, BOLT could benefit from sampling states rather than computing all the possible boosting directions, allowing to scale up to real-world problems. However, sampling

states introduces new challenges for the theoretical guarantees that have to be explored, because such an algorithm will not be able to verify the strictly optimistic property. However, maybe a probably optimistic property can be verified as in BOSS. Regardless this theoretical entanglement, experiments for a sampled-based version of BOLT are needed to explore the scalability of this approach to real-world applications.

# Chapter 7

# Optimally Learning Transition Models

## Contents



*"Man can learn nothing except by going from the known to the unknown."*
— Claude Bernard

Similarly to the information-gathering problems of Section 4.1, learning the transition model of an MDP independently of the utility function—if it exists at all—can be a very useful task in some domains. For example, this can be used for learning the transition model in a batch process, where in a first stage the objective is to choose the good actions for optimizing the information gathering process, and afterwards, in a second stage, the objective is to earn rewards (Şimşek and Barto, 2006). Moreover, there are some cases where the reward function is unaccessible or not defined, such as models for simulations or model refinement, where the objective is only to learn a good model, no matter which task the model will be used for in the future.

This chapter is focused on learning the hidden model of an MDP by observing state transitions online, under an active exploration strategy. This problem is a decision-making problem, where the best strategy must be selected in order to optimize the learning process. To do so, this chapter proposes to combine the results of Chapters 4, 5 and 6, by using the Bayesian RL machinery with information-gathering objectives.

Specifically, the learning problem can be cast as an information-gathering problem using rewards that depend on the belief over the model. To do this, a performance criterion must be defined to measure the quality of the distributions produced by different policies. Using this criterion, the information-based rewards are derived and used to find smart exploration policies through Bayesian RL. Due to the intractability of computing the optimal Bayesian value function (see Section 5.2), the best that can be done is to near-optimally solve this problem using approximate BRL algorithms.

## 7.1 MDP Model Learning

Learning stochastic MDP models is an easy task when an exploration policy is given. In this case, the history of transitions can be seen as the data, and the problem of finding the optimal parameters for the selected distribution over the models can be solved by using likelihood maximization like presented as Chapter 2.

In contrast, in this chapter the objective is to actively learning the transition model, which raises a sequential control problem. This amounts to finding a policy that explores optimally an MDP in order to acquire the best distribution over the possible models. Therefore, in *MDP Model Learning* (MML) the objective is to find a policy for optimally exploring the dynamics of an unknown MDP, where typically, a sequence of actions is needed to reach specific states from which new informative samples can be obtained. This is a complex and interesting problem since one has to reason on an imperfect model in order to improve that same model. However, this problem has been only marginally explored to the author's knowledge.

One of the few works in this domain is about learning Dynamic Bayesian Networks (DBNs) for representing factored MDPs (Jonsson and Barto, 2007), where the authors conclude that actively learning the transition models is a challenging problem and new techniques are needed to address this problem properly. Also, it is important to notice that MML differs from classical active supervised learning, because the dynamics of the MDP prevent from querying any sample at any time like in (Kapoor et al., 2007).

### 7.1.1 MML as a Bayesian RL Problem

Please note that for a given policy, the learning process is straightforward using the Bayesian setting with FDMs, because the likelihood maximization for the joint Dirichlet distribution corresponds to the sequential Bayes update of the belief $\mathbf{b}$ of Section 5.2. Therefore, the optimal policy depends on the criterion used to compare two joint Dirichlet distributions produced by different policies.

To find the optimal policy, the active learning task can be cast as a BRL problem with *information-based rewards*, where these rewards are derived from the performance criterion. In other words, the classical definition of BRL is extended to support rewards that depend on the belief $\mathbf{b}$. For this scenario the Bellman equation takes the following form:

$$V_H(\mathbf{b}, s) = \max_a \left[ \sum_{s'} Pr(S' = s' | S = s, A = a, B = \mathbf{b})(\rho(\mathbf{b}, a, \mathbf{b}') + V_{H-1}(\mathbf{b}', s')) \right], \tag{7.1}$$

with $\mathbf{b}'$ the posterior belief after the Bayes update with $(s, a, s')$, and $\rho(\mathbf{b}, a, \mathbf{b}') = \rho(s, a, s', \mathbf{b})$ the immediate information-based reward. Within this formulation the problem of actively learning MDP models can be optimally solved using a dynamic programming technique. Yet, as in normal BRL, computing the exact value function is intractable because of the large branching factor of the tree expansion, so approximation techniques are needed to address this problem.

### 7.1.2 Derived Rewards

In order to define the information-based rewards needed for Equation 7.1, the analytical expressions of the performance criterion will be used to derive analytical expressions for immediate reward functions. If the problem has a finite horizon, the performance criterion could be used directly as a reward in the final step, whereas the rewards would be zero for the rest of the steps. Yet, this type of reward functions forces to expand the complete tree in order to obtain non-zero rewards, which turns out to be extremely expensive for large horizons.

An alternative is to define substantial immediate rewards at each step by using reward shaping. As rewards are defined over a transition $(s, a, s')$ and the current belief-parameter $\phi$, the Bayesian update can be used for computing the performance difference between the current belief and the posterior belief. This is a standard reward shaping technique, which allows decomposing a potential function—here the performance criterion—in immediate rewards for each step, with the property of preserving the optimality of the generated policy (Ng et al., 1999).

**Definition 7.1.1** (Derived Rewards)**.** *Let $D(\mathbf{b}_t, \mathbf{b}_0)$ be a distance between the initial prior and the posterior parameters after $t$ Bayes updates such that maximizing this distance amounts to maximizing the gain of information. From here, the derived reward can be defined as follows,*

$$\rho(s, a, s', \mathbf{b}_t) = D(\mathbf{b}_{t+1}, \mathbf{b}_0) - D(\mathbf{b}_t, \mathbf{b}_0),$$

*where $\mathbf{b}_{t+1}$ is the set of parameters after the transition $(s, a, s')$ from $\mathbf{b}_t$.*

In some favorable cases, the performance criterion complies with the triangular *equality*, meaning that the derived rewards can be simply computed as

$$\rho(s, a, s', \mathbf{b}_t) = D(\mathbf{b}_t, \mathbf{b}_{t+1}), \tag{7.2}$$

removing the dependency from the initial prior.

Please recall that the Bayes update only modifies one state-action pair per update under the FDM prior, meaning that only one state-action pair component of the distribution will change per update. This provides important simplifications when computing the difference of value after a transition.

### 7.1.3 Performance Criteria

Assuming that the real model is unknown, comparing two distributions produced by different policies is not a trivial task, because there is no *a priori* best criterion. Therefore, three information-based criteria were selected under the constraint that they can be computed analytically: the *variance difference*, the *entropy difference* and the *Bhattacharyya distance*.

#### Variance Difference

The first criterion is based on the simple intuition that low-variance distributions are more informative than high-variance ones. Unfortunately, the variance is not a one-dimensional metric for the multivariate distribution over the models: it is a heavily sparse matrix of size $|\mathcal{S}|^2|\mathcal{A}| \times |\mathcal{S}|^2|\mathcal{A}|$. Similarly to A-optimality experimental design (Fedorov, 1972), here the sum of marginal variances (the trace of the matrix) is considered as a sufficient metric.

Using FDMs, the variance of the $i$-th element of a Dirichlet distribution is given by

$$\sigma_i^2(\phi) = \frac{\phi(i)(\|\phi\|_1 - \phi(i))}{\|\phi\|_1^2(\|\phi\|_1 + 1)}. \tag{7.3}$$

Then, the *variance difference* is defined as follows,

$$D_V(\mathbf{b}_t, \mathbf{b}_0) = \sum_{s,a} \sum_{s'} (\sigma_{s'}^2(\phi_{s,a}^{(0)}) - \sigma_{s'}^2(\phi_{s,a}^{(t)})).$$

#### Entropy Difference

As presented in Chapter 4, computing the entropy of beliefs is a natural way of quantifying the information of a distribution.

The entropy of a multivariate random variable distributed as a Dirichlet distribution with parameters $\phi$ is given by

$$H(\phi) = \log(B(\phi)) + (\|\phi\|_1 - N)\psi(\|\phi\|_1) - \sum_{j=1}^{N} (\phi_j - 1)\psi(\phi_j),$$

where $B(\cdot)$ is the generalized beta function, $N$ is the dimensionality of the vector $\phi$, and $\psi(\cdot)$ is the digamma function[31]. For the case of the FDM prior over the possible models, the independence between state-action pairs can be used to compute each state-action entropy separately, as stated in the following proposition.

31. The digamma function corresponds to the derivate of the gamma function.

**Proposition 7.1.1.** *The entropy of a FDM model distribution $T$ with parameters $\phi$ is the sum of the entropy for each state-action pair variable: $H(\phi) = \sum_{s,a} H(\phi_{s,a})$.*

*Proof.*

$$
\begin{aligned}
H(\phi) &= -\int Pr(M = \mu|\phi) \log(Pr(M = \mu|\phi)) d\mu \\
&= -\underbrace{\int_{\Delta}\int_{\Delta}\cdots\int_{\Delta}}_{|\mathcal{S}|\times|\mathcal{A}|} \left(\prod_{s,a} Pr(M_{s,a} = \mu_{s,a}|\phi_{s,a})\right) \left(\sum_{\hat{s},\hat{a}} \log(Pr(M_{\hat{s},\hat{a}} = \mu_{\hat{s},\hat{a}}|\phi_{\hat{s},\hat{a}}))\right) \left(\prod_{\bar{s},\bar{a}} d\mu_{\bar{s},\bar{a}}\right) \\
&= -\sum_{\hat{s},\hat{a}} \int_{\Delta} Pr(M_{\hat{s},\hat{a}} = \mu_{\hat{s},\hat{a}}|\phi_{\hat{s},\hat{a}}) \log(Pr(M_{\hat{s},\hat{a}} = \mu_{\hat{s},\hat{a}}|\phi_{\hat{s},\hat{a}})) d\mu_{\hat{s},\hat{a}} \\
&\quad \underbrace{\int_{\Delta}\int_{\Delta}\cdots\int_{\Delta}}_{|\mathcal{S}|\times|\mathcal{A}|-1} \prod_{s\neq\hat{s},a\neq\hat{a}} [Pr(M_{s,a} = \mu_{s,a}|\phi_{s,a}) d\mu_{s,a}] \\
&= -\sum_{\hat{s},\hat{a}} \int_{\Delta} Pr(M_{\hat{s},\hat{a}} = \mu_{\hat{s},\hat{a}}|\phi_{\hat{s},\hat{a}}) \log(Pr(M_{\hat{s},\hat{a}} = \mu_{\hat{s},\hat{a}}|\phi_{\hat{s},\hat{a}})) d\mu_{\hat{s},\hat{a}} \\
&= \sum_{s,a} H(\phi_{s,a}).
\end{aligned}
$$

□

Then, the *entropy difference* between a belief $\mathbf{b}_t$ parameterized by $\phi^{(t)}$ after applying $t$ Bayes updates, and its initial prior $\phi^{(0)}$ is defined as

$$D_H(\mathbf{b}_t, \mathbf{b}_0) = H(\phi^{(0)}) - H(\phi^{(t)}).$$

### Bhattacharyya Distance

The two measures described above attempt to quantify how much information the distribution contains. Following Chapter 4, one may think automatically about the KL-divergence or the Fisher Information Matrix as semi-measures for relative information. However, as stated in (Rauber et al., 2008), the last two are inappropriate for Dirichlet distributions, because an analytical solution for the integral does not exist or because the non-existence for some specific values.

The Chernoff distance [32] is a means to compare two distributions with parameters $\phi$ and $\phi'$, based on the Chernoff information. It is defined as

$$C_\lambda(\phi, \phi') = -\log\left(\int_{\mathcal{X}} Pr(X = x|\phi)^\lambda Pr(X = x|\phi')^{1-\lambda} dx\right),$$

with $\lambda \in [0, 1]$.

In the case of Dirichlet distributions, the Chernoff distance takes the form

$$C_\lambda(\phi, \phi') = -\log\left(\frac{B(\lambda\phi + (1-\lambda)\phi')}{B(\phi)^\lambda B(\phi')^{1-\lambda}}\right).$$

This equation can be computed using sums of $\log\Gamma(\cdot)$ functions as presented in (Rauber et al., 2008). However, the selection of the $\lambda$ parameter has an important impact on the computation of the distances for Dirichlet distributions, because it will produce non-integer $\Gamma(\cdot)$ evaluations, which are expensive to approximate. Fortunately, if $\lambda = 1/2$ then the Chernoff distance becomes the *Bhattacharyya distance* [33],

32. The Chernoff distance is not a proper distance, because it does not always comply with the triangle inequality.

33. The Chernoff "distance" with $\lambda = 1/2$ is actually symmetric, meaning that the Bhattacharyya distance is a proper distance measure.

with the advantage that $\Gamma(n+1/2)$ has a known analytical form. As for the entropy, note that the Chernoff distance (and by extension the Bhattacharyya distance) can be computed also for each state-action pair independently for FDMs.

**Proposition 7.1.2.** *The Chernoff distance between two FDM beliefs* $\mathbf{b}$ *and* $\mathbf{b}'$ *with parameters* $\phi$ *and* $\phi'$*, is the sum of the Chernoff distances for each state-action pair variable:* $C_\lambda(\phi, \phi') = \sum_{s,a} C_\lambda(\phi_{s,a}, \phi'_{s,a})$.

*Proof.*

$$\begin{aligned}
C_\lambda(\phi, \phi') &= -\log\left(\int Pr(\mu|\phi)^\lambda Pr(\mu|\phi')^{1-\lambda} d\mu\right) \\
&= -\log\left(\underbrace{\int_\Delta \int_\Delta \cdots \int_\Delta}_{|\mathcal{S}|\times|\mathcal{A}|} \prod_{s,a} \left[Pr(\mu_{s,a}|\phi_{s,a})^\lambda Pr(\mu_{s,a}|\phi'_{s,a})^{1-\lambda} d\mu_{s,a}\right]\right) \\
&= -\log\left(\prod_{s,a}\left[\int_\Delta Pr(\mu_{s,a}|\phi_{s,a})^\lambda Pr(\mu_{s,a}|\phi'_{s,a})^{1-\lambda} d\mu_{s,a}\right]\right) \\
&= \sum_{s,a} C_\lambda(\phi_{s,a}, \phi'_{s,a}).
\end{aligned}$$

□

Therefore, the *Bhattacharyya distance* between the belief $\mathbf{b}_t$ parametrized by $\phi_t$ after applying $t$ Bayes updates and its initial prior $\mathbf{b}_0$, can be defined as

$$D_B(\mathbf{b}_t, \mathbf{b}_0) = \sum_{s,a} C_{1/2}(\phi^{(t)}_{s,a}, \phi^{(0)}_{s,a}).$$

This last performance criterion can be interpreted as a metric of the *fairness of knowledge* of the model. If the model knowledge is biased to certain parts of it, then the Bhattacharyya distance of the rest of the model will be low, and therefore it will produce a low performance. In other words, this criterion prefers distributions with information on the whole model rather than specific parts of it.

### 7.1.4 From Criteria to Rewards

The first two criteria presented in Section 7.1.3 comply with the triangular *equality*, so Equation 7.2 can be used to compute their respective derived reward functions. Even though this is not always true for the *Bhattacharyya* distance, Equation 7.2 will also be used for simplicity, knowing that optimality is not preserved for this specific case.

Therefore, after some trivial algebra (detailed in Appendix A.3), the *variance*, *entropy* and *Bhattacharyya* instant rewards can be obtained. To present these expressions, the helper variables $x = \phi_{s,a}(s')$ and $y = \|\phi_{s,a}\|_1$ are used for all the rewards, and also $z = \|\phi_{s,a}\|_2^2$ for the variance reward. Thus:

$$\begin{aligned}
\rho_V(s, a, s', \mathbf{b}) &= \frac{1}{y+1} - \frac{z}{y^2(y+1)} - \frac{y^2 + 2y - z - 2x}{(y+1)^2(y+2)}, \\
\rho_H(s, a, s', \mathbf{b}) &= \log\left(\frac{y}{x}\right) + \frac{|\mathcal{S}|}{y} - \sum_{j=x}^{y} \frac{1}{j}, \\
\rho_B(s, a, s', \mathbf{b}) &= \log\left(\frac{\Gamma(x)\sqrt{x}}{\Gamma(x+1/2)}\right) - \log\left(\frac{\Gamma(y)\sqrt{y}}{\Gamma(y+1/2)}\right).
\end{aligned}$$

Also, a very simple information-based reward function will be used in the experiments, motivated by the exploration bonus of BEB (Kolter and Ng, 2009). The *state-action count* reward only focuses on the

difference of information from one state-action pair to another, and is simply defined as

$$\rho_S(s, a, s', \mathbf{b}) = \frac{1}{\|\phi_{s,a}\|_1} = \frac{1}{y}.$$

This reward is easier to compute than the other three, but preserves the same principle of quantifying the information gain of a Bayes update. In fact, this reward function optimizes the performance criterion

$$D_S(\mathbf{b}_t, \mathbf{b}_0) = \sum_{s,a} (\psi(\|\phi_{s,a}^{(t)}\|) - \psi(\|\phi_{s,a}^{(0)}\|)),$$

which turns out to be an expression appearing in both the *entropy difference* and the *Bhattacharyya distance*.

A key property of the rewards presented above is that they tend to zero for an infinite belief evolution, meaning that there is no more to learn at this stage. Specifically, the last two rewards $\rho_B$ and $\rho_S$ are always positive and decreasing functions with the belief evolution, while the first two $\rho_V$ and $\rho_H$ can have negative values, but their absolute values are always decreasing, all of them converging to zero in the limit.

## 7.2 Solving BRL with Information-based Rewards

It is clear that the algorithms introduced in Section 5.2 will require some modifications to work with information-based rewards. For example, BEETLE uses a polynomial representation of the value function, where rewards are scalar factors multiplying monomials. In this setup, rewards will be functions multiplying monomials, which makes off-line planning even more complex.

Please recall the EXPLOIT algorithm from Section 5.2.4. This algorithm, which is one of the simplest online algorithms for BRL, consists in solving the MDP corresponding to the current expected model or, in other words, iterating over the Bayesian value function without performing the Bayes update.

For the information-based reward scenario, EXPLOIT value function takes the form

$$V_H(\mathbf{b}, s) = \max_a \left[ \sum_{s'} Pr(S' = s' | S = s, A = a, \mathbf{b}) \left( \rho(s, a, s', \mathbf{b}) + V_{H-1}(\mathbf{b}, s') \right) \right],$$

where the MDP to solve is defined by a transition model $T(s, a, s') = \phi_{s,a}(s')/\|\phi_{s,a}\|$ and reward function $R(s, a, s') = \rho(s, a, s', \mathbf{b})$. By considering now the information-based rewards presented in Section 7.1.2, solving this MDP will not provide a lower bound—nor an upper bound—of the Bayesian value function, but only an approximation. This simple algorithm will exploit the current information about the belief to explore parts of the model where the information is still weak, and despite the sub-optimality of the approximation, this algorithm exhibits a fair exploration behavior where all state-action pairs are visited infinitely often in the limit.

## 7.3 Experiments

### 7.3.1 Experimental Setup

Three small MDP models were selected to be learned, taken from the BRL state of the art: the classic *Bandit* problem (Gittins, 1979), and the *Chain* and *Loop* problems (Strens, 2000). For each problem, several transition models were tested, varying the transition probabilities of the arcs to test the proposed technique under different scenarios. Figure 7.1 summarizes the structure of each problem (without the transition probabilities). In the 2-state **Bandit MDP** (Figure 7.1), the action $a$ corresponds to pulling one arm, and $b$ corresponds to pulling the other one. State $s_1$ represents winning, meanwhile $s_2$ means losing. This MDP has independence between arms (unknown to the agent), and there is a different unknown probability of success for each of them: $Pr(a) = 0.8$ and $Pr(b) = 0.7$.

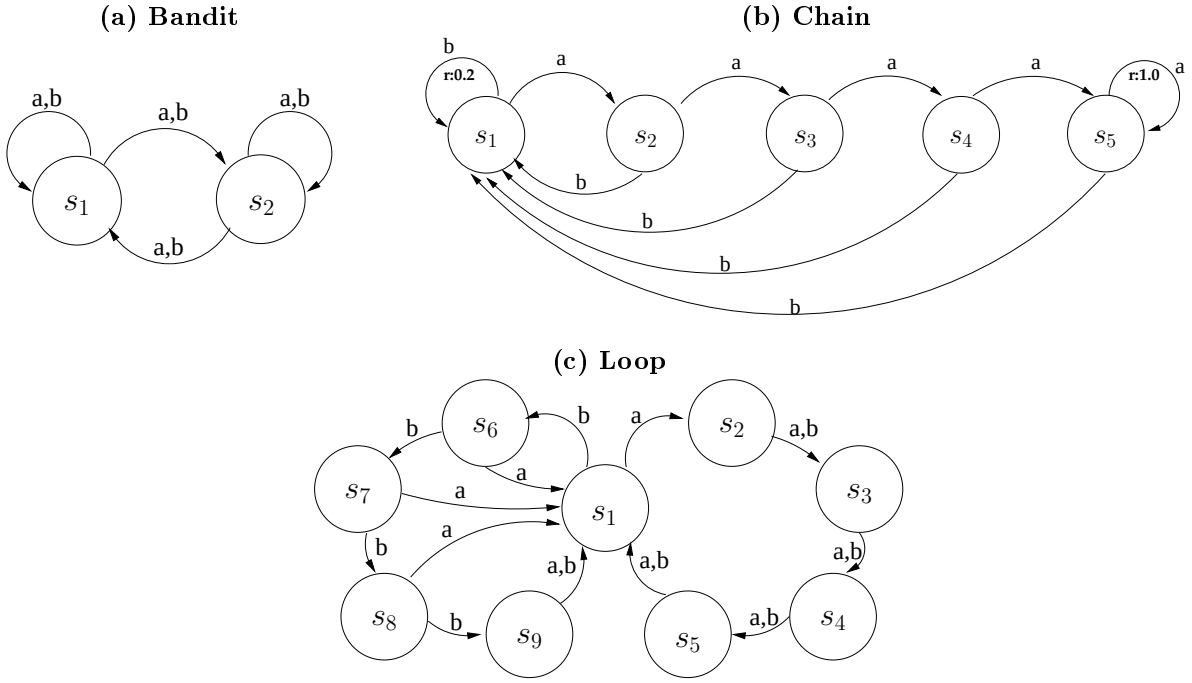



Figure 7.1: MDP models used for the experiments (without transition probabilities). In (a) the 2-armed bandit problem (see Section 1.3), in (b) the 5-state chain problem (see Section 6.4.1) and in (c) the loop problem from (Strens, 2000).

In the 5-state **Chain MDP** (Figure 7.1), every state is connected to the state $s_1$ by taking action $b$ and every state $s_i$ is connected to the next state $s_{i+1}$ with action $a$, except state $s_5$ that is connected to itself. In the *normal* case, the agent can "slip" at each time step with a probability of 0.2, performing the opposite action as intended. The *deterministic* case was also tested when the probability of slipping is zero. Finally, a *mixed* version was constructed, where the probability of slipping is zero for action $b$, but 0.5 for action $a$. These two variations decrease the chances to arrive "by luck" to the states at the right of the chain.

The last one is a 9-state **Loop MDP** (Figure 7.1), which consists in two loops of length 5. Taking action $a$ repeatedly causes traversal of the right loop. Similarly, taking action $b$ repeatedly causes the traversal of the left loop. Taking action $b$ inside the right loop is equivalent to taking action $a$, but taking action $a$ in the left loop causes a return to the initial state $s_1$. All transitions are deterministic.

The initial conditions of the problems are that it starts at state $s_1$ with the uniform distribution over the transition models as an initial prior. Other priors can be used—such as informative or structured priors—but for simplicity only the uniform one will be used for this chapter.

For evaluating the behavior of EXPLOIT, two other policies were considered, namely the RANDOM and GREEDY policies. The RANDOM policy chooses homogeneously a random action at each step, while the GREEDY policy selects the action with largest expected immediate reward.

The three performance criteria have been tested, where the rewards for EXPLOIT and GREEDY are the respective derived rewards of Section 7.1.2 $\rho_V$, $\rho_H$ and $\rho_B$ depending on the evaluated performance criterion. Also, the *state-action count* reward $\rho_S$ was tested for each criterion and experiment.

For approximately solving the finite horizon MDPs within EXPLOIT, the planning horizon was truncated to a small value of $h = \min(2|\mathcal{S}|, H)$. This choice was made to be similar in execution time with RANDOM and GREEDY, and because there is no guarantees that a larger $h$ provides always better results.

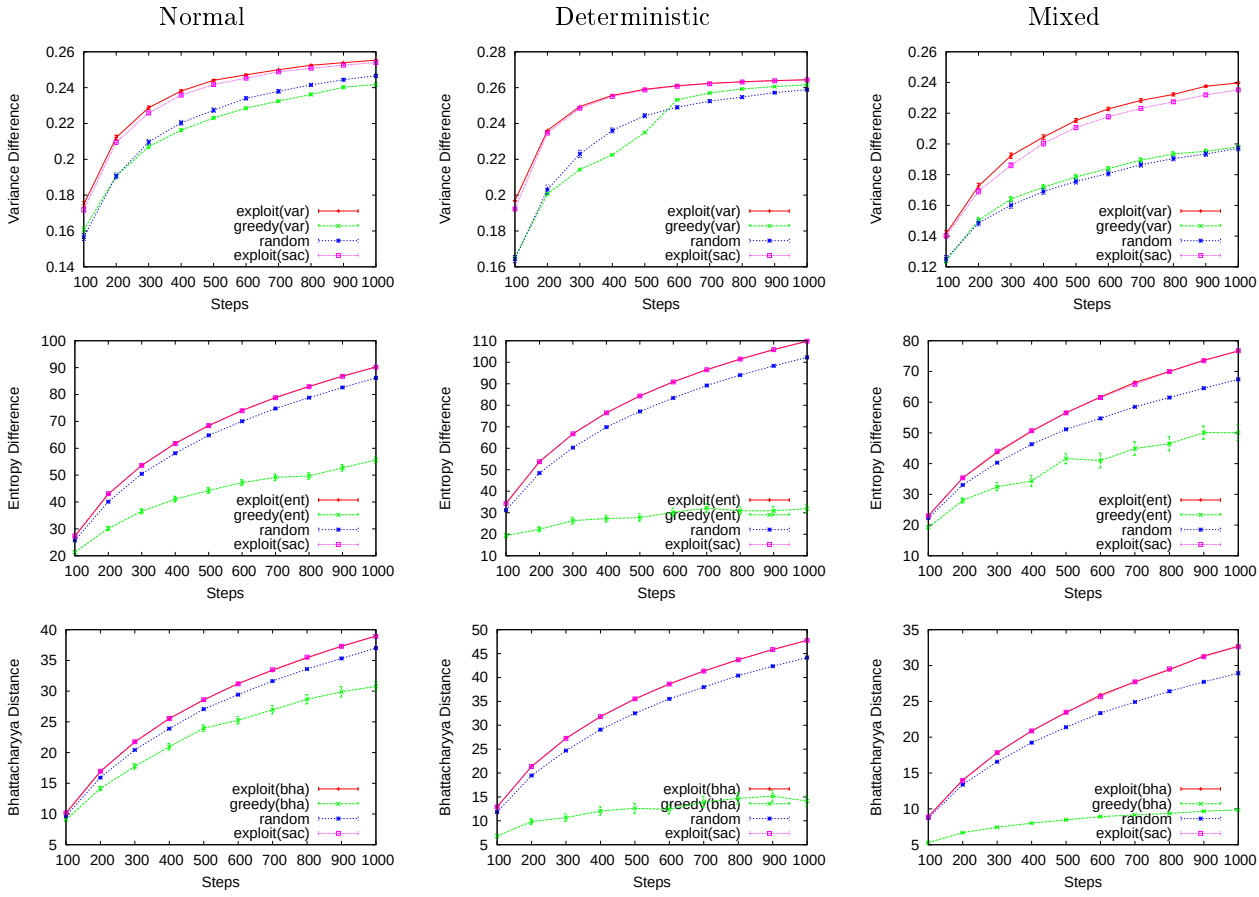



Figure 7.2: Mean performance over 100 trials versus time steps, for the *Chain* problem with different models and the three performance criteria. For each plot, the RANDOM strategy (in blue $*$), the GREEDY strategy (in green $\times$), and the EXPLOIT algorithm with the derived reward (in red $+$) and with the *state-action count* reward (in magenta $\square$) are shown.

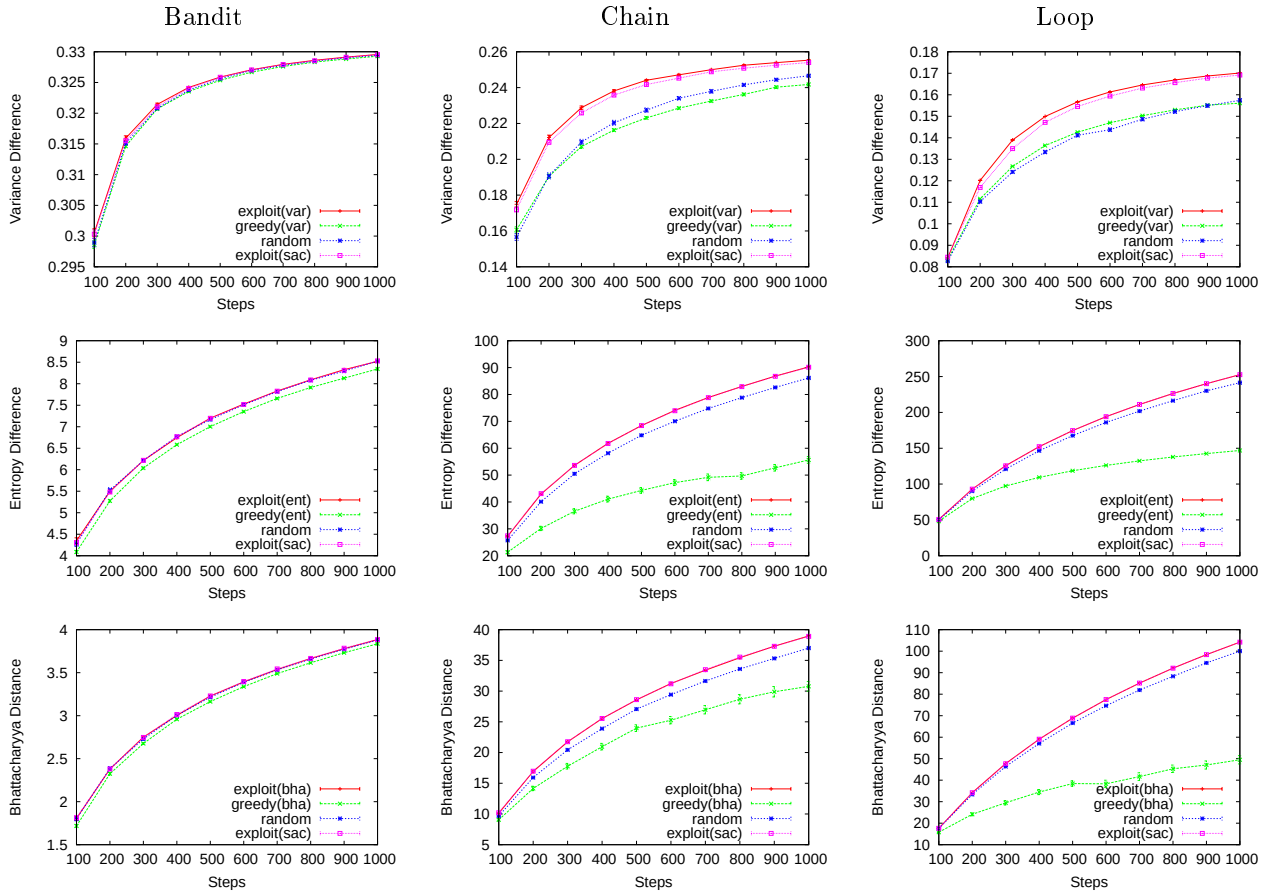



Figure 7.3: Mean performance over 100 trials versus time steps, for the *Bandit*, *Chain* and *Loop* problems, and the three performance criteria. For each plot, the RANDOM strategy (in blue), the GREEDY strategy (in green) and EXPLOIT algorithm with the derived reward (in red) and with the *state-action count* reward (in magenta) are shown.

### 7.3.2 Results

All the strategies were tested on each problem for the first 100 to 1000 steps, and for each performance criterion. Figure 7.2 shows the average performance over 100 trials plotted with their respective 95% confidence interval for the *Chain* problem. Figure 7.3 summarizes the same kind of results for the *Bandit* and *Loop* problems.

Even though *Chain* is a small problem, it is interesting because an intelligent exploration behavior is needed to learn the model. This can be seen along the three criteria, where the GREEDY policy behaves very poorly because a lookahead strategy is needed to arrive to some states. Even though the RANDOM strategy behaves fairly well, information-based strategies outperform this simple technique in all three criteria: for a desired solution quality, several hundred steps more are needed by RANDOM to achieve the same results. Even more, for the variance criterion, it seems to be very difficult for RANDOM to achieve the same quality in the short-term.

An other important result is that the *state-action count* reward behaves similarly well as the respective derived rewards. This means that the derived rewards can be replaced by this computationally cheap reward with no much performance loss. Indeed, performing a cross-experiment for the rewards and criteria shows that all information-based rewards behave similarly well for all criteria.

In terms of the variability of the results, it can seen that GREEDY is the only strategy with a significant variability, while all others present very little variability after a few hundred steps. In particular, the

*Chain* problem produces results with more variability than *Loop* due to the stochastic transition function, but even in that case, the difference between information-based techniques and RANDOM is much larger than the confidence interval.

For the *Bandit* problem there is not much difference between the algorithms, and the behavior is the same through the different criteria. This is because the optimal policy for exploring a fully connected MDP corresponds to fairly selecting the available actions, which resembles the RANDOM policy.

On the *Loop* problem the results resemble the ones presented for the *Chain*: information-based rewards outperform the two simple algorithms. Yet, the improvements of this approach compared to RANDOM are milder than in the *Chain*, because a simple exploration strategy is sufficient for this problem.

## 7.4 Conclusions

This chapter has presented a sound and original way of modeling the problem of actively learning a stochastic MDP model with arbitrary dynamics, by casting the problem as a BRL utility maximization problem with information-based rewards.

### 7.4.1 Summary of Contributions

Three performance criteria that are commonly used to compare probability distributions were employed, namely the variance, the entropy and the Bhattacharyya distance. For each performance criterion, an information-based reward was derived, such that, in the first two cases, the accumulated rewards correspond exactly to the performance criterion. Also, a simple reward function—the *state-action count*—was introduced based on previous work on normal BRL. Even though the formulation—in theory—allows solving the problem optimally, the intractability of computing the optimal Bayesian value function leads to using sub-optimal algorithms such as EXPLOIT. The experiments show that this simple technique produces better results than selecting actions randomly, which is the baseline technique for exploring unknown MDP models. Also, the experiments show that there is no need for selecting complex derived rewards (at least for EXPLOIT) in order to obtain good results; the *state-action count* behaves nearly as well as the theoretically derived rewards.

## 7.5 Future Work

Despite the good results, this chapter leaves several open questions about the possibilities of modeling the active learning of MDP models using BRL. For instance, deepening the analysis on the relationship between the *state-action count* criterion and the other criteria might help defining a more advanced reward shaping technique to derive computationally inexpensive rewards. Also, exploring other techniques used for normal BRL could improve the results as they approach the optimal solution. For example, belief-lookahead techniques can be used for refining the myopic policies proposed here, or maybe some other myopic technique could produce better results.

A natural extension is to encode prior knowledge as a structured prior, such as with DBNs in (Jonsson and Barto, 2007), or with parameter tying in (Poupart et al., 2006). This would dramatically speed up the learning process by making a much more efficient use of data, while not involving major modifications in the solution techniques.

Adapting more elaborate algorithms such as BOLT (see Chapter 6) to the information-gathering scenario is also possible, requiring similar modifications to the ones applied to EXPLOIT. Actually, these experiments are currently being carried out by the author of this dissertation.

One last interesting extension will be to apply the ideas of this chapter to *POMDP Model Learning*. Similarly to MML, a POMDP model may be learned using information-based rewards and the Bayes-Adaptive POMDP framework described in (Ross et al., 2011). This problem is particularly challenging because the information obtained through the observations is entangled: it carries information about the transition function and about the observation function at the same time. This amounts to maintaining several explanations for the same observation, and to propose smart information-gathering strategies for dealing with entangled information.

# Concluding Remarks

The research reported in this dissertation has investigated several aspects of sequential information-gathering decision problems, including POMDPs with information-based rewards in Chapter 4, RL problems with information-based rewards in Chapter 7, and optimistic exploration on Bayesian RL in Chapter 6. The topics addressed here were confined to certain constraints and to certain types of problems, in order to limit the scope of the dissertation. For each topic, a proper theoretical formulation was presented under these constraints and empirical studies were conducted, leading to novel and insightful results that are summarized in each chapter's conclusion. Also, future work was presented for each one of these chapters, describing natural extensions, theoretical entanglements and the research directions that have to be followed to apply these results to real-world problems.

This chapter presents a brief discussion on the results as a whole, showing how this thesis contributes to answering the questions asked in Section 1.2.

## 1 How to represent, measure and update the information held by an agent?

At first glance the representation question seems generally solved. Chapter 2 shows how to represent information using probability theory, and how to update it (and decide) using statistical inference. Moreover, the Bayesian view allows to represent and update information under a well-defined framework. The question about measuring information is addressed in Chapters 4 and 7 by using information theory over the selected representation.

However, this question is much more complex in reality. Unfortunately, within probability theory there is no magical representation for information, probabilistic models being always subject to some assumptions and simplifications. Indeed, the whole zoology of probability distributions is nothing more than different assumptions and simplifications for representing information. One example is the discussion of Bayesian priors for BRL in Section 5.2.1, where choosing FDM priors simplifies the whole process of decision and learning, but is not suitable for continuous state spaces, or for heavily structured problems. The same type of choices must be made when defining the belief-state and the initial belief in POMDPs. This dissertation tried to be as generic as possible within some reasonable limits. Yet, extending the presented results to more general models and looser assumptions is clearly a possible future work.

On top of that, measuring information poses additional problems. The elegant axiomatic derivation in (Shannon, 1948) shows that the entropy is the unique measure that complies with the desired properties for an information measure. However, other authors propose similar measures (and semi-measures) derived from similar axioms or generalizations, such as the Rényi entropy, the KL-divergence, the Chernoff information or the Fisher information. Selecting the proper measure will depend on the used representation, the problem's objective and the ability to compute the measure. Additionally, log-based measures like entropy are complicated functions to work with, in the sense that analytical results are often non-closed expressions. One of the main contributions of this dissertation is to empirically show through Chapters 4 and 7 that simpler (theoretically related) information measures can be used for information-gathering decision problems without a significant performance loss, like the linear reward for $\rho$POMDPs or the state-action count reward for MML. This interchangeability property is surely non-true for applications that need a very fine-grained solution, but most likely will provide better results than

naive strategies like random policies or information-greedy ones.

A particularly important result in this line is the error bound of a PWLC approximation of a convex $\alpha$-Hölderian function (see Section 4.3.7). This result shows that these log-based information measures can be successfully approximated by a set of linear functions with an arbitrary precision, and therefore linear methods can be used within a bounded error. This contribution goes beyond the domain of information-gathering decision problems, and could be useful in other applications of information theory.

## 2 Is (implicit) information gathering a solved issue in sequential decision problems?

The fact that an optimal solution can be theoretically found for sequential decision problems under uncertainty suggests that implicit information gathering is a solved problem, because it is embedded in the same optimal solution. Indeed, such a solution gathers only the information needed by the problem so as to maximize the expected return. However, the intractability of computing an optimal solution leads to solving the problem approximately in most cases. Therefore, some information-gathering procedure is often needed by these approximated techniques. For example, PB solvers must collect informative belief-points, which amounts to gathering information. In the case of Bayesian RL, approximate algorithms often rely on exploration methods that are implicit forms of information gathering. In this context, Chapter 6 offers a new information-gathering method based on the Bayes Boost idea.

Moreover, the relationship between this embedded information gathering and information theory is still an open question. More precisely, the information-theoretical underpinnings of how this optimal solution can optimize the needed information while maximizing the rewards might help develop smarter approximate algorithms. This dissertation presents some hints about this relationship in Section 4.3.6, where the relationship between explicit information gathering and active classification is discussed.

## 3 Is (explicit) information gathering a sequential decision problem by itself?

Despite the advances on implicit information gathering, this dissertation is mainly focused on explicit information gathering (Chapters 4 and 7). This dissertation confirms, through a previous work inspection, that both pure information-gathering problems and cost-sensitive ones are relevant and interesting problems. The proposed models for information gathering are sound and well-defined, in the sense that they are derived from a proper performance criterion, and therefore optimally solvable in theory. Also, the solution techniques proposed here were theoretically and empirically justified, so even though future or ad-hoc techniques may outperform these methods, the contribution of this dissertation is still important, as it offers a proper alternative to the naive—yet vastly used—techniques like random or myopic strategies.

# Research Directions

This section summarizes the main research directions proposed by this dissertation, showing that it opens specific technical extensions that can be tackled in the short-term, and more fundamental questions for the long-term research.

### Short-term Directions

In **Chapter 4** it is proposed to use an information-based collection method for PB algorithms rather than using a random collection. Also, extending the $\rho$POMDPs to factored representations like MOMDPs may help accelerate the solving process. Finally, it is proposed to conduct more cost-sensitive experiments to analyze the impact of the mixing parameter.

In **Chapter 6** it is proposed to use dynamic $\eta$ bonuses for achieving a tighter optimism. Also, extending BOLT to other prior families like Hyperdirichlets may help exploiting the structured information of the problem. Finally, it is proposed to extend the optimistic local transitions for the PORL problem.

In **Chapter 7** it is proposed to use more elaborate algorithms, such as BOLT, for MML. Also, extending the chapter's ideas to POMDP Model Learning and structured priors is suggested.

## Long-term Directions

The **$\rho$POMDP formalism** presents a way to model and to solve sequential information-gathering problems, but the relationship between this kind of problems and usual planning problems under uncertainty is still an open question. Understanding the underpinnings of information gathering may help to construct new POMDP solvers by efficiently gathering information.

The **optimistic local transitions** were used in this dissertation for gathering information in small domains with only few degrees of freedom. However, real-world applications will require extending these results to continuous or large state and action spaces, where information gathering becomes more complex and crucial. The robustness and simplicity of BOLT are desirable qualities when solving complex real-world problems. So, maintaining these properties while scaling up is an interesting challenge that needs further research.

The **MDP Model Learning** problem was defined using justified and formal information-theoretic measures. However, the *state-action count* experiments show that simple rewards may be sufficient to address the problem. Similarly to $\rho$POMDPs, the relationship between the information gathering in MML and BRL is still an open question, so understanding this relationship could help not only to improve the presented techniques for MML problems, but also to construct better BRL algorithms through smart information gathering.

# Bibliography


Mauricio Araya-López, Olivier Buffet, Vincent Thomas, and François Charpillet. A POMDP extension with belief-dependent rewards. In *Advances in Neural Information Processing Systems 23 (NIPS)*, 2010a.

Mauricio Araya-López, Vincent Thomas, Olivier Buffet, and François Charpillet. A closer look at MOMDPs. In *Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2010b.

Mauricio Araya-López, Vincent Thomas, Olivier Buffet, and François Charpillet. Des POMDPs avec des variables d'état visibles. In *Actes des cinquièmes journées francophones planification, décision, apprentissage pour la conduite de systèmes (JFPDA)*, 2010c.

Mauricio Araya-López, Olivier Buffet, Vincent Thomas, and François Charpillet. Active learning of MDP models. In *Proceedings of the 9th European Workshop on Reinforcement Learning (EWRL)*, 2011a.

Mauricio Araya-López, Vincent Thomas, Olivier Buffet, and François Charpillet. Une extension des POMDP avec des récompenses dépendant de l'état de croyance. In *Conférence Francophone d'Apprentissage 2011*, May 2011b. [Best Paper Award].

Mauricio Araya-López, Vincent Thomas, Olivier Buffet, and François Charpillet. Apprentissage actif de modèle de MDP. In *Actes des sixèmes journées francophones planification, décision, apprentissage pour la conduite de systèmes (JFPDA)*, 2011c.

Mauricio Araya-López, Olivier Buffet, and Vincent Thomas. Near-optimal BRL using optimistic local transitions. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012a.

Mauricio Araya-López, Olivier Buffet, and Vincent Thomas. BRL quasi-optimal à l'aide de transitions locales optimistes. In *Actes des septièmes journées francophones planification, décision, apprentissage pour la conduite de systèmes (JFPDA)*, 2012b.

John Asmuth and Michael L. Littman. Learning is planning: near Bayes-optimal reinforcement learning via Monte-Carlo tree search. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 19–26, Corvallis, Oregon, 2011. AUAI Press.

John Asmuth, Lihong Li, Michael L. Littman, Ali Nouri, and David Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 19–26, Corvallis, Oregon, 2009. AUAI Press.

Jean Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration-exploitation trade-off using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 2008.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, May 2002. ISSN 0885-6125.

Orly Avner, Shie Mannor, and Ohad Shamir. Decoupling exploration and exploitation in multi-armed bandits. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A. Ngo. Monte Carlo value iteration for continuous-state POMDPs. In *Proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR)*. Springer, 2010.

Ruzena Bajcsy. Active perception vs. passive perception. In *Proceedings of the 3rd Workshop on Computer Vision: Representation and Control*, pages 55–59. Computer Society Press, 1985.

Ruzena Bajcsy. Active perception. *Proceedings of the IEEE, Special issue on Computer Vision*, 76(8): 996–1005, August 1988.

Andrew G. Barto, Steven J. Bradtke, and Satinder P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1-2):81–138, 1995.

Richard Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60:503–516, 1954.

Richard Bellman. *Dynamic Programming*. Dover Publications, March 1957. ISBN 0486428095.

Dimitri Bertsekas and John Tsitsiklis. *Neurodynamic Programming*. Athena Scientific, 1996.

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

Blai Bonet and Hector Geffner. Solving POMDPs: RTDP-Bel vs. point-based algorithms. In Craig Boutilier, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1641–1646, 2009.

Blai Bonet and Héctor Geffner. Solving large POMDPs using real time dynamic programming. In *Proceedings of the AAAI Fall Symposium on POMDPs*, 1998.

Ronen I. Brafman and Moshe Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 3:213–231, 2003.

Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Active mobile robot localization by entropy minimization. In *Second Euromicro Workshop on Advanced Mobile Robotics*, pages 155–162, 1997.

George Casella and Roger Berger. *Statistical Inference*. Duxbury Resource Center, June 2001. ISBN 0534243126.

Anthony R. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University, Providence, RI, USA, 1998.

Anthony R. Cassandra, Michael L. Littman, and Nevin L. Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 54–61, 1997.

Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sums of observations. *Annals of Mathematical Statistics*, 23:409–507, 1952.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, August 1991. ISBN 0471062596.

Özgür Şimşek and Andrew G. Barto. An intrinsic reward mechanism for efficient exploration. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, pages 833–840, New York, NY, USA, 2006. ACM.

Trevor Darrell and Alex Pentland. Active gesture recognition using partially observable Markov decision processes. In *Proceedings of the International Conference on Pattern Recognition*, pages 984–988, 1996.

Richard Dearden, Nir Friedman, and Stuart Russell. Bayesian Q-learning. In *Proceedings of the 15th National Conference on Artificial intelligence (AAAI)*, pages 761–768, Menlo Park, CA, USA, 1998.

Richard Dearden, Nir Friedman, and David Andre. Model based Bayesian exploration. In *Proceeding of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 150–159, 1999.

Morris DeGroot. *Optimal Statistical Decisions*. Wiley, Newark, NJ, 1970.

Persi Diaconis and Donald Ylvisaker. Conjugate priors for exponential families. *Annals of Statistics*, 7 (2):269–281, 1979. ISSN 0090-5364.

Christos Dimitrakakis. Tree exploration for Bayesian RL exploration. In *CIMCA/IAWTIC/ISE*, 2008.

Michael O. Duff. *Optimal learning: computational procedures for Bayes-adaptive Markov decision processes*. PhD thesis, 2002. AAI3039353.

Adam Eck and Leen-Kiat Soh. Evaluating POMDP rewards for active perception. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1221–1222, Richland, SC, 2012.

Valerii V. Fedorov. *Theory of Optimal Experiments*. Academic Press, New York, 1972.

Ronald A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London*, 222:309–368, 1922.

Ronald A. Fisher. Theory of statistical estimation. *Proceedings of the Cambridge Philosophy Society*, 22: 700–725, 1925.

Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Active Markov localization for mobile robots. *Robotics and Autonomous Systems*, 1998.

Mohammad Ghavamzadeh and Yaakov Engel. Bayesian actor-critic algorithms. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 297–304, 2007.

John C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society*, 41(2):148–177, 1979.

Russell Greiner, Adam J. Grove, and Dan Roth. Learning cost-sensitive active classifiers. *Artificial Intelligence*, 139(2):137–174, 2002. ISSN 0004-3702.

Arthur Guez, David Silver, and Peter Dayan. Efficient Bayes-Adaptive Reinforcement Learning using Sample-Based Search. In *Advances in Neural Information Processing Systems 25 (NIPS)*, 2012.

Greg Hager. Information maps for active sensor control. Technical report, 1987. URL `http://books.google.fr/books?id=vjsMHAAACAAJ`.

Greg Hager. *Active reduction of uncertainty in multisensor systems*. PhD thesis, Philadelphia, PA, USA, 1988. AAI8824744.

Greg Hager and Max Mintz. Estimation procedures for robust sensor control. In *Proceedings of the 3rd Conference on Uncertainty in Artificial Intelligence (UAI)*, New York, NY, 1987. Elsevier Science.

Robin K. Hankin. A generalization of the Dirichlet distribution. *Journal of Statistical Software*, 33(11): 1–18, 2010. ISSN 1548-7660.

Milos Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research (JAIR)*, 13:33–94, 2000.

Alfred O. Hero III, David A. Castan, Douglas Cochran, and Keith Kastella. *Foundations and Applications of Sensor Management*. Springer, 2007. ISBN 0387278923, 9780387278926.

Kenneth J. Hintz. A measure of the information gain attributable to cueing. *IEEE Transactions on Systems, Man and Cybernetics*, 21(2):434 –442, mar/apr 1991. ISSN 0018-9472.

Eric J. Horvitz, John S. Breese, and Max Henrion. Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, 2:247–302, 1988.

Ronald A. Howard. *Dynamic Programming and Markov Processes.* MIT˜Press, Cambridge, Massachusetts, 1960.

Yoshiteru Ishida. Active diagnosis by self-organization: An approach by the immune network metaphor. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1084–1091, 1997.

Tommi Jaakkola, Michael Jordan, and Satinder Singh. Convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6:1185–1201, 1994.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 99:1563–1600, August 2010. ISSN 1532-4435.

Robin Jaulmes, Joelle Pineau, and Doina Precup. Active learning in partially observable Markov decision processes. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 601–608, 2005.

Shihao Ji and Lawrence Carin. Cost-sensitive feature acquisition and classification. *Pattern Recognition*, 40(5):1474–1485, 2007. ISSN 0031-3203.

Anders Jonsson and Andrew Barto. Active learning of dynamic Bayesian networks in Markov decision processes. In *Proceedings of the 7th International Conference on Abstraction, Reformulation, and Approximation*, pages 273–284, Berlin, Heidelberg, 2007. Springer-Verlag.

Leslie P. Kaelbling, Anthony R. Cassandra, and James A. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996a.

Leslie P. Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research (JAIR)*, 4:237–285, 1996b.

Ashish Kapoor, Eric Horvitz, and Sumit Basu. Selective supervision: Guiding supervised learning with decision-theoretic active learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 877–882, 2007.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. In *Machine Learning*, volume 49, pages 260–268, 1998.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 282–293. Springer, 2006.

Anfrey Kolmogorov. Logical basis for information theory and probability theory. *IEEE Transactions on Information Theory*, 14(5):662 – 664, sep 1968. doi: 10.1109/TIT.1968.1054210.

J. Zico Kolter and Andrew Ng. Near-Bayesian exploration in polynomial time. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009.

Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

Chris Kreucher, Keith Kastella, and Alfred O. Hero III. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, 2005. ISSN 0165-1684.

Vikram Krishnamurthy. Algorithms for optimal scheduling and management of hidden Markov model sensors. *IEEE Transactions on Signal Processing*, 50(6):1382–1397, 2002.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:49–86, 1951.

Hanna Kurniawati, David Hsu, and Wee S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems IV*, 2008.

Johannes Lenhard. Models and statistical inference: The controversy between Fisher and Neyman-Pearson. *British Journal for The Philosophy of Science*, 57:69–91, 2006. doi: 10.1093/bjps/axi152.

Michael L. Littman. *Algorithms for sequential decision-making*. PhD thesis, Providence, RI, USA, 1996.

William S. Lovejoy. Computationally feasible bounds for partially observed Markov decision processes. *Operations Research*, 39(1):162–175, 1991.

Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1-2):5–34, 2003.

Andrey Markov. Extension of the Limit Theorems of Probability Theory to a Sum of Variables Connected in a Chain. In R. Howard, editor, *Dynamic Probabilistic Systems (Volume I: Markov Models) (1971)*, chapter Appendix B, pages 552–577. John Wiley & Sons, Inc., New York City, 1907.

Lyudmila Mihaylova, Tine Lefebvre, Herman Bruyninckx, Klaas Gadeyne, and Joris De Schutter. Active sensing for robotics - a survey. In *Proceedings of the 5th International Conference On Numerical Methods and Applications*, pages 316–324, 2002.

Lyudmila Mihaylova, Tine Lefebvre, Herman Bruyninckx, and Joris De Schutter. *NATO Science Series on Data Fusion for Situation Monitoring, Incident Detection, Alert and Response Management*, volume 198, chapter Active Robotic Sensing as Decision Making with Statistical Methods, pages 129–143. IOS Press, 2006.

Marvin Minsky. Steps toward artificial intelligence. In *Computers and Thought*, pages 406–450. McGraw-Hill, 1961.

George E. Monahan. A survey of partially observable Markov decision processes. *Management Science*, 28:1–16, 1982.

Jerzy Neyman and Egon Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London*, 231(694-706):289–337, 1933.

Andrew Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 278–287. Morgan Kaufmann, 1999.

Sylvie C.W. Ong, Shao W. Png, David Hsu, and Wee S. Lee. POMDPs for robotic tasks with mixed observability. In *Proceedings of Robotics: Science and Systems V (RSS)*, 2009.

Ronald Ortner. Optimism in the face of uncertainty should be refutable. *Minds and Machines*, 18(4): 521–526, 2008.

Christos Papadimitriou and John Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

Jerônimo Pellegrini and Jacques Wainer. On the use of POMDPs to model diagnosis and treatment of diseases. In *IV Encontro Nacional de Inteligencia Artificial, 2003, Campinas. IV Encontro Nacional de Inteligencia Artificial.*, 2003.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025–1032, August 2003.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research (JAIR)*, 27:335–380, 2006.

Josep M. Porta, Nikos Vlassis, Matthijs Spaan, and Pascal Poupart. Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research (JMLR)*, 7:2329–2367, 2006.

Pascal Poupart. *Exploiting Structure to Efficiently Solve Large Scale Partially Observable Markov Decision Processes.* PhD thesis, University of Toronto, 2005.

Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete Bayesian reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 2006.

Pascal Poupart, Kee-Eung Kim, and Dongho Kim. Closing the gap: Improved bounds on optimal POMDP solutions. In Fahiem Bacchus, Carmel Domshlak, Stefan Edelkamp, and Malte Helmert, editors, *Proceeding of International Conference on Automated Planning and Scheduling (ICAPS)*, 2011.

Martin Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley-Interscience, April 1994. ISBN 0471619779.

Thomas W. Rauber, Tim Braun, and Karsten Berns. Probabilistic distance measures of the Dirichlet and Beta distributions. *Pattern Recognition*, 41(2):637–645, 2008.

Steven Reece. Strategies for active sensor management. *Active Sensors for Local Planning in Mobile Robotics, World Scientific Series in Robotics and Intelligent Systems*, 26:271–289, 2001.

Alfréd Rényi. On measures of entropy and information. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability*, pages 547–561, 1960.

Mohammad Rezaeian. Sensor scheduling for optimal observability using estimation entropy. In *Proceedings of the 5th IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 307–312, Washington, DC, USA, 2007. IEEE Computer Society.

Stéphane Ross and Brahim Chaib-draa. AEMS: An anytime online search algorithm for approximate policy refinement in large POMDPs. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

Stéphane Ross and Joelle Pineau. Model-based Bayesian reinforcement learning in large structured domains. In *Proceeding of the 24th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 476–483, 2008.

Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-draa. Online planning algorithms for POMDPs. volume 32, pages 663–704, 2008.

Stéphane Ross, Joelle Pineau, Brahim Chaib-draa, and Pierre Kreitmann. A bayesian approach for learning and planning in partially observable markov decision processes. *Journal of Machine Learning Research (JMLR)*, 12:1729–1770, 2011. ISSN 1532-4435.

Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robots. In *Advances in Neural Information Processing Systems 12 (NIPS)*, volume 12, pages 1043–1049, 1999.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.

Romesh Saigal. On piecewise linear approximations to smooth mappings. *Mathematics of Operations Research*, 4(2):153–161, 1979.

Zeyn A. Saigol, Richard W. Dearden, Jeremy L. Wyatt, and Bramley J. Murton. Information-lookahead planning for AUV mapping. In *Proceedings of the International Joint Conference on Artifical intelligence (IJCAI)*, pages 1831–1836, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Guy Shani, Ronen I. Brafman, and Solomon E. Shimony. Forward search value iteration for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, 2012.

Claude E. Shannon. A mathematical theory of communication. *The Bell System technical journal*, 27: 379–423, 1948.

Olivier Sigaud and Olivier Buffet, editors. *Markov Decision Processes and Artificial Intelligence*. ISTE - Wiley, 2010. ISBN: 978-1-84821-167-4.

David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 2164–2172, 2010.

Robert Sim and Nocholas Roy. Active exploration planning for SLAM using extended information filters. In *Proceedings of the 20th Conference Uncertainty in Artificial Intelligence (UAI)*, 2004.

Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J. Kaiser. Efficient informative sensing using multiple robots. volume 34, pages 707–755, 2009.

Richard D. Smallwood and Edward J. Sondik. The optimal control of partially observable Markov decision processes over a finite horizon. *Operation Research*, 21:1071–1088, 1973.

Trey Smith and Reid G. Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2004.

Trey Smith and Reid G. Simmons. Point-based POMDP algorithms: Improved analysis and implementation. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

Edward J. Sondik. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research*, 26:282–304, 1978.

Jonathan Sorg, Satinder Singh, and Richard Lewis. Variance-based rewards for approximate Bayesian reinforcement learning. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.

Matthijs Spaan. Cooperative active perception using POMDPs. In *AAAI 2008 Workshop on Advancements in POMDP Solvers*, July 2008.

Matthijs Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research (JAIR)*, 24:195–220, 2005.

Alexander L. Strehl and Michael L. Littman. A theoretical analysis of model-based interval estimation. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, 2005.

Alexander L. Strehl, Lihong Li, and Michael L. Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research (JMLR)*, 10:2413–2444, December 2009. ISSN 1532-4435.

Malcolm Strens. A Bayesian framework for reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 943–950, 2000.

Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 2010. ISBN 9781608454921. URL `http://books.google.fr/books?id=qwtphfl7U74C`.

István Szita and Csaba Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

Sebastian Thrun. Efficient exploration in reinforcement learning. Technical report, Pittsburgh, PA, USA, 1992.

Sebastian Thrun. Monte Carlo POMDPs. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12 (NIPS)*, pages 1064–1070. MIT Press, 2000.

Leslie G. Valiant. A theory of the learnable. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 436–445, New York, NY, USA, 1984. ISBN 0-89791-133-4.

Michael P. Vitus, Wei Zhang, Alessandro Abate, Jianghai Hu, and Claire J. Tomlin. On sensor scheduling of linear dynamical systems with error bounds. In *Proceeding of the Conference on Decision and Control*, pages 1318–1323, 2010.

Julia Vogel and Kevin Murphy. A non-myopic approach to visual search. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision*, pages 227–234, Washington, DC, USA, 2007. IEEE Computer Society.

John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. ISBN 0691119937.

Abraham Wald. *Statistical decision functions*. Wiley publications in statistics. Wiley, 1950.

Thomas J. Walsh, István Szita, Carlos Diuk, and Michael L. Littman. Exploring compact reinforcement-learning representations with linear regression. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2009.

Thomas J. Walsh, Sergiu Goschin, and Michael L. Littman. Integrating sample-based planning and model-based reinforcement learning. In *Proceedings of the National conference on Artificial Intelligence (AAAI)*, 2010.

Tao Wang, Daniel Lizotte, Michael Bowling, and Dale Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *Prooceding of the 22nd International Conference on Machine Learning (ICML)*, pages 956–963, 2005.

Tao Wang, Pascal Poupart, Michael Bowling, and Dale Schuurmans. Compact, convex upper bound iteration for approximate pomdp planning. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 1245–1251, 2006.

Christopher J. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.

Jason L. Williams. *Information theoretic sensor management*. PhD thesis, Cambridge, MA, USA, 2007.

Håkan L. S. Younes, Michael L. Littman, David Weissman, and John Asmuth. The first probabilistic track of the international planning competition. *Journal of Artificial Intelligence Research (JAIR)*, 24 (1):851–887, December 2005. ISSN 1076-9757.

Feng Zhao, Jaewon Shin, and James Reich. Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, 19(2):61–72, 2002.

Alice X. Zheng, Irina Rish, and Alina Beygelzimer. Efficient test selection in active diagnosis via entropy approximation. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

Karl Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174 – 205, 1965. ISSN 0022-247X.

# Appendixes

# Appendix A

# Technical Proofs

## Contents



*"We are servants rather than masters in mathematics."*
— Charles Hermite.

## A.1 Chapter 4: Deferred Proofs

### A.1.1 Stand-alone Proof of $\kappa$ Convexity

**Lemma A.1.1.** *Let $w \in \mathbb{R}^n$ and $f : \mathbb{R}^n \mapsto \mathbb{R}$ a convex function. If $\kappa(w) = \|w\|_1 f(\frac{w}{\|w\|_1})$, then $\kappa(w)$ is also a convex function.*

*Proof.* As stated in the main text, one can use the perspective and linear-fractional convexity preserving operations to directly prove that $\kappa$ is convex. However, this can also be proved by only using the convexity of $f$:

$$\begin{aligned}
\kappa(\alpha x + (1-\alpha)y) &= \|\alpha x + (1-\alpha)y\|_1 f(\frac{\alpha x + (1-\alpha)y}{\|\alpha x + (1-\alpha)y\|_1}) \\
&= \|\alpha x + (1-\alpha)y\|_1 f(\frac{\alpha\|x\|_1}{\|\alpha x + (1-\alpha)y\|_1} \cdot \frac{x}{\|x\|_1} + \frac{(1-\alpha)\|y\|_1}{\|\alpha x + (1-\alpha)y\|_1} \cdot \frac{y}{\|y\|_1}) \\
&\leq \|\alpha x + (1-\alpha)y\|_1 (\frac{\alpha\|x\|_1}{\|\alpha x + (1-\alpha)y\|_1} f(\frac{x}{\|x\|_1}) + \frac{(1-\alpha)\|y\|_1}{\|\alpha x + (1-\alpha)y\|_1} f(\frac{y}{\|y\|_1})) \\
&= \alpha\|x\|_1 f(\frac{x}{\|x\|_1}) + (1-\alpha)\|y\|_1 f(\frac{y}{\|y\|_1}) \\
&= \alpha\kappa(x) + (1-\alpha)\kappa(y).
\end{aligned}$$

□

### A.1.2 $\alpha$-Hölderian functions are Lipchitzian in $\Delta_\eta$

This section presents the proof of Lemma 4.3.3, which shows that a convex $\alpha$-Hölderian function is Lipchitzian when a margin from the boundaries is excluded.

**Lemma 4.3.3.** *Let $\eta > 0$ and $f$ be an $\alpha$-Hölder (with constant $K_\alpha$), bounded and convex function from $\Delta$ to $\mathbb{R}$, $f$ being differentiable everywhere in $\Delta^o$ (the interior of $\Delta$). Then, for all $b \in \Delta_\eta$, $\|\nabla f(b)\|_1 \leq K_\alpha \eta^{\alpha-1}$.*

Before proving Lemma 4.3.3, an equivalent result in the 1-dimensional case is presented.

**Lemma A.1.2.** *Let $x_a, x_b \in \mathbb{R}$ ($x_a < x_b$), and $\eta \in (0, x_b - x_a)$ be three scalars. Let also $f$ be a $\alpha$-Hölder (with constant $K_\alpha$), bounded and convex function from $[x_a, x_b]$ to $\mathbb{R}$, $f$ being differentiable everywhere on $(x_a, x_b)$. Then $f$ is $K_\alpha \eta^\alpha$-Lipschitz on $[x_a + \eta, x_b - \eta]$.*

*Proof.* For any $x \in [x_a + \eta, x_b]$ (see Fig. A.1 for an illustration):

$$\begin{aligned} f'(x) &\geq f'(x_a + \eta) && \text{(By convexity of } f\text{)} \\ &\geq \frac{f(x_a + \eta) - f(x_a)}{\eta} && \text{(For the same reason)} \\ &\geq -\frac{|f(x_a + \eta) - f(x_a)|}{\eta} \\ &\geq -K_\alpha \eta^{\alpha-1} && \text{(Because } f \text{ is } \alpha\text{-Hölder).} \end{aligned}$$

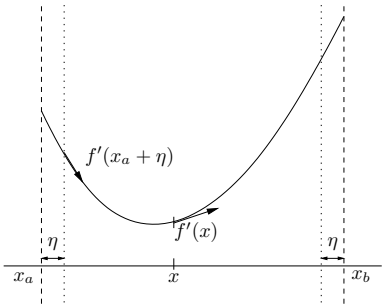



Figure A.1: Illustration for the proof of Lemma A.1.2

Similarly, for any $x \in [x_a, x_b - \eta]$,

$$\begin{aligned} f'(x) &\leq f'(x_b - \eta) \\ &\leq \frac{f(x_b) - f(x_b - \eta)}{\eta} \\ &\leq \frac{|f(x_b) - f(x_b - \eta)|}{\eta} \\ &\leq K_\alpha \eta^{\alpha-1}. \end{aligned}$$

Thus, for any $x \in [x_a + \eta, x_b - \eta]$, $|f'(x)| \leq K_\alpha \eta^{\alpha-1}$, so that $f$ is $K_\alpha \eta^{\alpha-1}$-Lipschitz on $[x_a + \eta, x_b - \eta]$. □

Now it can be shown how the above property extends to an $n$-simplex (using any norm).

*Proof of Lemma 4.3.3.* Let $b$ be a point in $\Delta_\eta$ and let $\mathbf{u}$ be a unit vector parallel to the hyperplane containing $\Delta$. As shown on Fig. A.2, the line going through $b$ and directed by $\mathbf{u}$ intersects $\Delta$ on the closed segment $S_\mathbf{u} = [b + x_a \mathbf{u}, b + x_b \mathbf{u}]$ ($x_a < 0 < x_b$). Thus, a function $g_u$ can be defined from $[x_a, x_b]$

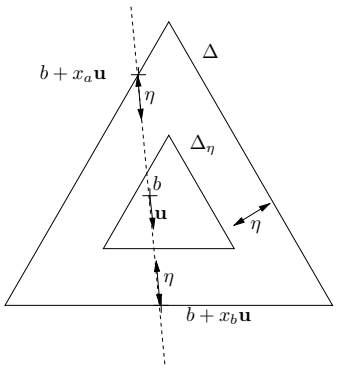



Figure A.2: Illustration for the proof of Lemma 4.3.3

to $\mathbb{R}$ such that $g_\mathbf{u} : x \mapsto f(b + x\mathbf{u})$. $g_\mathbf{u}$ is then a $\alpha$-Hölder (with same constant), bounded and convex function from $[x_a, x_b]$ to $\mathbb{R}$, $g_\mathbf{u}$ being differentiable everywhere on $(x_a, x_b)$.

Please note that the shortest distance (in any norm) between $\Delta_\eta$ and $bd(\Delta)$ (the boundary of $\Delta$) is lower bounded by $\eta$.[34] The intersection of $\Delta_\eta$ with $S_\mathbf{u}$ is thus a segment $S'_\mathbf{u} \subseteq [b+(x_a+\eta)\mathbf{u}, b+(x_b-\eta)\mathbf{u}]$. By applying Lemma A.1.2 to $g_\mathbf{u}$ in $[x_a + \eta, x_b - \eta]$, the derivative of $g_\mathbf{u}$ at point 0 (and therefore the directional derivative of $f$ at $b$ along $\mathbf{u}$) is bounded (in absolute value) by $K_\alpha \eta^{\alpha-1}$. This property holding for any $\mathbf{u}$, then

$$\forall b \in \Delta_\eta,\ \|\nabla f(b)\| \leq K_\alpha \eta^{\alpha-1}.$$

□

The above lemmas naturally extend to the case where $f$ is piecewise differentiable. Please note that the norm of the gradient of Lemma 4.3.3 is defined as norm-1 for the uses of this paper, but any $p$-norm can be used, as stated in the proof.

## A.2 Chapter 6: PAC-BAMDP Analysis of BEB

This section shows that BEB is PAC-BAMDP, extending the analysis provided in (Kolter and Ng, 2009) to infinite horizons.

### A.2.1 Main Result

Lemma 4 of (Kolter and Ng, 2009) state that if $\beta \geq 2H^2$, the BEB algorithm is always optimistic compared to the optimal Bayesian value function for the undiscounted reward case. This can be easily extended to the discounted reward case, where the optimism is maintained for $\beta \geq 2H(1-\gamma^H)/(1-\gamma)$. However, for a larger $\beta = 2H^2 \geq 2H(1-\gamma^H)/(1-\gamma)$ this is also true.

**Theorem A.2.1** (BEB is PAC-BAMDP)**.** *Let $\mathbf{A}_t$ denote the policy followed by* BEB *at time $t$ with $\beta = 2H^2$. Let also $s_t$ and $\mathbf{b}_t$ be the corresponding state and belief at that time. Then, with probability at least $1-\delta$,* BEB *is $\epsilon$-close to the optimal Bayesian policy*

$$\mathbb{V}^{\mathbf{A}_t}(s_t, \mathbf{b}_t) \geq \mathbb{V}^*(s_t, \mathbf{b}_t) - \epsilon$$

*for all but $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|\beta^2}{\epsilon^2(1-\gamma)^2}\right) = \tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^4}{\epsilon^2(1-\gamma)^2}\right)$ time steps.*

34. In $p$-norm, the distance between $\Delta_\eta$ and $bd(\Delta)$ is the distance between the points $(\eta, \frac{1-\eta}{n-1}, \ldots, \frac{1-\eta}{n-1})$ and $(0, \frac{1}{n-1}, \ldots, \frac{1}{n-1})$, i.e., $d(\Delta_\eta, bd(\Delta)) = (\eta^p + (n-1)\frac{\eta^p}{(n-1)^p})^{1/p} = \eta(1 + \frac{1}{(n-1)^{p-1}})^{1/p} \geq \eta$.

### A.2.2 BEB Mixed Bound

Let $\tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t)$ be the evaluation of the BEB policy $\mathbf{A}_t$ at time $t$ using a *mixed* value function. In this case, $\tilde{R}(s,a,s') = \hat{R}(s,a,s',\mathbf{b}_t)$ the reward with bonus, and $\tilde{T}(s,a,s') = T(s,a,s',\mathbf{b}_t)$ the expected transition model of $\mathbf{b}_t$.

**Lemma A.2.2** (BEB Mixed Bound)**.** *The difference between the value obtained by* BEB *and the value obtained by the* mixed *value function under the policy generated by* BEB*, $\mathbf{A}_t$, with $\beta = 2H^2$ is bounded by*

$$V_H^{\text{BEB}}(s_t, \mathbf{b}_t) - \tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t, \mathbf{b}_t) \leq \frac{2\beta(1-\gamma^H)}{m(1-\gamma)}. \tag{A.1}$$

*where $m$ is the minimum count for a state-action pair to be in $K$.*

*Proof.* Following the same induction technique as for the proof of Lemma 6.3.3, assume that the difference between two evaluations at iteration $i$ is bounded by a maximum quantity $\Delta_i$. Again, the difference $\Delta_{i+1}$ has two cases. The first case is when $(s,a) \notin K$, and, by the same arguments as in the proof of Lemma 6.3.3, the different is bounded by

$$\Delta_{i+1}^{(\notin K)} \leq \gamma\Delta_i.$$

The second case is when $(s,a) \in K$, where the probabilities and rewards differ,

$$\begin{aligned}
\Delta_{i+1}^{(\in K)} &= V_{i+1}^{\text{BEB}}(s, \mathbf{b}_t) - \tilde{\mathbb{V}}_{i+1}^{\mathbf{A}_t}(s, \mathbf{b}) \\
&= \sum_{s'} T(s,a,s',\mathbf{b}_t)(\hat{R}(s,a,s',\mathbf{b}_t) + \gamma V_i^{\text{BEB}}(s,\mathbf{b}_t)) - \sum_{s'} T(s,a,s',\mathbf{b})(R(s,a,s') + \gamma\tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')) \\
&\leq \sum_{s'} T(s,a,s',\mathbf{b}_t)\left(\hat{R}(s,a,s',\mathbf{b}_t) + \gamma V_i^{\text{BEB}}(s,\mathbf{b}_t) - R(s,a,s') - \gamma\tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')\right) \\
&\quad + \sum_{s'} |T(s,a,s',\mathbf{b}_t) - T(s,a,s',\mathbf{b})| \left(R(s,a,s') + \gamma\tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')\right) \\
&\leq \sum_{s'}\left[T(s,a,s',\mathbf{b}_t)\left(\frac{\beta}{1+\|\phi_{s,a}^{(t)}\|} + \gamma\Delta_i\right) + \frac{(1-\gamma^i)}{(1-\gamma)}\sum_{s'}|T(s,a,s',\mathbf{b}_t) - T(s,a,s',\mathbf{b})|\right] \\
&\leq \frac{\beta}{(1+\|\phi_{s,a}^{(t)}\|)} + \gamma\Delta_i + \frac{2H(1-\gamma^i)}{(1-\gamma)(1+\|\phi_{s,a}^{(t)}\|)} \\
&\leq \frac{2\beta}{m} + \gamma\Delta_i.
\end{aligned}$$

Here, the third step is due to the property

$$\sum_x p(x)f(x) - \sum q(x)g(x) \leq \sum_x p(x)(f(x) - g(x)) + \sum_x |p(x) - q(x)|g(x),$$

which holds if all functions are positive. The next step is obtained by using the definitions of $\hat{R}$ and $\Delta_i$ in the left term, and considering the maximum possible value of $R(s,a,s') + \gamma\tilde{\mathbb{V}}_i^{\mathbf{A}_t}(s,\mathbf{b}')$. The fifth step is obtained by Lemma 3 of (Kolter and Ng, 2009), where the sum of the absolute differences $\sum_s' |T(s,a,s',\mathbf{b}_t) - T(s,a,s',\mathbf{b})| \leq \frac{2H}{(1+\|\phi_{s,a}^{(t)}\|)}$. The last step is due to the facts that $(1-\gamma^i)/(1-\gamma) \leq H$, and that $1 + \|\phi_{s,a}^{(t)}\| \geq m$ because the analyzed state-action pairs are in $K$. Now, the maximum difference at iteration $i+1$ can be safely defined as the worst case: $\Delta_{i+1} = \max(\Delta_{i+1}^{(\notin K)}, \Delta_{i+1}^{(\in K)}) = \frac{2\beta}{m} + \gamma\Delta_i$. By applying this equation repeatedly with the base step $\Delta_0 = 0$, because $V_0^{\text{BEB}}(s,\mathbf{b}_t) = \tilde{\mathbb{V}}_0^{\mathbf{A}_t}(s,\mathbf{b}) = 0$, the desired result is obtained. □

### A.2.3 Theorem Proof

With these lemmas the main Theorem A.2.1 can be proved, showing that BEB is PAC-BAMDP in the discounted infinite horizon case (without modifying the algorithm to stop monitoring beliefs as in (Kolter and Ng, 2009)).

*Proof of Theorem A.2.1.* Consider the induced inequality (Lemma 6.3.2) with $\mathbf{A}_t$ the policy generated by BEB at time $t$, and $\tilde{\mathbb{V}}$ a *mixed* value function using BEB's update when $(s,a) \notin K$. As BEB's bonus is always decreasing, $\tilde{R}_{max} = 2\beta$ $(> 1+\beta)$ can be safely defined. Lemma A.2.2 is defined for a computing horizon $H$, so in order to use this result first one needs to notice that a truncated summation is always smaller than the infinite summation. Starting from these facts, and assuming normalized rewards, it follows that

$$\begin{aligned}
\mathbb{V}^{\mathbf{A}_t}(s_t,\mathbf{b}_t) &\geq \mathbb{V}_H^{\mathbf{A}_t}(s_t,\mathbf{b}_t) \\
&\geq \tilde{\mathbb{V}}_H^{\mathbf{A}_t}(s_t,\mathbf{b}_t) - \frac{2\beta(1-\gamma^H)}{(1-\gamma)} Pr(A_K) \\
&\geq V_H^{\text{BEB}}(s_t,\mathbf{b}_t) - \frac{2\beta(1-\gamma^H)}{m(1-\gamma)} - \frac{2\beta(1-\gamma^H)}{(1-\gamma)} Pr(A_K) \\
&\geq \mathbb{V}_H^{*}(s_t,\mathbf{b}_t) - \frac{2\beta(1-\gamma^H)}{m(1-\gamma)} - \frac{2\beta(1-\gamma^H)}{(1-\gamma)} Pr(A_K) \\
&\geq \mathbb{V}^{*}(s_t,\mathbf{b}_t) - \frac{2\beta(1-\gamma^H)}{m(1-\gamma)} - \frac{2\beta(1-\gamma^H)}{(1-\gamma)} Pr(A_K) - \frac{\gamma^H}{(1-\gamma)}
\end{aligned}$$

where the $3^{rd}$ step is due to Lemma A.2.2 (accuracy), the $4^{th}$ step to Lemma 4 of (Kolter and Ng, 2009) (optimism), and the last step to Lemma 2 of (Kearns and Singh, 1998) (error bound between finite and infinite horizon computation) [35].

Here, the error between $\mathbb{V}^*(s_t,\mathbf{b}_t)$ and $\mathbb{V}^{\mathbf{A}_t}(s_t,\mathbf{b}_t)$ depends on $H, \beta, m$ and $\gamma$, besides the probability $Pr(A_K)$. To simplify the analysis, assume that $\frac{\gamma^H}{(1-\gamma)} = \frac{\epsilon}{2}$ and fix $m = \frac{8\beta}{\epsilon(1-\gamma)}$.

There are two possible cases for $Pr(A_K)$. First suppose that $Pr(A_K) > \frac{1}{m} = \frac{\epsilon(1-\gamma)}{8\beta}$, then by the Hoeffding and union bounds (Valiant, 1984), this occurs in no more than

$$O\left(\frac{|\mathcal{S}||\mathcal{A}|m}{Pr(A_K)} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta}\right) = O\left(\frac{|\mathcal{S}||\mathcal{A}|\beta^2}{\epsilon^2(1-\gamma)^2} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta}\right)$$

time steps with probability $1-\delta$. By using the complexity notation that neglects logarithms the desired sample complexity of the theorem is obtained. This bound is derived from the fact that, if the event $A_K$ occurs more than $|\mathcal{S}||\mathcal{A}|m$ times, then all the state-action pairs are known [36] and the agent will never escape from $K$ anymore. The second case is when $Pr(A_K) \leq \frac{1}{m}$, where

$$\begin{aligned}
\mathbb{V}^{\mathbf{A}_t}(s_t,\mathbf{b}_t) &\geq \mathbb{V}^{*}(s_t,\mathbf{b}_t) - \frac{2\epsilon(1-\gamma^H)}{8} - \frac{2\epsilon(1-\gamma^H)}{8} - \frac{\epsilon}{2} \\
&\geq \mathbb{V}^{*}(s_t,\mathbf{b}_t) - \frac{\epsilon}{4} - \frac{\epsilon}{4} - \frac{\epsilon}{2} \\
&= \mathbb{V}^{*}(s_t,\mathbf{b}_t) - \epsilon
\end{aligned}$$

which verifies the proposed theorem. □

35. Lemma 2 of (Kearns and Singh, 1998) is presented for normal MDPs, but its applicability to BAMDPs is straightforward by using the same arguments.

36. Here the information already encoded in the prior is not considered, so this bound will be much more tight in practice.

## A.3 Chapter 7: Information-based Reward Derivations

This section shows how the derived rewards of Section 7.1.4 were obtained. For each performance criterion, a $\rho(s, a, s', \mathbf{b})$ function was derived for FDM priors. All of them can be finally expressed in terms of the three helper variables $x = \phi_{s,a}(s')$, $y = \|\phi_{s,a}\|_1$ and $z = \|\phi_{s,a}\|_2^2$, where $\phi$ is the FDM parameter vector for $\mathbf{b}$ and $(s, a, s')$ is the observed transition.

### A.3.1 The Variance Difference

The objective is to find an analytical expression for the variance difference,

$$\begin{aligned}\rho_V(s, a, s', \mathbf{b}_t) &= D_V(\mathbf{b}_t, \mathbf{b}_{t+1}) \\ &= \sum_{\hat{s},\hat{a}} \sum_{\hat{s}'} (\sigma^2_{\hat{s}'}(\phi_{\hat{s},\hat{a}}) - \sigma^2_{\hat{s}'}(\phi'_{\hat{s},\hat{a}})).\end{aligned}$$

As for FDMs, $\phi'$ differs from $\phi'$ only in that $\phi'_{s,a}(s') = \phi_{s,a}(s') + 1$, the expression can be trivially simplified to

$$\rho_V(s, a, s', \mathbf{b}_t) = \sum_{\hat{s}'} (\sigma^2_{\hat{s}'}(\phi_{s,a}) - \sigma^2_{\hat{s}'}(\phi'_{s,a})),$$

because for all $\hat{s} \neq s$ and $\hat{a} \neq a$, the parameters of $\phi'_{\hat{s},\hat{a}} = \phi'_{\hat{s},\hat{a}}$.

Now by using the variance definition for Dirichlet distributions of Equation 7.3, the variance difference

$$\begin{aligned}\rho_V(s, a, s', \mathbf{b}_t) &= \sum_{\hat{s}'} \left( \frac{\phi_{s,a}(\hat{s}')(\|\phi_{s,a}\|_1 - \phi_{s,a}(\hat{s}'))}{\|\phi_{s,a}\|_1^2(\|\phi_{s,a}\|_1 + 1)} - \frac{\phi'_{s,a}(\hat{s}')(\|\phi'_{s,a}\|_1 - \phi'(\hat{s}'))}{\|\phi'_{s,a}\|_1^2(\|\phi'_{s,a}\|_1 + 1)} \right) \\ &= \frac{\|\phi_{s,a}\|_1^2 - \|\phi_{s,a}\|_2^2}{\|\phi_{s,a}\|_1^2(\|\phi_{s,a}\|_1 + 1)} - \frac{\|\phi'_{s,a}\|_1^2 - \|\phi'_{s,a}\|_2^2}{\|\phi'_{s,a}\|_1^2(\|\phi'_{s,a}\|_1 + 1)} \\ &= \frac{\|\phi_{s,a}\|_1^2 - \|\phi_{s,a}\|_2^2}{\|\phi_{s,a}\|_1^2(\|\phi_{s,a}\|_1 + 1)} - \frac{(\|\phi'_{s,a}\|_1 + 1)^2 - (\|\phi'_{s,a}\|_2^2 - \phi_{s,a}(s')^2 + (\phi_{s,a}(s') + 1)^2)}{(\|\phi_{s,a}\|_1 + 1)^2(\|\phi'_{s,a}\|_1 + 2)}.\end{aligned}$$

This expression can be rewritten using the helper variables and simplified as follows:

$$\begin{aligned}\rho_V(s, a, s', \mathbf{b}_t) &= \frac{y^2 - z}{y^2(y+1)} - \frac{(y+1)^2 - (z - x^2 + (x+1)^2)}{(y+1)^2(y+2)} \\ &= \frac{1}{(y+1)} - \frac{z}{y^2(y+1)} - \frac{y^2 + 2y + 1 - z + x^2 - x^2 - 2x - 1}{(y+1)^2(y+2)} \\ &= \frac{1}{(y+1)} - \frac{z}{y^2(y+1)} - \frac{y^2 + 2y - z - 2x}{(y+1)^2(y+2)}.\end{aligned}$$

### A.3.2 The Entropy Difference

Similarly to the variance difference, the objective is to find an analytical expression for the entropy difference

$$\begin{aligned}\rho_H(s, a, s', \mathbf{b}_t) &= D_H(\mathbf{b}_t, \mathbf{b}_{t+1}) \\ &= \sum_{\hat{s},\hat{a}} \left( H(\phi_{\hat{s},\hat{a}}) - H(\phi'_{\hat{s},\hat{a}}) \right) \\ &= H(\phi_{s,a}) - H(\phi'_{s,a}).\end{aligned}$$

By using the definitions of $H$, $B(\cdot)$ and knowing that $\psi(x+1) = \frac{1}{x} + \psi(x)$, the expression can be rewritten as follows:

$$\begin{aligned}
\rho_H(s,a,s',\mathbf{b}_t) &= \log(B(\phi_{s,a}) + (\|\phi_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi_{s,a}\|_1) - \sum_{\hat{s}'}(\phi_{s,a}(\hat{s}') - 1)\psi(\phi_{s,a}(\hat{s}')) \\
&\quad .\log(B(\phi'_{s,a})) - (\|\phi'_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi'_{s,a}\|_1) + \sum_{\hat{s}'}(\phi'_{s,a}(\hat{s}') - 1)\psi(\phi'_{s,a}(\hat{s}')) \\
&= \log\left(\frac{\prod_{\hat{s}'}\Gamma(\phi_{s,a}(\hat{s}')}{\Gamma(\|\phi_{s,a}\|_1)}\right) + (\|\phi_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi_{s,a}\|_1) - \sum_{\hat{s}'}(\phi_{s,a}(\hat{s}') - 1)\psi(\phi_{s,a}(\hat{s}')) \\
&\quad - \log\left(\frac{\prod_{\hat{s}'}\Gamma(\phi'_{s,a}(\hat{s}')}{\Gamma(\|\phi'_{s,a}\|_1)}\right) - (\|\phi'_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi'_{s,a}\|_1) + \sum_{\hat{s}'}(\phi'_{s,a}(\hat{s}') - 1)\psi(\phi'_{s,a}(\hat{s}')) \\
&= \log\left(\frac{\prod_{\hat{s}'}\Gamma(\phi_{s,a}(\hat{s}'))}{\Gamma(\|\phi_{s,a}\|_1)} \cdot \frac{\Gamma(\|\phi'_{s,a}\|_1)}{\prod_{\hat{s}'}\Gamma(\phi'_{s,a}(\hat{s}'))}\right) \\
&\quad + \left((\|\phi_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi_{s,a}\|_1) - (\|\phi'_{s,a}\|_1 - |\mathcal{S}|)\psi(\|\phi'_{s,a}\|_1)\right) \\
&\quad + \left(\sum_{\hat{s}'}(\phi'_{s,a}(\hat{s}') - 1)\psi(\phi'_{s,a}(\hat{s}')) - (\phi_{s,a}(\hat{s}') - 1)\psi(\phi_{s,a}(\hat{s}'))\right) \\
&= \log\left(\frac{\|\phi_{s,a}\|_1}{\phi_{s,a}(s')}\right) + \left(-\psi(\|\phi_{s,a}\|_1) - 1 + \frac{|\mathcal{S}|}{\|\phi_{s,a}\|_1} - \frac{1}{\|\phi_{s,a}\|_1}\right) + \left(1 + \psi(\phi_{s,a}(\hat{s}'))\right).
\end{aligned}$$

This expression can be rewritten using the helper variables and simplified as follows:

$$\begin{aligned}
\rho_H(s,a,s',\mathbf{b}_t) &= \log\left(\frac{y}{x}\right) - \psi(y) + \frac{|\mathcal{S}|}{y} - \frac{1}{y} + \psi(x) \\
&= \log\left(\frac{y}{x}\right) + \frac{|\mathcal{S}|}{y} - \sum_{j=x}^{y}\frac{1}{j}.
\end{aligned}$$

This last step is obtained by using the harmonic interpretation of the digamma function, $\psi(x) = -\gamma + \sum_{j=1}^{x-1}\frac{1}{j}$, where $\gamma$ is the Euler-Mascheroni constant.

### A.3.3 The Bhattacharyya Distance

For the Bhattacharyya distance, the analytically derived reward is obtained by using the definitions of the Chernoff information, the Beta function and the Gamma function, as follows:

$$\begin{aligned}
\rho_B(s,a,s',\mathbf{b}_t) &= D_B(\mathbf{b}_t,\mathbf{b}_{t+1}) \\
&= \sum_{\hat{s},\hat{a}} C_{1/2}(\phi_{\hat{s},\hat{a}},\phi'_{\hat{s},\hat{a}} \\
&= C_{1/2}(\phi_{s,a},\phi'_{s,a}) + \sum_{\hat{s}\neq s,\hat{a}\neq a} C_{1/2}(\phi_{\hat{s},\hat{a}},\phi_{\hat{s},\hat{a}}) \\
&= -\log\left(\frac{B(1/2\phi_{s,a}+1/2\phi'_{s,a})}{B(\phi_{s,a})^{1/2}B(\phi'_{s,a})^{1/2}}\right) + \cancel{\sum_{\hat{s}\neq s,\hat{a}\neq a} -\log\left(\frac{B(\phi_{s,a})}{B(\phi_{s,a})}\right)} \\
&= -\log\left(\frac{\prod_{\hat{s}'}\Gamma(\phi_{s,a}(\hat{s}')+1/2\mathbb{I}(\hat{s}',s'))}{\Gamma(\|\phi_{s,a}\|+1/2)}\cdot\left(\frac{\Gamma(\|\phi_{s,a}\|)}{\prod_{\hat{s}'}\Gamma(\phi_{s,a}(\hat{s}'))}\right)^{\frac{1}{2}}\cdot\left(\frac{\Gamma(\|\phi_{s,a}\|+1)}{\prod_{\hat{s}'}\Gamma(\phi_{s,a}(\hat{s}')+\mathbb{I}(\hat{s}',s'))}\right)^{\frac{1}{2}}\right) \\
&= -\log\left(\frac{\Gamma(\phi_{s,a}(s')+1/2)}{\Gamma(\|\phi_{s,a}\|+1/2)\Gamma(\phi_{s,a}(s'))}\cdot\left(\frac{\Gamma(\|\phi_{s,a}\|)}{1}\right)^{\frac{1}{2}}\cdot\left(\frac{\Gamma(\|\phi_{s,a}\|+1)\Gamma(\phi_{s,a}(s'))}{\Gamma(\phi_{s,a}(s')+1)}\right)^{\frac{1}{2}}\right).
\end{aligned}$$

This expression can be rewritten using the helper variables and simplified as follows:

$$\begin{aligned}
\rho_B(s,a,s',\mathbf{b}_t) &= -\log\left(\frac{\Gamma(x+1/2)}{\Gamma(y+1/2)\Gamma(x)}\cdot\left(\frac{\Gamma(y)}{1}\right)^{\frac{1}{2}}\cdot\left(\frac{\Gamma(y+1)\Gamma(x)}{\Gamma(x+1)}\right)^{\frac{1}{2}}\right) \\
&= -\log\left(\frac{\Gamma(x+1/2)}{\Gamma(y+1/2)\Gamma(x)}\cdot\left(\frac{\Gamma(y)}{1}\right)^{\frac{1}{2}}\cdot\left(\frac{\Gamma(y+1)\Gamma(x)}{\Gamma(x+1)}\right)^{\frac{1}{2}}\right) \\
&= \log\left(\frac{\Gamma(x)\sqrt{x}}{\Gamma(x+1/2)}\right) - \log\left(\frac{\Gamma(y)\sqrt{y}}{\Gamma(y+1/2)}\right).
\end{aligned}$$

# Annexe B

# Résumé étendu

## Contents



*"Rien de beau ne peut se résumer."* — Paul Valéry

Ce chapitre présente un résumé en français du mémoire de thèse de Mauricio ARAYA-LOPEZ intitulé : *"Near-Optimal Algorithms for Sequential Information-Gathering Decision Problems"*, c'est-à-dire *"Des algorithmes presque optimaux pour les problèmes de décision séquentielle à des fins de collecte d'information"*.

## B.1 Introduction

De manière générale, ce mémoire s'intéresse à la prise de décision rationnelle dans des situations complexes. On y adopte le point de vue de l'intelligence artificielle, cherchant des algorithmes génériques pour automatiser une telle prise de décision. Les problèmes abordés ici mettent en jeu plus précisément des séquences de décisions, l'évolution du système considéré étant incertaine du fait de sa dynamique stochastique (le hasard influence cette évolution) et éventuellement du fait qu'on n'a, à chaque instant, qu'une information incomplète quant à la situation actuelle (les moyens de perception sont limités).

De tels problèmes de prise de décision séquentielle dans l'incertain sont typiquement formalisés comme des processus de décision markoviens (MDP) éventuellement partiellement observables (POMDP). Dans le cas où le modèle du système est mal connu (par exemple sa dynamique), on va parler de problèmes d'apprentissage par renforcement, l'optimisation des décisions nécessitant des interactions avec le système réel, et donc un apprentissage par essai-erreur.

Les problèmes de prise de décisions séquentielle usuels requièrent dans un certain nombre de cas d'agir de manière à acquérir des informations. C'est typiquement le cas :

- dans les POMDP, parce que certaines actions permettront d'en apprendre plus sur la situation courante pour, ensuite, mieux contrôler le système, et
- en apprentissage par renforcement, parce qu'il faut en permanence chercher le meilleur compromis entre effectuer l'action la plus prometteuse (exploiter les connaissances dont on dispose) et effectuer une action qui apportera de l'information sur le modèle (explorer pour acquérir des connaissances).

La collecte d'information (sur l'état courant d'une part, sur le modèle d'autre part) est alors un moyen de résoudre un problème dont l'objet est plutôt de contrôler l'état d'un système.

A l'inverse, le présent travail de thèse s'est principalement intéressé aux problèmes où la collecte d'information est une fin en soi. Cela a amené à aborder les questions suivantes :

- comment modifier le formalisme des POMDP pour exprimer des problèmes de collecte d'information ? (ce qui a amené à l'introduction des $\rho$POMDP) et quels algorithmes employer pour résoudre ces problèmes ?
- peut-on définir des tâches d'apprentissage par renforcement dont le but soit d'apprendre le modèle ? comme on le verra, cette question a amené naturellement à l'apprentissage par renforcement bayésien avec modèle (BRL) ;
- dans la famille des algorithmes de BRL qui sont "optimistes face à l'incertain", est-il possible d'en écrire un qui soit "optimiste *à propos* de l'incertitude", laquelle concerne généralement la dynamique du système ? (ce qui a amené à l'introduction de l'algorithme BOLT) quelles sont les propriétés de cet algorithme ?

Ce mémoire (et son résumé) introduit d'abord aux chapitres 2-3 (section B.2) le problème de la prise de décision séquentielle sous incertitude, et donc les MDP et POMDP. Le chapitre 4 (section B.3) qui suit introduit les $\rho$POMDP et montre comment adapter simplement des algorithmes usuels de résolution. Vient alors le chapitre 5 (section B.4) sur l'apprentissage par renforcement, lequel fournit les connaissances utiles au chapitre 6 (section B.5) sur l'algorithme BOLT ainsi qu'au chapitre 7 (section B.6) sur l'apprentissage actif de modèle. Certaines preuves sont développées en annexe.

## B.2 Prise de décision séquentielle sous incertitude

La prise de décision séquentielle consiste à trouver les meilleures lois possibles pour contrôler l'évolution d'un système dynamique (à temps discret) dont on connaît le fonctionnement (le modèle). Les travaux dans ce domaine diffèrent selon les hypothèses de travail faites : système déterministe ou stochastique, observabilité totale ou partielle, un ou plusieurs "preneurs de décision" (collaborant ou non), algorithme hors-ligne ou en-ligne... Nous allons considérer ici le cas de systèmes stochastiques, aussi bien avec observabilité totale que partielle, ce qui correspond aux formalismes des processus de décision markoviens totalement ou partiellement observables (MDP et POMDP).

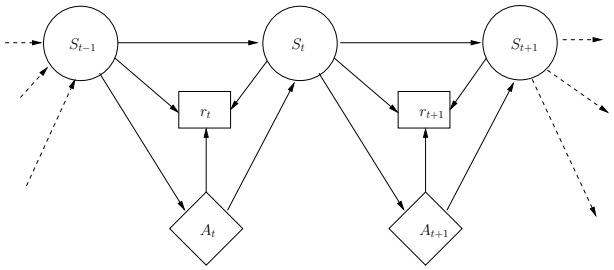



FIGURE B.1 – Une vue en 3 étapes du modèle graphique d'un processus de décision markovien.

### B.2.1 Processus de décision markoviens

Un *processus de décision markovien* (MDP) (Puterman, 1994) est défini par un tuple $\langle \mathcal{S}, \mathcal{A}, T, r, s_0 \rangle$ décrivant :

- l'ensemble $\mathcal{S}$ des états possibles du système à contrôler ;
- l'ensemble $\mathcal{A}$ des actions qui peuvent être effectuées pour influencer la dynamique de ce système ;
- la fonction de transition $T(s, a, s')$ décrivant la probabilité de transiter de l'état $s$ vers l'état $s'$ quand l'action $a$ est effectuée ($T(s, a, s') = Pr(S_{t+1} = s' | S_t = s, A_{t+1} = a)$) ;
- la fonction de récompense $r(s, a, s')$ décrivant la récompense reçue quand la transition $s \xrightarrow{a} s'$ a lieu ; et
- un état initial $s_0 \in \mathcal{S}$.

La figure B.1 montre un processus de décision markovien vu comme un modèle graphique, en ne représentant que 3 pas de temps.

Un tel processus vérifie la *propriété de Markov*, c'est-à-dire que les prévisions quant à son évolution futures peuvent être faites sans mémoire des états et actions passées du moment que l'état courant est connu :

$$Pr(S_{t+1} | S_t, A_{t+1}, S_{t-1}, A_t, \cdots, S_0) = Pr(S_{t+1} | S_t, A_{t+1}).$$

Cette propriété permettra ici de prendre des décisions sans avoir besoin de mémoire du passé.

Comme on peut le remarquer, la définition de ce qu'est un MDP ne détaille pas complètement quel problème il faut résoudre. On a bien une fonction de récompense, mais ce n'est pas un critère à optimiser (ou une propriété à satisfaire). A l'instar de nombreux travaux, nous allons considérer un critère total décompté, et donc chercher une *politique* de décision $\pi$ –dont on précisera la définition ultérieurement– maximisant la quantité :

$$E_H^\pi \left[ \sum_{t=0}^{H} \gamma^t R_t | S_0 = s_0 \right],$$

où $\gamma \in [0; 1]$ est un facteur d'atténuation traduisant une dévalorisation des récompenses futures, $H \in 0..\infty$ est l'horizon temporel –éventuellement infini– du problème, et $R_t$ est la variable aléatoire associée à la récompense reçue à l'instant $t$. Si le facteur $\gamma$ est parfois justifié par le problème à résoudre, le choix d'un $\gamma$ strictement inférieur à 1 est souvent imposé, dans les problèmes à horizon infini, par la nécessité que la somme des récompenses (le *retour*) ne diverge pas.

On pourrait se ramener à un problème générique d'optimisation avec comme espace de recherche celui des politiques, ce qui est parfois une bonne solution. Il reste toutefois généralement préférable d'exploiter le caractère temporel de ce problème de décision. Nous allons donc voir maintenant comment la programmation dynamique permet de résoudre ce problème, en commençant par le cas des horizons temporels finis.

#### Horizon temporel fini

A horizon fini, la décision à prendre dans un état $s$ peut logiquement dépendre du temps restant (de l'horizon $n$). En effet, certaines récompenses futures importantes ne seront accessibles que s'il reste

suffisamment de temps pour les atteindre. Cela nous amène à définir ici une politique $\pi$ comme une application $\pi : \mathcal{S} \times \mathbb{N} \to \mathcal{A}$.[37] Pour des raisons pratiques, on écrira $\pi_n(s)$ au lieu de $\pi(s, n)$.

L'espérance du retour si l'on suit la politique $\pi$ sur un horizon $n$ à partir d'un état $s$ est appelée *valeur* :

$$\begin{aligned} V_n^\pi(s) &= E_\pi \left[ \sum_{t=0}^{n} \gamma^t R_t \middle| S_0 = s \right], \\ &= E_\pi \left[ \sum_{t=0}^{n} \gamma^t r(S_t, \pi_{n-t}(S_t), S_{t+1}) \middle| S_0 = s \right]. \end{aligned}$$

On peut alors déduire une forme récurrente, et expliciter son calcul :

$$\begin{aligned} & V_0^\pi(s) = 0 \\ \forall n > 0, \quad & V_n^\pi(s) = E_\pi \left[ \sum_{t=0}^{n} \gamma^t r(S_t, \pi_{n-t}(S_t), S_{t+1}) \middle| S_0 = s \right], \\ & = E_\pi \left[ r(S_0, \pi_n(S_0), S_1) + \sum_{t=1}^{n} \gamma^t r(S_t, \pi_{n-t}(S_t), S_{t+1}) \middle| S_0 = s \right], \\ & = E_\pi \left[ r(S_0, \pi_n(S_0), S_1) + \gamma V_{n-1}^\pi(S_1) \middle| S_0 = s \right], \\ & = \sum_{s' \in \mathcal{S}} T(s, \pi_n(s), s') \left[ r(s, \pi_n(S_0), s') + \gamma V_{n-1}^\pi(s') \right]. \end{aligned}$$

Ce calcul de $V_n^\pi$ en fonction de $V_{n-1}^\pi$ définit l'*opérateur de Bellman pour la politique* $\pi$ :

$$V_n^\pi = \mathbb{B}^\pi V_{n-1}^\pi.$$

Pour des raisons pratiques, on introduit la *fonction de valeur d'action* associée à la politique $\pi$, définie par :

$$\begin{aligned} & Q_0^\pi(s, a) = 0 \\ \forall n > 0, \quad & Q_n^\pi(s, a) = \sum_{s' \in \mathcal{S}} T(s, a, s') \left[ r(s, \pi_n(s), s') + \gamma V_{n-1}^\pi(s') \right], \\ \text{d'où} \quad & V_n^\pi(s) = Q_n^\pi(s, \pi(s)). \end{aligned}$$

La *programmation dynamique* repose sur le *principe d'optimalité de Bellman* (Bellman, 1954) d'après lequel une action $a$ est optimale dans l'état $s$ à horizon $n$ si elle maximise l'espérance de la récompense instantanée, plus la *valeur* optimale de l'état suivant (et donc à horizon $n-1$). En notant $V_n^*(s)$ la valeur optimale dans l'état $s$ et pour un horizon $n$, ce principe s'écrit :

$$\begin{aligned} V_n^*(s) &= \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} T(s, a, s') \left[ r(s, a, s') + \gamma V_{n-1}^*(s') \right] \\ &= \max_{a \in \mathcal{A}} Q_n^*(s, a), \end{aligned}$$

où $Q_n^*(s, a)$ est la *fonction de valeur d'action optimale*, ce qui permet de définir *l'opérateur d'optimalité de Bellman* :

$$V_n^* = \mathbb{B}^* V_{n-1}^*.$$

On a ainsi une procédure récurrente pour calculer la fonction de valeur optimale. Une stratégie gourmande par rapport à la fonction d'action valeur donne alors l'ensemble des actions optimales :

$$\pi_n^*(s) \in \arg\max_{a \in \mathcal{A}} Q_n^*(s, a).$$

37. Il est démontré que, dans les cas qui nous intéressent ici, il n'est pas nécessaire de considérer des politiques stochastiques, c'est-à-dire associant à chaque état une distribution de probabilité sur les actions.

**Horizon temporel infini**

L'horizon temporel infini peut être considéré comme le cas limite de l'horizon temporel fini quand celui-ci croît. Quand $\gamma < 1$, la fonction valeur (d'action) tend alors vers une fonction limite $V^*_\infty$ ($Q^*_\infty$) à laquelle est associée une politique $\pi^*_\infty$. On obtient ainsi une politique stationnaire (on omettra souvent l'indice d'horizon).

$V^*_\infty$ peut être aussi vu comme l'unique solution de :

$$\forall s \in \mathcal{S},\ V(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} T(s,a,s') \left[ r(s,a,s') + \gamma V(s') \right].$$

Le fait que $\gamma$ soit strictement plus petit que 1 garantit que les opérateurs de Bellman (d'optimalité, ou pour une politique donnée) sont contractants, donc qu'il existe un unique point fixe à ces opérateurs, et qu'un tel point fixe est le point limite obtenu en itérant ces opérateurs.

L'algorithme d'itération sur la valeur (*Value Iteration* ou VI) consiste précisément à itérer l'opérateur d'optimalité de Bellman jusqu'à ce qu'un certain critère d'arrêt soit atteint. Cet algorithme est dit *synchrone* parce que les valeurs associées à tous les états sont calculées avant de passer à la prochaine itération.

**Algorithm 7:** Itération sur les valeurs (synchrone) (VI)

**Input**: $T$, $R$, $\gamma$, $s_0$, $V_0$
**Output**: $\hat{V}^*$

```
n ← 1;
tant que V n'a pas convergé faire
    pour chaque s ∈ S faire
        V*_n(s) ← 𝔹(V*_{n−1}, s);
    fin
    n ← n + 1;
fin
```

Une variante plus efficace, parce qu'utilisant toute nouvelle valeur dès qu'elle est calculée, et plus simple à implémenter consiste à n'utiliser qu'une table de valeurs $V(\cdot)$ comme dans l'algorithme d'itération sur la valeur *asynchrone* (*Asynchronous VI* ou AVI) présenté dans l'algorithme 8.

**Algorithm 8:** Itération sur la valeur asynchrone (AVI)

**Input**: $T$, $R$, $\gamma$, $s_0$, $V_0$
**Output**: $V^*$

```
V ← V_0;
tant que V n'a pas convergé faire
    pour chaque s ∈ S faire
        V(s) ← 𝔹(V, s);
    fin
fin
```

Un critère courant est de s'arrêter dès que l'erreur de Bellman $\|V^*_n - V^*_{n-1}\|_\infty$ est plus petite que $\frac{\epsilon}{1-\gamma^2}$ pour $\epsilon > 0$ donné, ce qui garantit que :

$$\begin{aligned} \|V^* - V_n\|_\infty &= \max_{s \in \mathcal{S}} |V^*(s) - V_n(s)| \\ &\leq \epsilon. \end{aligned}$$

On notera aussi que, si ces deux algorithmes n'exploitent pas la connaissance de l'état initial, d'autres le font et permettent ainsi de réduire les calculs effectués.

### B.2.2 Processus de décision markoviens partiellement observables

Dans de nombreux problèmes, l'état du système considéré n'est pas directement observable. On va modéliser ceux-ci comme des MDP partiellement observables (POMDP) (Åström, 1965; Sigaud and Buffet,

2010), définis par un uplet $\langle \mathcal{S}, \mathcal{A}, \mathcal{Z}, T, O, r, \mathbf{b}_0 \rangle$, où :
- $\mathcal{S}$, $\mathcal{A}$, $T$ et $r$ sont définis comme pour les MDP ;
- $\mathcal{Z}$ est un ensemble d'observations possibles ;
- $O$ est une fonction d'observation donnant la probabilité de percevoir une observation $z$ quand une action $a$ amène dans un état $s'$ : $O(s', a, z) = Pr(Z_t = z | S_t = s', A_t = a)$ ; et
- $\mathbf{b}_0 \in \Delta$ est une distribution de probabilité sur les états possibles –appelée *croyance initiale* (ou *belief*)– donnant la connaissance sur l'état initial.

Dans un POMDP, on ne peut prendre des décisions optimales en fonction de l'état courant puisque celui-ci n'est pas connu. En outre, prendre une décision en fonction de la dernière observation reçue –ou des $N$ dernières observations (et actions)– ne garantira en général pas son optimalité. Il faut idéalement prendre une décision en fonction de l'historique complet des actions et observations passées, ou d'une information suffisante.

La *croyance* courante $\mathbf{b}_t$ (distribution de probabilité sur les états possibles à l'instant $t$) est une telle information suffisante (et nécessaire, sauf cas particulier) pour prendre une décision optimale dans un POMDP. Sa mise à jour en fonction de la dernière action effectuée et de l'observation qui l'a suivie est une application directe de la règle de Bayes :

$$\mathbf{b}^{a,z}(s') \propto Pr(z|a, s') \sum_{s \in \mathcal{S}} Pr(s'|a, s)\mathbf{b}(s).$$

En fait, on peut ramener le problème de la prise de décision dans un POMDP au problème de la prise de décision dans un MDP dans lequel
- l'espace d'état est l'espace $\Delta$ des états de croyance du POMDP de départ ;
- l'espace des actions est le même ;
- la fonction de transition $\tau$ se déduit de $T$ et $O$ en utilisant la mise à jour de $b$ vue précédemment et :

$$\begin{aligned}\tau(\mathbf{b}, a, \mathbf{b}') &= \sum_{z \in \mathcal{Z}} \mathbb{I}(\mathbf{b}, \mathbf{b}^{a,z}) \left[ \sum_{s,s' \in \mathcal{S}} Pr(z|s', a) Pr(s'|s, a) Pr(s) \right] \\ &= \sum_{z \in \mathcal{Z}} \mathbb{I}(\mathbf{b}, \mathbf{b}^{a,z}) \left[ \sum_{s,s' \in \mathcal{S}} O(s', a, z) T(s, a, s') \mathbf{b}(s) \right],\end{aligned}$$

  où $\delta(\cdot, \cdot)$ est le delta de Kronecker ; et
- la fonction de récompense $\rho$ se déduit de $r$ par :

$$\rho(\mathbf{b}, a, \mathbf{b}') = \sum_{s,s' \in \mathcal{S}} \mathbf{b}(s)\mathbf{b}'(s')r(s, a, s').$$

*Note* – Dorénavant, de manière à clarifier la rédaction et sans perte de généralité, on considérera des récompenses $r(s, a)$ et $\rho(\mathbf{b}, a)$ (au lieu de $r(s, a, s')$ et $\rho(\mathbf{b}, a, \mathbf{b}')$).

On s'est donc, en théorie, ramené de la résolution d'un POMDP à celle d'un MDP, à ceci près que le MDP en question a un nombre d'états (de croyance) accessibles souvent infini. L'ensemble des états de croyance est un espace continu à $|\mathcal{S}| - 1$ dimensions. On ne peut donc recourir aux algorithmes usuels directement. De premières approches incluent par exemple :
- développer l'arbre des futurs possibles depuis $\mathbf{b}_0$ et utiliser la programmation dynamique dans des problèmes à horizon fini quand le facteur de branchement est suffisamment petit ;
- dans le cas de problèmes à horizon infini, décider de $a_t$ à chaque instant $t$ de décision en raisonnant sur un horizon fini (algorithme en ligne) comme précédemment ; ou
- discrétiser l'espace des croyances.

En pratique, de nombreux algorithmes reposent sur une propriété importante de la fonction de valeur (Smallwood and Sondik, 1973; Sondik, 1978) : pour tout horizon $h \geq 0$, la fonction de valeur $V_h^*$ est convexe et linéaire par morceaux (PWLC pour *Piecewise-Linear and Convex*). Cette propriété est immédiate pour $h = 0$ puisque $V_0^*(\mathbf{b}) = 0$ pour tout $\mathbf{b}$, et s'obtient par récurrence pour $h > 1$ parce que

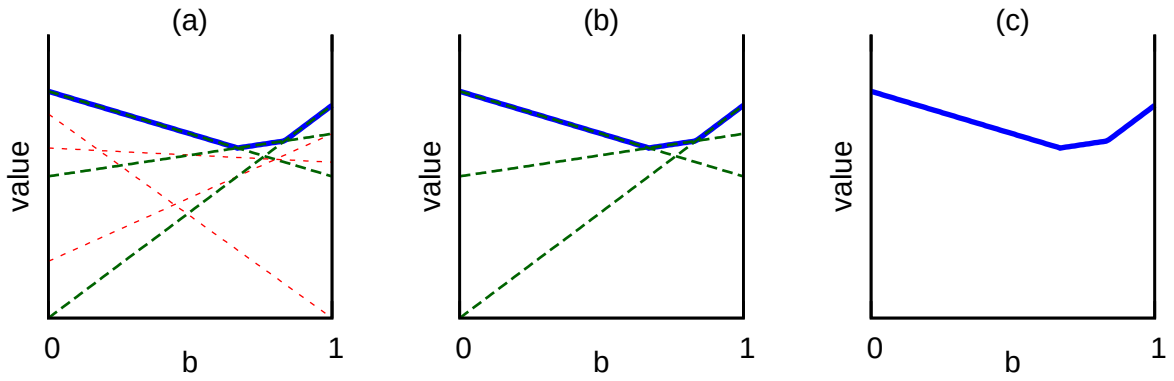



FIGURE B.2 – Représentation sous la forme d'un Γ-set (ensemble de vecteurs $\alpha$) d'une fonction de valeurs dans un 1-simplex avec (a) et sans (b) les hyperplans superflus. (c) montre la vraie fonction de valeur. Ici, (b) est une représentation parcimonieuse de (c).

- les opérateurs max, + et × préservent la linéarité et la convexité, et
- la fonction de récompense est linéaire en $\mathbf{b}$.

On notera que, dans les problèmes à horizon infini, la fonction de valeur stationnaire optimale est aussi convexe, mais pas nécessairement linéaire par morceaux.

Il est alors naturel d'écrire, à chaque itération, la fonction de valeur comme l'enveloppe convexe d'un ensemble de droites. En pratique, chacune est associée à un vecteur $\alpha$ qui la détermine (ainsi qu'à une action), et on écrit :

$$V_h^*(\mathbf{b}) = \max_{\alpha \in \Gamma_h} \mathbf{b}^\top \alpha.$$

La figure B.2 (a) montre une telle enveloppe décrivant la fonction de valeur présentée en figure B.2 (c). On peut ainsi dériver un algorithme de programmation dynamique exact à horizon fini, et une version exacte de l'algorithme d'itération sur la valeur dans les cas à horizon infini.

Malheureusement le nombre d'$\alpha$-vecteurs croît de manière exponentielle à chaque itération (Shani et al., 2012). Toutefois, un grand nombre d'entre eux sont superflus, parce que les hyperplans associés sont dominés en tout point par d'autres hyperplans, comme illustré par la figure B.2 (b). Une tâche importante est donc de débarrasser (élaguer) l'ensemble $\Gamma_h$ des $\alpha$-vecteurs dominés. Cela peut se faire à différentes étapes des calculs, ce qui a donné lieu à différents algorithmes tels que *Batch Enumeration* (Monahan, 1982) et *Incremental Pruning* (IP) (Cassandra et al., 1997).

Une famille importante d'algorithmes (Shani et al., 2012) va encore plus loin en élaguant aussi nombre d'hyperplans non complètement dominés. Il s'agit donc d'approcher au mieux la fonction de valeur avec un nombre limité d'$\alpha$-vecteurs, identifiés à des "points de croyances" (*belief points*) particuliers de $\Delta$. Un point clef dans ces algorithmes *à base de points* est la méthode de sélection des points en question. On citera parmi les algorithmes de référence de la littérature PBVI (Pineau et al., 2003), PERSEUS (Spaan and Vlassis, 2005) et HSVI2 (Smith and Simmons, 2005), ainsi que de plus récents comme FSVI (Shani et al., 2007), SARSOP (Kurniawati et al., 2008) et GapMin (Poupart et al., 2011).

On notera que les algorithmes exacts (avec élagage du strict nécessaire) ne se servent pas de la connaissance de la croyance initiale, alors que les algorithmes à base de points se servent souvent de cette connaissance pour sélectionner des points utiles intelligemment.

Que ce soit pour la résolution de MDP ou de POMDP, de nombreuses autres approches existent. Notre présentation s'est concentrée sur les approches les plus courantes, sur lesquelles les travaux qui suivent vont reposer.

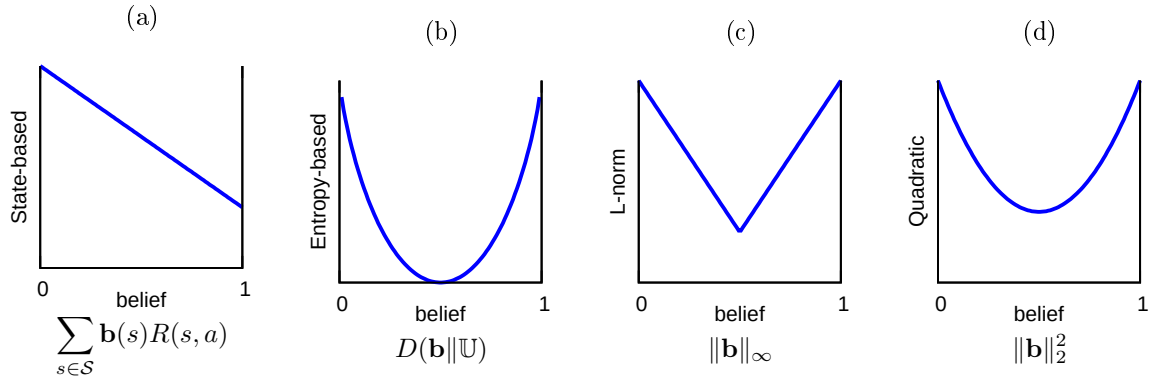



FIGURE B.3 – Récompenses dépendant de l'information dans un 1-simplex : (a) dépendant de l'état, (b) divergence KL, (c) norme $L_\infty$ et (d) carré de la norme $L_2$.

## B.3 Problèmes de décision pour la collecte d'informations

Les processus de décision markoviens, qu'ils soient complètement ou partiellement observables, sont faits pour contrôler l'état d'un système, lui faire atteindre un état but ou le maintenir dans un certain régime de fonctionnement. Toutefois, dans certains problèmes sous observabilité partielle, l'objectif peut être simplement d'acquérir de l'information, de réduire l'incertitude sur une variable cachée. C'est en pratique à cela que s'intéressent certains travaux sur la conception d'expérimentations (*experiments design*), les capteurs actifs ou le diagnostic actif. [38] Mais, sauf dans certains cas, ces problèmes ne peuvent être formulés directement à l'aide des POMDP, simplement parce que la fonction de récompense dépend de l'état du système, pas de l'information disponible (de la croyance). C'est ce qui nous a amené à proposer une extension de ce formalisme, les $\rho$POMDP, et à étudier la recherche de solutions de ces problèmes de prise de décision séquentielle.

### B.3.1 Problèmes et formalisation

Dans un POMDP, la récompense $r$ dépend des état $s$ et action $a$ courants, le belief MDP induit voyant sa fonction de récompense $\rho$ –dépendant de la croyance et de l'action– calculée comme une espérance sur les états possibles. Dans nos problèmes, nous souhaitons que la récompense soit toujours une fonction de la croyance, à laquelle on a d'ailleurs accès, plutôt que de l'état. On va donc directement définir la fonction $\rho$ qui sera directement utilisée aussi bien dans notre nouveau type de POMDP –appelé $\rho$POMDP– que dans le belief MDP induit. La figure B.3 présente quelques formes de fonctions $\rho$ possibles définies ici sur un 1-simplex. On notera que ce nouveau formalisme ne fait qu'étendre celui des POMDP. Tout POMDP peut être trivialement ré-écrit comme un $\rho$POMDP.

Dans les problèmes qui nous intéressent, la récompense doit mesurer une information. Il peut par exemple s'agir de l'opposé de l'entropie de Shannon ($H(\mathbf{b}) = \sum_{s\in\mathcal{S}} -\mathbf{b}(s)\log(\mathbf{b}(s))$), de la divergence de Kullback-Leibler ou de la distance de Chernoff. Il y a toutefois une variété de scénarios possibles, tels que :

**diagnostic :** dans un problème de diagnostic, on effectue une séquence d'actions de manière à acquérir de l'information sur une variable cachée statique, et ce jusqu'à terminaison (souvent volontaire) du processus ; seule la qualité de l'information obtenue au final est importante ; la récompense peut être nulle (ou associée au coût des actions) le reste du temps ;

**surveillance :** dans un problème de surveillance, on doit indéfiniment maintenir une certaine qualité de l'information relative à certaines variables cachées dynamiques ; la récompense est donc liée à

38. Si l'information n'est jamais une fin en soi, il est de nombreuses situations dans lesquelles il est pertinent de poser un problème en termes de recherche d'informations.

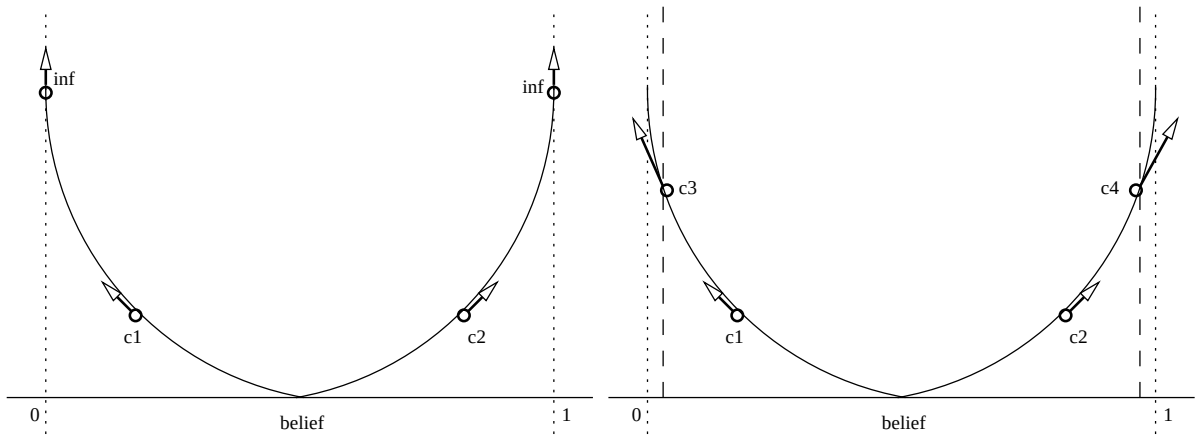



FIGURE B.4 – A gauche, une fonction non lipschitzienne mais convexe. A droite, la pente obtenue en s'éloignant du bord.

chaque pas de temps à une mesure d'information ;

**localisation :** dans un problème de localisation (de soi), une entité mobile dans un certain espace doit se déplacer jusqu'à savoir suffisamment bien où elle se trouve ; ici la variable cachée est non seulement dynamique, mais en plus contrôlée ; comme dans le problème du diagnostic, c'est surtout l'information finale que l'on va optimiser.

Il reste maintenant à voir comment résoudre les problèmes ainsi définis.

### B.3.2 Calcul de la fonction de valeur

Les approches présentées en section B.2.2 reposent sur le fait que, pour un POMDP donné et un horizon $h$, $V_h^*$ est linéaire par morceaux et convexe (PWLC), ce qui reposait sur la linéarité de la fonction $\rho$ induite à partir de $r$. Or $\rho$ n'est en général plus linéaire dans les $\rho$POMDP, ce qui remet en cause les résultats établis.

Dans le cas général, $\rho$ n'est elle-même pas nécessairement PWLC, ce qui rend d'emblée invalide la propriété fondamentale de $V_h^*$. Par contre on montre très facilement que, si $\rho$ est PWLC, $V_h^*$ est PWLC pour tout $h$.

Raisonner sur les mises à jour de $V_h^*$ de manière à obtenir une solution en forme fermée n'est de manière générale plus possible. Par contre on peut facilement faire l'hypothèse que la fonction $\rho$ doit être convexe (c'est souvent le cas) et que, dans ce cas, $V_h^**$ est aussi convexe. L'extension de la propriété fondamentale de $V_h^*$ au cas d'une récompense PWLC amène donc à se demander s'il ne serait pas possible d'approcher $\rho$, et donc $V_h^*$ par des fonctions PWLC.

Borner l'erreur en approximation d'une fonction convexe dérivable (au moins par morceaux) et de gradient borné est aisé. Malheureusement un certain nombre de fonctions permettant de mesurer l'information ont un gradient non borné, comme par exemple la divergence de Kullback-Leibler (parce que la dérivée de $f(x) = x\log(x/c)$ tend vers l'infini quand $x$ tend vers 0), et tel qu'illustré par la figure B.4 (partie de gauche). Nombre de ces fonctions sont par contre $\alpha$-hölderienne pour un $\alpha \in [0;1)$ donné, c'est-à-dire qu'il existe $K_\alpha > 0$ tel que, pour tout $\mathbf{b}, \mathbf{b}'$,

$$|\rho(\mathbf{b}) - \rho(\mathbf{b}')| \leq K_\alpha \|\mathbf{b} - \mathbf{b}'\|_1^\alpha.$$

Un premier résultat théorique important est une borne d'erreur sur l'approximation PWLC d'une fonction convexe $\alpha$-hölderienne, en fonction de la densité de l'ensemble $\mathcal{B}$ de points échantillonnés pour construire cette approximation. [39]

39. Densité de $\mathcal{B}$ : $\delta_\mathcal{B} = \min_{\mathbf{b}\in\Delta} \max_{\mathbf{b}'\in\mathcal{B}} \|\mathbf{b} - \mathbf{b}'\|_1$.

**Theorem B.3.1.** *Soit $\rho$ une fonction continue et convexe sur $\Delta$, dérivable partout dans $\Delta^o$ (l'intérieur de $\Delta$), et satisfaisant la condition $\alpha$-Hölder avec une constante $K_\alpha$. L'erreur de l'approximation $\rho_\mathcal{B}$ est majorée par $C{\delta_\mathcal{B}}^\alpha$, où $C$ est une constante scalaire qui dépend de $\alpha$.*

Ce résultat s'obtient en n'échantillonnant des points que s'ils sont au moins à une certaine distance des bords de $\Delta$, comme illustré par la figure B.4. Il permet ensuite de borner l'erreur sur la fonction de valeur obtenue avec des mises à jour exactes.

**Theorem B.3.2.** *Soit $\bar{V}_h$ la fonction de valeur obtenue en employant l'approximation PWLC $\rho_\mathcal{B}(\mathbf{b})$, et soit $V_h^*$ la fonction de valeur optimale, toutes deux prises pour l'horizon $h$. L'erreur en approximation est majorée par*

$$\|\bar{V}_h - V_h^*\|_\infty \leq \frac{C{\delta_\mathcal{B}}^\alpha}{1-\gamma}.$$

On peut ensuite naturellement étendre ce résultat au cas de l'application de l'algorithme approché PBVI.

**Theorem B.3.3.** *Soit $\hat{V}_h$ la fonction de valeur calculée en employant l'approximation PWLC de $\rho$ et l'algorithme PBVI, et soit $V_h^*$ la fonction de valeur optimale, toutes deux prises pour l'horizon $h$. L'erreur en approximation est majorée par*

$$\|\hat{V}_h - V_h^*\|_\infty \leq \frac{(R_{max} - R_{min} + C{\delta_\mathcal{B}}^\alpha)\delta_\mathcal{B} + C{\delta_\mathcal{B}}^\alpha}{1-\gamma}.$$

Ces résultats permettent de résoudre des $\rho$POMDP à l'aide d'algorithmes à base de points, lesquels ont prouvé leur efficacité pour les POMDP. On notera toutefois que certaines heuristiques (par exemple $Q$-MDP) perdent leur intérêt.

### B.3.3 Expérimentations

Des expérimentations des algorithmes obtenus ont été conduites sur deux problèmes illustrés par la figure B.5 : *Camera Clean* (dans trois variantes : diagnostic, surveillance et localisation) et *Rock Diagnosis* (problème inspiré de *Rock Sampling* (Smith and Simmons, 2004)). Différentes tailles de problèmes ont été considérées, ainsi que différents algorithmes, afin de comparer les approches à base de points (avec un seul critère d'optimisation, mais en essayant trois récompenses différentes pour la phase de planification) avec des approches myopes et des décisions prises au hasard.

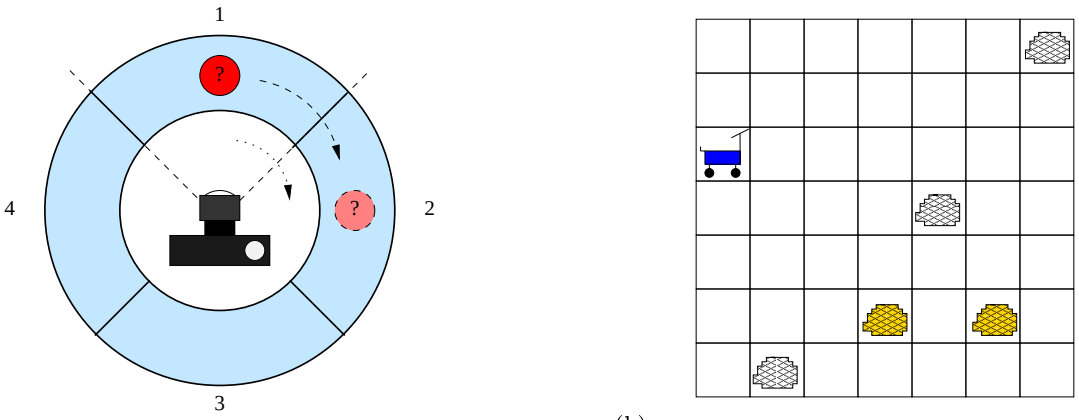
Les résultats obtenus montrent que l'utilisation de récompenses dépendant de l'information est généralement préférable, avec une meilleure efficacité de celle fondée sur l'entropie si une approximation dense est possible, et de la récompense "linéaire" si l'approximation doit être parcimonieuse.

### B.3.4 Travaux futurs

Un point important est de chercher des méthodes efficaces pour échantillonner des points pour l'approximation de $\rho$, comme cela a été fait pour l'approximation de $V$. Cela amènerait par exemple à employer des ensembles de points dynamiques et une sélection visant à minimiser l'erreur d'approximation.

Aussi, du fait de la complexité des algorithmes employés, il parait nécessaire d'exploiter la structure des problèmes considérés, par exemple quand certaines variables d'état sont visibles et d'autres cachées, ce qui conduit au formalisme des MDP à observabilité mixte (Araya-López et al., 2010bc; Ong et al., 2009).

Enfin, des expérimentations supplémentaires permettraient d'encore mieux comprendre les résultats dans le cas sensible aux coûts, et étudier en particulier l'effet du paramètre $\eta$ qui permet de passer des POMDP classiques à l'acquisition d'information.




(a) (b)

FIGURE B.5 – (a) Représentation graphique du problème *Camera-clean* à 4 zones, dans lequel une cible peut se déplacer en avançant d'une zone à l'autre, et la caméra doit trouver cette cible, mais en ayant besoin de nettoyer régulièrement sa lentille. (b) Représentation graphique du problème *Rock Diagnosis* avec 5 pierres et une grille de côté 7. Les pierres jaunes sont les "bonnes" pierres (avec une valeur scientifique) et les autres sont les "mauvaises" pierres (sans valeur scientifique), mais le type de chaque pierre n'est pas connu de l'agent. Notons que, pour un diagnostic, une "bonne" pierre n'est pas plus importante qu'une "mauvaise" puisque l'objectif est de les distinguer toutes, et non d'échantillonner seulement les "bonnes" comme dans le problème *Rock Sampling* original.

## B.4 Apprentissage par renforcement

L'apprentissage par renforcement consiste, comme on l'a fait jusqu'ici, à chercher une politique optimale pour un problème de décision séquentielle, mais dans une situation où le modèle n'est pas connu. Il faut donc acquérir des informations sur la dynamique du système (et éventuellement la fonction de récompense) en interagissant avec celui-ci. Il s'agit donc en un sens d'apprendre par essai-erreur. Mais il n'est pas question d'apprendre dans une première phase, et d'exploiter les connaissances obtenues dans une seconde phase. Exploration et exploitation sont deux problématiques concomitantes. L'agent a donc à gérer à chaque instant le dilemme exploration-exploitation, c'est-à-dire choisir entre agir pour explorer le système (et acquérir des connaissances) et agir pour exploiter les connaissances acquises.

### B.4.1 Cas avec modèle bayésien

On va s'intéresser ici aux approches d'apprentissage par renforcement qui cherchent le vrai modèle $\mu$ dans l'ensemble des modèles possibles $\mathcal{M}$ (d'autres approches optimisant directement une politique). En particulier, on va considérer le cas d'une estimation (d'un apprentissage) bayésien du modèle, c'est-à-dire qu'on maintient à chaque instant une distribution de probabilité sur les modèles possibles, chaque transition étant exploitée pour mettre à jour cette distribution par simple application de la règle de Bayes. On est donc dans le cadre de l'apprentissage par renforcement bayésien (avec modèle) ou BRL (*(model-based) Bayesian Reinforcement Learning*), et on associe une variable aléatoire $M$ au modèle. En outre, il faut définir une croyance initiale sur le modèle, l'*a priori* de l'agent sur les modèles possibles.

### B.4.2 Représentation

Sans perte de généralité, on va supposer ici la fonction de récompense connue. Considérant des ensembles d'états et d'actions finis, on va naturellement modéliser la fonction de transition par une distribution discrète (catégorielle) pour chaque paire état-action.

Le cadre du BRL implique que l'on maintient une distribution de probabilité sur l'ensemble de ces $|\mathcal{S}| \times |\mathcal{A}|$ distributions. On va typiquement travailler avec des distributions conjuguées, de sorte que la mise à jour induite par toute transition nous fait rester dans la même famille de distributions, avec un

calcul simple des nouveaux paramètres. Une approche naïve est d'employer une distribution de Dirichlet par couple état-action (factorisant ainsi le modèle global en $|\mathcal{S}| \cdot |\mathcal{A}|$ modèles locaux). En notant $\theta^{s,a}$ la variable aléatoire associée au couple $(s,a)$, et $\phi_{s'}^{s,a}$ le paramètre associé à la transition $s \xrightarrow{a} s'$, on a :

$$E\left[Pr(s'|s,a)|b\right] = E\left[\theta_{s'}^{s,a}|b\right] = \frac{\phi_{s'}^{s,a}}{\|\phi^{s,a}\|_1}.$$

Et la mise à jour des paramètres se fait par l'incrément de $\phi_{s'}^{s,a}$ quand la transition $s \xrightarrow{a} s'$ a lieu.

D'autres distributions sont toutefois préférables en pratique, de manière à exploiter la structure inhérente à chaque problème pour accélérer l'apprentissage.

### B.4.3 Problème d'optimisation

Le cadre du BRL permet de définir un critère d'optimisation comme suit Duff (2002); Dimitrakakis (2008) :

**Definition B.4.1.** *Soit $\phi$ les paramètres a priori d'une distribution sur les modèles du MDP, tels que $Pr(M) = \mathbb{P}_\phi$. L'évaluation bayésienne $\mathbb{V}$ d'une politique $\pi$ est alors l'espérance de la fonction de valeur étant donnés les paramètres $\phi$ de la distribution sur les modèles :*

$$\mathbb{V}^\pi(s,\phi) = E_M[V_M^\pi(s)|\phi] = \int_{\mu\in\mathcal{M}} V_\mu^\pi(s)\mathbb{P}_\phi(\mu)d\mu.$$

Cette formulation de l'apprentissage par renforcement permet d'avoir un problème d'optimisation bien défini. A l'inverse les autres formulations usuelles ne disposent pas de données suffisantes pour gérer le dilemme exploration-exploitation. Par contre le BRL requiert un a priori (une distribution initiale) sur les modèles possibles.

Tout problème de BRL peut être vu comme un POMDP (ou un *Bayes-Adaptive* MDP) dans lequel l'espace des états est augmenté d'une variable cachée correspondant aux paramètres du modèle. On peut mesurer la complexité du BRL en notant que l'espace d'état obtenu est infini. Le belief MDP correspondant est donc continu et infini-dimensionel, ce qui exclut d'utiliser la plupart des algorithmes pour POMDP usuels.

### B.4.4 Approches de résolution

On peut distinguer trois grands types d'approches pour traiter les problèmes de BRL :

***belief-lookahead*** **:** ces premières approches cherchent explicitement à traiter le vrai problème d'optimisation posé (Poupart et al., 2006; Dimitrakakis, 2008), mais avec des méthodes approchées ;

***undirected*** **:** ces approches effectuent des actions d'exploration sans tenir compte de la connaissance courante ; on peut citer par exemple l'emploi de stratégies $\epsilon$-gourmandes (une action est prise au hasard avec probabilité $\epsilon \in (0;1]$) ;

***optimistic*** **:** ici l'exploration suit l'idée qu'il faut être optimiste face à l'incertain, ce qui incite à réduire l'incertitude sur le modèle (Brafman and Tennenholtz, 2003; Kolter and Ng, 2009; Sorg et al., 2010; Asmuth et al., 2009).

#### BEB et BOSS

Un exemple d'approche optimiste est BEB (*Bayesian Exploration Bonus*). Dans celle-ci, à chaque instant de décision, un MDP est résolu en utilisant le modèle de transition moyen courant, et le modèle moyen de récompense augmenté d'un bonus incitant à l'exploration :

$$r'(s,a) = r(s,a) + \frac{\beta}{1+n(s,a)},$$

où $\beta > 0$ est une constante, et $n(s,a)$ est le nombre d'occurrences de l'action $a$ dans l'état $s$. On notera que, même si l'incertitude est liée à la fonction de transition, c'est à travers la fonction de récompense que BEB est optimiste.

Un second exemple est BOSS (*Best Of Sampled Set*). Ici aussi, un MDP est résolu à chaque instant de décision. Mais celui-ci est obtenu en échantillonnant $m$ modèles d'après la distribution courante, et en dérivant de ceux-ci un modèle optimiste.

### B.4.5 Algorithmes probablement approximativement corrects (PAC)

A défaut d'être optimaux, on souhaiterait que les algorithmes que l'on emploie soient *approximativement corrects*, c'est-à-dire que, pour un $\epsilon > 0$ donné, et pour tout $s \in \mathcal{S}$,

$$V_T^*(s) - V_T^{\mathcal{A}}(s) < \epsilon,$$

où $V_T^*$ est la fonction de valeur optimale réelle pour la fonction de transition (inconnue) $T$, et $V_T^{\mathcal{A}}$ est la fonction de valeur associée à l'algorithme $\mathcal{A}$. Mais cette propriété ne peut être garantie tout le temps : l'exploration nécessite d'effectuer indéfiniment des actions arbitrairement sous-optimales. On va donc demander à ce que cette inégalité soit vraie tout le temps sauf pour un nombre borné d'instants. Là encore, il suffit que le hasard génère une séquence de transitions trompeuse pour que, quelle que soit la borne, elle soit dépassée. Une solution est alors de requérir que cette borne soit garantie avec une probabilité supérieure à $1-\delta$ (pour $\delta \in [0;1)$), ce qui amène à dire que l'algorithme $\mathcal{A}$ est *probablement approximativement correct*.

Plus précisément, un algorithme $\mathcal{A}$ est PAC-MDP si, avec une probabilité supérieure à $1-\delta$, il génère une politique $\epsilon$-proche de l'optimum à chaque instant sauf pour un nombre de pas de temps polynomial.

Un défaut de la propriété PAC-MDP est de reposer sur la fonction de valeur "idéale" dans le cas où l'on connaît le modèle. Une alternative est de se référer à la fonction de valeur bayésienne. On parle alors d'algorithme PAC-BAMDP (pour *Bayes-Adaptive* MDP) ou *quasi-bayésien* (*near-Bayesian*).

## B.5 Transitions Locales Optimistes

On peut classifier les approches optimistes pour le BRL selon deux critères principaux :

- comment se traduit cet optimisme ? dans BEB, c'est la fonction de récompense qui est optimiste ; dans BOSS, c'est le modèle qui est optimiste ;
- l'approche est-elle déterministe ou stochastique ? dans BEB, la fonction de récompense est générée de façon déterministe ; dans BOSS, le modèle optimiste est le résultat d'un échantillonnage.

La contribution présentée ici est un algorithme qui, comme BOSS, traduit l'optimisme dans la fonction de transition, et, comme BEB, est déterministe. L'optimisme dans la fonction de transition nous semble intéressante dans la mesure où c'est là que réside l'incertitude. Le déterminisme, lui, permet un "meilleur contrôle" de l'algorithme.

### B.5.1 Algorithme proposé

Comme BEB et BOSS, l'algorithme proposé résout un MDP à horizon $H$ (même si le problème considéré est à horizon infini) à chaque instant de décision. La fonction de récompense n'est pas altérée, mais la fonction de transition est obtenue en cherchant une variante optimiste de la fonction de transition moyenne. Plus précisément, dans le cas de distributions de Dirichlet, on cherche, pour chaque couple état-action $(s,a)$ (supposés indépendants), quel prochain état $\sigma$ serait le plus favorable, de manière à augmenter sa probabilité. Du fait des hypothèses d'indépendance, cela conduit à résoudre un nouveau MDP dans lequel

- l'espace d'action est augmenté, une action $\alpha$ contenant une action $a$ et un prochain état $\sigma$, et

– le calcul de la probabilité d'une transition $s \xrightarrow{\alpha=(a,\sigma)} s'$ est donné par

$$T(s,(a,\sigma),s') = \begin{cases} \frac{\phi^{s,a}_{s'}+\eta}{\|\phi^{s,a}\|_1} + \eta & \text{si } \sigma = s', \\ \frac{\phi^{s,a}_{s'}}{\|\phi^{s,a}\|_1} + \eta & \text{si } \sigma \neq s', \end{cases}$$

où $\eta > 0$ est une constante.

En résumé, cet algorithme prend des décisions en étant optimiste quant au modèle de transition local (pour chaque couple état-action), d'où l'acronyme BOLT pour "Bayesian Optimistic Local Transition". On notera que le paramètre $\eta$ de BOLT est le pendant du paramètre $\beta$ de BEB, même si une identification n'est pas possible, comme on le verra plus loin.

Ce premier algorithme suppose un problème réel à horizon infini et, du fait que $\eta$ n'est pas infini, il ne garantit pas d'avoir un modèle aussi favorable que le vrai modèle. Il n'est donc que probablement optimiste. Par contre, dans le cas d'un horizon réel fini $H$, on peut employer un bonus égal au temps restant ($\eta = H$), ce qui permet d'être strictement optimiste.

### B.5.2 Propriétés

Sans entrer dans les détails techniques, les propriétés importantes démontrées de manière théorique dans ce travail sont que :

– BEB est PAC-BAMDP pour les problèmes à horizon infini (la même propriété ayant précédemment été établie pour les problèmes à horizon fini) ;
– BOLT est PAC-BAMDP pour les problèmes à horizon infini ; et
– BOLT pour horizon réel fini est strictement optimiste.

**Theorem B.5.1** (BEB est PAC-BAMDP). *Notons $\mathbf{A}_t$ la politique suivie par* BEB *au temps $t$ avec $\beta = 2H^2$. Soient aussi $s_t$ et $\mathbf{b}_t$ les état et croyance correspondant à ce pas de temps. Alors, avec probabilité au moins $1-\delta$,* BEB *est $\epsilon$-proche de la politique bayésienne optimale*

$$\mathbb{V}^{\mathbf{A}_t}(s_t,\mathbf{b}_t) \geq \mathbb{V}^*(s_t,\mathbf{b}_t) - \epsilon$$

*pour tous les pas de temps sauf* $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|\beta^2}{\epsilon^2(1-\gamma)^2}\right) = \tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^4}{\epsilon^2(1-\gamma)^2}\right)$.

**Theorem B.5.2** (BOLT est PAC-BAMDP). *Notons $\mathbf{A}_t$ la politique suivie par* BOLT *au temps $t$ avec $\eta = H$. Soient aussi $s_t$ et $\mathbf{b}_t$ les état et croyance correspondant à ce pas de temps. Alors, avec probabilité au moins $1-\delta$,* BOLT *est $\epsilon$-proche de la politique bayésienne optimale*

$$\mathbb{V}^{\mathbf{A}_t}(s_t,\mathbf{b}_t) \geq \mathbb{V}^*(s_t,\mathbf{b}_t) - \epsilon$$

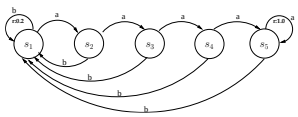
*pour tous les pas de temps sauf* $\tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|\eta^2}{\epsilon^2(1-\gamma)^2}\right) = \tilde{O}\left(\frac{|\mathcal{S}||\mathcal{A}|H^2}{\epsilon^2(1-\gamma)^2}\right)$.

### B.5.3 Etude expérimentale

BEB et BOLT (pris ici dans le cas d'un horizon réel infini) ont certaines ressemblances importantes, de sorte que les résultats théoriques obtenus impliquent que $\eta$ et $\beta$ doivent être pris dans un certain intervalle de valeurs (lié à l'horizon fictif $H$, lequel dépend lui-même de $\epsilon$ et $\gamma$).

Les études expérimentales que nous avons conduites ont toutefois permis de mettre en évidence une grande différence de comportement entre les deux algorithmes. En effet, sur les quelques problèmes abordés (voir figure B.6), si BEB atteint les meilleurs résultats, c'est sur une plage du paramètre $\beta$ très réduite, alors que BOLT a de bons résultats sur une plage assez grande. Sur un nouveau problème mal connu, il serait donc a priori préférable d'employer BOLT avec une valeur $\eta$ quelconque. Et sur un problème ressemblant à d'autres problèmes déjà rencontrés, on devrait choisir BEB avec une valeur $\beta$ ayant déjà fait ses preuves. Ceci dit, il est possible qu'une certaine valeur de $\beta$ s'avère robuste pour une grande variété de problèmes.

Les résultats expérimentaux ont aussi montré qu'EXPLOIT, un algorithme très simple sans exploration, peut s'avérer très efficace sur certains problèmes.



(a)

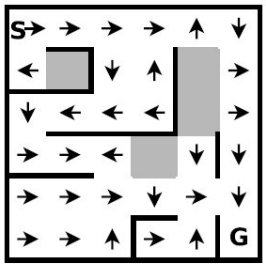
*paint*(X): reward = −1
**PRE:** none
**ADD:** *Painted*(X) **DEL:** none **0.6**
**ADD:** *Painted*(X), *Scratched*(X) **DEL:** none **0.3**
**ADD:** none **DEL:** none **0.1**

*polish*(X): reward = −1
**PRE:** none
**ADD:** none **DEL:** *Painted*(X) **0.2**
**ADD:** none **DEL:** *Scratched*(X) **0.2**
**ADD:** *Polished*(X) **DEL:** *Painted*(X), *Scratched*(X) **0.3**
**ADD:** *Polished*(X) **DEL:** *Painted*(X) **0.2**
**ADD:** none **DEL:** none **0.1**

*shortcut*(X): reward = −1
**PRE:** none
**ADD:** *Painted*(X), *Polished*(X) **DEL:** none **0.05**
**ADD:** none **DEL:** none **0.95**

*done*(X): reward = 10
**PRE:** *Painted*(X), *Polished*(X),¬*Scratched*(X)
**ADD:** *Finished*(X) **DEL:** none **1.0**

(b)




(c)

FIGURE B.6 – (a) Problème de la chaîne à 5 états (sans les probabilités). En général, l'action $a$ fait avancer dans la chaîne alors que l'action $b$ fait retourner à l'état de départ $s_1$. (b) Définition STRIPS du problème *Paint/Polish* (Younes et al., 2005). (c) Le problème *Marble Maze* dans lequel on doit apprendre à trouver la sortie d'un labyrinthe alors que les déplacements sont incertains. Les flèches correspondent à une politique optimale quand les probabilités de transition sont connues.

### B.5.4 Remarques et conclusion

BOLT est un algorithme optimiste simple, assez facile à étudier, et PAC-BAMDP. Des pistes de recherche intéressantes consisteraient à étendre son utilisation à des distributions plus complexes exploitant mieux la structure des problèmes rencontrés, et à mieux comprendre ce qui le rend plus robuste (à des changements de paramètres) que BEB.

En outre, un point délicat est l'évaluation expérimentale des algorithmes d'apprentissage par renforcement. L'usage est de ne considérer, pour chaque problème,

- qu'un jeu de paramètres du modèle, alors qu'on voudrait justement que l'apprentissage soit robuste à la variabilité des paramètres, et
- qu'un jeu de paramètres (plus ou moins optimisé) de l'algorithme, alors que l'utilisateur n'est pas supposé pouvoir les régler à l'avance.

## B.6 Apprendre des modèles de transition de manière optimale

Les $\rho$POMDP permettent de modéliser des problèmes dans lesquels l'objectif n'est pas (uniquement) de contrôler l'état d'un système, mais (aussi) d'acquérir de l'information. De la même manière, une tâche d'apprentissage par renforcement avec modèle pourrait ne pas avoir comme objectif de contrôler au mieux l'état du système considéré, mais d'apprendre au mieux la dynamique du système. Cette problématique conduit en un sens à l'intersection des deux sujets principaux étudiés dans cette thèse : les $\rho$POMDP et le BRL.

L'apprentissage (par renforcement) d'un modèle de transition suppose d'abord de spécifier un critère d'optimisation, donc avant tout une mesure de qualité du modèle (mais aussi un horizon temporel). Il faut alors traduire ce critère en une fonction de récompense, et enfin choisir un algorithme de résolution. Nous allons brièvement évoquer ces différents points.

### B.6.1 Critère d'optimisation

On suppose ici qu'il n'y a pas de coûts associés aux actions, et que la qualité du modèle est une fonction $f(\theta)$.

Dans le cas d'un problème à horizon fini ou indéfini, c'est logiquement la qualité du modèle une fois un état terminal atteint qui nous intéresse : $f(\theta^{term})$.

Dans le cas d'un problème à horizon infini, la qualité moyenne est maximale du moment que l'algo-

rithme $\mathcal{A}$ explore infiniment tous les couples état-action :

$$\lim_{T\to\infty} E_{\mathcal{A}}\left[\frac{1}{T}\sum_{t=1}^{T} f(\theta^t)\right] = f_{\max}.$$

Il est donc préférable de choisir un critère qui tient compte de la vitesse d'apprentissage, par exemple en optimisant

$$E\left[\sum_{t=1}^{\infty} \gamma^t f(\theta^t)\right].$$

Pour ce qui est de la fonction $f$ qui servira de critère, différents choix sont possibles, tels que la différence de variance entre modèle initial et modèle final, la différence d'entropie entre modèle initial et modèle final, ou la distance de Bhattacharyya entre modèle initial et modèle final. On notera que, dans ces trois cas, $f(\theta^t)$ mesure une distance $D(\theta^0, \theta^t)$ entre deux modèles.

### B.6.2 Fonction de récompense

Dans les problèmes à horizon infini, il suffit a priori de donner à l'instant $t$ une récompense égale à $\rho(s, a, s', \theta^t) = D(\theta^0, \theta^{t+1})$ pour satisfaire le critère choisi.

**Calcul raisonnable en pratique ?**

Dans les problèmes à horizon fini ou indéfini, ne récompenser que l'état terminal pose des problèmes pratiques : nombre d'algorithmes sont plus efficaces si on peut fournir des récompenses au plus tôt. Une solution est alors de façonner la fonction de récompense comme proposé par Ng et al. (1999), c'est-à-dire en écrivant $\rho(s, a, s', \theta^t) = D(\theta^{t+1}, \theta^0) - D(\theta^t, \theta^0)$. En pratique, certaines des mesures considérées vérifient l'égalité triangulaire, ce qui permet d'écrire $\rho(s, a, s', \theta^t) = D(\theta^{t+1}, \theta^t)$, quantité souvent beaucoup plus facile à calculer que la formule d'origine. Ici, on va *abusivement* considérer que cette égalité triangulaire est toujours vérifiée.

### B.6.3 Algorithme

Le cadre de l'apprentissage actif de modèle est plus compliqué que l'apprentissage par renforcement bayésien avec modèle, du fait de l'usage d'une fonction de récompense complexe.

Par contre, il n'est pas nécessaire de provoquer l'exploration. La fonction de récompense ne vise de toute façon qu'à faire explorer l'environnement. On peut donc utiliser le très simple algorithme EXPLOIT, lequel, à chaque instant de décision, repose son choix d'action sur une politique solution du modèle moyen courant.

Etant dans un cadre dérivé de l'apprentissage par renforcement bayésien, il est naturel d'aussi envisager l'utilisation d'algorithmes dédiés à ce cadre, et donc en particulier ici BOLT.

### B.6.4 Expérimentations

Des expérimentations ont été conduites sur trois problèmes de référence –*Bandit*, *Chain* et *Loop*– illustrés par la figure B.7, dans des versions plus ou moins déterministes pour *Chain*. EXPLOIT et BOLT ont ainsi été comparés à des comportements aléatoire et gourmand, et ce, en employant différentes mesures d'incertitude.

On observe le plus souvent que le comportement gourmand est le pire, parfois de loin, étant généralement dominé par le comportement aléatoire, lui-même dominé par EXPLOIT et BOLT. Cette différence est bien moins visible sur le problème *Bandit*, ce qui s'explique par la structure fortement connectée de ce MDP.

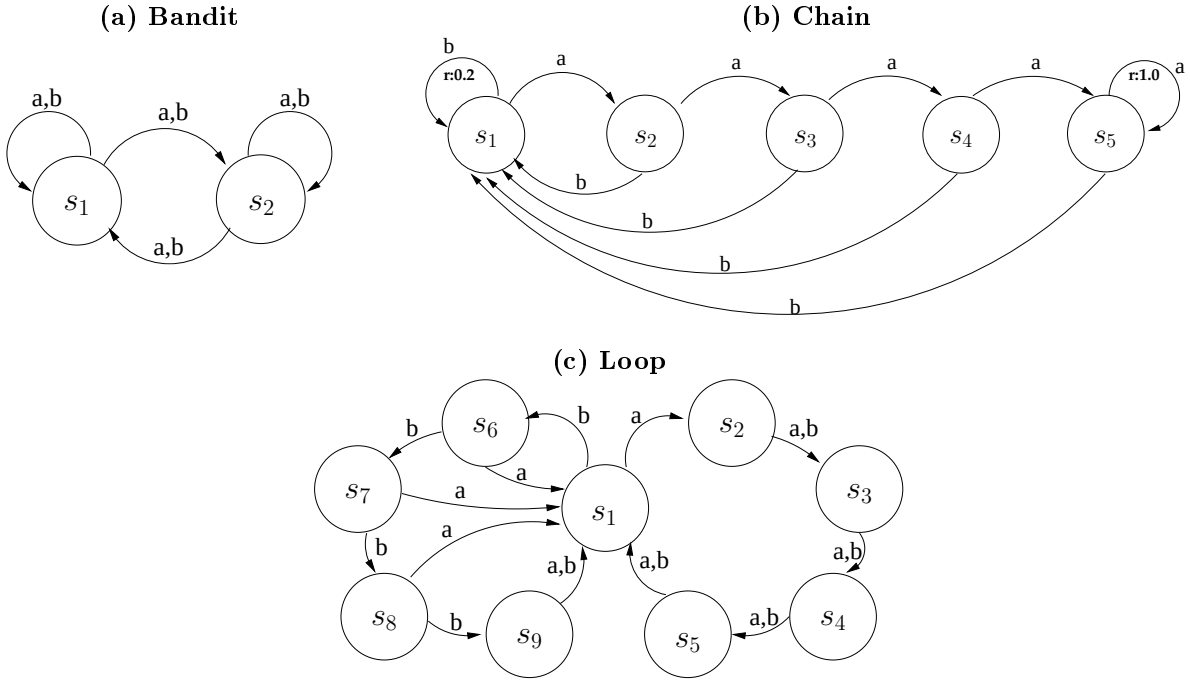



FIGURE B.7 – Modèles des MDP utilisés pour les expérimentations (sans les probabilités de transition). Dans (a) le problème du bandit à 2 bras (voir section 1.3), dans (b) le problème de la chaîne à 5 états (voir section 6.4.1) et dans (c) le problème de la boucle de (Strens, 2000).

### B.6.5 Travaux futurs

Une première piste à explorer est la relation entre le très simple, mais efficace, critère *state-action count* et les trois critères introduits dans cette thèse. Ceci permettrait éventuellement de proposer des fonctions de récompenses pertinentes et peu coûteuses computationnellement.

Une autre piste naturelle est d'introduire des connaissances *a priori* sous la forme de distributions *a priori* telles que des DBN dans (Jonsson and Barto, 2007), ou en liant des paramètres comme dans (Poupart et al., 2006). Cela accélérerait considérablement le processus d'apprentissage en faisant un usage bien plus efficace des données, et ce, sans modifier notablement les techniques de résolution.

## B.7 Conclusion

La recherche rapportée dans cette dissertation a examiné plusieurs aspects des problèmes de décisions séquentiels pour la collecte d'information, dont les POMDP avec récompense dépendant de l'information au chapitre 4, les problèmes d'apprentissage par renforcement avec récompense dépendant de l'information au chapitre 7, et l'exploration optimiste pour l'A/R bayésien au chapitre 6. Les sujets traités ici ont été confinés à certaines contraintes et certains types de problèmes de manière à limiter le champ de cette dissertation. Pour chaque sujet, une formulation théorique appropriée a été présentée sous ces contraintes et des études empiriques ont été conduites, menant à des résultats nouveaux et instructifs qui sont résumés dans les conclusions de chaque chapitre. Des travaux futurs ont aussi été présentés pour chacun de ces chapitres, décrivant des extensions naturelles, des nœuds théoriques et des directions de recherche futures qu'il faudra suivre pour appliquer ces résultats à des problèmes réels.

Ce chapitre présente une brève de ces résultats dans leur ensemble, montrant comment cette thèse contribue à répondre aux questions posées en section 1.2.

## B.8 Comment représenter, mesurer et mettre à jour l'information détenue par un agent ?

A première vue la question de la représentation semble généralement résolue. Le chapitre 2 montre comment représenter l'information en utilisant la théorie des probabilités, et comment la mettre à jour (et décider) en utilisant l'inférence statistique. De plus, la vue bayésienne permet de représenter et de mettre à jour l'information épistémique dans un cadre bien défini. La question de la mesure de l'information est traitée aux chapitres 4 et 7 en utilisant la théorie de l'information avec la représentation choisie.

Pourtant cette question est bien plus complexe en réalité. Malheureusement, dans le cadre de la théorie des probabilités il n'y a pas de représentation magique pour l'information, les modèles probabilistes étant toujours sujets à certaines hypothèses et simplifications. En effet, la zoologie complète des distributions de probabilité n'est rien de plus qu'une variété d'hypothèses et de simplifications pour représenter l'information. Un exemple est la discussion des distributions a priori pour l'A/R bayésien en section 5.2.1, où le choix de priors FDM simplifie l'ensemble du processus de décision et d'apprentissage, mais n'est pas approprié pour des espaces d'état continus ou pour des problèmes fortement structurés. Le même type de choix doit être fait quand on définit l'état de croyance et la croyance initiale dans les POMDP. Cette dissertation a essayé d'être aussi générique que possible dans des limites raisonnables. Toutefois, étendre les résultats présentés à des modèles plus généraux et des hypothèses plus faibles est clairement un travail futur possible.

En outre, mesurer l'information pose des problèmes supplémentaires. La dérivation axiomatique élégante de (Shannon, 1948) montre que l'entropie est l'unique mesure qui satisfait les propriétés désirées pour une mesure d'information. Pourtant, d'autres auteurs ont proposé des mesures (et semi-mesures) similaires dérivées d'axiomes similaires ou de généralisations, telles que l'entropie de Rényi entropy, la divergence de Kullback-Leibler, l'information de Chernoff ou l'information de Fisher. Sélectionner la mesure appropriée dépendra de la représentation utilisée, de l'objectif du problème, et de la capacité à calculer cette mesure. En plus, les mesures reposant sur le logarithme comme l'entropie sont des fonctions avec lesquelles il est compliqué de travailler dans le sens que les résultats analytiques sont rarement des expressions en forme close. Une des principales contributions de cette dissertation est de montrer empiriquement à travers les chapitres 4 et 7 que des mesures d'information plus simples (mais liées théoriquement) peuvent être utilisées dans le cadre de problèmes de décision pour la collecte d'information sans perte de performance importante, comme la récompense linéaire pour $\rho$POMDP ou la récompense *state-action count* pour l'apprentissage actif de modèle. Cette propriété d'interchangeabilité ne vaut sûrement pas pour des applications qui ont besoin d'une solution très fine, mais fournira vraisemblablement de meilleurs résultats que des stratégies naïves comme des politiques aléatoires ou des politiques gourmandes en information.

Un résultat particulièrement important dans ce domaine est la borne d'erreur d'une approximation PWLC d'une fonction $\alpha$-hölderienne (voir section 4.3.7). Ce résultat montre que ces mesures d'information reposant sur le logarithme peuvent être approchées par un ensemble de fonctions linéaires avec une précision arbitraire, et donc que des méthodes linéaires peuvent être utilisées avec une erreur bornée. Cette contribution dépasse le domaine des problèmes de décision pour la collecte d'information, et pourrait être utile dans d'autres applications de la théorie de l'information.

## B.9 La collecte (implicite) d'information est-elle une question résolue dans les problèmes de décision séquentielle ?

Le fait qu'une solution optimale puisse être trouvée théoriquement pour des problèmes de décision séquentielle sous incertitude suggère que la collecte implicite d'information est un problème résolu parce qu'elle est incorporée dans cette même solution optimale. En effet, une telle solution collecte seulement l'information requise pour le problème de manière à maximiser le retour espéré. Pourtant, le caractère insoluble du calcul d'une solution optimale conduit dans la plupart des cas à résoudre le problème de manière approchée. En conséquence, une procédure de collecte d'information est souvent requise par ces méthodes approchées. Par exemple, les solveurs à base de points collectent des points de croyance

informatifs, ce qui équivaut à collecter de l'information. Dans le cas de l'A/R bayésien, les algorithmes approchés reposent souvent sur des méthodes d'exploration qui sont des formes implicites de collecte d'information. Dans ce contexte, le chapitre 6 offre une nouvelle méthode de collecte d'information basée sur l'idée d'un "coup de pouce" bayésien.

En outre, la relation entre cette collecte d'information incorporée et la théorie de l'information reste une question ouverte. Plus précisément, savoir comment, du point de vue de la théorie de l'information, cette solution optimale peut optimiser le besoin en information tout en maximisant les récompenses pourrait aider à développer des algorithmes approchés plus intelligents. Cette dissertation présente quelques éléments à propos de cette relation en section 4.3.6, section qui discute du lien entre collecte explicite d'information et classification active.

## B.10 La collecte (explicite) d'information est-elle un problème de décision séquentielle ?

Malgré les avancées sur la collecte d'information implicite, cette dissertation se concentre principalement sur la collecte explicite d'information (chapitres 4 et 7). Cette dissertation confirme, en examinant des travaux antérieurs, qu'aussi bien les problèmes de collecte d'information purs que ceux sensibles aux coûts sont des problèmes pertinents et intéressants. Les modèles proposés pour la collecte d'information sont corrects et bien définis dans le sens qu'ils sont dérivés d'un critère de performance propre, et donc résoluble de manière optimale en théorie. Aussi, les techniques de résolution proposées ici ont été justifiée théoriquement et empiriquement de sorte que, même si des techniques futures, éventuellement ad-hoc, peuvent les dépasser, la contribution de cette dissertation reste importante puisqu'elle offre une alternative propre aux techniques naïves –mais largement employées– telles que les stratégies aléatoires ou myopes.

## B.11 Directions de recherche

Cette section résume les principales directions de recherche proposées dans cette dissertation, montrant qu'elle ouvre des extensions techniques spécifiques qui peuvent être traitées à court terme, et des questions plus fondamentales pour la recherche à long terme.

### Directions à court terme

Dans le **chapitre 4** il est proposé d'utiliser une méthode de collecte fondée sur l'information pour des algorithmes à base de points plutôt que d'utiliser une collecte aléatoire. Etendre les $\rho$POMDP à des représentations factorisées telles que les MOMDP peut aussi aider à accélérer le processus de résolution. Finalement, il est proposé de mener d'avantages d'expérimentations sensibles aux coûts pour analyser l'impact du paramètre de mélange.

Dans le **chapitre 6** il est proposé d'utiliser des bonus $\eta$ dynamiques pour obtenir un optimisme plus fin. Aussi, étendre BOLT à d'autres familles de priors telles que les distributions hyperdirichlet pourrait aider à exploiter l'information structurée du problème. Finalement, il est proposés d'étendre les transitions locales optimistes pour des problèmes d'A/R partiellement observables.

Dans le **chapitre 7** il est proposé d'utiliser des algorithmes plus élaborés, tels que BOLT, pour le MML. Il est aussi suggéré d'étendre les idées du chapitre à l'apprentissage de modèles POMDP et des distributions plus structurées.

### Directions à long terme

Le **formalisme $\rho$POMDP** présente une manière de modéliser et de résoudre les problèmes de collecte d'information séquentielle, mais la relation entre ces types de problèmes et les problèmes de planification usuels sous incertitude reste une question ouverte. Comprendre les fondements de la collecte d'information pourrait aider à construire de nouveaux solveurs de POMDP en collectant de l'information efficacement.

Les **transitions locales optimistes** ont été utilisées dans cette dissertation pour collecter de l'information dans de petits domaines avec seulement peu de degrés de liberté. Toutefois des applications réelles vont nécessiter d'étendre ces résultats à des espaces d'état ou d'action larges ou continus, dans lesquels l'information deviendra plus complexe et cruciale. La robustesse et la simplicité de BOLT sont des qualités désirables pour résoudre des problèmes réels complexes. Un défi intéressant et qui nécessite des recherches futures est donc de préserver ces propriétés en passant à l'échelle.

Le problème de l'**apprentissage de modèle de MDP** a été défini en utilisant des mesures d'information justifiées et formelles. Toutefois, les expérimentations avec *state-action count* montrent que des récompenses simples peuvent être suffisantes pour traiter ce problème. Comme pour les $\rho$POMDP, la relation entre la collecte d'information en MML et l'A/R bayésien reste une question ouverte, de sorte que comprendre cette relation pourrait mener non seulement à l'amélioration des techniques présentées pour les problèmes de MML, mais aussi à la conception de meilleurs algorithmes de BRL à travers une collecte d'information intelligente.

# Résumé

Le formalisme des MDP, comme ses variantes, sert typiquement à contrôler l'état d'un système par l'intermédiaire d'un agent et de sa politique. Lorsque l'agent fait face à des informations incomplètes, sa politique peut effectuer des actions pour acquérir de l'information typiquement (1) dans le cas d'une observabilité partielle, ou (2) dans le cas de l'apprentissage par renforcement. Toutefois cette information ne constitue qu'un moyen pour contrôler au mieux l'état du système, de sorte que la collecte d'informations n'est qu'une conséquence de la maximisation de la performance escomptée.

Cette thèse s'intéresse au contraire à des problèmes de prise de décision séquentielle dans lesquels l'acquisition d'information est une fin en soi. Plus précisément, elle cherche d'abord à savoir comment modifier le formalisme des POMDP pour exprimer des problèmes de collecte d'information et à proposer des algorithmes pour résoudre ces problèmes. Cette approche est alors étendue à des tâches d'apprentissage par renforcement consistant à apprendre activement le modèle d'un système. De plus, cette thèse propose un nouvel algorithme d'apprentissage par renforcement bayésien, lequel utilise des transitions locales optimistes pour recueillir des informations de manière efficace tout en optimisant la performance escomptée.

Grâce à une analyse de l'existant, des résultats théoriques et des études empiriques, cette thèse démontre que ces problèmes peuvent être résolus de façon optimale en théorie, que les méthodes proposées sont presque optimales, et que ces méthodes donnent des résultats comparables ou meilleurs que des approches de référence. Au-delà de ces résultats concrets, cette thèse ouvre la voie (1) à une meilleure compréhension de la relation entre la collecte d'informations et les politiques optimales dans les processus de prise de décision séquentielle, et (2) à une extension des très nombreux travaux traitant du contrôle de l'état d'un système à des problèmes de collecte d'informations.



**Mots-clés:** collecte d'informations, transitions optimistes, POMDP, apprentissage par renforcement bayésien, apprentissage du modèle d'un MDP, problèmes de prise de décision séquentielle, modèles bayésiens



# Abstract

The MDP formalism and its variants are usually used to control the state of a system through an agent and its policy. When the agent confronts incomplete information, its policy can perform actions to gather information, such as in (1) the partially observable state case, or in (2) the reinforcement learning scenario. However, the acquired information is only a means to better control the system state, so the information gathering is only a consequence of maximizing the expected return.

On the contrary, the purpose of this dissertation is to study sequential decision problems where acquiring information is an end in itself. More precisely, it first covers the question of how to modify the POMDP formalism to model information-gathering problems and which algorithms to use for solving them. This idea is then extended to reinforcement learning problems where the objective is to actively learn the model of the system. Also, this dissertation proposes a novel Bayesian reinforcement learning algorithm that uses optimistic local transitions to efficiently gather information while optimizing the expected return.

Through bibliographic discussions, theoretical results and empirical studies, it is shown that these information-gathering problems are optimally solvable in theory, that the proposed methods are near-optimal solutions, and that these methods offer comparable or better results than reference approaches. Beyond these specific results, this dissertation paves the way (1) for understanding the relationship between information-gathering and optimal policies in sequential decision processes, and (2) for extending the large body of work about system state control to information-gathering problems.



**Keywords:** Information-Gathering, Optimistic Transitions, POMDP, Bayesian Reinforcement Learning, MDP Model Learning, Sequential Decision Problems, Bayesian Models