On the Generation of Representations for Reinforcement Learning

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
SUN Yi

under the supervision of
Jürgen Schmidhuber

June 2012

Dissertation Committee

Prof. Marcus Hutter	Australia's National University, Australia
Prof. Marco Wiering	University of Groningen, the Netherlands
Prof. Rich Sutton	University of Alberta, Canada
Prof. Rolf Krause	Università della Svizzera Italiana, Switzerland
Prof. Kai Hormann	Università della Svizzera Italiana, Switzerland

Dissertation accepted on 15 June 2012

Research Advisor	PhD Program Director
Jürgen Schmidhuber	**The PhD program Director *pro tempore***

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

SUN Yi
Lugano, 15 June 2012

To my beloved

Abstract

Creating autonomous agents that learn to act from sequential interactions has long been perceived as one of the ultimate goals of Artificial Intelligence (AI). Reinforcement Learning (RL), a subfield of Machine Learning (ML), addresses important aspects of this objective. This dissertation investigates a particular problem encountered in RL called *representation generation*. Two related subproblems are considered, namely *basis generation* and *model learning*, concerning which we present three pieces of original research.

The first contribution considers a particular basis generation method called *online kernel sparsification* (OKS). OKS was originally proposed for recursive least squares regression, and shortly thereafter extended to RL. Despite the popularity of the method, important theoretical questions are still to be answered. In particular, it was unclear how the size of the OKS dictionary, or equivalently the number of basis functions constructed, grows in relation to the amount of data available. Characterizing this growth rate is crucial to understanding OKS, both in terms of its computational complexity and, perhaps more importantly, the generalization capability of the resulting linear regressor or value function estimator. We investigate this problem using a novel formula expressing the expected determinant of the kernel Gram matrix in terms of the eigenvalues of the covariance operator. Based on this formula, we are able to connect the cardinality of the dictionary with the eigen-decay of the covariance operator. In particular, we prove that under certain technical conditions, the size of the dictionary will always grow sub-linearly in the number of data points, and, as a consequence, the kernel linear regressor or value function estimator constructed from the resulting dictionary is consistent.

The second contribution turns to a different class of basis generation methods, which make use of reward information. We introduce a new method called V-BEBF. V-BEBF relies on a principle that is different from that of previous approaches based on Bellman error basis function (BEBF), in which approximations to *the value function of the Bellman error*, rather than to the Bellman error itself as in BEBF, are added as new basis functions. This approach is justified

by a simple yet previously unspotted insight, i.e., V-BEBF, if computed exactly, is in fact the error in value estimation, and therefore its addition to the existing set of basis functions immediately allows the value function to be represented accurately. We demonstrate that V-BEBF is a promising alternative to BEBF, especially when the discount factor approaches 1, in which case it is proven that BEBF, even if computed exactly, can be very inefficient. Limited experiments, where both V-BEBFs and BEBFs are approximated using linear combinations of the input features, are also conducted, and the result is in line with the theoretical finding.

The last contribution focuses on model learning, especially learning the transition model of the environment. The problem is investigated under a Bayesian framework, where the learning is done by probabilistic inference, and the learning progress is measured using Shannon information gain. In this setting, we show that the problem can be formulated as an RL problem, where the reward is given by the immediate information gain resulting from performing the next action. This shows that the model-learning problem can in principle be solved using algorithms developed for RL. In particular, we show theoretically that if the environment is an MDP, then near optimal model learning can be achieved following this approach.

Acknowledgements

This work was carried out during the years 2007-2012, at the Dalle Molle Institute for Artificial Intelligence (IDSIA).

First I express my deepest gratitude to my supervisor, Jürgen Schmidhuber, who is always providing me with inspiring guidance while giving me immense freedom. I will always be grateful to Faustino Gomez. Without his patience, meticulousness and professionalism, this dissertation, as well as much of the work it is based on, would hardly have been possible. I have also been fortunate that Marco Zaffalon, with his great kindness and tolerance, gave me the chance to pursue my PhD in IDSIA.

Secondly, I would like to thank all my colleagues in IDSIA: Daan Wierstra, Tom Schaul, Julian Togelius and Cassio de Campos, with whom I spent hours and hours in interesting and often absurd discussions; Tobias Glasmachers, who, as a true mathematician, patiently answered my questions and read my lengthy and often not so well written proofs; Bas Steunebrink, who gave me a lot of valuable advice on writing and brightened the days by bringing the delicious Dutch stroopwafels and the not so delicious liquorice; and Vincent Graziono and Jan Koutnik, with whom I shared exciting cycling experiences. Special thanks to Cinzia Daldini, our secretary, who helped me greatly with all the procedures. Needless to say, the list is still by far too short and I owe my gratitude to all the people who helped me in this wonderful institute.

Thirdly, I am grateful for my dissertation committee, Marcus Hutter, Marco Wiering, Rich Sutton, Kai Hormann and Rolf Krause, for the constructive comments. Special thanks to Marcus Hutter, who has shown such a level of dedication and professionalism by examining all the proofs, and Rich Sutton, who has pushed me to perfect all the important details that I was too lazy to do otherwise.

Last but definitely not the least, I owe my gratitude to my parents, whose constant support always make me feel warm during my lengthy stay in Switzerland. I would also like to thank all the friends that I met, and the girls who have passed through my life, even briefly. Together we had great fun and the happiness will always be missed.

Contents

Chapter 1

Introduction

Creating autonomous agents that learn to act from sequential interactions has long been perceived as one of the ultimate goals of Artificial Intelligence (AI) [Legg, 2008; Hutter, 2005]. Reinforcement Learning (RL) addresses important aspects of this objective by answering the following question:

> *How should an agent interact with a dynamic environment, so as to maximize the prospect of future reward collected from the environment?*

Reinforcement learning is closely related to how animals and humans act and learn. Without a teacher, solely from occasional pain and pleasure signals, RL agents must discover how to interact with an unknown environment to maximize the pleasure and avoid the pain.

This dissertation addresses a particular problem encountered in RL called *representation generation*. Two related subproblems are considered, namely *basis generation* and *model learning*. We present original research concerning the first topic in Chapter 3 and Chapter 4 and the second one in Chapter 5.

This chapter provides a bird's eye view over the rest of the dissertation in an informal way. After introducing a few necessary concepts in Section 1.1, we explain the general problem of representation generation in Section 1.2, and then outline the main contributions in Section 1.3.

1.1 Reinforcement Learning

We consider interactions between an agent and its environment taking place in discrete time cycles $t = 1, 2, \ldots$, as depicted in Figure 1.1. In cycle t, the

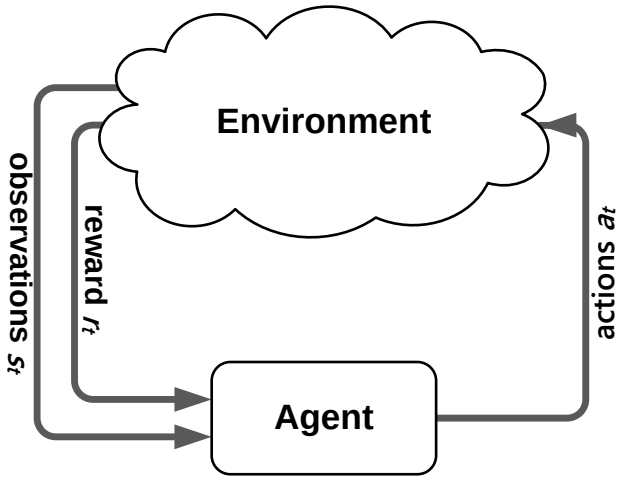

Figure 1.1. **Agent-environment interaction in RL**. At each cycle t, the agent observes s_t, takes action a_t, and then receives a real-value reward r_t.

agent receives from the environment an *observation* s_t, then sends back an *action* a_t. In doing this, the agent collects a real-valued *reward* r_t. In RL, the goal of the agent is to maximize the sum of the rewards received. Comparing to other problems considered in Machine Learning, e.g., supervised learning, RL problems are considerably more challenging, particularly because the reward are usually delayed and sparse.

Traditionally, RL problems are formulated in abstract finite state and action spaces forming so-called *Markov decision processes* (MDPs), where it is assumed that at each moment the observation s_t, to which we also refer as the *state* in this case, contains all relevant information for predicting future observations and rewards in response to any of the agent's actions. Under this assumption, globally optimal *policies* are known to exist and can be represented by (deterministic) mappings from state to action spaces. It is also known that the optimal policies maximize the so-called *value-function*

$$v^{\pi}(s) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{\tau-1} r_t \,\middle|\, s_1 = s\right],$$

which corresponds to the exponentially discounted sum of reward (with discount factor $\gamma \in [0,1)$) when the agent starts from state s and follows policy π. Moreover, given the *model* of the MDP, i.e., the conditional distribution dictating how future observations and rewards are generated, there are efficient

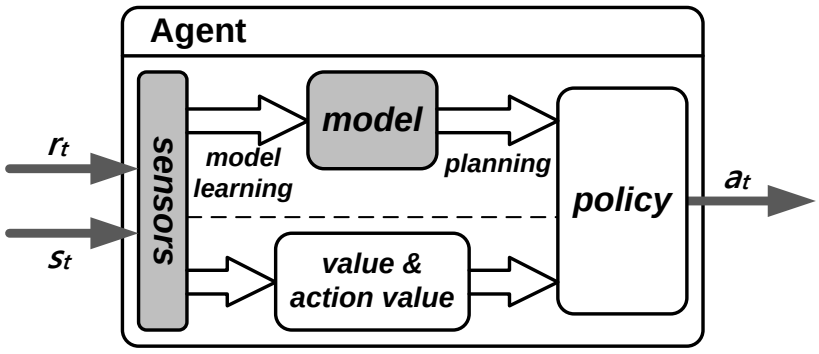

Figure 1.2. **General structure of an RL agent**. After the inputs are processed by the sensor block for dimensionality reduction, the agent either follows the model-based approach (the upper link in the diagram), where it learns a model of the environment and then plans using the model, or adopts the model-free approach (the lower link) and learns the policy through value or action-value functions. The two gray blocks, *sensor* and *model*, can all be viewed as the internal representations created by the agents, and hence are the focus of this dissertation.

algorithms that find the optimal policies in time polynomial in the number of states and actions.

In RL, we are mainly concerned with the scenario where the agent has no, or only partial, knowledge about the model. To achieve the goal, the agent must interact with the environment to gather the necessary information. There are two general approaches to this problem, summarized in Figure 1.2. *Model-based* methods (upper branch in the diagram) first learn an (approximated) model of the environment (the *model-learning* step), then find the optimal policy on the learned model (the *planning* step). *Model-free* algorithms (lower branch), bypass the model-learning step and learn the value function directly from the interactions, primarily using a family of algorithms called *Temporal Difference* (TD; [Sutton, 1988]) learning. The policy can then be obtained from the value function.

In most real-world RL problems, the cardinality of the state space is too large to permit explicit storage of the value function. In addition, it is often too expensive to gather the amount of data required to reliably estimate the value function (a phenomenon known as *overfitting* in Machine Learning). In such cases, it is common to reduce the dimensionality of observation via the sensor

block in Figure 1.2, then approximate the value function using some tractable form. This approach is generally referred to as *function approximation* (FA) in RL. A popular FA technique is *linear function approximation* (LFA), where the value function is represented by

$$v(s) \simeq \sum_{n=1}^{N} \theta_n \phi_n(s). \tag{1.1}$$

Here $\phi_1, \ldots, \phi_N$ are called the *basis functions*, and $\theta_1, \ldots, \theta_N$ are the corresponding coefficients. Usually the basis functions can be computed on the fly, and only the N weights have to be learned, where N is usually much smaller than the number of states. Hence, both the storage and overfitting issues can be alleviated by using LFA.

1.2 Representation Generation

The past several decades have seen significant advances in RL. In particular, important progress has been made in a number of directions concerning MDP. In the model-based case, efficient planning algorithms have been developed which are capable of solving MDPs with millions of states. In model-free RL, the family of TD methods, when combined with function approximators, has demonstrated convincing performance in a wide range of real-world problems [e.g., see Tesauro, 1995; Crites and Barto, 1996; Silver et al., 2012]. In particular, efficient algorithms have been developed for estimating the weights in LFA.

As the state-of-the-art of RL advances rapidly, numerous new challenges emerge. In particular, we focus on the following two problems (see the two gray blocks in Figure 1.2):

- In model-based RL, the model learning problem is still by and large open. In particular, since the efficiency of the model learning is affected by the policy that the agent follows to gather information, it is important to select the actions intelligently. Traditionally, in model learning, the data is collected by following random policies, which proves to be inefficient due to the random walk behavior generated, especially when the state space is large. To address this issue, model learning methods based on active learning [Fedorov, 1972; Thrun and Möller, 1991; Jonsson and Barto, 2007], artificial curiosity [Schmidhuber, 1990, 1991; Storck et al., 1995; Wyatt et al., 2011; Ross et al., 2011; Schmidhuber, 2010] and intrinsic reward [Singh et al., 2004; Özgür Şimşek and Barto, 2006] have been proposed. However, the theoretical understanding of these approaches is still lacking.

- In model-free RL using LFA, the majority of the effort has been devoted to estimating the weights over a pre-defined set of basis functions (Equation 1.1). Usually, the basis functions are designed by domain experts after close examination of the problem to be solved. However, as the tasks grow in complexity, hand-crafting basis functions becomes increasingly difficult, and it becomes necessary to consider automatic basis function generation. The literature on automatic basis generation is relatively new, including the earlier work on basis adaptation [Menache et al., 2005], Bellman Error Basis Functions (BEBFs) [Keller et al., 2006; Parr et al., 2007; Mahadevan and Liu, 2010], Proto-value functions [Mahadevan et al., 2007], and algorithmic methods for feature extraction [Hutter, 2009b]. In addition, there is growing trend to study the basis generation problem using Machine Learning methods developed for regression and dimensionality reduction [Engel, 2005; Kolter and Ng, 2009a; Boots and Gordon, 2010]. At the same time, a number of open problems arise concerning the theoretical properties of these algorithms, which call for further investigation.

The two topics above can be summarized into a single, general problem: how to form effective internal representations for RL in environments that exhibit high dimensionality. The representation can be a model of the environment in the model-learning problem, or the set of basis functions in the model-free setting. Up to now, this general problem is far from solved, and will remain a central focus in RL research.

1.3 Contributions

In this dissertation, we present three pieces of original research centered around representation generation, after providing the necessary background concerning RL in Chapter 2. More specifically, Chapter 3 and 4 are devoted to the basis generation problem encountered in the model-free case, and Chapter 5 focuses on the problem of model-learning (see Figure 1.3). The following is an overview of these three, primarily theoretical, contributions:

- In Chapter 3, we study a particular basis generation method called *online kernel sparsification* (OKS). OKS was originally proposed by Engel et al. [2004] for recursive least squares regression, and shortly thereafter extended to RL [Engel, 2005]. Despite the popularity of the method, important theoretical questions are still to be answered. In particular, it was unclear how the size of the OKS dictionary, or equivalently the number

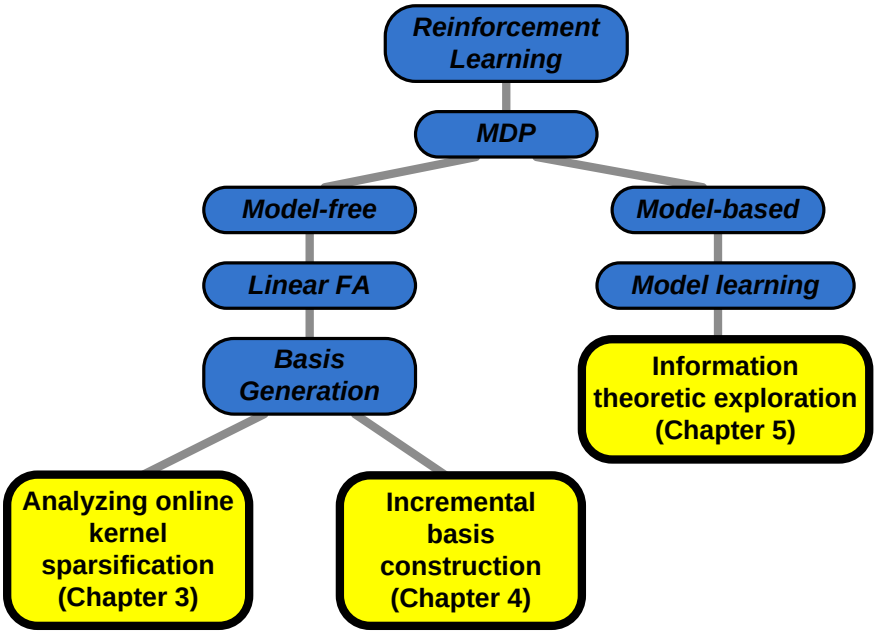

Figure 1.3. **Contributions in Tree Diagram**

of basis functions constructed, grows in relation to the amount of data available. Characterizing this growth rate is crucial for the understanding of OKS, both on its computational complexity and, perhaps more importantly, the generalization capability of the resulting linear regressor or value function estimator.

We investigate this problem using a novel formula expressing the expected determinant of the kernel Gram matrix in terms of the eigenvalues of the covariance operator. Based on this formula, we are able to connect the cardinality of the dictionary with the eigen-decay of the covariance operator. In particular, we prove that under certain technical conditions, the size of the dictionary will always grow sub-linearly in the number of data points, and, as a consequence, the kernel linear regressor or value function estimator constructed from the resulting dictionary is consistent.

- Chapter 4 turns to a different class of basis generation methods, which make use of the reward information, where we introduce a new method called V-BEBF. V-BEBF relies on a principle that is different from that of Bellman error basis function (BEBF), in which the approximations to *the value function of the Bellman error*, rather than to the Bellman error itself

as in BEBF [see e.g. Parr et al., 2007], are added as new basis functions. This approach is justified by a simple yet previously unspotted insight, i.e., V-BEBF, *if computed exactly*, is in fact the error in value estimation, and therefore its addition to the existing set of basis functions immediately allows the value function to be represented.

We demonstrate that V-BEBF is a promising alternative to BEBF, especially when the discount factor γ approaches 1, in which case it is proven that BEBF, even if computed exactly, can be very inefficient. Limited experiments, where both V-BEBFs and BEBFs are approximated using linear combinations of the input features, are also conducted, and the result is in line with the theoretical finding.

- Chapter 5 focuses on model learning, especially learning the transition model of the environment. The problem is investigated under a Bayesian framework, where the learning is done by probabilistic inference, and the learning progress is measured using Shannon information gain. In this setting, we show that the problem can be formulated as an RL problem, where the reward is given by the immediate information gain resulting from performing the next action. This shows that the model-learning problem can in principle be solved using algorithms developed for RL. In particular, we show theoretically that if the environment is an MDP, then near optimal model learning can be achieved following this approach.

Chapter 3 and 4 are based on two publications [Sun et al., 2012, 2011a] in the International Conference on Machine Learning (ICML), while Chapter 5 is based on a publication [Sun et al., 2011b] in the Third Conference on Artificial General Intelligence (AGI), but with significant extension; in particular, the proof of optimality in the MDP case is new.

Chapter 2

Reinforcement Learning at a Glance

This chapter serves as a mini-introduction to RL, in the hope of providing the necessary background to the research presented in the following chapters. Indeed, due to the broadness of the topic and the rapid advances in the field, this introduction is not intended to be comprehensive. In particular, we confine ourselves to (finite) Markov Decision Processes (MDPs), and contend with the linear case when it comes to Function Approximation (FA). For a more thorough treatment, the reader is recommended to refer to excellent text books such as Sutton and Barto [1998]; Bertsekas and Tsitsiklis [1996] and Bertsekas [2007].

We begin the chapter with an overview of the RL problem in Section 2.1, where MDPs are formally defined, and then survey a number of variations and extensions. Section 2.2 reviews (asymptotically) exact solutions to RL, including Dynamic Programming (DP) methods in the model-based case and Temporal Difference (TD) methods in the model-free case. Section 2.3 is devoted to function approximation, in particular Linear FA (LFA), where algorithms for both weight estimation and basis generation in LFA are introduced.

2.1 The Reinforcement Learning Problem

Most frequently, the problem of RL is investigated under the framework of MDPs. It is no exaggeration to say that MDPs have become the default setting in RL, while results obtained therein are generally regarded as the 'backbone' of the field.

2.1.1 Markov Decision Process

Formally, a *Markov Decision Process* is described by a 5-tuple $(\mathscr{S}, \mathscr{A}, P, R, \gamma)$, where:

- $\mathscr{S}$ is a finite set called the *state space*,
- $\mathscr{A}$ is a finite set called the *action space*,
- P is a *transition model*, such that $P(s'|s,a)$ is a conditional probability mass function over $s' \in \mathscr{S}$, i.e., $P(s'|s,a) \geq 0$ and $\sum_{s' \in \mathscr{S}} P(s'|s,a) = 1$ for any given $s \in \mathscr{S}$ and $a \in \mathscr{A}$,
- R is a *reward model*, i.e., a real-valued function $R(s,a)$ over $\mathscr{S} \times \mathscr{A}$, and
- $\gamma \in [0,1)$ is a *discount factor*.

We refer to P and R together as the *model*. Without loss of generality, assume that the elements in the state space and action space are so labeled that $\mathscr{S} = \{1,\ldots,S\}$ and $\mathscr{A} = \{1,\ldots,A\}$, with S and A the respective number of states and actions. As a result, any function f over $\mathscr{S}$ can be represented by an S-by-1 column vector with the s-th element $f(s)$, and thus we will refer to f both as a function and a vector depending on the context. Similarly, a function g over $\mathscr{S} \times \mathscr{A}$ is treated equivalently as an SA-by-1 column vector, where the entries can be arranged in any agreed order, for example

$$\left[g(1,1),\ldots,g(1,A),\ldots,g(S,1),\ldots,g(S,A)\right]^\top.$$

An MDP entails a description of the environment and a specification of the goal of learning. Regarding the former, we consider interactions between an agent and the environment taking place in discrete time cycles $t = 1,2,\ldots$. In cycle t, the agent receives from the environment a *state* $s_t \in \mathscr{S}$, then sends back an *action* $a_t \in \mathscr{A}$. In doing this, the agent collects a *reward* $r_t = R(s_t, a_t)$.

The defining characteristic of an MDP is the following assumption [see e.g., Sutton and Barto, 1998, Section 3.5].

Assumption 2.1 (Markov property) *For any $t \geq 1$ and any sequence of the form $s_1 a_1 \cdots s_t a_t s_{t+1}$, where $s_1,\ldots,s_{t+1} \in \mathscr{S}$, $a_1,\ldots,a_t \in \mathscr{A}$, we have*

$$\mathbb{P}\left[s_{t+1}|s_1 a_1 \cdots s_t a_t\right] = P(s_{t+1}|s_t, a_t), \quad \textit{and} \quad r_t = R(s_t, at).$$

Assumption 2.1 says that in an MDP, previous states and actions have no effect on the transition probability, and we may view s_{t+1} as a random sample from the transition model $P\left(\cdot|s_t, a_t\right)$. In other words, s_t contains all relevant information for predicting the states and rewards beyond time t, in response to any of the agent's actions.

During the interaction, the agent has to select an action at each time step. The rule according to which the agent chooses its actions are referred to as the *policy*. In MDP, a policy π is a mapping from the state space to the set of probability mass functions over $\mathscr{A}$, such that $\pi(a|s)$ is the probability of selecting action a when the current state is s. We also denote $\pi(s)$ to be a unique action chosen if the policy is *deterministic*.

Policy π, together with the transition model of the environment and a given initial state s_1, jointly define a probability distribution over all sequences of the form $s_1 a_1 \cdots s_t a_t$, such that

$$\mathbb{P}^\pi\left[s_1 a_1 \cdots s_t a_t | s_1\right] = \left[\prod_{\tau=1}^{t} \pi\left(a_\tau|s_\tau\right)\right]\left[\prod_{\tau=2}^{t} P\left(s_\tau|s_{\tau-1}, a_\tau\right)\right],$$

where we use the superscript to highlight the dependency on the policy. The *value function* of a policy π, denoted v^π, is thereby defined as a real-valued function on $\mathscr{S}$, such that

$$v^\pi(s) = \lim_{t\to\infty} \mathbb{E}^\pi\left[\sum_{\tau=1}^{t} \gamma^{\tau-1} r_\tau \middle| s_1 = s\right].$$

Here the expectation $\mathbb{E}^\pi$ is taken with respect to the distribution $\mathbb{P}^\pi$ and initial state s. The limit always exists thanks to the boundedness of $R(s,a)$ and the fact that $\gamma < 1$, and we may simply write

$$v^\pi(s) = \mathbb{E}^\pi\left[\sum_{\tau=1}^{\infty} \gamma^{\tau-1} r_\tau \middle| s_1 = s\right]. \tag{2.1}$$

The value function of a policy π, when evaluated at a particular state s, is the expectation of the exponentially discounted sum of all future rewards, assuming that the agent starts from s and follows π.

A closely related concept is the so-called *action-value function*, denoted q^π, which is defined as a mapping from $\mathscr{S} \times \mathscr{A}$ to $\mathbb{R}$, such that

$$q^\pi(s,a) = \mathbb{E}^\pi\left[\sum_{\tau=1}^{\infty} \gamma^{\tau-1} r_\tau \middle| s_1 = s, a_1 = a\right].$$

The action-value $q^\pi(s,a)$ amounts to the expected discounted sum of future rewards, assuming that the agent starts from state s, performs action a, and then follows policy π afterwards. It is easy to see that

$$v^\pi(s) = \sum_{a \in \mathscr{A}} \pi(a|s)\, q^\pi(s,a).$$

We are now ready to specify the goal of RL in MDP, which is to

> *find an optimal policy π^*, such that v^{π^*} is maximal in the sense that $v^{\pi^*}(s) \geq v^\pi(s)$ for any policy π and any $s \in \mathscr{S}$.*

By definition all optimal policies share the same value function. We thereby denote v^* to be this optimal value function. In addition, the optimal policies also define a unique optimal action-value function, which we denote q^*, such that $q^*(s,a) \geq q^\pi(s,a)$ for any $s \in \mathscr{S}$, $a \in \mathscr{A}$ and any policy π.

It is known [see e.g., Sutton and Barto, 1998, Section 3.8] that for every MDP there exists at least one deterministic policy that is optimal. Therefore, the learning problem can in principle be solved by brute-force search over all deterministic policies, while much more efficient alternatives will be discussed later in Section 2.2.

2.1.2 Beyond MDP

MDP has a number of attractive theoretical properties, and thus receives much attention in RL research. However, this relatively simple framework is limited in many ways, to which various extensions are proposed in the literature. In this section, we briefly survey some of these variations and extensions, in order to provide the readers a slightly more complete picture of the field. We will not go into the details of the topics mentioned, and content ourselves with explaining the basic idea and motivation.

When confined to MDP, there are some minor variations in the formulation. The first is on the definition of the reward model. Depending on the problem, reward models can be changed to $R(s)$ or $R(s,a,s')$. Moreover, it is common to view $R(s,a)$ as random variables [e.g., see Szepesvári, 2010]. This assumption is central in *bandit problems* [e.g., see Agrawal, 1995; Garivier and Cappè, 2011] where $S = 1$ and the goal is to find a^* with $\mathbb{E}[R(1,a^*)]$ maximized. Clearly, the problem is trivial if R is deterministic. In the general setting where $S > 1$, all these modifications will not alter our discussion significantly. The second variation is on the definition of value function. If the task is *episodic*, i.e., the

interactions stop after a finite number of steps, say T, then the value function can be defined as

$$v^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=1}^{T} r_t \middle| s_1 = s \right],$$

where the discount is removed [see e.g., Sutton and Barto, 1998, Section 3.3]. Sometimes it is also beneficial to consider the value function defined as the average reward [e.g., see Mahadevan, 1996; Jaksch et al., 2010]:

$$v^\pi(s) = \lim_{T\to\infty} \mathbb{E}^\pi \left[\frac{1}{T} \sum_{t=1}^{T} r_t \middle| s_1 = s \right].$$

The basic transition structure of an MDP can also be altered to formulate a number of other learning problems studied in RL. Examples include *multi-objective* RL, which considers several reward functions instead of one, and tries to find the set of admissible policies that are non-dominated [Barrett and Narayanan, 2008]; *multi-agent* RL studies the scenario where multiple agents act in a single environment [Busoniu et al., 2008]; *inverse* RL tries to recover the reward model instead of the value function from examples generated by following (near) optimal policies [Ng and Russell, 2000; Ramachandran and Amir, 2007; Ziebart et al., 2008]; and *risk-sensitive* RL tries to maximize the value function while at the same time reduce the fluctuation in return [Mihatsch and Neuneier, 2002; Mannor and Tsitsiklis, 2011].

Representing the transition model of an MDP requires $O\left(S^2A\right)$ parameters. Sometimes, the transition model exhibits a certain 'factored' that allows it to be represented by much fewer parameters. *Factored MDPs* exploit this idea by assuming that the state s_t is described by a set of variables $x_t^{(1)}, \ldots, x_t^{(n)}$, and the transition model is given as

$$\mathbb{P}\left(x_{t+1}^{(1)}, \ldots, x_{t+1}^{(n)} | x_t^{(1)}, \ldots, x_t^{(n)}, a_t\right) = \prod_{i=1}^{n} p_i\left(x_{t+1}^{(i)} | x_t^{(1)}, \ldots, x_t^{(n)}, a_t\right),$$

where each p_i is parameterized by m parameters. Assuming $x_t^{(1)}, \ldots, x_t^{(n)}$ are all binary, then the state space is of size 2^n, while the number of parameters describing the transition model is merely nm—exponentially smaller than the size of the state space if m is polynomial in n. A rich literature is devoted to factored MDPs, and efficient solutions have been developed [e.g., see Guestrin et al., 2003; Szita and Lörincz, 2009; Chakraborty and Stone, 2011].

A further extension to MDPs is to drop the assumptions of discrete time as well as finite state and action space. In particular, *continuous-time MDP* remove

the discrete time assumption [Guo and Hernández-Lerma, 2009], and MDPs with continuous state and action spaces are also studied, particularly in the context of control and robotics [e.g., see van Hasselt and Wiering, 2007, for a survey].

Perhaps the most significant deviation from the MDP framework is to remove the Markov Property (Assumption 2.1). Problems belonging to this class are generally referred to as *non-Markovian*, where, instead of receiving the state s_t at each step, the agent receives an observation o_t, which by itself is not enough to recover the state information. As a result, the agent has to make use of potentially all the history information (i.e., memory). Non-Markovian problems are hard, and only approximate solutions can be expected.

Roughly speaking, there are three methodologies for solving non-Markovian problems. The first is to reduce the problem to a Markovian one by extracting rich feature representations from the history observations, which are then treated as the states of the environment. The feature representations can be generated explicitly, e.g., by recurrent neural networks [Szita et al., 2006] or more generally by searching over program spaces [Hutter, 2009a,b], as well as implicitly, e.g., by feature maps induced by metric kernels defined over history observations [McCallum, 1996]. This approach will be referred to later in Section 2.3.2.

The second methodology is generally referred to as *policy search*, where we search for a good policy from a pre-defined class of functions, (in the most general case all computable policies [see Hutter, 2005; Veness et al., 2011].) Here the policies are evaluated solely based on their empirical performance, and the search can take various forms, including optimization, Bayesian updating, or even random guessing. One of the most successful classes in this family are the so-called *policy gradient* methods (see for example [Baxter and Barlett, 2001; Wierstra and Schmidhuber, 2007; Wierstra et al., 2007; Peters and Schaal, 2008], including the author's work [Rückstieß et al., 2010]), and the closely related evolutionary optimization methods (see e.g., [Yao, 1999], as well as the author's own work [Sun et al., 2009b,a; Glasmachers et al., 2010]). It has been demonstrated that these algorithms are capable of solving complex non-Markovian control tasks.

Policy search methods treat the environment as a black box, and only rely on it for the empirical evaluation of the policy. To this extent, these algorithms are *model-free*, in that they do not need to construct an explicit model of the environment. In contrast, the third methodology, which we referred to as *model-based* approach, learns the model of the environment first (the *model-learning* step), then the policy can be obtained by planning on the learned model (the

planning step). Here the model classes may be recurrent neural networks (RNN; see e.g., [Lin and Mitchell, 1993; Whitehead and Lin, 1995; Schäfer, 2008]) or probabilistic graphical models (PGM; see e.g., [Toussaint et al., 2006; Vlassis et al., 2009]). The models are usually learned by gradient descent or stochastic search in the RNN case, and by maximum likelihood estimation, in particular variations of Expectation-Maximization (EM; [Dempster et al., 1977]) method, in the case of PGMs (e.g., [Vlassis et al., 2009], see also the author's own work in learning RNNs using EM-type algorithms [Unkelbach et al., 2009]). Planning on the learned models is usually hard, and the policies are obtained from the model using either policy search or various forms of approximate planning algorithms [see e.g., Sutton, 1990; Mauá and de Campos, 2011; Mauá et al., 2012].

A particularly well-studied framework for non-Markovian RL is the *Partially Observable MDP* (POMDP; [see e.g., Murphy, 2000]), where the environment is assumed to follow the generative model

$$o_t \sim p_{\text{obs}}\left(\cdot|x_t\right), \quad x_{t+1} \sim p_{\text{state}}\left(\cdot|x_t, a_t\right).$$

Here x_t is a 'hidden state' variable assumed to be finite, and p_{obs} and p_{state} are two conditional distributions respectively describing how the observations are generated and how the hidden states are updated. A closely related extension to the POMDP is the so-called *Predictive State Representation* (PSR; [Littman et al., 2001]), where the observations are generated according to a stochastic multiplicative automata

$$o_t \sim \text{mul}\left(\xi_t\right), \quad \xi_{t+1} = \Gamma_{o_t,a_t}\xi_t,$$

where ξ_t is a probability vector, i.e., a non-negative vector summing up to 1, and $\text{mul}\left(\xi_t\right)$ is the multinomial distribution parameterized by ξ_t. In PSR, ξ_t can be viewed as the state, which is updated at each time step by left multiplication with matrix Γ_{o_t,a_t} depending on the action a_t and observation o_t. POMDP is a special case of PSR, where ξ_t represents the conditional distribution of x_t given the history. Learning POMDPs can be done via EM. However, recent studies have shown that PSRs (and thus POMDPs) can also be learned using spectral methods [Rosencrantz et al., 2004; James and Singh, 2004; McCracken and Bowling, 2005; Wolfe et al., 2005], yielding superior performance. Exact planning in POMDPs (and thus PSRs) is theoretically hard [Mauá and de Campos, 2011], however, efficient heuristic, approximate algorithms exist (e.g., see [Monahan, 1982; Kaelbling et al., 1998; Zhang, 2001; Fleck, 2004; Ross et al., 2008] for POMDP and [Izadi and Precup, 2003; Boots et al., 2009] for PSR).

2.2 Exact Solutions to MDP

In this section we review algorithms for solving MDPs, based on the Bellman equation (introduced below), and are 'exact' in the sense that they guarantee convergence to the optimal value or action-value functions, or at least to the value or action-value functions of the policy being evaluated.

As stated in Section 2.1.2, solutions to RL problems can generally be partitioned into model-based and model-free, depending on whether they require model-learning as an intermediate step. In Section 2.2.2, we focus on the planning step in the model-based methods, introducing the so-called DP solutions, while the model-learning step will be investigated later in Chapter 5. Model-free solutions to MDPs follow in Section 2.2.3.

2.2.1 Bellman Equations

From the definition of v^π, it is straightforward to verify the following *Bellman equation*

$$v^\pi(s) = \sum_{a \in \mathscr{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v^\pi\left(s'\right) \right]. \tag{2.2}$$

In fact, v^π is the only solution to the above equation. A similar Bellman equation holds for the action-value function

$$q^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) \sum_{a' \in \mathscr{A}} \pi\left(a'|s'\right) q^\pi\left(s',a'\right), \tag{2.3}$$

where the solution q^π is also unique. It can also be seen that the value and action-value functions are related through:

$$q^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v^\pi\left(s'\right).$$

Bellman equations form a finite set of linear equations with $v^\pi(s)$ or $q^\pi(s,a)$ as unknowns. Therefore, v^π and q^π, though defined as infinite sums (see Equation 2.1), can be computed in time polynomial in S and A.

Perhaps the most fundamental results concerning MDP is the following theorem established by Bellman [1952], which states that the optimal value and action-value functions, and consequently the optimal policies, can be characterized by the so-called Bellman optimality equations.

Theorem 2.1 (Bellman) *The optimal value function v^* and optimal action-value function q^* are respectively the unique solutions to the following* Bellman optimality equations

$$v^*(s) = \max_{a \in \mathscr{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v^*\left(s'\right) \right], \tag{2.4}$$

and

$$q^*(s,a) = R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) \max_{a' \in \mathscr{A}} q^\pi\left(s',a'\right). \tag{2.5}$$

Moreover, an optimal, deterministic policy π^* can be obtained by choosing greedily with respect to $q^*(s,a)$, such that

$$\pi^*(s) = \operatorname*{argmax}_{a \in \mathscr{A}} q^*(s,a), \tag{2.6}$$

or equivalently

$$\pi^*(s) = \operatorname*{argmax}_{a \in \mathscr{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v^*\left(s'\right) \right], \tag{2.7}$$

where ties may be broken in an arbitrary way.

Theorem 2.1 reduces the problem of MDP planning to the computation of optimal value or action-value functions, from which the optimal policies can be 'read out' using Equation 2.7 or 2.6. It also implies that the optimal value function can be obtained by solving the following Linear Programming (LP) problem [d'Epenoux, 1963]:

$$\min \sum_{s \in \mathscr{S}} v^*(s) \text{ s.t.}$$
$$v^*(s) \geq R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v^*\left(s'\right), \forall s \in \mathscr{S}, a \in \mathscr{A}.$$

This linear program contains S variables with $S \times A$ constraints, therefore can be solved in time polynomial in S and A [Littman et al., 1995]. This in turn shows that MDP planning is solvable in polynomial time.

2.2.2 Model-based Planning Algorithms

The LP solution to MDP offers little insight into the structure of the problem, and is often found to be less efficient than the algorithms presented in this section

[see Sutton and Barto, 1998, Section 4.7], which are instances of a family of solutions based on dynamic programming (DP).

Roughly speaking, all DP solutions work by iteratively updating $v^{(i)}$ or $q^{(i)}$, which are estimations to the target value or action-value functions. In iteration i, $v^{(i)}$ (or $q^{(i)}$) is updated to $v^{(i+1)}$ (or $q^{(i+1)}$) by applying an operator $\mathscr{T}_i$. The set of operators $\{\mathscr{T}_i\}$ is so designed that the sequence $\mathscr{T}_1\mathscr{T}_2\cdots$ forms a contraction, whose fixed point is v^* (or q^*) when the goal is to find an optimal policy, and v^π (or q^π) when we want to evaluate a given policy. Almost exclusively, the operators $\mathscr{T}_i$ are variations of the right hand side of the Bellman optimality equations or Bellman equations.

We start from the evaluation of v^π, which may be done by applying S operators $\mathscr{T}^{(1)},\ldots,\mathscr{T}^{(S)}$ cyclically over an initial value estimate. The operator $\mathscr{T}^{(s)}$ is so defined that $v(s)$ is updated according to the right hand side of the Bellman equation

$$v(s) \leftarrow \sum_{a\in\mathscr{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'\in\mathscr{S}} P\left(s'|s,a\right) v\left(s'\right) \right],$$

and all other $v(s')$ for $s' \neq s$ remain the same. Another way is to write

$$\mathscr{T}^{(s)}v = v + \underbrace{\left(\sum_{a\in\mathscr{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'\in\mathscr{S}} P\left(s'|s,a\right) v\left(s'\right) \right] - v(s) \right)}_{\varepsilon_v(s)} e_s, \quad (2.8)$$

where e_s is the function over $\mathscr{S}$ such that $e_s(s) = 1$ and $e_s(s') = 0$ for $s \neq s'$. The term inside the parentheses, $\varepsilon_v(s)$, corresponds to the difference between the right hand side and the left hand side of the Bellman equation, and we call ε_v the *Bellman error* associated with v. It is straightforward to see that $v = v^\pi$ if and only if the Bellman error becomes zero, and the above procedure may be viewed as an attempt to reduce the Bellman error iteratively, one state at a time. It is known that the sequence $\left\{v^{(i)}\right\}$ converges to v^π. In fact, the convergence to v^π continues to hold even if we mix $\mathscr{T}^{(s)}$ arbitrarily, provided that every $\mathscr{T}^{(s)}$ gets applied from time to time.

Similarly, to evaluate the action-value function q^π, we may devise a set of operators $\mathscr{T}^{(s,a)}$ for $s \in \mathscr{S}$ and $a \in \mathscr{A}$, such that

$$\mathscr{T}^{(s,a)}q = q + \left(R(s,a) + \gamma \sum_{s'\in\mathscr{S}} P\left(s'|s,a\right) \sum_{a'\in\mathscr{A}} \pi\left(a'|s'\right) q\left(s',a'\right) - q(s,a) \right) e_{s,a}, \quad (2.9)$$

where $e_{s,a}$ is defined as $e_{s,a}(s,a) = 1$ and $e_{s,a}(s',a') = 0$ for $s' \neq s$ or $a' \neq a$. Again, the term in the parentheses can be viewed as a variant of Bellman error corresponding to the action-value function, and the convergence to q^π holds provided that each $\mathcal{T}^{(s,a)}$ is applied infinitely often.

We now turn our attention to methods for MDP planning, namely computing v^* or q^*. The most basic algorithm is *policy iteration*, which repeatedly alternates between *policy evaluation*, i.e., computing $v^{(i)}$ (or $q^{(i)}$) with respect to the current policy $\pi^{(i)}$ using the above mentioned procedures, and *policy improvement*, where we update the policy greedily with respect to the current estimation $v^{(i)}$ (or $q^{(i)}$) as

$$\pi^{(i+1)}(s) = \operatorname*{argmax}_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P\left(s'|s,a\right) v^{(i)}\left(s'\right) \right], \tag{2.10}$$

or in the action-value case simply

$$\pi^{(i+1)}(s) = \operatorname*{argmax}_{a \in \mathcal{A}} q^{(i)}(s,a). \tag{2.11}$$

It is known [e.g., see Sutton and Barto, 1998, Section 4.3] that the sequence $v^{(i)}$ (or $q^{(i)}$) converges to the optimal value function v^* (or optimal action-value function q^*).

The general structure of policy iteration, i.e., alternating between policy evaluation and policy improvement, is shared among a wide variety of algorithms, often beyond the scope of MDP, although the details of the two steps may vary significantly. For example, in policy evaluation, the value or action-value function may be estimated from sample trajectories instead being of computed directly from the model, and the value function may be represented by a (parametric or non-parametric) function instead of in tabular form. In policy improvement, the policy may also be represented by some function approximator, and the update may be less greedy than Equation 2.10 or 2.11, e.g., applying various types of soft-max or following gradient steps. In addition, the policy evaluation and policy improvement steps might be interleaved at a finer granularity to speed up the convergence [see Sutton and Barto, 1998, Section 4.5], where it is possible (and often preferred) to perform policy improvement after only a few iterations of policy evaluation. In particular, *value iteration* pairs one iteration of policy evaluation with one step of policy improvement, by applying cyclically a set of S operators $\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(S)}$, defined as

$$\mathcal{T}^{(s)} v = v + \left(\max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P\left(s'|s,a\right) v\left(s'\right) \right] - v(s) \right) e_s.$$

Similarly, in the action-value case we may compute q^* by applying a set of operators of the form

$$\mathscr{T}^{(s,a)}q = q + \left(R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) \max_{a \in \mathscr{A}} q\left(s',a'\right) - q(s,a) \right) e_{s,a}, \quad (2.12)$$

which leads to Q-learning [Watkins, 1989; Watkins and Dayan, 1992]; to be explained in the next section.

2.2.3 Model-free Algorithms

In this section we introduce Temporal Difference (TD) methods, which are among the most successful model-free algorithms for solving MDPs, and are often regarded as the key achievement setting the field of RL apart. The common characteristic of TD methods is to learn the value or action-value function from past experience using stochastic versions of the DP updates.

The first member in the family of TD methods, to which we refer simply as *TD*, is for policy evaluation [Sutton, 1988], i.e., computing v^π for a given policy π. If the model is known, then v^π can be obtained by applying Equation 2.8, which we write down again for clarity:

$$\begin{aligned}\mathscr{T}^{(s)}v &= v + \left(\sum_{a \in \mathscr{A}} \pi(a|s) \left[R(s,a) + \gamma \sum_{s' \in \mathscr{S}} P\left(s'|s,a\right) v\left(s'\right) \right] - v(s) \right) e_s \\ &= v + \varepsilon_v(s)\, e_s.\end{aligned}$$

In the model-free setting, the aim is to estimate v^π from a *sample trajectory*, i.e., a sequence of the form $s_1 a_1 r_1 s_2 a_2 r_2 \cdots$, or more generally from a set of *samples*, namely 4-tuples of the form $\left(s_t a_t r_t s'_t\right)$ where s'_t is the state following s_t. The samples may be obtained from a single sample trajectory, in which case $s'_t = s_{t+1}$, or from multiple trajectories. TD estimates v^π by applying three key modifications to the DP procedure:

1. Since $\mathscr{T}^{(s)}$ can be arranged in arbitrary order, we may as well arrange them according to the order in which states are experienced (or samples are provided), namely, we apply the operators in the order $\mathscr{T}^{(s_1)}, \mathscr{T}^{(s_2)}, \ldots$, in accordance with the sample trajectory.

2. When applying $\mathscr{T}^{(s_t)}$, we replace the Bellman error $\varepsilon_v\left(s_t\right)$ with the fol-

lowing sample estimation:

$$\varepsilon_v(s_t) = \underbrace{\sum_{a\in\mathscr{A}} \pi(a|s_t) R(s_t,a)}_{\simeq r_t} + \gamma \underbrace{\sum_{a\in\mathscr{A}} \pi(a|s_t) \sum_{s'\in\mathscr{S}} P(s'|s_t,a) v(s')}_{\simeq v(s_{t+1})} - v(s_t)$$

$$\simeq r_t + \gamma v(s_{t+1}) - v(s_t).$$

Define the *temporal difference error* as

$$\varepsilon_t = r_t + \gamma v^{(t)}(s_{t+1}) - v^{(t)}(s_t),$$

then the updating formula becomes

$$v^{(t+1)} \leftarrow v^{(t)} + \varepsilon_t e_{s_t}.$$

3. Finally, as the update is contaminated by sample noise, we reduce it by introducing a small *learning rate* α_t. This leads to the TD updating formula:

$$v^{(t+1)} \leftarrow v^{(t)} + \alpha_t \varepsilon_t e_{s_t}. \tag{2.13}$$

It has been shown that under fairly standard conditions, e.g., the learning rate satisfying $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$, the sequence $v^{(t)}$ will converge to v^π almost surely [Dayan and Sejnowski, 1994]. In the following we always assume that the learning rate $\{\alpha_t\}$ satisfies these conditions.

TD estimates value functions from samples in a model-free fashion. However, making use of the obtained value functions in policy improvement steps still requires a model (see Equation 2.10). The 'control' versions of TD methods, on the other hand, eliminate the need for models altogether by estimating instead the action-value functions, which by themselves provide enough information for policy improvement (see Equation 2.11). In particular, applying the above mentioned transformations to Equation 2.9 yields *SARSA* (State-Action-Reward-State-Action; [Rummery and Niranjan, 1994; Rummery, 1994])

$$q^{(t+1)} = q^{(t)} + \alpha_t \left(r_t + \gamma q^{(t)}(s_{t+1}, a_{t+1}) - q(s_t, a_t)\right) e_{s_t,a_t},$$

and to Equation 2.12 leads to *Q-learning* [Watkins, 1989; Watkins and Dayan, 1992]:

$$q^{(t+1)} = q^{(t)} + \alpha_t \left(r_t + \gamma \max_{a\in\mathscr{A}} q(s_{t+1}, a) - q(s_t, a_t)\right) e_{s_t,a_t}.$$

In SARSA, the learning is *on-policy*, that is, the sample trajectory has to be generated by following the near greedy policy (to be explained below) with respect to

the current action-value estimation $q^{(t)}$ to ensure convergence to q^*. In contrast, Q-learning is *off-policy*, where convergence to q^* holds irrespective of the policy generating the sample trajectory, as long as all state-action pairs get visited infinitely often.

SARSA and Q-learning store the current estimate of the action-value function. As a result, at any time, a greedy policy can be read out using Equation 2.11. It is tempting to use this policy to generate future samples. However, this often leads to poor performance due to the *exploration exploitation dilemma*, which states that if we act completely greedily according to the current estimation of the value or action-value function, then we may confine ourselves to a small part of the state-space and miss the opportunity of hitting states giving large rewards. With respect to SARSA and Q-learning, this means that if we always choose the action a_t that maximizes the current $q\left(s_t, a_t\right)$, then $q\left(s_t, a\right)$ for $a \neq a_t$ may never get updated due to bad initial value. To solve this problem, policy improvement needs to be 'less greedy' than in Equation 2.11, in order to allow all state-action pairs to be tried often enough. Two types of policy improvement steps are commonly used:

ε-greedy : at time t, with probability $1-\varepsilon_t$, choose

$$a_t = \underset{a}{\operatorname{argmax}}\, q\left(s_t, a\right),$$

and with probability ε_t, choose a random action is chosen. If ε_t goes to zero at a speed that allows all (s, a) pairs visited infinitely often, then both SARSA and Q-learning will produce the optimal policy asymptotically.

softmax : sample the actions from Boltzmann distributions, where action a_t is chosen with the probability proportional to

$$\exp\left(\frac{1}{\beta_t} q\left(s_t, a_t\right)\right),$$

i.e., sub-optimal actions are selected exponentially less often. As with ε_t, if β_t, the temperature, is decrease at a speed that is not too fast to allow all (s, a) pairs visited infinitely often, then both SARSA and Q-learning produce the optimal policy.

In addition, the *optimistic initialization* trick may also be used, i.e., initialize the action-value table with large numbers, so that the unvisited state-action pairs will always be preferred [see e.g., Strehl et al., 2006; Kolter and Ng, 2009b]. More sophisticated algorithms are designed for the bandit case, assuming that the random variables $R(1, a)$ are well-behaved (e.g., bounded, see [Agrawal, 1995; Garivier and Cappè, 2011]).

2.3 Function Approximation

The algorithms presented in the previous section find solutions by iteratively updating the value or action-value table, and therefore their complexity scales at least linearly in the cardinality of the state space. This explicit dependency limits the usefulness of these algorithms for three reasons:

1. In many real-world applications, the cardinality of the state space is exceedingly large, especially when the state space is defined combinatorially. For example, if the observation consists of n bits, then the potential number of states will be $S = 2^n$. In such cases, storing the value or action-value function is impossible.

2. Particular to the model-free methods, even if the value or action-value table can be store, all the entries still have to be learned. This amounts to estimating at least S real-valued parameters from sample trajectories. If S is large, the length of the sample trajectory required for reliable estimation becomes impractically long, while the value or action-value function computed from trajectories of limited length is of high risk and cannot be expected to generalize well.

3. In some cases, maintaining the value and action-value function in tables prohibits us from incorporating important domain knowledge. For example, in control problems, continuous state and action spaces are often discretized, and hence the value or action-value functions tend to be smooth with respect to the metric inherited from the continuous space. This bias may be incorporated to regularize the solution, which improves generalization and reduces the required sample size. However, this cannot be done using the algorithms mentioned in the previous section.

For these reasons, it is often necessary to replace the tabular representation with a (parameterized or non-parameterized) estimator, *even if storing the value or action-value function is viable*. This technique is generally referred to as *Function Approximation* (FA).

Usually, FA is combined with TD methods to provide practical solutions to MDPs with large state spaces. However, it should be pointed out that not only does FA reduce the computational cost and improve the generalization, it also stretches the applicability of TD methods. For example, in combination with function approximators that take continuous input, TD methods can solve MDPs with continuous state spaces [e.g., see van Hasselt and Wiering, 2007; Maei

et al., 2009]. Moreover, if the function approximator is history dependent, namely the value function is estimated from both the current and previous observations, then the combination of FA and TD methods can potentially be used to solve non-Markovian problems [e.g., see Sutton, 1990; Whitehead and Lin, 1995; Szita et al., 2006; Schäfer, 2008; Boots and Gordon, 2010].

Generally speaking, there are two approaches for incorporating FA in TD methods. The first is to adapt TD methods to FA. The convergence of the resulting algorithms relies on the preservation of the contraction properties of the operators, which is often problematic (e.g., see [Baird, 1999] for an example where Q-learning with FA diverges.) The second approach starts from treating the RL problem as an optimization problem, where the objective is to minimize some measure of Bellman error. Note that Bellman error vanishes when the value function estimates approach the true value function, and in general small Bellman error implies good value function estimates, even if the error cannot be reduced to zero because the chosen class of estimators is not rich enough to represent the true value function. Often, the optimization point of view is preferred for three reasons:

1. Formulating RL as an optimization problem may produce analytical solutions, especially in the linear case. The best example is the Least Square TD (LSTD) algorithm to be described in Section 2.3.1.

2. The optimization point of view leads to (stochastic) gradient descent type of algorithms, which are often easier to analyze and enjoy better convergence properties, especially when the function approximators are nonlinear [e.g., see Maei et al., 2009].

3. It is easy to incorporate various regularizers into the objective function, making the solution generalize better. This is particularly useful in the under-sampled case, where the number of parameters in the function approximator is more than, or at least comparable to, the number of samples. A good example is the LARS-TD algorithm by Kolter and Ng [2009a], where L_1 regularizations are added to enforce sparse solutions.

In this section, we review FA methods in RL, focusing exclusively on Linear Function Approximation (LFA), where the value or action-value function is represented as a linear combination of a set of basis functions. This bias towards LFA is for two reasons: First, compared with other techniques, LFA is better understood theoretically. Second, the work presented in this dissertation (Chapter

3 and 4) is all based on LFA, making it more relevant than its non-linear counterpart. However, it is worth pointing out that non-linear FA methods are by no means unimportant, and they are at least as widely used as LFA, demonstrating great practicality in various applications including the famous Backgammon example [Tesauro, 1995].

The basic idea of LFA is to approximate the value function v^π as

$$v^\pi(s) \simeq \sum_{n=1}^{N} \theta_n \phi_n(s),$$

where $\phi_1, \ldots, \phi_N$ are N real-valued functions defined over $\mathcal{S}$, called the *basis functions*, and $\theta_1, \ldots, \theta_N$ are their respective weights. Similarly, for the action-value function, we have

$$q^\pi(s,a) \simeq \sum_{n=1}^{N} \theta_n \phi_n(s,a).$$

For simplicity, in the rest of this section we focus on the problem of evaluating value function v^π with fixed policy π. Results obtained here can readily be extended to the action-value case, albeit with slightly complicated notations for incorporating the action parameter [Lagoudakis and Parr, 2003; Busoniu et al., 2010]. These algorithms form the policy evaluation step in policy iteration.

When the policy is fixed, it is often beneficial to marginalize out the actions and consider the resulting Markov chain defined over the state space. We thereby define a *Markov reward process* (MRP) as the following 4-tuple $\left(\mathcal{S}, P^\pi, r^\pi, \gamma\right)$, where $\mathcal{S}$ and γ are defined as before, while P^π is a *transition kernel* of the Markov chain defined over $\mathcal{S}$, such that

$$P^\pi\left(s'|s\right) = \sum_{a \in \mathcal{A}} \pi(a|s) P\left(s'|s,a\right)$$

and r^π is a real-valued *reward function* on $\mathcal{S}$ containing the expected reward

$$r^\pi(s) = \sum_{a \in \mathcal{A}} R(s,a) \pi(a|s).$$

With a fixed policy, we drop the superscripts π, and write v^π, P^π, and r^π as v, P, and r, respectively. The MRP admits a particularly convenient matrix representation, of which we make heavy usage, especially in Chapter 4. Namely, we interpret functions over $\mathcal{S}$ as column vectors as before, and treat P as an S-by-S matrix with the (s,s')-th entry $P(s'|s)$. It is easy to check that the Bellman equation can be written as

$$v = r + \gamma P v,$$

or $v = \left(I - \gamma P\right)^{-} r$, where we denote Q^{-} the inverse of matrix Q. Denote $L = I - \gamma P$, then $v = L^{-} r$.

Let $\Phi = \left[\phi_1, \ldots, \phi_N\right]$ be the S-by-N *feature matrix* and $\theta = \left[\theta_1, \ldots, \theta_N\right]^{\top}$ be an N-by-1 weight vector, then $\Phi\theta$ approximates the value function v, while the Bellman equation now becomes

$$\Phi\theta = r + \gamma P \Phi\theta,$$

and the Bellman error is simply

$$\varepsilon = r + \gamma P\Phi\theta - \Phi\theta = r - L\Phi\theta.$$

It is straightforward to see that if we have $N = S$ linearly independent basis functions, then Φ becomes an S-by-S invertible matrix, and the Bellman equation can be solved exactly, giving

$$\theta = \Phi^{-} L^{-} r.$$

Moreover, if $\Phi = I$, then the problem become exactly the same as in the tabular case, with $\theta_s = v(s)$ for state s.

Applying LFA involves two steps, namely (i) constructing the set of basis functions Φ, and (ii) estimating the weights θ. The second step has been well studied in the literature, and we will review some of the key results in Section 2.3.1. The first step, being one of the main topics of the present dissertation, is an open problem, and we briefly survey the literature in Section 2.3.2.

2.3.1 Estimating Weights

In this section, we assume Φ is given, and focus on the problem of computing θ. In the general case that $N \ll S$, the equation above cannot be solved exactly for all reward functions.

Let $\omega(s)$ be the transpose of the s-th row of Φ, i.e., $\omega(s) = \left[\phi_1(s), \ldots, \phi_N(s)\right]^{\top}$, then $\omega(s)$ is the vector of *features* observed when the state is s. Suppose we are given a set of T samples from the MRP

$$\left(s_1, s_1', r_1\right), \left(s_2, s_2', r_2\right), \ldots, \left(s_T, s_T', r_T\right),$$

where s_t' is the state following s_t, i.e., $s_t' \sim P\left(\cdot | s_t\right)$. Let p be the probability mass function on $\mathcal{S}$ from which s_t is sampled and D the diagonal matrix with diagonal entries p. Note that if the samples are obtained from a single trajectory, then $s_t' = s_{t+1}$, and if moreover the Markov chain is ergodic, then p approaches the stationary distribution of the chain with increasing T, and in the limit case

$p^\top = p^\top P$. However, in the general case, the samples may be gathered from different pieces of trajectories so that $s' \neq s_{t+1}$. Below we write $\omega_t = \omega\left(s_t\right)$ and $\omega'_t = \omega\left(s'_t\right)$ for simplicity.

To compute θ, we may follow the first approach mentioned above, namely adapting TD to the LFA case. The key insight to this approach is to realize that the vector e_s, with $e_s(s) = 1$ and $e_s(s') = 0$ for $s' \neq s$, can be viewed as the vector of features observed in state s, with v being the weight vector. Substituting e_{s_t} with ω_t, and $v^{(t)}$ with $\theta^{(t)}$ in Equation 2.13 produces the TD update in the LFA case

$$\theta^{(t+1)} = \theta^{(t)} + \alpha_t \varepsilon_t \omega_t.$$

This update usually works well in practice, however, in the case where p deviates too far from the stationary distribution of P, the algorithm may diverge (see Sutton et al. [2009] for an explanation.)

Another class of methods for computing θ comes from the optimization point of view, where we define some objective function measuring the Bellman error. The default choice is to use quadratic objective functions of the form

$$J(\theta) = \varepsilon^\top \left(L^{-\top} K L^{-}\right) \varepsilon = (v - \Phi\theta)^\top K (v - \Phi\theta),$$

where K is some positive semi-definite matrix and we add L^- to both sides of K for notational convenience. The matrix $L^{-\top} K L^-$ determines how the Bellman error in different states is weighted. It is also straightforward to see from the second equality above that J measures the *value error*, i.e., the deviation of $\Phi\theta$ from the true value function v. Provided that $\Phi^\top K \Phi$ is invertible, the optimal weights minimizing J can be obtained analytically as

$$\hat{\theta} = \left(\Phi^\top K \Phi\right)^{-} \left(\Phi^\top K v\right).$$

To enable estimation of $\hat{\theta}$ from finite samples, the following three relations are used repeatedly:

$$\Phi^\top D P \Phi = \mathbb{E}\left[\omega_t \left(\omega'_t\right)^\top\right] = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \omega_t \left(\omega'_t\right)^\top,$$

$$\Phi^\top D \Phi = \mathbb{E}\left[\omega_t \omega_t^\top\right] = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \omega_t \left(\omega_t\right)^\top,$$

$$\Phi^\top D r = \mathbb{E}\left[r_t \omega_t\right] = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} r_t \omega_t,$$

where the expectation is taken with respect to distribution p. It can also be seen that the following relation holds

$$\Phi^\top P^\top D P \Phi = \mathbb{E}\left[\mathbb{E}\left[\omega_t' \left(\omega_t''\right)^\top \middle| s_t\right]\right],$$

where ω_t' and ω_t'' are the feature vectors corresponding to two *independent* transitions from s_t. As a result, estimating $\Phi^\top P^\top D P \Phi$ from the samples is nearly impossible, especially when S is large.

In the literature, there have been several different choices of K corresponding to different algorithms, the earliest being the *residual gradient* algorithm proposed by Baird [1995], where $K_{rg} = L^\top D L$. The optimal weights are thus given by

$$\begin{aligned}\hat{\theta}_{rg} &= \left(\Phi^\top L^\top D L \Phi\right)^- \left(\Phi^\top L^\top D r\right) \\ &= \left(\gamma^2 \Phi^\top P^\top D P \Phi - \gamma \Phi^\top D P \Phi - \gamma \Phi^\top P^\top D \Phi + \Phi^\top D \Phi\right)^- \\ &\cdot \left(\Phi^\top D r - \gamma \Phi^\top P^\top D r\right).\end{aligned}$$

The drawback of this choice is immediately clear from the previous discussion, as the term $\Phi^\top P^\top D P \Phi$ cannot be estimated from sample trajectories.

A significantly better choice is to set

$$K_{lstd} = L^\top \Pi_D D \Pi_D L,$$

where Π_D is a projection operator defined as $\Pi_D = \Phi\left(\Phi^\top D \Phi\right)^- \Phi^\top D$. This choice gives the *Least-Square TD* (LSTD; [Bradtke et al., 1996; Boyan, 2002]) algorithm, which in some sense has become the standard choice in LFA. The optimal weights in this case are given by

$$\begin{aligned}\hat{\theta}_{td} &= \left(\Phi^\top D L \Phi\right)^- \left(\Phi^\top D r\right) \\ &= \left(\Phi^\top D \Phi - \gamma \Phi^\top D P \Phi\right)^- \left(\Phi^\top D r\right),\end{aligned}$$

which can be estimated using samples as

$$\tilde{\theta}_{td} = \left(\frac{1}{T}\sum_{t=1}^{T} \omega_t \left(\omega_t - \gamma \omega_t'\right)^\top\right)^- \left(\frac{1}{T}\sum_{t=1}^{T} \omega_r r_t\right).$$

The reason of choosing the subscript 'td' will become clear from later discussion. The computational complexity of LSTD is $O\left(N^2 T\right)$, since the first term on the right hand side is a sum of rank-one updates and the inverse can be updated efficiently using the Woodbury identity (e.g., see [Boyan, 2002]). It is

also worth pointing out that various versions of kernelized LSTD have been proposed, where the features are of infinite dimensionality and the computational complexity of FA usually scales with $O\left(T^3\right)$, e.g., see Ormoneit and Sen [2002]; Engel [2005]; Xu [2006]; Xu et al. [2007]; Engel et al. [2003, 2005], and Taylor and Parr [2009] for a unified view.

When the number of features N is large, sometimes even the cost of $O\left(N^2\right)$ computations per sample is still too high to be affordable. There are three types of solutions:

1. If we make the assumption that the features are sparse, i.e., for each ω_t there are at most M entries which are non-zero, then iLSTD [Geramifard et al., 2006] can be applied to reduce the computation cost to $O\left(ZN + M^2\right)$ per time step, where Z is an algorithm parameter chosen such that $Z \ll N$.

2. From the LSTD objective function, Sutton et al. [2009] developed two gradient-descent type algorithms, namely GTD-2 and TDC, which have complexity $O(N)$ per sample update. A related algorithm was developed earlier in Sutton et al. [2008], where a different objective function is used:

$$K_{gtd} = \left(\Phi^\top DL\right)^\top \left(\Phi^\top DL\right).$$

   The optimal weight with respect to K_{gtd} is also $\hat{\theta}_{td}$, yet the gradient update takes a different form. Despite its low computational complexity, the gradient descent type of algorithm usually has a higher sample complexity when compared to LSTD, meaning that they require much more iterations (either over new samples or repeatedly over old samples) to reach the same accuracy as LSTD.

3. We can reduce the number of features from N to a much smaller number say M, then perform LSTD on the M features obtained. This amounts to performing feature dimensionality reduction, and is one of the main focuses of this dissertation. We defer the detailed discussion of this methodology to Section 2.3.2.

It is worth pointing out that a particularly interesting connection between TD and LSTD can be spotted from the result above. Namely, if we rewrite the definition of $\hat{\theta}_{td}$ as

$$\left(\Phi^\top D\Phi - \gamma\Phi^\top DP\Phi\right)\hat{\theta}_{td} = \Phi^\top Dr,$$

or

$$\mathbb{E}\left[\omega_t\left(\omega_t - \gamma\omega_{t+1}\right)^\top\right]\hat{\theta}_{td} = \mathbb{E}\left[\omega_t r_t\right],$$

then $\hat{\theta}_{td}$ is the fixed point of the following iteration

$$\begin{aligned}\theta^{(t+1)} &\leftarrow \theta^{(t)} + \alpha_t \left(\mathbb{E}\left[\omega_t r_t\right] - \mathbb{E}\left[\omega_t \left(\omega_t - \gamma\omega_{t+1}\right)^\top\right] \theta^{(t)} \right) \\ &= \theta^{(t)} + \alpha_t \left(\mathbb{E}\left[\omega_t \left(r_t + \gamma\omega_{t+1}^\top \theta^{(t)} - \omega_t^\top \theta^{(t)} \right)\right]\right) \\ &= \theta^{(t)} + \alpha_t \mathbb{E}\left[\varepsilon_t \omega_t\right].\end{aligned}$$

Therefore, if $\mathbb{E}\left[\varepsilon_t\omega_t\right]$ is replaced with its sample estimate $\varepsilon_t\omega_t$, then we get the TD update.

2.3.2 Generating Basis Functions

The effectiveness of LFA depends crucially on the choice of the set of basis functions $\phi_1, \ldots, \phi_N$ for the following reasons:

1. With a poor selection of basis functions, the value function cannot be well represented within the space spanned by $\phi_1, \ldots, \phi_N$, or equivalently the column space of Φ, even if we have infinite number of samples. To see this, note that the optimal value function estimation in LFA is given by

$$\begin{aligned}\Phi\hat{\theta} &= \Phi \left(\Phi^\top K \Phi\right)^{-} \Phi^\top K v \\ &= \Pi_K v,\end{aligned}$$

   where $\Pi_K = \Phi \left(\Phi^\top K \Phi\right)^{-} \Phi^\top K$ is the projection operator such that

$$\Pi_K v = \underset{\hat{v} \in \operatorname{span}\left\{\phi_1, \ldots, \phi_N\right\}}{\operatorname{argmin}} (\hat{v} - v)^\top K (\hat{v} - v).$$

   If the basis functions are chosen improperly, e.g., in the extreme case $\phi_n^\top K v = 0$ for $i = 1, \ldots, n$, then the error $v - \Pi_K v$ will be large.

2. The convergence rate of TD (as well as the gradient methods) may vary significantly among different choices of basis functions, even if they all span the same subspace. Note that the TD updates can be viewed as the stochastic approximation of

$$\theta^{(t+1)} = \theta^{(t)} + \alpha \left(\Phi^\top D r - \Phi^\top D L \Phi \theta^{(t)}\right),$$

   and the convergence rate of the recursion depends on the condition number of $\Phi^\top D L \Phi$. Indeed, if we replace Φ with a different feature matrix ΦW, where W is an N-by-N invertible matrix, then ΦW has the same column space as Φ, yet the condition number of $\Phi^\top D L \Phi$ and $W^\top \Phi^\top D L \Phi W$ can be very different.

Traditionally, the set of basis functions are constructed by experts after careful examination of the problem. However, as the size and complexity of the problem grows, hand-crafting basis functions becomes increasingly difficult, and it becomes necessary to resort to methods that generate basis functions automatically.

Depending on the information used, automatic basis generation methods can be roughly partitioned into two classes. *Reward-sensitive* methods make direct use of the reward (or equivalently, the Bellman error or TD error), while *reward-insensitive* methods generate basis functions relying only on information contained in the observations. The benefit of reward-sensitive methods is that the basis functions are tuned specifically to suit the task at hand, and thus accurate value function estimations may be achieved by using a very small set of such basis functions. However, the basis functions learned usually cannot be used across different tasks, even if the dynamics of the environment, i.e., the transition model, remains the same. In contrast, reward-insensitive methods tend to exhibit less dependency on the tasks, and the basis functions generated are more likely to reflect the structure of the transition dynamics. These basis functions can then be used across different tasks, albeit with degraded performance. It must be pointed out that even in the reward-insensitive methods, there can still be implicit dependency on the reward, as the policy that the agent follows to gather experience may be guided by the reward.

It is also possible to partition the methods according to the way in which the basis functions are generated. In *batch-based* methods, the set of basis functions is constructed at once from a given set of samples, while in *online* methods, the set is built up incrementally. Batch-based methods usually give better solutions thanks to the fact that the information contained in the samples can be processed altogether, while online methods have the benefit of allowing the set of basis functions to be continuously adapted.

In this section, we are particularly concerned with a family of automatic basis generation methods that take the form of *linear basis projection* (LBP). More specifically, these methods aim at finding an N-by-M matrix W, where $M \ll N$, so that the feature fed to the LFA at each time step becomes

$$\upsilon = W^\top \omega,$$

which is of size M instead of N. Equivalently, this amounts to replacing the original feature matrix Φ with $\Psi = \Phi W$, which is S-by-M dimensional, and the columns of Ψ are the constructed basis functions. This particular methodology is motivated by the following scenarios:

1. As stated above, for complex problems, it is often hard to design a small set of basis functions that are sufficient in representing the value function. However, it is usually possible to generate a large number of 'raw' basis functions, where each one of them is deemed 'marginally' related to estimating the value function [Boots and Gordon, 2010].

2. Sometimes the observations contain enough information about the state, yet the value function depends on the observations in a non-linear way. In this case, it is common to transform the observations into high dimensional features, in the hope that the value function can be represented as a linear function over features. A classical example is the kernel trick [Schölkopf and Smola, 2002], where the observations are mapped into elements in some infinite dimensional feature space, so that non-linear functions over observations can be represented as linear functions in the feature space. Another important example is to use a large number of randomly generated non-linear features, e.g., neural networks, which are proven to be capable of universal approximation [Huang et al., 2006].

3. As pointed out in Section 2.1.2, when dealing with non-Markovian problems, it is often desirable to create a large set of history dependent features, and regress the value function using LFA. For example, it has been shown in Legenstein and Maass [2007] that certain types of randomly generated recurrent neural networks are universal predictors, so that a wide class of history dependent functions, which may include the value function, can be linearly regressed to high accuracy using the activations of the network of sufficiently large scale [Szita et al., 2006].

In all these cases, we end up with a large number of features. It has been pointed out that direct regression on these features brings with it computational issues and affects the generalization in the finite sample case, which can be solved by LBP.

When restricted to reward-sensitive approaches, there are three classes of methods:

LARS-TD. LARS-TD [Kolter and Ng, 2009a] is a batch-based method which works by incorporating L_1 regularization into the objective function, namely,

$$J = (v - \Phi\theta)^\top K (v - \Phi\theta) + \kappa \|\theta\|_1 .$$

Here κ is a parameter controlling the strength of the regularization. It is well known in the regression literature that L_1 regularization tends to

produce 'sparse' solutions (e.g., see [Hastie et al., 2004],). As a result, v will typically be dependent only on a handful of basis functions, even if the set of basis functions is large. In other words, the algorithm implicitly constructs a W such that $W^\top \omega$ contains only a small number of entries in ω. LARS-TD is particularly suitable to the scenario where most of the features are irrelevant to the problem being solved.

Predictive-state TD. PS-TD [Boots and Gordon, 2010] is a family of batch-based methods. In PS-TD, the matrix W is so constructed that $W^\top \omega$ preserves the most information flowing between the 'past' and the 'future'. Here past and future may be interpreted in different ways. For example, the past and the future may refer respectively to one or several features observed before and after the present moment. One may also choose to include the reward signal into the past or future, and thus PS-TD can be reward-sensitive or reward-insensitive, depending on the design.

If we restrict the past and future to be the current and the next feature respectively, then the matrix W can be constructed by performing singular value decomposition (SVD):

$$\Phi^\top D P \Phi = U \Lambda V^\top,$$

where U, V are orthogonal matrices and Λ is a diagonal, positive semi-definite matrix with diagonal entries sorted in descending order. PS-TD constructs W as the first M columns of U. It can be shown that $W^\top \omega$ preserves the maximum cross variance between the past and future Creutzig et al. [2009].

Bellman error basis functions. BEBF [Wu and Givan, 2005; Keller et al., 2006; Parr et al., 2007; Mahadevan and Liu, 2010] is a family of online basis construction methods, where approximations of the current Bellman error are repeatedly added as the new basis functions, i.e.,

$$\phi_{bebf} \simeq r + \gamma P \tilde{v} - \tilde{v},$$

where ϕ_{bebf} is the next basis function to be added, while $\tilde{v}$ is the current value function estimation. The approximation can be done using either linear or nonlinear methods. When confined to the linear case, BEBF amounts to incrementally constructing W, one column at a time, such that the new column w^* to be added is obtained as

$$w^* = \operatorname*{argmin}_w \left\| \Phi w - \varepsilon_{W,\vartheta} \right\|,$$

where $\|\cdot\|$ is some norm measuring the scale of the difference between Φw and the current Bellman error $\varepsilon_{W,\vartheta}$ obtained from feature matrix ΦW and M dimensional weight ϑ, i.e.,

$$\varepsilon_{W,\vartheta} = r + \gamma P \Phi W \vartheta - \Phi W \vartheta.$$

BEBF serves as the starting point of the work presented in Chapter 4, where we propose a new family of online basis construction methods that have better theoretical properties than BEBF.

The reward-insensitive methods for feature dimensionality reduction can roughly be partitioned into two classes.

Proto-value functions. Proto-value functions [Mahadevan et al., 2007] are batch-based methods creating basis functions that reflect the clustering structure of the state space. The original proto-value basis functions are defined to be the eigenvectors corresponding to small eigenvalues of the Graph Laplacian

$$I - P,$$

or its symmetrized version. It is known from the spectral clustering literature (e.g., see [von Luxburg, 2006; Filippone et al., 2008]), that such eigenvectors correspond to clusters in the state space. This method can be extended to the feature case, in which we construct each column of W to be an eigenvector corresponding to near-zero eigenvalues of

$$\Phi^\top D \Phi - \Phi^\top D P \Phi.$$

Unsupervised learning methods. In principle, feature dimensionality reduction in RL can be done by applying any of the unsupervised dimensionality reduction methods developed in the i.i.d. setting to the set of features $\{\omega_t\}$. Indeed, if the Markov chain defined by P satisfies certain conditions (e.g., ergodic, which is in fact stronger than needed), then with increasing number of samples, the set $\{\omega_t\}$ can be treated as i.i.d. without much problem.

A number of methods belonging to this class have been investigated in the literature. For example, one may use *random projection*, i.e., W is a matrix with entries randomly generated [Ghavamzadeh et al., 2010]. The performance guarantee is obtained from the fact that the random projection approximately preserves the norm of the high dimensional features,

a result known as the Johnson-Lindenstrauss Lemma (e.g., see [Dasgupta and Gupta, 2003] for an elementary proof). Another example is to apply principle component analysis (PCA); this technique is so widely used that sometimes it is viewed as a default step.

A particularly interesting method we would like to mention is the following greedy, incremental procedure, where we maintain a dictionary $\mathcal{D}$ made of a subset of features seen previously. At each time step t, we check how well ω_t can be represented by the span of $\mathcal{D}$ by examining

$$\min_{\omega \in \operatorname{span} \mathcal{D}} \left\| \omega_t - \omega \right\|^2.$$

If this quantity is larger than some pre-specified constant α, then we add ω_t to $\mathcal{D}$. The matrix W can then be constructed as follows. Namely, assume $\mathcal{D} = \left\{ \omega^{(1)}, \ldots, \omega^{(M)} \right\}$ and let $B = \left[\omega^{(1)}, \ldots, \omega^{(M)} \right]$, then

$$W = B^\top \left(B^\top B \right)^{-},$$

so that the new features become

$$v = \left(B^\top B \right)^{-} B\omega,$$

which are in fact the coefficients when representing ω using a linear combination of $\omega^{(1)}, \ldots, \omega^{(M)}$. This technique works, even if the features are of infinite dimensionality, where we may use the kernel trick to compute $W^\top \omega$ analytically. In that case, the procedure is referred to as online kernel sparsification (OKS) [Engel et al., 2004]. We will study the theoretical properties of OKS in Chapter 3, focusing on the speed with which the dictionary grows.

Chapter 3

Understanding Basis Size Growth in Online Kernel Sparsification

In many RL problems, the value functions depend nonlinearly on the observations, and thus LFA is either ineffective due to the nonlinearity, or impossible when the observations are not presented in vector form (e.g., strings and graphs). *Kernelized function approximation* (kernel FA; [Ormoneit and Sen, 2002; Xu et al., 2005; Engel, 2005; Xu, 2006; Engel et al., 2003, 2005; Xu et al., 2007]) solves these problems by first transforming observations into high dimensional features using the kernel trick [Schölkopf and Smola, 2002] and then applying LFA in the feature space.

A significant drawback of kernel FA (or kernel methods in general) is that the usual computational complexity scales cubic in the number of observations [Taylor and Parr, 2009], rendering the method intractable for large data sets. In addition, the number of parameters to be estimated is the same as the number of observations, causing overfitting. To alleviate these problems, dimensionality reduction is often incorporated.

In this chapter we focus on *Online Kernel Sparsification* (OKS; [Engel et al., 2004]), a simple, online dimensionality reduction technique for kernel methods. Intuitively speaking, OKS incrementally builds a dictionary consisting of 'important' observations which are sufficient in approximately representing in the feature space all previous observations. Dimensionality reduction is then achieved by transforming observations into feature vectors whose dimensionality equals the size of the dictionary. Particular to RL, the resulting feature vectors can then be fed to LFA.

In spite of its wide usage in regression [Duy and Peters, 2010], classification [Slavakis et al., 2008] and reinforcement learning [Engel, 2005; Xu, 2006], the

theoretical understanding of OKS is still lacking. In particular, little is known about how the size of the dictionary, or equivalently the dimensionality of the output features, grows with the number of observations. Gaining insight into this problem is important, since such growth affects both the computational complexity and the generalization of the associated kernel methods.

The main contribution of the chapter is a novel approach for analyzing the growth rate of the OKS dictionary, assuming i.i.d. observations. Specifically, we discover a formula that expresses the expected determinant of the Gram matrix in terms of the eigenspectrum of the covariance operator. This allows us to connect the cardinality of the dictionary with the eigen decay of the covariance operator. In particular, we show that under certain technical conditions, the size of the OKS dictionary will always grow sub-linearly in the number of observations, and, as a consequence, the kernel regressor constructed from the resulting dictionary is statistically consistent, i.e., converging to the true regression function with an increasing number of observations.

It should be pointed out that although we perform the analysis under the i.i.d. assumption, which is the case for regression and classification but not RL, the results obtained are still pertinent to kernel FA, particularly when the set of observations is large and thus can be treated approximately i.i.d.

The rest of the chapter is organized as follows: Section 3.1 briefly reviews the background and then formulates the research question. Section 3.2 describes the first step of our analysis, establishing a number of theoretical properties concerning the determinant of the Gram matrix, including its expectation, decay, and moments. We then proceed to the second step, in Section 3.3, to analyze the growth of the dictionary size using the results from Section 3.2. Section 3.4 discusses the results and directions for future research.

3.1 Background

We review some basic facts about kernels and refer the readers to Schölkopf and Smola [2002] for more detail. Let $\mathcal{O}$ be the set of observations. A *positive definite kernel* (PD kernel, or simply *kernel*) $\ell : \mathcal{O} \times \mathcal{O} \mapsto \mathbb{R}$ is a symmetric, real-valued function over pairs of observations, such that for any integer $k > 0$ and[1]

[1]We use $o_{1:k}$ as a short hand for $o_1, \ldots, o_k$.

$o_{1:k} \in \mathcal{O}$, the k-by-k *Gram matrix*

$$G_k(o_1, \ldots, o_k) = \begin{bmatrix} \ell(o_1, o_1) & \cdots & \ell(o_1, o_k) \\ \vdots & \ddots & \vdots \\ \ell(o_k, o_1) & \cdots & \ell(o_k, o_k) \end{bmatrix},$$

is positive semi-definite, i.e., $c^\top G_k c \geq 0$ for any k-by-1 column vector c. PD kernels can be defined over vector spaces, where commonly used examples include radial basis functions (RBFs)

$$\ell(o, o') = \exp\left(-\frac{\|o - o'\|^2}{2\sigma^2}\right), \quad \sigma > 0$$

and polynomial kernels

$$\ell(o, o') = \left(1 + o^\top o'\right)^p, \quad p > 0,$$

as well as for discrete objects such as strings [e.g., Lodhi et al., 2002] and graphs [e.g., Kondor and Lafferty, 2002].

For any PD kernel ℓ, we may view the univariate function $\ell(o, \cdot)$ as an element in some uniquely defined (real) Hilbert space $\mathcal{H}$ called the *reproducing kernel Hilbert space* (RKHS). Roughly speaking, elements in $\mathcal{H}$ are functions of the form

$$f(\cdot) = \int \xi(o)\, \ell(o, \cdot)\, do,$$

which is essentially a weighted sum of $\ell(o, \cdot)$'s with the respective weights $\xi(o)$. The precise meaning of the integration depends on the observation space $\mathcal{O}$. The inner product $\langle \cdot, \cdot \rangle$ in $\mathcal{H}$ satisfies the *reproducing property*

$$\ell(o, o') = \left\langle \ell(o, \cdot), \ell(o', \cdot) \right\rangle, \quad \forall o, o' \in \mathcal{O},$$

and thus for any $f \in \mathcal{H}$, the evaluation at o can be viewed as an inner product, i.e.,

$$\begin{aligned} f(o) &= \int \xi(o')\, \ell(o, o')\, do' \\ &= \int \xi(o') \left\langle \ell(o, \cdot), \ell(o', \cdot) \right\rangle do' \\ &= \left\langle \int \xi(o')\, \ell(o', \cdot)\, do', \ell(o, \cdot) \right\rangle \\ &= \left\langle f, \ell(o, \cdot) \right\rangle. \end{aligned}$$

In RKHS, the set $\{\ell(o,\cdot) : o \in \mathcal{O}\}$ plays two roles. On one hand, elements therein are *basis functions* since every $f \in \mathcal{H}$ is their linear combination. On the other hand, each $\omega(o) = \ell(o,\cdot)$ can be interpreted as the (potentially infinite dimensional) *feature* associated with obervation o, such that every nonlinear function over the observations becomes a linear function over the features. This latter interpretation inspires the so-called *kernel trick*, where we transform algorithms that depend on finite dimensional feature vectors through their dot products into algorithms operating on the observation space $\mathcal{O}$ by replacing the dot product with the kernel $\ell(\cdot,\cdot)$. For example, applying the kernel trick on least square regression gives the (unregularized) kernel least square (KLS):

$$f(o) = y^\top G_k^- g_k(o),$$

where $g_k(o) = \left[\ell\left(o_1, o\right), \ldots, \ell\left(o_k, o\right)\right]^\top$, G_k is the Gram matrix over $o_{1:k}$, and $y = \left[y_1, \ldots, y_k\right]^\top$ with y_i being the response for observation o_i. Similarly, applying the kernel trick on LSTD gives the (unregularized) kernelized function approximation [Xu et al., 2005]

$$v(o) = r^\top G_k \left(G_k H_k G_k\right)^- g_k(o),$$

where $r = \left[r_1, \ldots, r_k\right]^\top$ is the vector containing the rewards collected at each time step, and

$$H_k = \begin{bmatrix} 1 & -\gamma & & \\ & 1 & -\gamma & \\ & & \ddots & -\gamma \\ & & & 1 \end{bmatrix}$$

encodes the temporal correlation between adjacent observations.

It is easy to see that in both KLS and kernel FA, the computational complexity scales at least quadratic in k, stemming from the need for inverting G_k. This renders the algorithms infeasible for large k. Furthermore, in both algorithms, the solutions take the form

$$\sum_{i=1}^{k} \theta_i \ell\left(o_i, o\right) = \left\langle \sum_{i=1}^{k} \theta_i \ell\left(o_i, \cdot\right), \omega(o) \right\rangle,$$

where $\theta_{1:k}$ are k weights over the basis functions $\ell\left(o_1, \cdot\right), \ldots, \ell\left(o_k, \cdot\right)$ minimizing the mean-square errors. This amounts to empirical risk minimization [see e.g., Bousquet et al., 2004] with the number of parameters to be estimated the same as the number of observations, causing overfitting. Particular to kernel

methods, this is accompanied by the numerical instability resulting from inverting large G_k, which is often ill-conditioned due to the fast decay of the eigenspectrum.

Usually, there are two families of solutions to the above mentioned problems [see e.g., Bousquet et al., 2004]. The first one is through *regularization*, e.g., to introduce a positive definite matrix B, typically $B = \delta I$ for some $\delta > 0$, and replace the inversion of G_k and $G_k H_k G_k$ with that of $G_k + B$ and $G_k H_k G_k + B$, respectively. This improves both the generalization and numerical stability, but does not affect the computational cost. The second solution is through *dimensionality reduction*. Specific to kernel methods, this amounts to replacing G_k with a low rank approximation

$$\tilde{G}_k = U_k D_k U_k^\top,$$

where U_k is a k-by-m matrix ($m \ll k$) and D_k is an m-by-m invertible matrix [see e.g., Schölkopf et al., 1999; Williams and Seeger, 2000; Engel et al., 2004; Shi and Guo, 2010]. The inverse of G_k is then replaced by the pseudo inverse of $\tilde{G}_k$, which can be computed in time $O\left(m^2 k\right)$. Moreover, this also reduces the number of *independent* parameters to m, alleviating the overfitting problem. To see this, taking KLS for example, we have

$$f(o) = y^\top G_k^- g_k(o) \simeq y^\top \left(U_k^\dagger\right)^\top D_k^- \left(U_k^\dagger\right) g_k(o),$$

where $U_k^\dagger$ is the m-by-k pseudo inverse of U_k. The set of possible weights can thus be written as

$$\theta = \left[\theta_1, \ldots, \theta_k\right]^\top = \left(U_k^\dagger\right)^\top D_k^- \left(U_k^\dagger\right) y, \, \forall y \in \mathbb{R}^k,$$

forming an m-dimensional space instead of k. Replacing G_k with $\tilde{G}_k$ may also be viewed as a form of linear basis projection (see Section 2.3.2), where we reduce the number of basis functions from k to m using matrix $\left(U_k^\dagger\right)^\top$, so that only m weights need to be estimated.

In this chapter, we are concerned with a particular class of dimensionality reduction methods, called the dictionary based methods, where we maintain a *dictionary* of obervations $\mathscr{D} = \left\{o^{(1)}, \ldots, o^{(m)}\right\}$ taken from $o_{1:k}$, and approximate G_k by the following formula

$$G_k \simeq G_{km} G_{\mathscr{D}}^- G_{km}^\top,$$

or its variations [e.g., Bach and Jordan, 2005; Shi and Guo, 2010]. Here $G_{\mathscr{D}}$ is the Gram matrix over $\mathscr{D}$, and G_{km} is the k-by-m matrix with the (i, j)-th entry $\ell\left(o_i, o^{(j)}\right)$.

Generally speaking, there are two approaches to constructing the dictionary. The first is the *Nyström method* [Williams and Seeger, 2000], where a randomly selected dictionary is used. The second, which is the concern of this chapter, is OKS, where the dictionary is built up incrementally by incorporating new features that cannot be represented well (in the least squares sense) using the current dictionary. More precisely, let $\omega_i = \omega\left(o_i\right)$, $\omega^{(i)} = \omega\left(o^{(i)}\right)$, and with a little abuse of notations rewrite $\mathscr{D} = \left\{\omega^{(1)}, \ldots, \omega^{(m)}\right\}$, then OKS adds $\omega_t = \omega\left(o_t\right) \in \mathscr{H}$ to $\mathscr{D}$ if

$$\min_{\omega \in \operatorname{span} \mathscr{D}} \left\|\omega_t - \omega\right\|^2 > \alpha. \tag{3.1}$$

Here $\alpha > 0$ is a hyper-parameter set by the user. It has been shown in Engel et al. [2004] that the left hand side of Equation 3.1 can be computed as

$$\min_{\omega \in \operatorname{span} \mathscr{D}} \left\|\omega_t - \omega\right\|^2 = \ell\left(o_t, o_t\right) - g_{\mathscr{D}}^{\top}\left(o_t\right) G_{\mathscr{D}}^{-} g_{\mathscr{D}}\left(o_t\right),$$

where $g_{\mathscr{D}}(o) = \left[\ell\left(o^{(1)}, o\right), \ldots, \ell\left(o^{(m)}, o\right)\right]^{\top}$. It can also be seen [e.g., from Akhiezer and Glazman, 1961, Section 1.7] that

$$\min_{\omega \in \operatorname{span} \mathscr{D}} \left\|\omega_t - \omega\right\|^2 = \frac{\det G_{\mathscr{D} \cup \{\omega_t\}}}{\det G_{\mathscr{D}}}. \tag{3.2}$$

Previous theoretical study by Engel et al. [2004] has revealed a number of attractive properties concerning OKS. In particular, it has been shown that the constructed dictionary is guaranteed to represent a major fraction of the leading eigenvectors of the Gram matrix G_k [Engel et al., 2004, Theorem 3.3]. In addition, it was proven that the dictionary stays finite if the set of possible features is compact, and thus admits a finite covering number [Engel et al., 2004, Theorem 3.1]. Yet, an important question remains open:

> *How does the size of the dictionary scale with the number of features if the set of possible features does* not *admit a finite covering number, or if the covering number is too large compared to the size of the data set?*

Answering this question allows us to: (1) estimate the computational complexity of OKS, and therefore the associated kernel methods, more accurately, and (2) characterize the generalization capability of the associated kernel methods, as the usual risk bounds are controlled by the quotient between the size of the dictionary and the number of observations [see e.g. Györfi et al., 2004].

In this chapter, we address this question under the assumption that the observations $o_{1:k}$ are given i.i.d. This is the case for regression and classification, but not for RL[2]. Our analysis proceeds in two steps:

1. We provide a novel formula expressing the *expected Gram determinant* $\mathbb{E}\left[\det G_k\right]$ over a set of i.i.d. observations in terms of the eigenvalues of the covariance operator. We then prove that the expected Gram determinant diminishes with the cardinality of the set faster than any exponential function.

2. We observe that the Gram determinant over the OKS dictionary is lower bounded by some exponential function in the size of the dictionary. However, since step 1 concludes that the chance of finding a big Gram matrix with large determinant is exceedingly small, the size of the dictionary must also stay small with high probability. Specifically, we show that the size of the dictionary will always grow sub-linearly in the number of data points, which implies consistency of KLS regressors constructed from the dictionary.

3.2 The Determinant of a Gram Matrix

We assume throughout the chapter that the RKHS $\mathscr{H}$ is *separable*, i.e., admitting a countable set of orthonormal basis. Let P be the distribution of the $\mathscr{H}$-valued random element $\omega(o)$ ($o \in \mathscr{O}$), and $\omega_1, \ldots, \omega_k$ be the features associated with i.i.d. observations $o_1, \ldots, o_k$, then $\omega_{1:k} \sim P$, i.i.d. Assume $\mathbb{E}_{\omega\sim P}\|\omega\|^2 < \infty$, and let $\mathscr{C} = \mathbb{E}_{\omega\sim P}[\omega \otimes \omega]$ be the (non-centered) *covariance operator*, where $\otimes$ denotes the tensor product. Let $\lambda_1 \geq \lambda_2 \geq \cdots$ be the eigenvalues of $\mathscr{C}$ sorted in descending order, then $\sum \lambda_i < \infty$ [Blanchard et al., 2007, Theorem 2.1]. The decay speed of the eigenvalues depends on both the choice of the kernel and the underlying distribution from which observations are sampled. For some important special cases, e.g., Gaussian kernels, the decay rate is known [see e.g., Bach and Jordan, 2002, Table 3, Appendix C.2, and the references therein].

With a little abuse of notations, we write $G_k\left(\omega_{1:k}\right)$ as the $k \times k$ Gram matrix with the (i,j)-th entry $\left\langle \omega_i, \omega_j \right\rangle = \ell\left(o_i, o_j\right)$, and let $\det G_k$ be the determinant of G_k. Clearly, $\det G_k$ is a random variable from $\mathscr{H}^k$ to $\mathbb{R}$. Moreover, $\det G_k$ has

[2]We speculate that if the underlying Markov reward process mixes fast enough, then the results obtained may also be extended to kernel FA. However, a rigorous analysis on the non-i.i.d. case is beyond the scope of this dissertation.

finite expectation since from Hadamard's inequality

$$0 \leq \mathbb{E}\left[\det G_k\right] \leq \mathbb{E}\left[\prod_{i=1}^{k} \left\|\omega_i\right\|^2\right] = \left(\mathbb{E}\left\|\omega\right\|^2\right)^k.$$

Let $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_k$ denote the eigenvalues of $k^{-1}G_k$ (and thus those of the *empirical covariance operator* $\tilde{\mathscr{C}}_k = k^{-1}\sum_{i=1}^{k} \omega_i \otimes \omega_i$) sorted in descending order. We assume the following condition.

Assumption 3.1 $\lim_{k\to\infty} \sum_{i=1}^{\infty} \left|\tilde{\lambda}_i - \lambda_i\right| = 0$, *a.s., where we take* $\tilde{\lambda}_i = 0$ *for* $i > k$.

The validity of this assumption will be discussed later in Section 3.4.1.

3.2.1 A Formula for the Expectation of Gram Determinant

Before presenting our first main result (Theorem 3.1), we introduce some additional notation. The *elementary symmetric polynomial*[3] of order k over n variables is defined as

$$\nu_{n,k}\left(\lambda_{1:n}\right) = k!\sum_{1\leq i_1<i_2<\cdots<i_k\leq n} \lambda_{i_1}\lambda_{i_2}\cdots\lambda_{i_k},$$

where the summation runs over all k-subsets of $\{1,\ldots,n\}$. We denote the infinite extension of $\nu_{n,k}$ as

$$\nu_k\left(\lambda_1,\lambda_2,\ldots\right) = k!\sum_{1\leq i_1<i_2<\cdots<i_k} \lambda_{i_1}\lambda_{i_2}\cdots\lambda_{i_k},$$

whenever the infinite sum exists. For simplicity, ν_k and $\nu_{n,k}$ denote both the function and their respective values with default argument $\left(\lambda_1,\lambda_2,\ldots\right)$, and we only write down the arguments when they differ from $\left(\lambda_1,\lambda_2,\ldots\right)$. Some of the useful properties of $\nu_{n,k}$ and ν_k are summarized in the following Lemma.

Lemma 3.1 *We have*

a) $\nu_{n,k} \geq \nu_{n-1,k} \geq 0$, *and* $\lim_{n\to\infty} \nu_{n,k} = \nu_k$.

b) $\nu_{n,k} = k\lambda_n\nu_{n-1,k-1} + \nu_{n-1,k}$,

c) $\nu_k^2 \geq \nu_{k-1}\nu_{k+1}$ *(Newton's inequality),*

d) $\nu_k^{\frac{1}{k}} \geq \nu_{k+1}^{\frac{1}{k+1}}$ *(Maclaurin's inequality).*

[3]Note that the standard definition does not have the $k!$ term.

Proof. We only prove the limit in a) exists. The other properties can be derived easily using the limit argument and the properties of elementary symmetric polynomials [e.g., see Niculescu, 2000]. In particular, c) is a direct consequence of Newton's inequality, and d) is a rephrase of Maclaurin's inequality.

Note that $\nu_{n,k}$ is a non-decreasing sequence of n. Moreover,

$$\nu_{n,k} = k! \sum_{1 \le i_1 < i_2 < \cdots < i_k \le n} \lambda_{i_1} \cdots \lambda_{i_k} < k! \left(\sum_{i=1}^{n} \lambda_i \right)^k$$

is bounded because $\sum \lambda_i < \infty$. Therefore the limit exists. ■

Note that property b) enables us to compute $\nu_{n,k}$ in $O(nk)$ time using dynamic programming. More precisely, this is done by initializing i) $\nu_{1,1} = \lambda_1$, ii) $\nu_{i,1} = \nu_{i-1,1} + \lambda_i$ for $i = 1, \ldots, n$, and iii) $\nu_{i,i} = i\lambda_i \nu_{i-1,i-1}$ for $i = 1, \ldots, k$, and then applying the recursion in b).

The following theorem gives an explicit representation of the expectation of $\det G_k$ in terms of the eigenvalues of $\mathscr{C}$.

Theorem 3.1 $\mathbb{E}\left[\det G_k\left(\omega_{1:k}\right)\right] = \nu_k$

That is, the expectation of the determinant of a Gram matrix built from k features is equal to the k-th order elementary symmetric polynomial over the eigenvalues of the covariance operator.

Proof. [4] Let $\omega_1, \ldots, \omega_n \sim P$, and $G_n = G_n\left(\omega_{1:n}\right)$ be the corresponding Gram matrix. Denote $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ the eigenvalues of $n^{-1}G_n$, so that $n\tilde{\lambda}_i$ are the eigenvalues of G_n. The characteristic polynomial of G_n is given by $f(\lambda) = \det\left(G_n - \lambda I\right)$. By definition,

$$\begin{aligned} f(-\lambda) &= \prod_{i=1}^{n} \left(\lambda + n\tilde{\lambda}_i\right) \\ &= \sum_{k=0}^{n} n^k \left(\sum_{1 \le i_1 < \cdots < i_k \le n} \tilde{\lambda}_{i_1} \cdots \tilde{\lambda}_{i_k} \right) \cdot \lambda^{n-k} \\ &= \sum_{k=0}^{n} n^k \frac{\nu_{n,k}\left(\tilde{\lambda}_{1:n}\right)}{k!} \cdot \lambda^{n-k}. \end{aligned}$$

[4] An alternative proof may be derived using the generator function of $\mathbb{E}\left[\det G_k\right]$ Martin [2007]. Unfortunately, the result is only briefly alluded to in the slides and no detailed documentation can be found till this moment.

Alternatively, we can express $f(-\lambda)$ using the determinants of the principal submatrices [see for example Meyer, 2001, pp.494], which are Gram matrices by themselves:

$$f(-\lambda) = \sum_{k=0}^{n} \sum_{\mathscr{I} \in [n]_k} \det G_k\left(\omega_{\mathscr{I}}\right) \cdot \lambda^{n-k},$$

where $[n]_k$ is the family of k-subsets in $\{1, \ldots, n\}$, and $\omega_{\mathscr{I}}$ denotes $\{\omega_i\}_{i \in \mathscr{I}}$. Divide the coefficients before λ^{n-k} by binomial coefficient $\binom{n}{k}$ to get the identity:

$$\binom{n}{k}^{-1} \sum_{\mathscr{I} \in [n]_k} \det G_k\left(\omega_{\mathscr{I}}\right) = \frac{(n-k)!n^k}{n!} \nu_{n,k}\left(\tilde{\lambda}_{1:n}\right).$$

The l.h.s. is a U-statistic [Serfling, 1980] with kernel $\det G_k$. Since $\mathbb{E}\left[\det G_k\right] < \infty$, the law of large numbers for U-statistics [Hoeffding, 1961] asserts that

$$\mathbb{E}\left[\det G_k\right] = \lim_{n \to \infty} \binom{n}{k}^{-1} \sum_{\mathscr{I} \in [n]_k} \det G_k\left(\omega_{\mathscr{I}}\right), \text{ a.s.}$$

Now consider the r.h.s. For the first term

$$\lim_{n \to \infty} \frac{(n-k)!n^k}{n!} = \lim_{n \to \infty} \frac{n}{n-1} \cdots \frac{n}{n-k+1} = 1.$$

For the second term, we have

$$\begin{aligned}
&\nu_{n,k}\left(\tilde{\lambda}_{1:n}\right) - \nu_k \\
&= \sum_{i=1}^{n} \left(\nu_{n,k}\left(\lambda_{1:i-1}, \tilde{\lambda}_{i:n}\right) - \nu_{n,k}\left(\lambda_{1:i}, \tilde{\lambda}_{i+1:n}\right)\right) \\
&+ \left(\nu_{n,k}\left(\lambda_1, \ldots, \lambda_n\right) - \nu_k\right).
\end{aligned}$$

Note that

$$\begin{aligned}
&\left|\nu_{n,k}\left(\lambda_{1:i-1}, \tilde{\lambda}_{i:n}\right) - \nu_{n,k}\left(\lambda_{1:i}, \tilde{\lambda}_{i+1:n}\right)\right| \\
&= \left|k\left(\tilde{\lambda}_i - \lambda_i\right) \nu_{n-1,k-1}\left(\lambda_{1:i-1}, \tilde{\lambda}_{i+1:n}\right)\right| \\
&\leq k\left|\tilde{\lambda}_i - \lambda_i\right| \nu_{n,k-1}\left(\lambda_{1:i-1}, \lambda_i, \tilde{\lambda}_{i+1:n}\right) \\
&\leq k\left|\tilde{\lambda}_i - \lambda_i\right| \nu^*_{n,k-1},
\end{aligned}$$

where

$$\nu^*_{n,k-1} = \nu_{n,k-1}\left(\max\left\{\tilde{\lambda}_1, \lambda_1\right\}, \ldots, \max\left\{\tilde{\lambda}_n, \lambda_n\right\}\right)$$

is bounded as $\sum \max\left\{\tilde{\lambda}_i, \lambda_i\right\} < \infty$. Therefore

$$\begin{aligned}
&\left|\nu_{n,k}\left(\tilde{\lambda}_{1:n}\right) - \nu_k\right| \\
&\leq k\nu^*_{n,k-1}\sum_{i=1}^{n}\left|\tilde{\lambda}_i - \lambda_i\right| + \left|\nu_{n,k}\left(\lambda_1, \ldots, \lambda_n\right) - \nu_k\right| \\
&\to 0, \text{ a.s.}
\end{aligned}$$

The first summand vanishes because of Assumption 3.1, and the second one diminishes because of Lemma 3.1 a). As a result,

$$\lim_{n\to\infty}\frac{(n-k)!n^k}{n!}\nu_{n,k}\left(\tilde{\lambda}_{1:n}\right) = \nu_k.$$

■

3.2.2 The Decaying Speed of $\mathbb{E}\left[\det G_k\right]$

It is not immediately obvious how $\nu_k = \mathbb{E}\left[\det G_k\right]$ behaves with increasing k. Here we provide a direct link between the speed with which ν_k approaches zero and the tail behavior of $\{\lambda_i\}$. The analysis is based on the following lemma.

Lemma 3.2 *Let* $\lambda^{(0)} = \sum \lambda_j$, *and* $\lambda^{(k)} = \sum_{j>k}\lambda_j$. *Then*

$$\log \nu_{k+s} - \log \nu_k \leq s \log \lambda^{(k)} + \log \binom{k+s}{k}.$$

Proof. Note that

$$\begin{aligned}
\nu_{k+s} &= \frac{(k+s)!}{k!s!}k!\sum_{1\leq i_1<\cdots<i_k}\lambda_{i_1}\cdots\lambda_{i_k}\cdot s!\sum_{i_k<j_1<\cdots j_s}\lambda_{j_1}\cdots\lambda_{j_s} \\
&= \binom{k+s}{k}k!\sum_{1\leq i_1<\cdots<i_k}\lambda_{i_1}\cdots\lambda_{i_k}\cdot\nu_s\left(\lambda_{i_k+1}, \lambda_{i_k+2}, \ldots\right)
\end{aligned}$$

Since λ_i is decreasing and $i_k \geq k$, we have for all i_k

$$\begin{aligned}
\nu_s\left(\lambda_{i_k+1}, \lambda_{i_k+2}, \ldots\right) &\leq \nu_s\left(\lambda_{k+1}, \lambda_{k+2}, \ldots\right) \\
&= \left(\nu_s\left(\lambda_{k+1}, \lambda_{k+2}, \ldots\right)^{\frac{1}{s}}\right)^s \\
&\leq \left(\nu_1\left(\lambda_{k+1}, \lambda_{k+2}, \ldots\right)\right)^s \\
&= \left(\sum_{j>k}\lambda_j\right)^s,
\end{aligned}$$

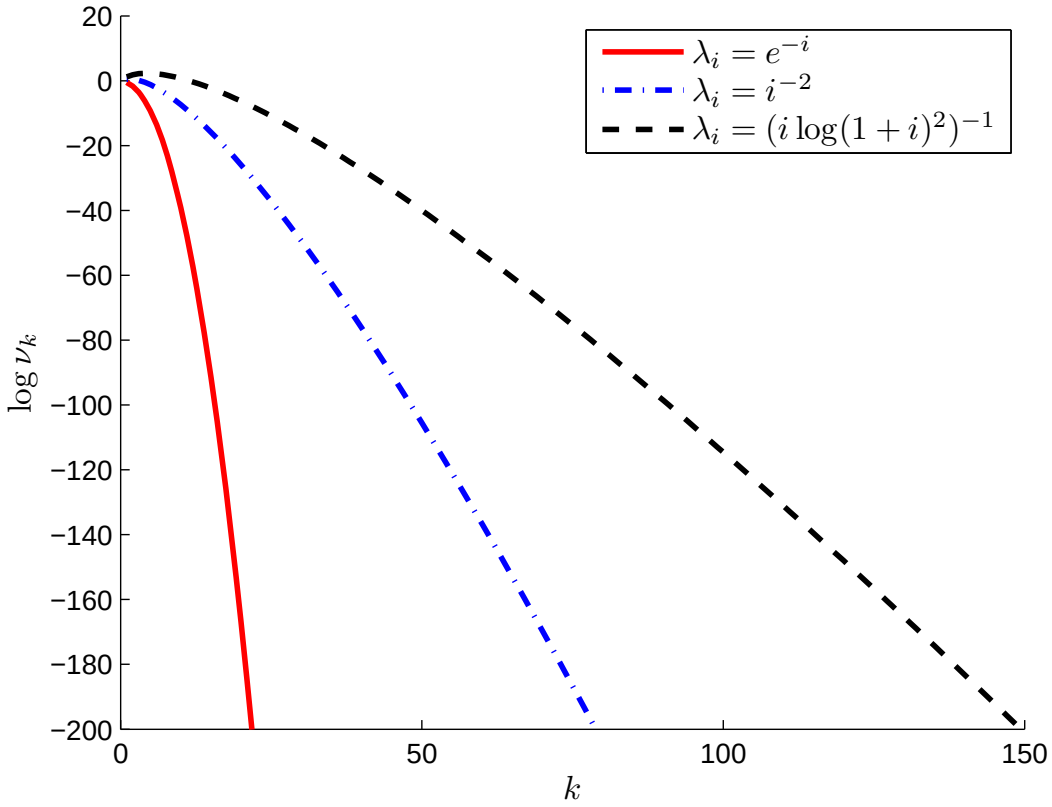

Figure 3.1. Illustration of the behavior of ν_k with exponential, polynomial and $n \log n$-type eigen decay.

where the last inequality is from Lemma 3.1 d). Therefore,

$$\nu_{k+s} \leq \binom{k+s}{k} \left(\lambda^{(k)}\right)^s \cdot \nu_k.$$

Taking the logarithm gives the desired result. ■

An immediate consequence is that ν_k converges to 0 faster than any exponential function.

Corollary 3.1 *For any $\alpha > 0$, $\lim_{k\to\infty} \alpha^{-k}\nu_k = 0$.*

Proof. Assume k is fixed and s is large. From Stirling's formula

$$\begin{aligned}\log\binom{k+s}{k} &= k\log\left(1+\frac{s}{k}\right) + s\log\left(1+\frac{k}{s}\right) + O\left(\log s\right)\\ &< s + O\left(\log s\right),\end{aligned}$$

where we use $\log(1+x) < x$ for all $x > -1$.

By Lemma 3.2,

$$\begin{aligned}&\log\nu_{k+s} - (k+s)\log\alpha\\ &\leq s\left[1 - \log\alpha + \log\lambda^{(k)}\right] + k\log(k+s) + \log\nu_k - k\log\alpha.\end{aligned}$$

Since $\sum\lambda_i < \infty$, we can pick a k^* such that $\log\left(\sum_{j>k^*}\lambda_j\right) < -2 + \log\alpha$, then

$$\begin{aligned}\lim_{k\to\infty}\log\nu_k - k\log\alpha &= \lim_{s\to\infty}\left(\log\nu_{k^*+s} - (k^*+s)\log\alpha\right)\\ &< \lim_{s\to\infty}\left(-s + k^*\log(k^*+s) + \log\nu_{k^*} - k^*\log\alpha\right)\\ &= -\infty,\end{aligned}$$

and thus $\lim_{k\to\infty}\alpha^{-k}\nu_k = 0$. ■

Remark 3.1 *We can also bound ν_k in terms of $\lambda^{(k)}$ using Lemma 3.2. (See Figure 3.1 for an illustration.) For exponential decay, i.e., $\lambda_i \sim O\left(\sigma^{-i}\right)$, we take $s = 1$, then*

$$\log\nu_k < -\frac{k^2}{2}\log\sigma + \log k! + O(k).$$

The bound is tight since for $\lambda_i = \sigma^{-i}$, direct computation gives

$$\nu_k = k!\sum_{i_1=1}^{\infty}\lambda_{i_1}\cdots\sum_{i_k=i_{k-1}+1}^{\infty}\lambda_{i_k} = \frac{k!\sigma^{-k}}{\prod_{i=1}^{k}\left(\sigma^i - 1\right)}.$$

Taking the logarithm and applying some algebra we get

$$\log \nu_k = -\frac{k^2}{2} \log \sigma + \log k! + O(k).$$

For polynomial decay, i.e., $\lambda_i \sim O\left(i^{-(1+p)}\right)$, $\sum_{i \geq k} i^{-(1+p)} \sim \frac{k^{-p}}{p}$, *we set* $s = k$, *then*

$$\log \nu_{2k} - \log \nu_k \leq k \log \lambda^{(k)} + \log \binom{2k}{k}.$$

Using Stirling's formula,

$$\log (2k)! - 2 \log k! = k \log 4 + O\left(\log k\right).$$

Therefore,

$$\log \frac{\nu_{2k}}{\nu_k} \leq -pk \log k + k \log \frac{4}{p} + O\left(\log k\right),$$

which characterizes the convergence of ν_k.

3.2.3 Bounding the Moments of the Gram Determinant

In this section we prove a simple result concerning the moment $\mathbb{E}\left[\left(\det G_k\right)^m\right]$, with the additional assumption that the kernel $\ell(\cdot, \cdot)$ is bounded, i.e.

$$\sup_{o, o' \in \mathcal{O}} \ell\left(o, o'\right) < \infty.$$

Note that for any $m \geq 1$, $\ell^{(m)}(o, o') = (\ell(o, o'))^m$ is still a bounded kernel. Let $\mathscr{H}^{(m)}$ be the RKHS associated with $\ell^{(m)}$ and denote $\lambda_1^{(m)} \geq \lambda_2^{(m)} \geq \cdots$ the eigenvalues of the corresponding covariance operator in $\mathscr{H}^{(m)}$. We have the following bound.

Theorem 3.2 $\mathbb{E}\left[\left(\det G_k\right)^m\right] \leq \nu_k\left(\lambda_1^{(m)}, \lambda_2^{(m)}, \ldots\right)$ *for* $m = 2, 3, \ldots$.

Proof. Let $A \circ B$ be the Hadamard product of A and B. We use the well-known fact: If A, B are positive semi-definite, then

$$\det(A \circ B) \geq \det(A) \det(B).$$

Repeating the process in the proof of Theorem 3.1, and applying

$$\det G_k^{(m)} \geq \left(\det G_k\right)^m$$

gives the result. ■

Remark 3.2 *Theorem 3.2 allows us to estimate empirically the bound of* $\mathbb{E}\left[\left(\det G_k\right)^m\right]$ *without enumerating all subsets of size* k. *Moreover, for RBF and polynomial kernels,* $\ell^{(m)}$ *remains RBF and polynomial, respectively. However, it remains unknown how* $\lambda_i^{(m)}$ *behaves in the general case.*

3.3 Analyzing Online Kernel Sparsification

In OKS, the dictionary $\mathscr{D}$ is initially empty. When a new feature ω arrives, it is added to the dictionary if (see Equation 3.2)

$$\frac{\det G_{\mathscr{D}\cup\{\omega\}}}{\det G_{\mathscr{D}}} > \alpha.$$

Our analysis is based on the key observation that

$$\det G_{\mathscr{D}} > \alpha^{|\mathscr{D}|}.$$

Since we have shown in the previous section that $\alpha^{-k}\mathbb{E}\left[\det G_k\right] \to 0$, the chance of finding a subset with the property that $\det G_{\mathscr{D}} > \alpha^{|\mathscr{D}|}$ will diminish as $|\mathscr{D}|$ grows, making a large dictionary unlikely.

More specifically, let $\omega_1, \ldots, \omega_n$ be n i.i.d. features from P, and let $\mathscr{D}_n$ be the dictionary constructed from $\omega_{1:n}$. Denote $[n]_k$ to be the family of all k-subsets of $\{1, \ldots, n\}$. For $\mathscr{A} \in [n]_k$, let

$$\rho_k\left(\omega_{\mathscr{A}}\right) = \mathbb{I}\left[\det G_k\left(\omega_{\mathscr{A}}\right) > \alpha^k\right],$$

where $\mathbb{I}[\cdot]$ is the indicator function. Define

$$k_n^* = \operatorname*{argmax}_k \left\{ \sum_{\mathscr{A}\in[n]_k} \rho_k\left(\omega_{\mathscr{A}}\right) > 0 \right\}.$$

Then clearly $\left|\mathscr{D}_n\right| \le k_n^*$, and we may study k_n^* instead of $|\mathscr{D}|$. Intuitively, k_n^* characterizes the dimensionality of the linear space spanned by $\omega_{1:n}$, because for any subset larger than k_n^* all ω can be represented within error α by the linear combination of $\omega_{\mathscr{A}}$ for some $|\mathscr{A}| \le k_n^*$.

To characterize k_n^* we study $\mathbb{P}\left[k_n^* \ge k\right]$. The following lemma shows that this probability is equal to the probability of the existence of k-subsets $\mathscr{A}$ with $\rho_k(\omega_{\mathscr{A}}) = 1$.

Lemma 3.3 $\mathbb{P}\left[k_n^* \ge k\right] = \mathbb{P}\left[\sum_{\mathscr{A}\in[n]_k} \rho_k\left(\omega_{\mathscr{A}}\right) > 0\right]$.

Proof. By definition

$$\begin{aligned}\mathbb{P}\left[\sum_{\mathscr{A}\in[n]_k} \rho_k\left(\omega_{\mathscr{A}}\right) > 0\right] &\le \mathbb{P}\left[\bigcup_{k'\ge k}\left\{\sum_{\mathscr{A}\in[n]_{k'}} \rho_{k'}\left(\omega_{\mathscr{A}}\right) > 0\right\}\right] \\ &= \mathbb{P}\left[k_n^* \ge k\right].\end{aligned}$$

Therefore the equality is not trivial.

From Theorem 5 in Cover and Thomas [1988],

$$\left(\frac{\det G_{k+1}\left(\omega_{1:k+1}\right)}{\alpha^{k+1}}\right)^{\frac{1}{k+1}} \leq \frac{1}{k+1}\sum_{\mathscr{A}\in[k+1]_k}\left(\frac{\det G_k\left(\omega_{\mathscr{A}}\right)}{\alpha^k}\right)^{\frac{1}{k}}.$$

Therefore, $\sum_{\mathscr{A}\in[n]_k}\rho_k\left(\omega_{\mathscr{A}}\right)=0$ implies $\sum_{\mathscr{A}\in[n]_{k'}}\rho_{k'}\left(\omega_{\mathscr{A}}\right)=0$ for all $k'\geq k$, and thus

$$\mathbb{P}\left[\sum_{\mathscr{A}\in[n]_k}\rho_k\left(\omega_{\mathscr{A}}\right)=0\right] \leq \mathbb{P}\left[\bigcap_{k'\geq k}\left\{\sum_{\mathscr{A}\in[n]_{k'}}\rho_{k'}\left(\omega_{\mathscr{A}}\right)=0\right\}\right].$$

Taking the complement on both sides,

$$\begin{aligned}\mathbb{P}\left[\sum_{\mathscr{A}\in[n]_k}\rho_k\left(\omega_{\mathscr{A}}\right)>0\right] &\geq \mathbb{P}\left[\bigcup_{k'\geq k}\left\{\sum_{\mathscr{A}\in[n]_{k'}}\rho_{k'}\left(\omega_{\mathscr{A}}\right)>0\right\}\right]\\ &= \mathbb{P}\left[k_n^*\geq k\right].\end{aligned}$$

■

We may now proceed to bound k_n^*, using basic tools from probability theory.

Theorem 3.3 $\mathbb{P}\left[\left|\mathscr{D}_n\right|\geq k\right]\leq\mathbb{P}\left[k_n^*\geq k\right]<\alpha^{-k}\binom{n}{k}\nu_k$.

Proof. Note that

$$\begin{aligned}\mathbb{E}\left[\sum_{\mathscr{A}\in[n]_k}\rho_k\left(\omega_{\mathscr{A}}\right)\right] &= \binom{n}{k}\mathbb{E}\left[\rho_k\right]\\ &= \binom{n}{k}\mathbb{P}\left[\det G_k>\alpha^k\right].\end{aligned}$$

From Markov's inequality,

$$\mathbb{P}\left[\det G_k>\alpha^k\right]<\frac{\mathbb{E}\left[\det G_k\right]}{\alpha^k}.$$

It then follows

$$\begin{aligned}\mathbb{P}\left[k_n^* \geq k\right] &= \mathbb{P}\left[\sum_{\mathscr{A}\in[n]_k} \rho_k\left(\omega_{\mathscr{A}}\right) \geq 1\right] \\ &\leq \mathbb{E}\left[\sum_{\mathscr{A}\in[n]_k} \rho_k\left(\omega_{\mathscr{A}}\right)\right] \\ &< \binom{n}{k}\frac{\mathbb{E}\left[\det G_k\right]}{\alpha^k}.\end{aligned}$$

Here we use the fact that ρ_k is $\{0,1\}$-valued, and apply Markov's inequality again. ■

Note that the proof only uses Markov's inequality, which usually provides bounds that are by no means tight. The possibility of strengthening the bound is discussed in the next section. However, even with this simple analysis, some interesting results for the size of $\mathscr{D}$ can be obtained. The first is the following corollary.

Corollary 3.2 *For any $\varepsilon \in (0,1]$,*

$$\lim_{n\to\infty} \mathbb{P}\left[\frac{k_n^*}{n} \geq \varepsilon\right] = 0.$$

Proof. For simplicity assume εn is an integer. Let $k = n\varepsilon$, then

$$\binom{n}{k}\frac{\nu_k}{\alpha^k} = \binom{\varepsilon^{-1}k}{k}\frac{\nu_k}{\alpha^k}.$$

Using Stirling's formula,

$$\begin{aligned}\log\binom{\varepsilon^{-1}k}{k} &= \log\left(\varepsilon^{-1}k\right)! - \log k! - \log\left(\left(\varepsilon^{-1}-1\right)k\right)! \\ &= \gamma k + O\left(\log k\right),\end{aligned}$$

where

$$\gamma = \frac{1}{\varepsilon}\log\frac{1}{\varepsilon} - \left(\frac{1}{\varepsilon}-1\right)\log\left(\frac{1}{\varepsilon}-1\right).$$

Therefore, following Corollary 3.1 and Theorem 3.3,

$$\begin{aligned}\lim_{n\to\infty} \mathbb{P}\left[\frac{k_n^*}{n} \geq \varepsilon\right] &< \lim_{n\to\infty,k=\varepsilon n} \binom{n}{k}\frac{\mathbb{E}\left[\det G_k\right]}{\alpha^k} \\ &= \lim_{k\to\infty} \binom{\varepsilon^{-1}k}{k}\frac{\nu_k}{\alpha^k} \\ &= 0.\end{aligned}$$

■

Remark 3.3 *By definition, Corollary 3.2 indicates that $n^{-1}k_n^* \to 0$ in probability, i.e., the size of the dictionary grows only sub-linearly with the number of observations. Assuming finite variance of the response variable, it immediately follows that the ordinary linear regressor constructed using features obtained from OKS is consistent, as the generalization error is controlled by $n^{-1}|\mathscr{D}|$ [e.g., see Györfi et al., 2004].*

The next corollary provides a bound given a finite number of observations.

Corollary 3.3 *For arbitrary $\delta > 0$ and*

$$n < \frac{\alpha k}{e}\left(\frac{\delta}{\nu_k}\right)^{\frac{1}{k}},$$

we have $\mathbb{P}\left[\left|\mathscr{D}_n\right| > k\right] < \delta$.

Remark 3.4 *It is possible to give a bound in k rather than n. However, such a bound requires the inversion of ν_k and complicates the notation.*

Proof. Assume $n = \varepsilon^{-1}k$. Rewrite Theorem 3.3 as

$$\mathbb{P}\left[\left|\mathscr{D}_{\varepsilon^{-1}k}\right| \geq k\right] < \alpha^{-k}\binom{\varepsilon^{-1}k}{k}\nu_k.$$

Using the simple relation $\binom{\varepsilon^{-1}k}{k} < \left(\varepsilon^{-1}e\right)^k$, we have

$$\log \alpha^{-k}\binom{\varepsilon^{-1}k}{k}\nu_k < k\left(1 - \log\alpha\right) - k\log\varepsilon + \log\nu_k.$$

Letting the r.h.s. equal $\log\delta$, it follows that

$$\log\frac{e}{\alpha} + \frac{1}{k}\log\frac{\nu_k}{\delta} = \log\varepsilon, \text{ and } \varepsilon = \frac{e}{\alpha}\left(\frac{\nu_k}{\delta}\right)^{\frac{1}{k}}.$$

■

Using Corollary 3.3, an upper bound on the dictionary size can be derived using $\{\lambda_i\}$, and the impact of α on the dictionary size can be analyzed.

From the previous discussion, if $\lambda_i \leq \sigma^{-i}$, then

$$\nu_k \leq \frac{k!\,(\sigma)^{-k}}{\prod_{i=1}^{k}\left(\sigma^i - 1\right)} = \frac{k!\,(\sigma)^{-k}}{\sigma^{\frac{k(k+1)}{2}}\prod_{i=1}^{k}\left(1 - \sigma^{-i}\right)},$$

or

$$\frac{1}{k}\log\nu_k \leq \frac{1}{k}\log k! - \frac{k}{2}\log\sigma - \frac{3}{2}\log\sigma - \frac{1}{k}\sum_{i=1}^{k}\log\left(1-\sigma^{-i}\right)$$
$$\leq -\frac{k}{2}\log\sigma + \log k - \frac{3}{2}\log\sigma + \frac{1}{k}\frac{\sigma}{(\sigma-1)^2},$$

where we use

$$-\sum_{i=1}^{k}\log\left(1-\sigma^{-i}\right) \leq \frac{\sigma}{(\sigma-1)^2}$$

Plugging this into Corollary 3.3, $n < \frac{\alpha}{\beta}\delta^{\frac{1}{k}}\sigma^{\frac{k}{2}}$, where β is some constant depending on σ, which implies $k \sim O\left(\log(n)\right)$. Similarly, for polynomial decay $n^{-(1+p)}$, we have for large k

$$\frac{1}{k}\log\frac{\nu_{2k}}{\nu_k} < -p\log k + \log\frac{4}{p},$$

and using Corollary 3.3, $n < \alpha\delta^{\frac{1}{k}}k^{1+p}$. Therefore, the dictionary size grows approximately at the rate of $n^{\frac{1}{1+p}}$. Note that the order of magnitude of these bounds coincides with the number of eigenvalues above a certain threshold [Bach and Jordan, 2002, Table 3].

3.4 Discussion

This chapter presented a rigorous theoretical analysis of how the dictionary in online kernel sparsification scales with respect to the number of observations, based on properties of the Gram matrix determinant. This work should lead to a better understanding of OKS, both in terms of its computational complexity, and the generalization capabilities associated with kernel regressors and kernel FA. Three additional points are discussed below concerning a) the validity of Assumption 3.1, b) how our results relate to the Nyström method, and c) how the analysis can be potentially developed further.

3.4.1 On Assumption 3.1

Under the mild condition $\mathbb{E}_{\omega\sim P}\|\omega\|^2 < \infty$, it can be seen that $\langle\cdot,\cdot\rangle$ is a Mercer kernel in the sense of Definition 2.15 in Braun [2005], and subsequently by Theorem 3.26 therein, it follows that the δ_2 distance (pp.116, Koltchinskii and Giné 2000) between $\left\{\tilde{\lambda}_i\right\}$ and $\{\lambda_i\}$ vanishes almost surely.

However, the convergence of the spectrum in δ_2 metric is insufficient for Theorem 3.1 to hold. To see this, consider the simple case where the kernel is defined as $\ell(x,x)=1$ and $\ell(x,x')=0$ for $x \neq x'$. Note that the RKHS induced by ℓ is non-separable, as there are uncountably many basis vectors if the support of x is uncountable. Nevertheless, it follows that all eigenvalues of $n^{-1}G_n$ are n^{-1} and all eigenvalues of $\mathscr{C}$ are zero. In this case, the δ_2 distance becomes $n^{-1} \to 0$, yet for fixed k, $\nu_{n,k} = \frac{n(n-1)\cdots(n-k)}{n^k} \to 1$, and thus the convergence to ν_k fails.

It is possible to drop Assumption 3.1 altogether and base the discussion on $\lim_{n\to\infty} \nu_{n,k}$ instead, where the limit always exists and equals to $\mathbb{E}\left[\det G_k\right]$. Otherwise, following the analysis by Gretton et al. [2009], we may provide sufficient conditions[5] to Assumption 3.1 using the following extension of the Hoffman–Wielandt inequality (Theorem 3, Bhatia and Elsner 1994)

$$\sum\nolimits_i \left|\tilde{\lambda}_i - \lambda_i\right| \leq \left\|\tilde{\mathscr{C}}_k - \mathscr{C}\right\|_{\mathrm{tr}},$$

where $\|\cdot\|_{\mathrm{tr}}$ denotes the trace norm. Using Proposition 12 in Harchaoui et al. [2008], the convergence of $\left\|\tilde{\mathscr{C}}_k - \mathscr{C}\right\|_{\mathrm{tr}}$ to zero can be established provided that i) $\mathscr{H}$ is a separable RKHS (e.g., an RKHS induced by a continuous kernel over a separable metric space; Steinwart et al. 2006) induced by some *bounded* kernel, and ii) the eigenspectrum of $\mathscr{C}$ satisfies $\sum_i \lambda_i^{\frac{1}{2}} < \infty$.

3.4.2 Comparison with Nyström Method

A similar approach to OKS for reducing the computational cost of kernel methods is the Nyström method [Williams and Seeger, 2000], where the dictionary consists of a subset of observations chosen at random. One distinction of the two methods, following from the analysis before, is that the dictionary from OKS satisfies $\det G_{\mathscr{D}} > \alpha^{|\mathscr{D}|}$, while the randomly selected subset $\tilde{\mathscr{D}}$, satisfies $\det G_{\tilde{\mathscr{D}}} \ll \alpha^{|\tilde{\mathscr{D}}|}$ for large $\mathscr{D}$. Therefore,

$$\frac{\det G_n}{\det G_{\mathscr{D}}} \gg \frac{\det G_n}{\det G_{\tilde{\mathscr{D}}}}.$$

Note that in Gaussian processes, $\log \frac{\det G_n}{\det G_{\mathscr{D}}}$ and $\log \frac{\det G_n}{\det G_{\tilde{\mathscr{D}}}}$ are respectively the conditional entropies of the data set given observations in $\mathscr{D}$ and $\tilde{\mathscr{D}}$ [see e.g. Rasmussen and Williams, 2006], which indicates that $\tilde{\mathscr{D}}$ captures much less information about the data sets.

[5] We thank Arthur Gretton for pointing this out in the ICML review.

The theoretical study of the Nyström method by Drineas and Mahoney [2005] suggests that $O\left(\alpha^{-4}k\right)$ observations are needed to approximate the first k eigenvectors well, which is linear in k, irrespective of the sample size. A recent study by Jin et al. [2012] shows that assuming bounded kernel, the spectral norm of the approximation error between the true and the approximated Gram matrix scales at $O\left(n\left|\mathscr{D}\right|^{-\frac{1}{2}}\right)$, and in the case of $\lambda_i \sim i^{-p}$, an $O\left(n\left|\mathscr{D}\right|^{1-p}\right)$ rate may be obtained. In comparison, the results obtained here are over the dictionary size $|\mathscr{D}|$, and the approximation error is controlled by α. In particular, assuming bounded kernel, the (i,j)-th entry of the difference between the true and approximated Gram matrix using OKS is bounded by

$$\left|\left\langle \omega_i, \omega_j \right\rangle - \left\langle \Pi_{\mathscr{D}}\omega_i, \Pi_{\mathscr{D}}\omega_j \right\rangle\right| < 2\sup\|\omega\|\sqrt{\alpha},$$

where $\Pi_{\mathscr{D}}$ denotes the projection operator into the space spanned by $\mathscr{D}$ and the inequality follows from Cauchy-Schwartz inequality. Using the fact that $\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty}$ for arbitrary matrix A, where $\|\cdot\|_2$, $\|\cdot\|_1$ and $\|\cdot\|_\infty$ respectively denote the spectral norm, maximum absolute column sum norm and maximum absolute row sum norm, we conclude that the spectral norm of the approximation error is controlled by $O\left(n\sqrt{\alpha}\right)$, which is a non-probabilistic bound and does not explicitly depend on the dictionary size.

3.4.3 On Strengthening the Bound

The proof of Theorem 3.2 uses Markov's inequality to bound both $\mathbb{P}\left[\det G_k > \alpha^k\right]$, and the probability of $\sum_{\mathscr{A}\in[n]_k} \rho_k(\mathscr{A}) \neq 0$. In practice, this bound is hardly satisfying. One possibility is to strengthen the bound by incorporating information from higher order moments [Philips and Nelson, 1995], i.e.,

$$\begin{aligned}\mathbb{P}\left[\det G_k > \alpha^k\right] &\leq \inf_{m\in\{1,2,\ldots\}} \frac{\mathbb{E}\left[\det G_k^m\right]}{\alpha^{km}} \\ &\leq \inf_{m\in\{1,2,\ldots\}} \frac{\nu\left(\lambda_1^{(m)}, \lambda_2^{(m)}, \ldots\right)}{\alpha^{km}}.\end{aligned}$$

However, analyzing $\lambda_i^{(m)}$ is difficult in general, and remains an open research question.

It is also possible to improve the second step, using concentration inequalities for configuration functions [Boucheron et al., 1999]. Let $\upsilon_1, \ldots, \upsilon_k$ be a subsequence of $\omega_{1:n}$. We say $\upsilon_{1:k}$ is *α-compatible*, if for $j = 1, \ldots, k$,

$$\frac{\det G_{\{\upsilon_1,\ldots,\upsilon_j\}}}{\det G_{\{\upsilon_1,\ldots,\upsilon_{j-1}\}}} > \alpha.$$

Note that the dictionary constructed by OKS is α-compatible, and the property of α-compatibility is *hereditary*, i.e., $v_{1:k}$ being α-compatible implies that all sub-sequences are also α-compatible. To see this, let $v_{i_1}, \ldots, v_{i_s}$ be a sub-sequence of $v_{1:k}$, then

$$\begin{aligned}
\frac{\det G_{\{v_{i_1},\ldots,v_{i_s}\}}}{\det G_{\{v_{i_1},\ldots,v_{i_s-1}\}}} &= \min_{v \in \operatorname{span}\{v_{i_1},\ldots,v_{i_{s-1}}\}} \left\| v_{i_s} - v \right\|^2 \\
&\geq \min_{v \in \operatorname{span}\{v_1,\ldots,v_{i_s-1}\}} \left\| v_{i_s} - v \right\|^2 \\
&= \frac{\det G_{\{v_1,\ldots,v_{i_s}\}}}{\det G_{\{v_1,\ldots,v_{i_s-1}\}}} \\
&> \alpha.
\end{aligned}$$

As a result, let Z_n denote the length of the longest sub-sequence in $\omega_{1:n}$ that is α-compatible, then $\left|\mathscr{D}_n\right| < Z_n$. By Theorem 2 in Boucheron et al. [1999], Z_n concentrates sharply around $\mathbb{E}\left[Z_n\right]$. Therefore, it is unlikely that $\left|\mathscr{D}_n\right|$ exceeds $\mathbb{E}\left[Z_n\right]$ by much. However, providing tight bounds for $\mathbb{E}\left[Z_n\right]$ is difficult and requires further study.

Chapter 4

Incremental Basis Construction from Temporal Difference Error

In this chapter we introduce a method for reward-sensitive, incremental basis construction, called *V-BEBF*. V-BEBF relies on a novel principle, where the *approximations* to *the value function of the Bellman error*, rather than to the Bellman error itself as in BEBF (see Section 2.3.2), are added as new basis functions (hence the name V-BEBF). This approach is justified by a simple yet previously unspotted insight, i.e., V-BEBF, *if computed exactly*, is in fact the error in value estimation, and therefore its addition to the existing set of basis functions immediately allows the value function to be represented without error.

This result transforms reward-sensitive basis contruction into a second value-function estimation problem, and suggests a natural framework for estimating value functions: a *primary* reinforcement learner estimates the value function using its present basis functions; it then sends its TD error to a *secondary* reinforcement learner, which interprets that error as a reward and estimates the corresponding value function, i.e., V-BEBF; the resulting V-BEBF estimation is then added to the set of basis functions used by the primary learner.

The main contribution of this chapter is twofold, namely (i) formulating V-BEBF, and (ii) demonstrating that V-BEBF is a promising alternative to BEBF (see Section 2.3.2), especially when the discount factor γ approaches 1, in which case we prove that BEBF can be very inefficient. Limited experiments are also conducted to compare the two methods, and the results are in line with the theoretical finding.

The rest of the chapter is organized as follows. Section 4.1 briefly recapitulates the background material. The formulation of V-BEBF is then detailed in Section 4.2, along with a theoretical comparison with BEBF. Section 4.3 de-

scribes two algorithms, where the idea of V-BEBF (as well as BEBF) is used for linear dimensionality reduction. A limited empirical study is presented in Section 4.4, followed by a discussion in Section 4.5.

4.1 Background

This section reviews some of the notation established in Section 2.3. A Markov Reward Process (MRP) is a 4-tuple $\langle \mathscr{S}, P, r, \gamma \rangle$, where $\mathscr{S} = \{1, \ldots, S\}$ is a state space, P is an S-by-S transition matrix, r is an S-by-1 (expected) reward function, and $\gamma \in [0, 1)$ is a discount factor. Let v be the value function of the MRP, then v satisfies the Bellman equation

$$v = r + \gamma P v,$$

which can be written as

$$v = L^{-} r, \tag{4.1}$$

with $L = I - \gamma P$.

In LFA, the value function is approximated via a linear combination of basis functions $\Phi = \left[\phi_1, \cdots, \phi_N\right]$ such that

$$v \simeq \sum\nolimits_{n=1}^{N} \theta_n \phi_n = \Phi\theta,$$

where $N \ll S$ and $\theta = \left[\theta_1, \ldots, \theta_N\right]^{\top}$ are the corresponding weights. The Bellman error of such approximation is thus given by

$$\varepsilon = r + \gamma P \Phi\theta - \Phi\theta = r - L\Phi\theta = L\left(v - \Phi\theta\right).$$

Clearly, $\varepsilon = 0$ if and only if $v = \Phi\theta$.

When the basis functions are given, the weights can be obtained using methods described in Section 2.3.1. In particular, we may compute the weights by minimizing quadratic objective functions of the form

$$J = \left(L^{-}\varepsilon\right)^{\top} K \left(L^{-}\varepsilon\right) = \left(v - \Phi\theta\right)^{\top} K \left(v - \Phi\theta\right), \tag{4.2}$$

either explicitly or though (stochastic or batch) gradient descent. The optimal weights are thus given by

$$\hat{\theta} = \left(\Phi^{\top} K \Phi\right)^{-} \Phi^{\top} K v,$$

with the corresponding value estimation $\hat{v} = \Pi_K v$, Bellman error

$$\hat{\varepsilon} = r - L\Phi\hat{\theta} = L\left(v - \Pi_K v\right),$$

and the minimum of the objective function

$$\hat{J} = \min_{\theta} J = v^\top K v - \left(\Pi_K v\right)^\top K \left(\Pi_K v\right). \tag{4.3}$$

Here $\Pi_A = \Phi \left(\Phi^\top A \Phi\right)^- \Phi^\top A$ is the projection operator with respect to the (semi) inner product $\langle v_1, v_2 \rangle_A = v_1^\top A v_2$ defined by arbitrary positive (semi) definite matrix A. Specific to LSTD [Bradtke et al., 1996] and TDC [Sutton et al., 2009], we set $K_{lstd} = L^\top \Pi_D D \Pi_D L$, where D is the diagonal matrix whose diagonal is the sample distribution of the states, and the optimal weights become

$$\hat{\theta}_{td} = \left(\Phi^\top D L \Phi\right)^- \Phi^\top D r,$$

which can be estimated from sample trajectories.

When it comes to basis construction, particularly the online, reward sensitive case, a significant fraction of the existing approaches make use of the so-called Bellman error basis functions (BEBFs; [Wu and Givan, 2005; Keller et al., 2006; Parr et al., 2007; Mahadevan and Liu, 2010]), in which the next basis function ϕ_{N+1} is constructed so that

$$\phi_{N+1} \simeq \hat{\varepsilon}_N,$$

where $\hat{\varepsilon}_N$ is the Bellman error corresponding to the optimal value estimation using basis functions $\phi_1, \ldots, \phi_N$. These methods capture the intuition that the "Bellman error, loosely speaking, point[s] towards the optimal value function" [Parr et al., 2007]. Because a sequence of normalized BEBFs form an orthonomal basis of the space in which the value function resides, any value function can be exactly represented given a sufficient number of BEBFs [Parr et al., 2007; Mahadevan and Liu, 2010].

4.2 V-BEBF

In this section, we pursue an entirely different idea for basis-function generation. Instead of constructing a *sequence* of basis functions that, in sufficient number, can eventually represent the value function, we aim to do so with a single new basis function, namely the V-BEBF. The rest of this section is organized as follows: Section 4.2.1 details the formulation of V-BEBF. Two related topics are then discussed, namely (i) the comparison with BEBF (Section 4.2.2), where we provide theoretical results demonstrating the potential ineffectiveness of BEBF, and (ii) issues related to approximating V-BEBF (Section 4.2.3), where a simple analysis on the approximation error is presented.

4.2.1 The Ideal Basis Function from TD-error

With a given set of basis functions, once the optimal weights have been found, no more improvement can be made to the value-function approximation. To further improve performance, additional basis functions can be added. If $\Phi_+ = [\Phi \vdots \phi]$, and $\theta_+ = [\theta^\top \vdots 1]^\top$, where $[A \vdots B]$ is the matrix with A and B juxtaposed, then the Bellman error becomes

$$\begin{aligned}\varepsilon_+ &= r - L\Phi_+\theta_+ = r - L\Phi\theta - L\phi \\ &= L\left(v - \Phi\theta - \phi\right) \\ &= L\left(L^-\varepsilon - \phi\right).\end{aligned}$$

Ideally, we should choose $\phi = L^-\varepsilon = v - \Phi\theta$, so that ε_+ is reduced to 0, and the value function is exactly represented by $v = \Phi_+\theta_+$.

The key insight is that $\phi = L^-\varepsilon$ is the solution of the Bellman equation $\phi = \varepsilon + \gamma P\phi$, namely, *$\phi$ is the value function of the original Markov chain when its reward function is equal to the Bellman error, ε*. Since ε is also the expectation of the TD-error, ϕ can be computed by solving a second reinforcement learning problem, in which the TD-error of the first problem is used as the reward in the second.

The choice of ϕ above depends on both Φ and the current weights θ, which change during learning. However, in the limit θ converges to $\hat{\theta}$, so we can remove this dependency and define the *Value Function of the Bellman Error*, V-BEBF, as $\hat{\phi} = L^-\hat{\varepsilon}$, or equivalently through the Bellman equation

$$\hat{\phi} = \hat{\varepsilon} + \gamma P\hat{\phi}. \tag{4.4}$$

We argue that $\hat{\phi}$ is the ideal basis function for two reasons. First, adding $\hat{\phi}$ allows the value function to be represented exactly, since by Equation 4.1

$$\begin{aligned}v &= L^-r = L^-\left(r - L\Phi\hat{\theta} + L\Phi\hat{\theta}\right) \\ &= L^-\left(\hat{\varepsilon} + L\Phi\hat{\theta}\right) \\ &= \hat{\phi} + \Phi\hat{\theta}.\end{aligned}$$

Second, as can also be seen from this last equation, adding $\hat{\phi}$ does not change any of the optimal weights for the existing basis functions. Therefore, when $\hat{\phi}$ is added, there is no need to re-adjust the weights learned previously. It is straightforward to verify that $\hat{\phi}$ (up to a non-zero multiplicative constant) is the only basis function possessing these two properties. Indeed,

$$0 = -\frac{1}{2}\nabla_\theta J\bigg|_{\theta=\hat{\theta}} = \Phi^\top K L^-\hat{\varepsilon} = \Phi^\top K\hat{\phi}, \tag{4.5}$$

indicating that $\hat{\phi}$ is orthogonal to the first N basis functions with respect to the (semi) inner product $\langle \cdot, \cdot \rangle_K$.

It must be pointed out that V-BEBF is no easier to compute than the value function itself (and so does BEBF). Yet, the usefulness of this idea comes from the following intuition, shared by previous work on BEBF [Wu and Givan, 2005, see e.g.,], that crude approximations to V-BEBFs (or BEBFs) may still yield good basis functions. This will be discussed further in Section 4.5.

4.2.2 Comparison with BEBFs

The idea of generating basis functions from Bellman error has been explored several times in the literature. In particular, Parr et al. [2007] carried out a theoretical study, where they pointed out that a sequence of normalized BEBFs, $\hat{\varepsilon}$ in our terminology[1], form an orthonormal basis in $\mathbb{R}^S$ with respect to $\langle \cdot, \cdot \rangle_D$, assuming that K_{lstd} is used in the objective function. As a consequence, repeatedly adding BEBFs eventually allows any value function to be represented exactly. A recent variation, namely the Bellman Average Reward Bases (BARBs), replaces the very first BEBF, r, with $P^\infty r$, where $P^\infty = \lim_{n\to\infty} P^n$. BARBs are equivalent to BEBFs if P is ergodic, yet demonstrate faster convergence otherwise, particularly when $\gamma \to 1$ [Mahadevan and Liu, 2010].

The formulation of V-BEBF bears a clear resemblance to BEBF in that it is also based on the Bellman error. However, there is a key difference. In the worst case, representing a value function requires *a whole sequence* of BEBFs, even if all BEBFs are computed exactly. In contrast, a single additional V-BEBF, if computed exactly, is sufficient to represent the value function. In fact, when the initial basis function set is empty, the first V-BEBF is simply $L^- r$, the value function itself.

The difference between V-BEBF and BEBF is captured by the following proposition, and illustrated by the example in Figure 4.1.

Proposition 4.1 *Consider the objective function J as defined in Equation 4.2, with K an arbitrary fixed positive definite matrix. Let $\hat{J}$ and $\hat{J}_+$ be the minimum of J as defined in Equation 4.3 corresponding to the basis functions Φ and $[\Phi \vdots \hat{\varepsilon}]$. Then*

$$\rho = \frac{\hat{J}_+}{\hat{J}} \leq \gamma^2,$$

with the equality holding iff r is chosen such that

[1] The sequence of BEBFs is constructed iteratively, such that $\phi_1 = r$, and $\phi_{k+1} = \hat{\varepsilon}^{(k)} = r - L\Phi^{(k)}\hat{\theta}^{(k)}$, where $\Phi^{(k)} = [\phi_1, \cdots, \phi_k]$ and $\hat{\theta}^{(k)}$ are the optimal weights.

1. $\Pi_K P v = \Pi_K P \Phi \hat{\theta}$;

2. $\hat{\phi}^\top K \hat{\phi} = \hat{\phi}^\top P^\top K P \hat{\phi}$;

3. $\hat{\phi}^\top K P \hat{\phi} = \gamma \hat{\phi}^\top K \hat{\phi}$.

Remark 4.1 *The example in Figure 4.1 shows that when the conditions in Proposition 4.1 are satisfied, the upper bound for ρ is indeed achieved.*

Proof. Let ϕ be the new basis function, then the minimum of the objective functions $\hat{J}$ and $\hat{J}_+$ without and with ϕ added are given respectively by Equation 4.3 and

$$\hat{J}_+ = v^\top K v - v^\top K \left[\Phi, \phi\right] \left(\left[\Phi, \phi\right]^\top K \left[\Phi, \phi\right] \right)^- \left[\Phi, \phi\right]^\top K v.$$

Change variables for simplicity: write $K = A^\top A$, $u = Av$, $\Psi = A\Phi$, $\psi = A\phi$, then $\hat{J} = u^\top (I - \Pi) u = u^\top V u$, where $\Pi = \Psi \left(\Psi^\top \Psi\right)^- \Psi^\top$, and $V = I - \Pi$. Note that $\Pi = \Pi^2$, $V = V^2$. From block matrix inversion [see e.g., Petersen and Pedersen, 2008, pp.45],

$$\begin{bmatrix} \Psi^\top \Psi & \Psi^\top \psi \\ \psi^\top \Psi & \psi^\top \psi \end{bmatrix}^- = \begin{bmatrix} \left(\Psi^\top \Psi - \frac{\Psi^\top \psi \psi^\top \Psi}{\psi^\top \psi}\right)^- & -\frac{\left(\Psi^\top \Psi\right)^- \Psi^\top \psi}{\psi^\top V \psi} \\ -\frac{\psi^\top \Psi \left(\Psi^\top \Psi\right)^-}{\psi^\top V \psi} & \frac{1}{\psi^\top V \psi} \end{bmatrix}$$

and by Kailath variant [see e.g., Petersen and Pedersen, 2008, pp.17],

$$\left(\Psi^\top \Psi - \frac{\Psi^\top \psi \psi^\top \Psi}{\psi^\top \psi}\right)^- = \left(\Psi^\top \Psi\right)^- + \frac{\left(\Psi^\top \Psi\right)^- \Psi^\top \psi \psi^\top \Psi \left(\Psi^\top \Psi\right)^-}{\psi^\top V \psi},$$

therefore

$$\begin{bmatrix} \Psi^\top \Psi & \Psi^\top \psi \\ \psi^\top \Psi & \psi^\top \psi \end{bmatrix}^- = \begin{bmatrix} \left(\Psi^\top \Psi\right)^- & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{\psi^\top V \psi} \zeta \zeta^\top$$

where $\zeta = \left[-\psi^\top \Psi \left(\Psi^\top \Psi\right)^-, 1\right]^\top$. It follows that

$$\hat{J}_+ = u^\top V u - \frac{u^\top V \psi \cdot \psi^\top V u}{\psi^\top V \psi}.$$

Note that here the additional basis function is the BEBF, so $\phi = \hat{\varepsilon} = LA^{-}Vu$, therefore

$$\begin{aligned}\rho = \frac{\hat{J}_{+}}{\hat{J}} &= 1 - \frac{\left(u^{\top}V\psi\right)\left(\psi^{\top}Vu\right)}{\left(\psi^{\top}V\psi\right)\left(u^{\top}Vu\right)} \\ &= 1 - \frac{\left(u^{\top}VALA^{-}Vu\right)^{2}}{\left(u^{\top}VA^{-\top}L^{\top}A^{\top}VALA^{-}Vu\right)\left(u^{\top}Vu\right)} \\ &= 1 - \frac{\left(z^{\top}KLz\right)^{2}}{\left(z^{\top}L^{\top}A^{\top}VALz\right)\left(z^{\top}Kz\right)},\end{aligned}$$

with $z = A^{-}Vu = A^{-}VAL^{-}r$. Also,

$$A^{\top}VA = A^{\top}\left(I - \Psi\left(\Psi^{\top}\Psi\right)^{-}\Psi^{\top}\right)A \prec A^{\top}A = K,$$

thus

$$\rho \leq 1 - \frac{\left(z^{\top}KLz\right)^{2}}{\left(z^{\top}L^{\top}KLz\right)\left(z^{\top}Kz\right)}.$$

And the equality holds if and only if

$$(I - V)ALz = (I - V)ALA^{-} \cdot VAL^{-}r = 0.$$

Simplify this equation gives the first condition.

Now assume that the equality holds, and expand $L = I - \gamma P$, then

$$\begin{aligned}\rho &= 1 - \frac{\left(z^{\top}Kz - \gamma z^{\top}KPz\right)^{2}}{z^{\top}Kz \cdot z^{\top}\left(I - \gamma P\right)^{\top}K\left(I - \gamma P\right)z} \\ &= \gamma^{2} \cdot \frac{\left(z^{\top}Kz\right)\left(z^{\top}P^{\top}KPz\right) - \left(z^{\top}KPz\right)^{2}}{z^{\top}Kz \cdot z^{\top}\left(I - \gamma P\right)^{\top}K\left(I - \gamma P\right)z}.\end{aligned}$$

Write $x = Az$, $y = APz = APA^{-}x$, then

$$\rho = 1 - \frac{\left\langle x, x - \gamma y\right\rangle^{2}}{\|x\|^{2}\left\|x - \gamma y\right\|^{2}} = \gamma^{2}\frac{\langle x, x\rangle\left\langle y, y\right\rangle - \left\langle x, y\right\rangle^{2}}{\|x\|^{2}\left\|x - \gamma y\right\|^{2}},$$

where $\left\langle x, y\right\rangle = x^{\top}y$ and $\|x\|^{2} = \langle x, x\rangle$. We show that

$$\|x\|^{2}\left\|x - \gamma y\right\|^{2} \geq \|x\|^{2}\left\|y\right\|^{2} - \left\langle x, y\right\rangle^{2}.$$

Indeed,

$$\begin{aligned}&\|x\|^2 \left\|x-\gamma y\right\|^2 - \|x\|^2 \left\|y\right\|^2 + \langle x,y\rangle^2 \\ &= \left(\gamma\|x\|^2 - \langle x,y\rangle\right)^2 + \left(1-\gamma^2\right)\|x\|^2 \left(\|x\|^2 - \left\|y\right\|^2\right),\end{aligned}$$

and notice that $y = APA^- x$ and P is a transition matrix, so all the eigenvalues of APA^- are smaller or equal than 1, therefore $\left\|y\right\|^2 \leq \|x\|^2$. Also, the equality holds iff: a) $\|x\|^2 = \left\|y\right\|^2$ and b) $\langle x,y\rangle = \gamma\|x\|^2$, which correspond to the second and the third condition. ■

In particular, when K is chosen as either $L^\top DL$ or D, ρ corresponds to the Mean Square Bellman error and the Mean Square Value Error, respectively, and the fraction of improvement from adding a BEBF is only $1-\gamma^2$ in the worst case. However, if the V-BEBF is added as the new basis function, the new minimum of J is always 0 since the value function is represented exactly.

An analysis in a similar spirit was proposed by Mahadevan and Liu [2010], who showed that the error in approximating the value function using the first m BEBFs is bounded in terms of the Chebyshev polynomial of degree m and the condition number of L. Our result, albeit less general, is simpler to interpret and allows construction of the worst-case scenario (see Figure 4.1 for an example).

4.2.3 Approximating V-BEBF

In theory, the V-BEBF can be computed as the value function of the Markov chain using TD-error as the reward, provided that the weights for the current basis functions are set optimally. In practice, the V-BEBF has to be approximated and three sources of errors are anticipated: (a) the exact representation of V-BEBF (as well as BEBF) requires storage of size S which is not available in the first place, and therefore function approximation must be used; (b) the convergence of θ to $\hat{\theta}$ happens only asymptotically, and thus only the TD-errors, ε, corresponding to the current θ can be used as the reward; and (c) error arises when estimating V-BEBF from a finite set of samples. Formally, let $\mathscr{B}$ be the set of functions from which the representations of V-BEBF approximators are chosen, and let

$$\hat{\phi}_\theta = \arg\min_{\phi\in\mathscr{B}} \left\|\phi - L^- \varepsilon\right\|$$

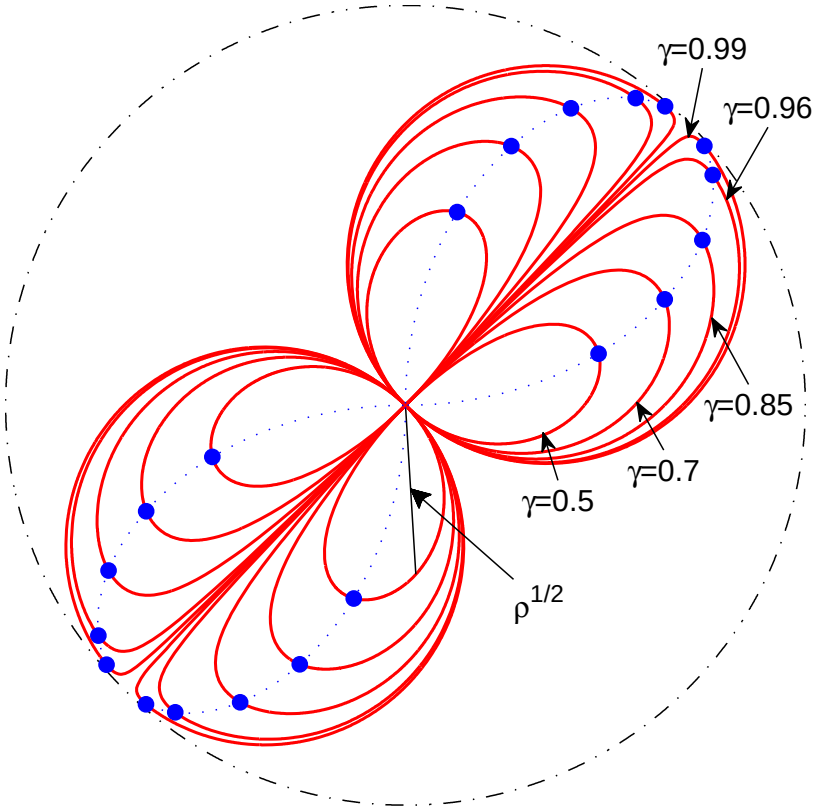

Figure 4.1. Consider a simple two-state MDP with transition matrix $P = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$, and let $K = L^\top DL$, so that J is the MSBE. Assume initially there are no basis functions. In this case the first BEBF is r, and the V-BEBF $L^- r$ is the value function itself. The figure shows how $\rho^{\frac{1}{2}}$ varies with γ and r. When r varies on the unit circle (dotted black circle), the distance between the points on the butterfly-shaped curve and the origin denotes $\rho^{\frac{1}{2}}$, and each curve corresponds to a different γ. The blue dots show where ρ achieves its maximum (computed from Proposition 4.1). It can be seen that when γ approaches 1, the worst-case ρ approaches the unit circle, which indicates less and less improvement when the BEBF is added. Note that ρ is always zero when r is exactly on $\left[\cos\frac{\pi}{4}, \sin\frac{\pi}{4}\right]^\top$ and $\left[\cos\frac{3\pi}{4}, \sin\frac{3\pi}{4}\right]^\top$, the eigendirections of L, but changes abruptly when r leaves the first eigendirection.

be the best-in-class given θ, then the error between an estimate ϕ and $\hat{\phi}$, the exact V-BEBF, is bounded by

$$\begin{aligned}\left\|\phi - \hat{\phi}\right\| &= \left\|\left(\phi - \hat{\phi}_\theta\right) + \left(\hat{\phi}_\theta - L^- \varepsilon\right) + \left(L^- \varepsilon - L^- \hat{\varepsilon}\right) + \left(L^- \hat{\varepsilon} - \hat{\phi}\right)\right\| \\ &\leq \underbrace{\left\|\hat{\phi}_\theta - L^- \varepsilon\right\|}_{\text{(a)}} + \underbrace{\left\|\Phi\hat{\theta} - \Phi\theta\right\|}_{\text{(b)}} + \underbrace{\left\|\phi - \hat{\phi}_\theta\right\|}_{\text{(c)}},\end{aligned}$$

where $\hat{\varepsilon}$ is the Bellman error corresponding to the optimal weight $\hat{\theta}$ and thus by definition $L^- \hat{\varepsilon} - \hat{\phi} = 0$, and

$$L^- \varepsilon - L^- \hat{\varepsilon} = \Phi\theta - \Phi\hat{\theta}$$

is the error in current value estimation. The three items in r.h.s.of the last inequality correspond to the three sources of error mentioned above. As a result, though only one V-BEBF is needed in principle, in practice we still rely on repeatedly adding new approximations of V-BEBFs to compensate for the error in the estimation, as well as changes in the policy and non-stationarities in the environment.

4.3 Incremental Basis Projection with V-BEBF

The result in the previous section suggests a natural way to perform value function approximation incrementally, whereby a primary learner receives reward from the environment, modifies its value function estimate over a set of basis functions, and propagates the TD-error to a secondary learner, which estimates the value function of the TD-error, i.e., an approximation of V-BEBF, which is then added to the existing set of basis functions used by the primary learner (Figure 4.2). The secondary learner can use any form of approximator for V-BEBF. If the secondary learner uses non-linear function approximation, then the framework above allows one to increase the representational power of LFA.

In this study, we confine ourselves to the simple, linear case where V-BEBF is combined with LBP (Section 2.3.2). More specifically, the primary reinforcement learner uses a set of N 'refined' basis functions $\Phi = \left[\phi_1, \cdots, \phi_N\right]$, with each ϕ_n being a *linear* combination of a set of M 'raw' basis functions, $\Psi = \left[\psi_1, \cdots, \psi_M\right]$, where $N \ll M$, i.e.,

$$\phi_n = \sum\nolimits_{m=1}^{M} w_{m,n}\psi_m = \Psi w_n.$$

Here $w_n = \left[w_{1,n}, \ldots, w_{M,n}\right]^\top$ are the mixing coefficients. Denote $W = \left[w_1, \ldots, w_N\right]$, then $\Phi = \Psi W$ takes the form of LBP. The new set of mixing coefficients w_{N+1}

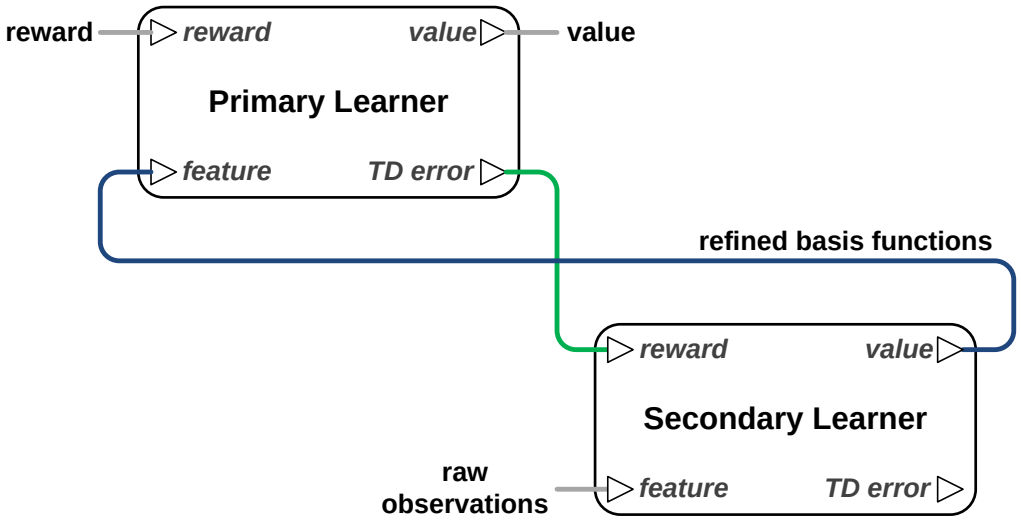

Figure 4.2. **The V-BEBF framework.** A primary learner receives reward from the environment, modifies its value function estimate over a set of refined basis functions, and propagates the TD-error to a secondary learner, that estimates V-BEBF, which then becomes the new refined basis function to be used by the primary learner.

are thereby generated by the secondary learner such that Ψw_{N+1} approximates the current V-BEBF, and are subsequently added as a new column of W, which amounts to adding V-BEBF as a new refined basis function. We refer to this approach as IBP-V (Incremental Basis Projection with V-BEBF), and to its BEBF counterpart, where w_{N+1} is constructed to approximate the current Bellman error, as IBP-B (IBP with BEBF).

Both IBP-V and IBP-B *linearly* reduce the feature dimensionality, but do not add to the representational power of the original set of raw basis functions. However, adopting the LBP framework allows us to focus on the difference between the fundamental ideas of V-BEBF and BEBF and avoid unnecessary complications caused by specific (nonlinear) implementations, though it must be pointed out that most of the practical value of V-BEBF and BEBF does lie in the construction of basis functions depending nonlinearly on the observations [Wu and Givan, 2005; Keller et al., 2006].

4.3.1 Batch IBP-V

We explore the case where the primary and the secondary learner use LSTD[2]. Let $\hat{\theta}_{raw} = \left(\Psi^\top D L \Psi\right)^- \Psi^\top D r$ be the LSTD solution using the raw basis functions, $\hat{\theta}_{ref} = \left(\Phi^\top D L \Phi\right)^- \Phi^\top D r$ be the LSTD solution using the refined basis functions. Then by definition the solution of V-BEBF is given by

$$\hat{w}_{vbebf} = \left(\Psi^\top D L \Psi\right)^- \Psi^\top D \hat{\varepsilon}_{ref} \tag{4.6}$$

$$= \hat{\theta}_{raw} - W \hat{\theta}_{ref}, \tag{4.7}$$

where $\hat{\varepsilon}_{ref} = r - L\Psi W \hat{\theta}_{ref}$ is the Bellman error. Note that $\hat{\theta}_{raw} = \hat{w}_{vbebf} + W\hat{\theta}_{ref}$, which indicates that a single LSTD solution of the V-BEBF allows the exact representation of $\hat{\theta}_{raw}$. Also, note that the LSTD solution of the BEBF can be derived by just replacing $\left(\Psi^\top D L \Psi\right)^-$ with $\left(\Psi^\top D \Psi\right)^-$ in Equation 4.6.

This naive implementation of IBP-V offers no computational advantage over directly computing $\hat{\theta}_{raw}$, since starting from an empty W, the first V-BEBF is exactly $\hat{\theta}_{raw}$. Without assumptions about the sparsity of the basis functions, the exact computation of $\hat{\theta}_{raw}$ requires $O\left(M^2 T\right)$ time, and $O\left(M^2\right)$ storage [Geramifard et al., 2006], which is often not feasible. For this reason, we consider using only $B \ll M$ randomly chosen raw basis functions to approximate each V-BEBF (see Algorithm 4.1). If N V-BEBFs are constructed in total to approximate the value function, then the overall computational cost is $O(MT) + O\left(NB^2 T\right) + O\left(N^3 T\right)$, where the three terms correspond respectively to the cost of computing M raw basis functions, N V-BEBFs, and the weights over the refined basis functions for N times. The storage cost is $O(NB) + O\left(B^2\right) + O\left(N^2\right)$, with the first term being the storage for W.

Algorithm 4.1 raises the issue of how to choose N, the number of refined basis functions to construct, and B, the number of raw basis functions used to generate each refined basis function. The following analysis shows that we can choose $N = B = (cM)^{\frac{1}{2}}$, where $c = -\log \epsilon$, with the guarantee that at least $1 - \epsilon$ fraction of the raw basis functions are covered when constructing the refined basis functions. In addition, with this choice of N, B, the computational complexity becomes $O\left((cM)^{\frac{3}{2}} T\right)$ in time, and $O\left(c^2 M\right)$ in storage. To see this, note that the probability of a raw basis function getting selected at least once is

[2]For succinctness, we use the exact sample distribution D and transition matrix P. But in practice, both D and P must be replaced by their finite sample approximations.

Algorithm 4.1 Batch-IBP-V[3]

Input: samples $\left(s_t, s_{t+1}, r_t\right)_{t=1}^{T}$, discount factor γ, raw basis functions $\psi_1, \ldots, \psi_M$, number of basis functions for each V-BEBF B, maximum number of refined basis functions N

Output: mixing matrix W, weights over refined basis functions θ

$W \leftarrow [\,]$; $\theta \leftarrow [\,]$
while $n < N$ **do**
  Select at random $U \subset \{1, \ldots, M\}$ with $|U| = B$
  for $t = 1$ to T **do**
    $\omega \leftarrow \left[\psi_1\left(s_t\right), \ldots, \psi_M\left(s_t\right)\right]$
    $\omega_p \leftarrow \left[\psi_1\left(s_{t+1}\right), \ldots, \psi_M\left(s_{t+1}\right)\right]$
    $\delta_t \leftarrow r_t + \gamma\omega_p W\theta - \omega W\theta$
  end for
  $w_r \leftarrow \mathrm{LSTD}\Big(\left(s_t, s_{t+1}, \delta_t\right)_{t=1}^{T}, \gamma, \Psi_{[U]}\Big)$
  $w \leftarrow 0_{M\times 1}$; $w_{[U]} \leftarrow w_r$; $W \leftarrow [W \vdots w]$
  $\theta \leftarrow \mathrm{LSTD}\Big(\left(s_t, s_{t+1}, r_t\right)_{t=1}^{T}, \gamma, \Psi W\Big)$
  $n \leftarrow n + 1$
end while

$1 - \left(1 - \frac{B}{M}\right)^N$, and our assumption requires

$$1 - \left(1 - \frac{B}{M}\right)^N \geq 1 - \epsilon, \text{ or } N \geq \frac{-\log\epsilon}{-\log\left(1 - \frac{B}{M}\right)}.$$

Assume $\frac{B}{M} = o(1)$, and note that $-\log(1-x) > x$. It suffices that $NB > cM$, and letting $N = B = (cM)^{\frac{1}{2}}$ gives the complexity result.

4.3.2 Online IBP-V

Batch IBP-V can be modified to work online, e.g., by replacing LSTD with iLSTD [Geramifard et al., 2006]. One may instead use the linear complexity methods such as TD or TDC [Sutton et al., 2009], (see Algorithm 4.2), so that the complexity of each time step is $O(MN)$ if all raw basis functions are used to estimate the V-BEBF or BEBF, or $O(BN)$ if only B raw basis functions are used. In practice, N may also be controlled by dynamically removing refined basis functions

[3] We assume the subroutine $\mathrm{LSTD}\big(samples, \gamma, basis\big)$ produces LSTD estimates of the weights for the provided samples, i.e., triples $\left(s_t, s_{t+1}, r_t\right)$, and basis functions. Also, for an arbitrary set of integers U, $w_{[U]}$ denotes the sub-vector with entries indexed by U.

Algorithm 4.2 Online-IBP-V[4]

Input: sample trajectory (s_t, s_{t+1}, r_t), discount factor γ, raw basis functions $\psi_1, \ldots, \psi_M$, steps before adding each V-BEBF C
Output: mixing matrix W, weights over refined basis functions θ

1: $c \leftarrow C$
2: $W \leftarrow []$; $\theta \leftarrow []$; $w \leftarrow 0_{M\times 1}$
3: **repeat**
4: $\quad \omega \leftarrow \left[\psi_1(s_t), \ldots, \psi_M(s_t)\right]$
5: $\quad \omega_p \leftarrow \left[\psi_1(s_{t+1}), \ldots, \psi_M(s_{t+1})\right]$
6: $\quad \delta \leftarrow r_t + \left(\gamma\omega_p - \omega\right) W\theta$
7: $\quad \theta \leftarrow \theta + \alpha\delta\left(\omega W\right)^\top$
8: $\quad w \leftarrow w + \alpha\left(\delta + \gamma\omega_p w - \omega w\right)\omega^\top$
9: $\quad c \leftarrow c - 1$
10: $\quad$ **if** $c = 0$ **then**
11: $\quad\quad c \leftarrow C$
12: $\quad\quad W \leftarrow [W \vdots w]$; $\theta \leftarrow [\theta^\top \vdots 0]^\top$; $w \leftarrow 0_{M\times 1}$
13: $\quad$ **end if**
14: **until** trajectory ends

whose weights are very small. If N is upper bounded by a constant, then online IBP-V is linear in the number of raw basis functions.

4.4 Experiments

We conduct two simple experiments to compare the performance of IBP-V and IBP-B, with the aim of illustrating the difference resulting from the principles underlying V-BEBF and BEBF. In both experiments, batch and online IBP-V were tested on randomly generated[5] MRPs with 500 states and a branching factor (the number of successor states) of 5. The reward at each state was drawn i.i.d. from a standard normal distribution. All experiments used binary raw basis functions over the states, that were generated by filling Ψ with i.i.d. Bernoulli variables

[4]For simplicity, assume TD is used in both the primary and secondary learner (see line 7 and 8), which can be replaced by other algorithms. Also, the learning rates are fixed to α.

[5]We use the method described in `http://webdocs.cs.ualberta.ca/~sutton/RandomMDPs.html`

with $p = 0.2$. The error was measured using

$$\eta(W, \theta) = \frac{\left(\hat{\theta}_{raw} - W\theta\right)^\top \Psi^\top D \Psi \left(\hat{\theta}_{raw} - W\theta\right)}{\hat{\theta}_{raw}^\top \Psi^\top D \Psi \hat{\theta}_{raw}}, \tag{4.8}$$

which is the normalized Mean Square Value Error (MSVE) with respect to the best possible value function approximation $\Psi\hat{\theta}_{raw}$ from the current set of raw basis functions. The initial W is set to empty so that the initial value of η is always 1.

For the batch case, IBP-V was compared with IBP-B, with the results from random feature projection [Ghavamzadeh et al., 2010] serving as a baseline. The number of raw basis functions, B, used to compute each V-BEBF or BEBF, and the number of refined basis functions N generated in total, were both set to $\sqrt{5M}$, following the analysis in Section 4.3.1.

For the online case, IBP-V was compared to IBP-B with $B = M$ (each refined basis function uses all of the raw basis functions). A refined basis function was added after every 2000 time steps. To provide a baseline, these two incremental approaches were compared to TD using only the raw basis functions. All TD updates[6] used the same learning rate 0.1, *where the baseline TD achieves the best performance* (see Section 4.5.4 for further discussion).

For all experiments, each method was run 50 times with $\gamma = \{0.9, 0.99, 0.999\}$ and $M = \{100, 200\}$ raw basis functions, for a total of six different comparisons in both the batch and online settings.

4.4.1 Results: Batch

Figure 4.3 plots the error, η, against the number of refined basis functions added. Each curve is the average of 50 runs, each of which is based on an independent sample of the MRP and the set of raw basis functions, and a trajectory of length $T = 5000$ is used. (The error bar is small thus is omitted). It can be seen that both IBP-V and IBP-B perform significantly better than random feature projection. It is also clear that when γ approaches 1, the difference between adding IBP-V and IBP-B increases dramatically—when $\gamma = 0.999$, the error decreases very slowly as BEBFs are added, which coincides with Proposition 4.1. For smaller γ, the difference between BEBF and V-BEBF diminishes, since BEBF can be seen as V-BEBF with discount factor 0. In addition, although each V-BEBF is constructed from a much smaller number of raw basis functions, (e.g.,

[6]Note that the secondary learner in IBP-B performs Least Mean Square [Widrow and Stearns, 1985] update, which is equivalent to TD with $\gamma = 0$.

$B = 31$ for $M = 200$), the error still decreases rapidly as the number of V-BEBFs is increased, and our heuristic choice of B and N is shown to be quite effective.

4.4.2 Results: Online

Figure 4.4 plots the error η against the number of time-steps. Note the 'stair' shape of the curves for both IBP-V and IBP-B caused by the abrupt drop in error each time a new refined basis function is added. It can be seen that when the discount factor approaches 1, the first few V-BEBFs added are far more effective than the BEBFs added. Also, the performance of IBP-V matches the baseline TD method after only a few basis functions are added, indicating that the basis functions constructed are indeed useful.

4.5 Discussion

We address four issues arising from the previous discussion, respectively on the interpretation of the contributions (Section 4.5.1), the practical benefit of V-BEBF (Section 4.5.2), the intepretation of the experimental results (Section 4.5.3), and the practical benefit of IBP-V (Section 4.5.4).

4.5.1 Interpretation of the Contributions

The main contribution of this chapter is twofold: (i) proposing V-BEBF for reward-sensitive basis construction, and (ii) demonstrating that V-BEBF is a promising alternative to BEBF, particularly when the discount factor $\gamma \to 1$.

It is worth pointing out that V-BEBF is a general methodology in parallel with BEBF, rather than a specific algorithm. Indeed, there are countless way in which V-BEBF (as well as BEBF) can be approximated. It is therefore foreseenable that the effectiveness of any particular implementation will be decided not only by the underlying principle, but also by the quality of the approximation.

The study in Section 4.2.2 compares the theoretical properties of V-BEBF and BEBF, *assuming they are computed exactly*. The analysis ignores the approximation error, and may best be intepreted as an attempt to provide insights into the potential effectiveness (or ineffectiveness) of the two methods. On the other hand, analysis of this nature is frequently done in the literature, and particular to BEBF, two previous theoretical studies by Parr et al. [2007] and Mahadevan and Liu [2010] make the same assumption.

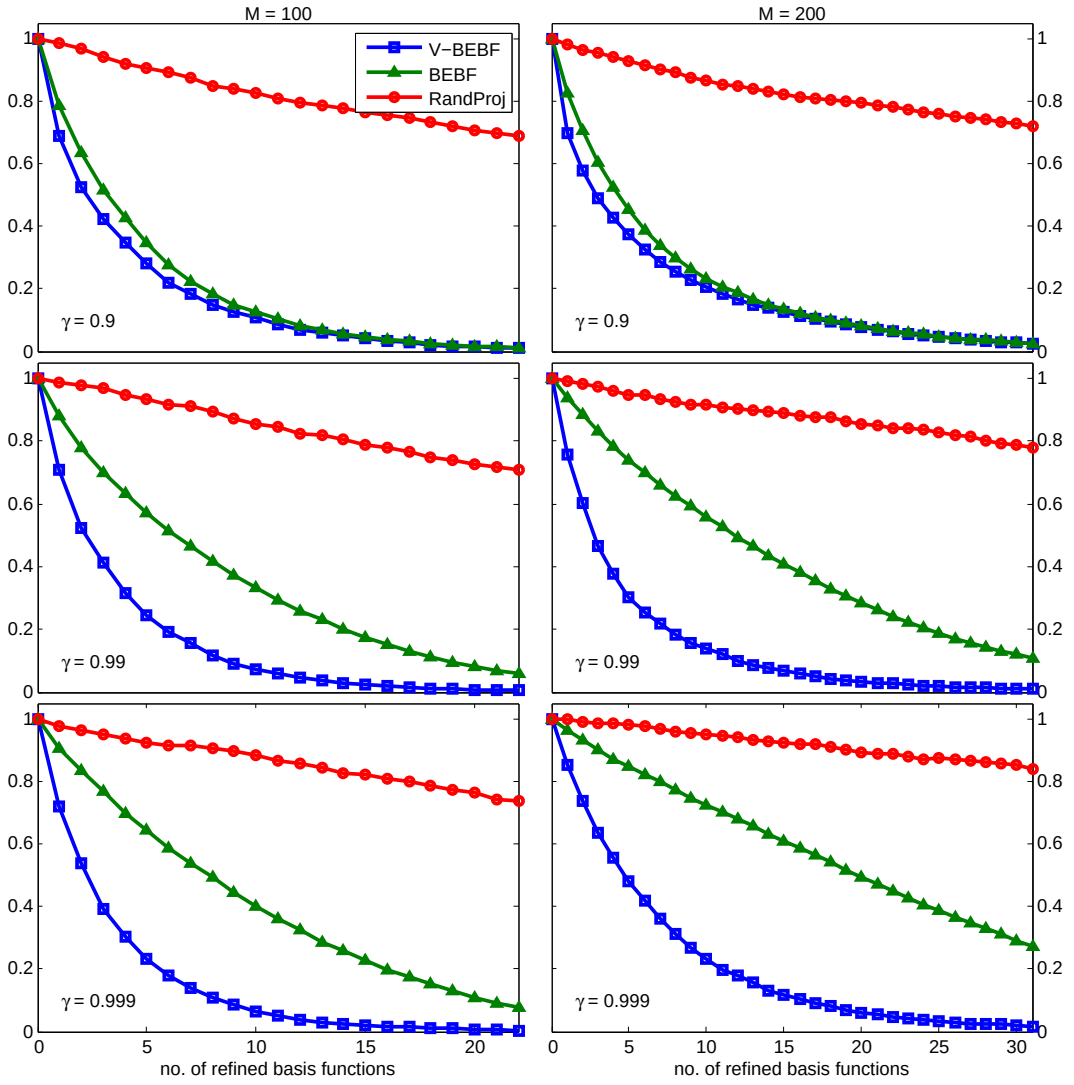

Figure 4.3. **Results for batch IBP**. Each graph shows the performance of the three methods compared, IBP-V, IBP-B, and random feature projection (RFP), in terms of the error η (Equation 4.8) against the number of refined basis function added (averaged over 50 runs).

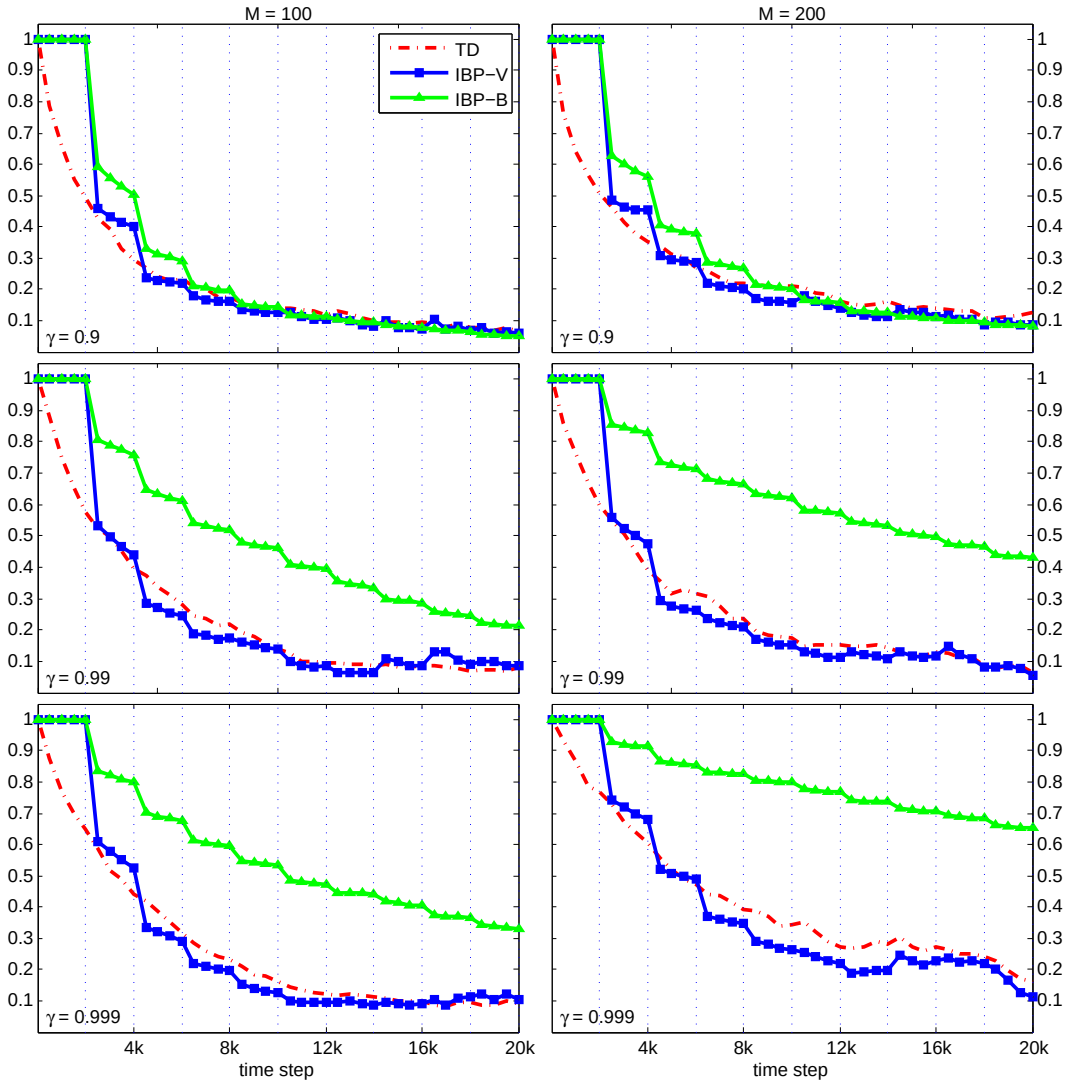

Figure 4.4. **Results for online IBP**. Each graph shows the performance of the three methods compared, IBP-V, IBP-B, and TD, in terms of η against time (averaged over 50 runs). The vertical dotted lines indicate the moments at which a new refined basis function is added to the primary learner. Notice that in both IBP-V and IBP-B, at each time a new basis function is added, the error drops abruptly.

4.5.2 Practical Benefit of V-BEBF

As pointed out in Section 4.2.1, computing V-BEBF (and also BEBF) *exactly* is as hard as computing the value function itself, if not harder. While this is indeed true, we argue that V-BEBF (BEBF) is nevertheless useful. The key insight is that crude approximations to V-BEBF (BEBF), which are obtained using far less computation or samples, often suffices in providing (a relatively small set of) useful (refined) basis functions. These basis functions can then be used by the primary learner to achieve good overall accuracy in value estimation, while keeping the cost (in terms of either computation or sample complexity) low. This is *partially* confirmed by our experiment:

- In the batch case, each refined basis function is generated using only a small fraction ($\sqrt{5M}$ compared to M) of raw basis functions. As a result, the computational cost of approximating each V-BEBF is $O(MT)$, rather than $O\left(M^2T\right)$. However, Figure 4.3 shows that a relatively small number of V-BEBFs ($\sqrt{5M}$) enables value-function approximation to high accuracy.

- In the online case, each approximate V-BEBF is generated from 2000 samples rather than from the whole trajectory (in our case 20000 samples). As a result, the approximation is by no means accurate. However, Figure 4.4 shows that with each initial V-BEBF added, the error drops steeply, resulting in the stair shape curve. In addition, the overall accuracy from the first 10 refined basis functions is on par with TD using the same learning rate and all raw basis functions, indicating that the approximate V-BEBFs are indeed good basis functions.

4.5.3 Interpretation of the Experimental Results

The empirical study in Section 4.4 is simple and mainly undertaken to illustrate the theoretical findings. In the batch experiment, both V-BEBFs and BEBFs are computed optimally in the least square sense, and therefore the performance difference is caused solely by the underlying principles. On the other hand, the online experiment is inconclusive, as many factors may interfere with the performance of the algorithms, e.g., separate tuning the learning rates for primary and secondary learners, or using different value estimation algorithms such as TD(λ). Nevertheless, both batch and online results in these experiments were in line with the theoretical finding, i.e., that the performance of BEBF degrades significantly when $\gamma \to 1$, in comparison with V-BEBF.

It is worth pointing out that the algorithms we used in the experiment, IBP-V and IBP-B, do not expand the set of raw basis functions. The practical value of these algorithms is unclear, as it is widely perceived that the main usage of basis construction is to exhance the representational power of LFA. However, we argue that the experiments are targeted toward revealing the difference between the fundamental ideas underlying V-BEBF and BEBF. To this end, these algorithms are suitable due to their simplicity, and the comparisons are *fair*, since both (refined) V-BEBFs and BEBFs are selected from the same set of possible functions, i.e., the linear span of the raw basis functions.

4.5.4 Practical Benefit of IBP-V

In some of our experiments, we found that IBP-V converges quicker than TD using the same learning rate. This raises an interesting question as to whether IBP-V can be used as a practical approach to trade off computational complexity with sample complexity.

To investigate this, we conducted an additional experiment, where we use the same setting as that of Section 4.4.2, except fixing $M = 200$ and $\gamma = 0.999$. Four algorithms are compared, TD, IBP-V, and IBP-B, and a new variation to IBP-V, called IBP-V-ls, where the primary learner uses LSTD instead of TD. We vary the learning rate of TD (from $\alpha = 0.01$ to $\alpha = 0.5$ with step size 0.01), and in each case the same α is inherited by IBP-V, IBP-B, and IBP-V-ls (secondary learner only). Results with selected learning rates are shown in Figure 4.5. It can be seen that with small learning rate ($\alpha = 0.01$ and $\alpha = 0.02$), IBP-V outperforms TD, while the performance becomes almost equal around $\alpha = 0.1$, where TD achieves the best performance. Further increasing α beyond 0.2 renders both algorithms unstable. In contrast, the performance of IBP-V-ls varies little with the learning rate, and is comparable with TD with the best learning rates. However, since the primary learner uses LSTD, IBP-V-ls does not suffer from the instability at high learning rates, even if the V-BEBFs are updated aggressively with $\alpha = 0.5$. The IBP-B algorithm, suffering from a high discount factor, does not perform as well with any learning rate, as suggested by the earlier analysis. Given that the optimal learning rate is usually not known in advance, these results indicate that IBP-V, in particular the IBP-V-ls variant, may be of practical value. However, the experiment is inconclusive and the question remains open for future study.

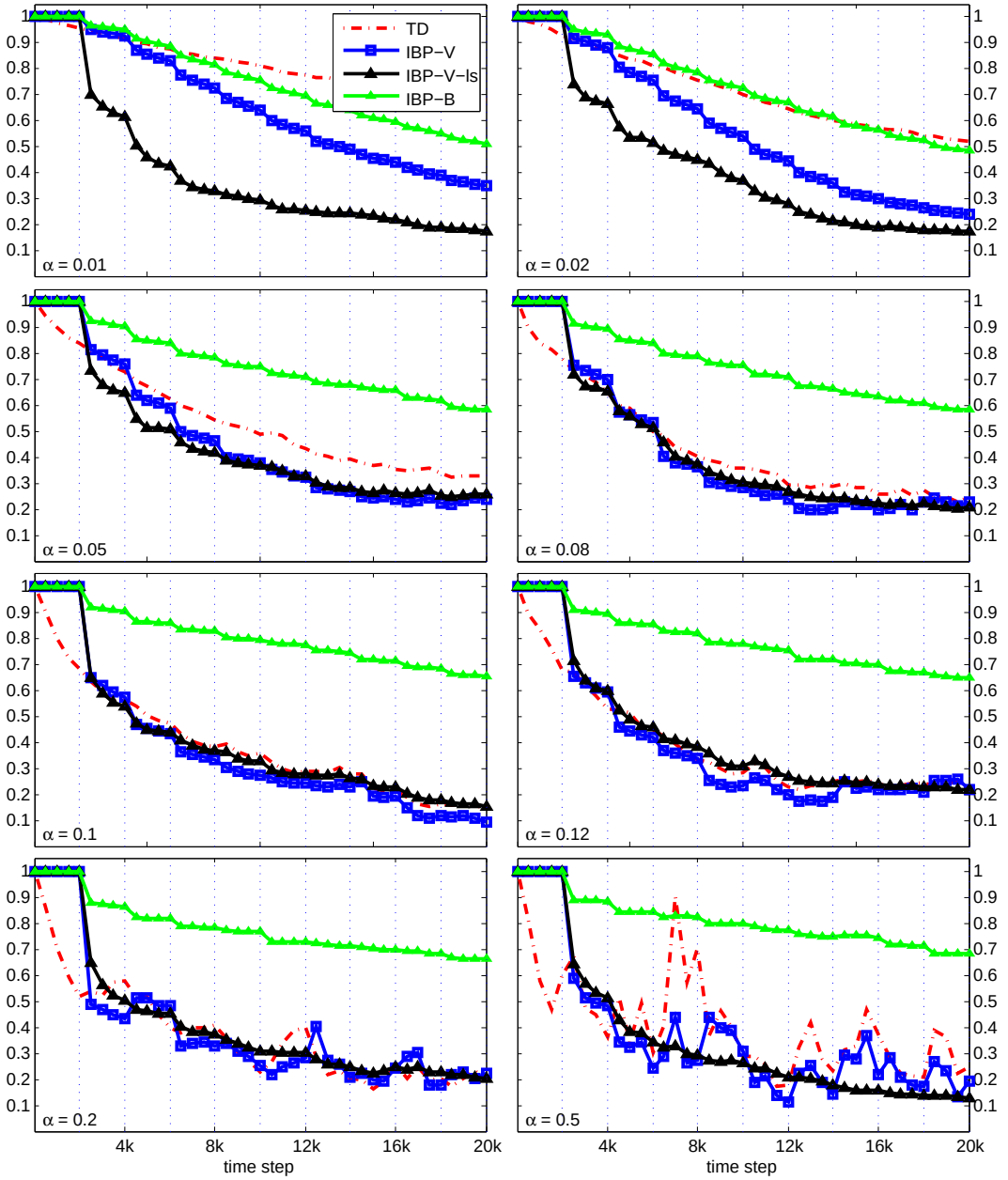

Figure 4.5. **Additional results on IBP.** Each graph shows the performance of the four methods compared, TD, IBP-V, IBP-B, and IBP-V-ls, in terms of η against time (averaged over 50 runs). In the experiment, we fix $M = 200$ and $\gamma = 0.999$, and vary the learning rate α (shown in the lower left corner).

Chapter 5

Information Theoretic Exploration in Dynamic Environments

Chapters 3 and 4 studied basis construction in model-free RL. In this chapter, we move to the model-based case and consider another representation generation problem, namely model learning. In particular, the following question is asked:

> *How should an agent plan its actions such that the knowledge about the environment accumulates as quickly as possible?*

We provide an answer to this question under a classical framework, in which the agent improves its model of the environment through Bayesian inference, and the learning progress is measured in terms of Shannon information. We show that the agent can optimally plan its actions by solving an RL problem in which the reward is given as the expected (immediate) information gain. We then concentrate on a special case, where the environment is finite and Markovian, and the agent's knowledge about the environment is represented by a collection of Dirichlet distributions over the transition probabilities of the Markovian model. In this case, we first prove the existence of optimal exploration strategies assuming an infinite planning horizon but with future information gain discounted, and then show that such strategies can be approximated well by solving a series of dynamic programming problems.

The idea of actively selecting actions to accelerate model learning has a long history [e.g., Fedorov, 1972; Schmidhuber, 1990, 1991; Thrun and Möller, 1991; Storck et al., 1995; Chaloner and Verdinelli, 1995; Özgür Şimşek and Barto, 2006; Schmidhuber, 2010; Orseau, 2011]. Primarily, this idea is pursued under the framework of *artificial curiosity* [see e.g., Schmidhuber, 1991, 2010] as well as *intrinsically motivated RL* [Singh et al., 2004; Stout et al., 2005;

Özgür Şimşek and Barto, 2006]. Measuring learning progress using Shannon information gain also dates back to Lindley [1956]; Fedorov [1972], and is re-introduced recently as *Bayesian surprise* [Itti and Baldi, 2006]. Our work combines these two ideas through rigorous derivation from first principles, in contrast to the more heuristic approaches [e.g., Storck et al., 1995; Ortega and Braun, 2010]. It is worth pointing out that in contrast to the exploration problem studied in model-free RL [see e.g., Kearns and Singh, 2002; Strehl and Littman, 2005; Kolter and Ng, 2009b], where the emphasis is on the balance between exploration and exploitation, *we do not address the exploitation aspects*, and focus solely on exploration side of the problem.

The rest of the chapter is organized as follows: Section 5.1 reviews the basic concepts and establishes the terminology; Section 5.2 presents our formulation of optimal Bayesian exploration; Section 5.3 focuses on exploration in MDP; Section 5.4 presents a simple illustrative example; Section 5.5 discusses the results. The proofs are detailed in Section 5.6.

5.1 Background

Suppose that the agent interacts with the environment in discrete time cycles $t = 1, 2, \ldots$. In each cycle, the agent performs an action a_t, then observes o_t. We assume no external reward signals are provided to the agent, hence the process is pure exploratory. A *history* h is either the empty string $\emptyset$ or a string of the form $a_1 o_1 \cdots a_t o_t$ for some t, and ha and hao refer to the strings resulting from appending a and ao to h, respectively.

5.1.1 Learning from Sequential Interactions

To facilitate the subsequent discussion under a probabilistic framework, we make the following assumptions:

Assumption 5.1 *The models of the environment under consideration are fully described by a random element* Θ *which* depends solely on the environment. *Moreover, the agent's initial knowledge about* Θ *is summarized by a prior density* $p(\theta)$.

Assumption 5.2 *The agent is equipped with a* conditional predictor $p(o|ha;\theta)$, *i.e. the agent is capable of refining its prediction in the light of information about* Θ.

Using $p(\theta)$ and $p(o|ha;\theta)$ as building blocks, it is straightforward to formulate learning in terms of probabilistic inference. From Assumption 5.1, given the history h, the agent's knowledge about Θ is fully summarized by $p(\theta|h)$. According to Bayes rule, $p(\theta|hao) = \frac{p(\theta|ha)p(o|ha;\theta)}{p(o|ha)}$, with $p(o|ha) = \int p(o|ha,\theta)\,p(\theta|h)\,d\theta$. The term $p(\theta|ha)$ represents the agent's current knowledge about Θ given history h and an additional action a. Since Θ depends solely on the environment, and, importantly, *knowing the action without subsequent observations cannot change the agent's state of knowledge about* Θ, then $p(\theta|ha) = p(\theta|h)$, and thus the knowledge about Θ can be updated using

$$p(\theta|hao) = p(\theta|h) \cdot \frac{p(o|ha;\theta)}{p(o|ha)}. \tag{5.1}$$

5.1.2 Information Gain as Learning Progress

Let h and h' be two histories such that h is a prefix of h'. The respective posteriors of Θ are $p(\theta|h)$ and $p(\theta|h')$. Using h as a reference point, the amount of information gained when the history grows to h' can be measured using the KL divergence between $p(\theta|h)$ and $p(\theta|h')$. This *information gain* from h to h' is defined as

$$g(h'\|h) = KL\left(p\left(\theta|h'\right)\|p\left(\theta|h\right)\right) = \int p\left(\theta|h'\right)\log\frac{p(\theta|h')}{p(\theta|h)}d\theta.$$

As a special case, if $h = \emptyset$, then $g(h') = g(h'\|\emptyset)$ is the *cumulative information gain* with respect to the prior $p(\theta)$. We also write $g(ao\|h)$ for $g(hao\|h)$, which denotes the information gained from an additional action-observation pair.

From an information theoretic point of view, the KL divergence between two distributions p and q represents the additional number of bits required to encode elements sampled from p, using optimal coding strategy designed for q. This can be interpreted as the degree of 'novelty' or 'surprise' caused by observing samples from p when expecting samples from q.

The key property of information gain for the treatment below is the following decomposition: Let h be a prefix of h' and h' be a prefix of h'', then

$$\mathbb{E}_{h''|h'}\,g\left(h''\|h\right) = g\left(h'\|h\right) + \mathbb{E}_{h''|h'}\,g\left(h''\|h'\right). \tag{5.2}$$

That is, the information gain is *additive in expectation*.

Having defined the information gain from trajectories ending with observations, one may proceed to define the *expected information gain* of performing action a, before observing the outcome o. Formally, the *expected information gain* of performing a with respect to the current history h is given by

$g(a\|h) = \mathbb{E}_{o|ha} g(ao\|h)$. A simple derivation gives

$$g(a\|h) = \sum_o \int p(o,\theta|ha) \log \frac{p(\theta|hao)\, p(o|ha)}{p(\theta|h)\, p(o|ha)} d\theta = I(O;\Theta|ha),$$

which shows that $g(a\|h)$ is the mutual information between Θ and the random variable O representing the unknown observation, conditioned on the history h and action a.

5.2 Bayesian Exploration in Dynamic Environments

In this section, we present our notion of Bayesian exploration in dynamic environments, with optimality results assuming a fixed limited life span for the agent. We then discuss conditions required to extend this to infinite time horizons.

5.2.1 Planning in Finite Time Horizon

Suppose that the agent has experienced history h, and is about to choose τ more actions in the future. Let π be a policy mapping the set of histories to the set of actions, such that the agent performs a with probability $\pi(a|h)$ given h. Define the *curiosity Q-value* $q_\pi^\tau(h,a)$ as the expected information gained from the additional τ actions, assuming that the agent performs a in the next step and follows policy π in the remaining $\tau - 1$ steps. Formally, for $\tau = 1$,

$$q_\pi^1(h,a) = \mathbb{E}_{o|ha} g(ao\|h) = g(a\|h),$$

and for $\tau > 1$,

$$\begin{aligned} q_\pi^\tau(h,a) &= \mathbb{E}_{o|ha} \mathbb{E}_{a_1|hao} \mathbb{E}_{o_1|haoa_1} \cdots \mathbb{E}_{o_{\tau-1}|h\cdots a_{\tau-1}} g\left(haoa_1o_1 \cdots a_{\tau-1}o_{\tau-1}\|h\right) \\ &= \mathbb{E}_{o|ha} \mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}|hao} g\left(haoa_1o_1 \cdots a_{\tau-1}o_{\tau-1}\|h\right), \end{aligned}$$

where $a_1, \ldots, a_{\tau-1}$ are determined by policy π.

The curiosity Q-value can be defined recursively. Applying Equation 5.2 for $\tau = 2$,

$$\begin{aligned} q_\pi^\tau(h,a) &= \mathbb{E}_{o|ha} \mathbb{E}_{a_1o_1|hao} g\left(haoa_1o_1\|h\right) \\ &= \mathbb{E}_{o|ha}\left[g(ao\|h) + \mathbb{E}_{a_1o_1|hao} g\left(a_1o_1\|hao\right)\right] \\ &= g(a\|h) + \mathbb{E}_{o|ha} \mathbb{E}_{a'|hao} q_\pi^1\left(hao, a'\right). \end{aligned}$$

And for $\tau > 2$,

$$\begin{aligned} q_\pi^\tau(h,a) &= \mathbb{E}_{o|ha}\mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}|hao} g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|h\right) \\ &= \mathbb{E}_{o|ha}\left[g(ao\|h) + \mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}} g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|hao\right)\right] \\ &= g(a\|h) + \mathbb{E}_{o|ha}\mathbb{E}_{a'|hao} q_\pi^{\tau-1}\left(hao,a'\right). \end{aligned} \tag{5.3}$$

Noting that Equation 5.3 bears great resemblance to the definition of action-value function in RL, with the one step expected information gain in the place of the reward. One can similarly define the *curiosity value* of a particular history as $v_\pi^\tau(h) = \mathbb{E}_{a|h} q_\pi^\tau(h,a)$, analogous to the value function, which can also be iteratively defined as $v_\pi^1(h) = \mathbb{E}_{a|h} g(a\|h)$, and

$$v_\pi^\tau(h) = \mathbb{E}_{a|h}\left[g(a\|h) + \mathbb{E}_{o|ha} v_\pi^{\tau-1}(hao)\right].$$

The curiosity value $v_\pi^\tau(h)$ is the expected information gain of performing the additional τ steps, assuming that the agent follows policy π. The two notations can be combined to write

$$q_\pi^\tau(h,a) = g(a\|h) + \mathbb{E}_{o|ha} v_\pi^{\tau-1}(hao). \tag{5.4}$$

This equation has an interesting interpretation: since the agent is operating in a dynamic environment, it has to take into account not only the immediate expected information gain of performing the current action, i.e., $g(a\|h)$, but also the expected curiosity value of the situation in which the agent ends up due to the action, i.e., $v_\pi^{\tau-1}(hao)$. As a consequence, *the agent needs to choose actions that balance the two factors in order to improve its total expected information gain*.

Now we show that there is an optimal policy π_*, which leads to the maximum cumulative expected information gain given any history h. To obtain the optimal policy, one may work backwards in τ, taking greedy actions with respect to the curiosity Q-values at each time step. Namely, for $\tau = 1$, let

$$q^1(h,a) = g(a\|h),\ \pi_*^1(h) = \arg\max_a g(a\|h), \text{ and } v^1(h) = \max_a g(a\|h),$$

such that $v^1(h) = q^1\left(h, \pi^1(h)\right)$, and for $\tau > 1$, let

$$q^\tau(h,a) = g(a\|h) + \mathbb{E}_{o|ha}\left[\max_{a'} q^{\tau-1}\left(a'|hao\right)\right] = g(a\|h) + \mathbb{E}_{o|ha} v^{\tau-1}(hao),$$

with $\pi_*^\tau(h) = \arg\max_a q^\tau(h,a)$ and $v^\tau(h) = \max_a q^\tau(h,a)$. We show that $\pi_*^\tau(h)$ is indeed the optimal policy for any given τ and h in the sense that the curiosity

value, when following π_*^τ, is maximized. To see this, take any other strategy π, first notice that

$$v^1(h) = \max_a g(a\|h) \geq \mathbb{E}_{a|h} g(a\|h) = v_\pi^1(h).$$

Moreover, assuming $v^\tau(h) \geq v_\pi^\tau(h)$,

$$\begin{aligned} v^{\tau+1}(h) &= \max_a \left[g(a\|h) + \mathbb{E}_{o|ha} v^\tau(hao) \right] \geq \max_a \left[g(a\|h) + \mathbb{E}_{o|ha} v_\pi^\tau(hao) \right] \\ &\geq \mathbb{E}_{a|h} \left[g(a\|h) + \mathbb{E}_{o|ha} v_\pi^\tau(hao) \right] = v_\pi^{\tau+1}(h). \end{aligned}$$

Therefore $v^\tau(h) \geq v_\pi^\tau(h)$ holds for arbitrary τ, h, and π. The same can be shown for curiosity Q-values, namely, $q^\tau(h,a) \geq q_\pi^\tau(h,a)$, for all τ, h, a, and π. It may be beneficial to write q^τ in explicit forms, namely,

$$q^\tau(h,a) = \mathbb{E}_{o|ha} \max_{a_1} \mathbb{E}_{o_1|haoa_1} \cdots \max_{a_{\tau-1}} \mathbb{E}_{o_{\tau-1}|h\cdots a_{\tau-1}} g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|h\right),$$

Now consider that the agent has a fixed life span T. It can be seen that at time t, the agent has to perform $\pi_*^{T-t}\left(h_{t-1}\right)$ to maximize the expected information gain in the remaining $T-t$ steps. Here $h_{t-1} = a_1o_1\cdots a_{t-1}o_{t-1}$ is the history at time t. However, from Equation 5.2,

$$\mathbb{E}_{h_T|h_{t-1}} g\left(h_T\right) = g\left(h_{t-1}\right) + \mathbb{E}_{h_T|h_{t-1}} g\left(h_T\|h_{t-1}\right).$$

Note that at time t, $g\left(h_{t-1}\right)$ is a constant, thus *maximizing the cumulative expected information gain in the remaining time steps is equivalent to maximizing the expected information gain of the whole trajectory with respect to the prior*. The result is summarized in the following proposition:

Proposition 5.1 *Let* $q_1(h,a) = \bar{g}(a\|h)$, $v_1(h) = \max_a q_1(h,a)$, *and*

$$q_\tau(h,a) = \bar{g}(a\|h) + \mathbb{E}_{o|ha} v_{\tau-1}(hao),\ v_\tau(h) = \max_a q_\tau(h,a),$$

then the policy $\pi_\tau^*(h) = \arg\max_a q_\tau(h,a)$ *is optimal in the sense that* $v_\tau(h) \geq v_\tau^\pi(h)$, $q_\tau(h,a) \geq q_\tau^\pi(h,a)$ *for any* π, τ, h *and* a. *In particular, for an agent with fixed life span* T, *following* $\pi_{T-t}^*\left(h_{t-1}\right)$ *at time* $t = 1,\ldots,T$ *is optimal in the sense that the expected cumulative information gain with respect to the prior is maximized.*

The definition of the optimal exploration policy is constructive, which means that it can be readily implemented, provided that the number of actions and possible observations is finite so that the expectation and maximization can be computed exactly. However, the cost of computing such a policy is $O\left(\left(n_o n_a\right)^\tau\right)$, where n_o and n_a are the number of possible observations and actions, respectively. Since the cost is exponential on τ, planning with a large number of look-ahead steps is infeasible, and approximation heuristics must be used in practice.

5.2.2 Subtlety of the Result

Intuitively, the definition of the curiosity (Q) values bears clear resemblance to their counterparts in RL. It might be tempting to think that the result is nothing more than solving the finite horizon RL problem using $g(ao\|h)$ as the reward signals. However, this is not the case.

First, note that the recursion (Equation 5.4) is a direct consequence of Equation 5.2, which is a special property of the KL divergence. The decomposition does not necessarily hold for other measures of information gain (e.g., mean square errors, etc.)

Second, it is worth pointing out that $g(ao\|h)$ behave differently from normal reward signals in the sense that they are *additive only in expectation*, while in the standard RL setup, the reward signals are assumed to be additive. Figure 5.1 emphasizes such difference. Note that even $g(ao\|h)$ are non-negative for any h, a, and o, the cumulative information gain $g(h)$ may not grow monotonically as h grows. It can also be seen from Figure 5.1 that the difference between the sum of one step information gain and the cumulative information gain can be large for the same history.

5.2.3 Extension to Infinite Horizon

Having to restrict the maximum life span of the agent is rather inconvenient. It is tempting to define the curiosity Q-value in the infinite time horizon case as the limit of curiosity Q-values with increasing life spans, $T \to \infty$. However, this cannot be achieved without additional technical constraints. For example, consider simple coin tossing. Assuming a $Beta(1,1)$ prior over the probability of seeing heads, then the expected cumulative information gain for the next T flips is given by

$$v^T(h_1) = I(\Theta; X_1, \ldots, X_T) \sim \log T.$$

With increasing T, $v^T(h_1) \to \infty$. To resolve this problem, we introduce a discount factor γ. Assume that the agent has a maximum τ actions left, but before finishing the τ actions it may be forced to leave the environment with probability $1-\gamma$ ($0 \le \gamma < 1$) at each time step. In this case, the γ-discounted curiosity Q-value is defined as $q_\pi^{1,\gamma}(h,a) = g(a\|h)$, and

$$\begin{aligned} q_\pi^{\tau,\gamma}(h,a) &= (1-\gamma)\, g(a\|h) + \gamma \left[g(a\|h) + \mathbb{E}_{o|ha} \mathbb{E}_{a'|hao} q_\pi^{\tau-1,\gamma}(hao, a') \right] \\ &= g(a\|h) + \gamma \mathbb{E}_{o|ha} \mathbb{E}_{a'|hao} q_\pi^{\tau-1,\gamma}(hao, a'). \end{aligned}$$

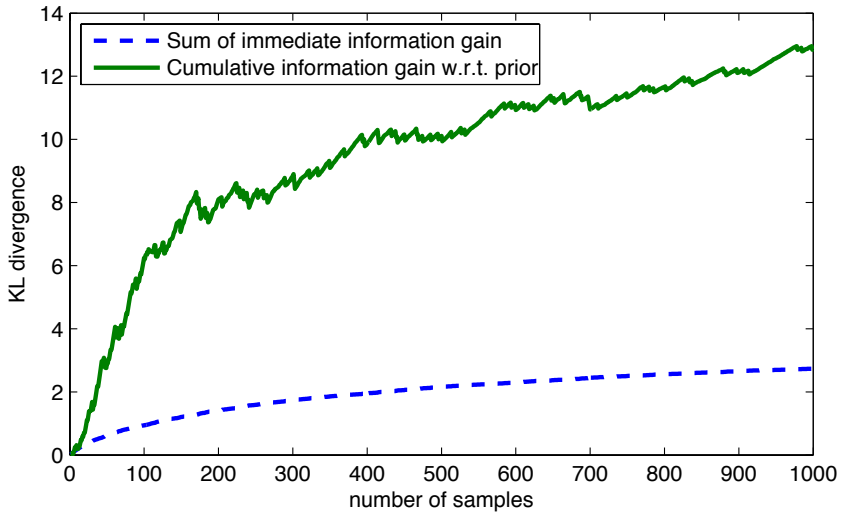

Figure 5.1. Illustration of the difference between the sum of one-step information gain and the cumulative information gain with respect to the prior. In this case, 1000 independent samples are generated from a distribution over finite sample space $\{1,2,3\}$, with $p(x=1)=0.1$, $p(x=2)=0.5$, and $p(x=3)=0.4$. The task of learning is to recover the mass function from the samples, assuming a Dirichlet prior $Dir\left(\frac{50}{3},\frac{50}{3},\frac{50}{3}\right)$. The KL divergence between two Dirichlet distributions are computed according to Penny [2001]. It is clear from the graph that the cumulative information gain fluctuates when the number of samples increases, while the sum of the one-step information gain increases monotonically. It also shows that the difference between the two quantities can be large.

One may also interpret $q_\pi^{\tau,\gamma}(h,a)$ as a linear combination of curiosity Q-values without the discount,

$$q_\pi^{\tau,\gamma}(h,a)=(1-\gamma)\sum_{t=1}^{\tau}\gamma^{t-1}q_\pi^t(h,a)+\gamma^\tau q_\pi^\tau(h,a),$$

where the curiosity Q-values with larger look-ahead steps are weighed exponentially less.

Extending the result in Proposition 5.1, the optimal discounted curiosity (Q) value can be computed recursively as

$$q^{1,\gamma}(h,a)=g(a\|h),\ v^{1,\gamma}(h)=\max_a q^{1,\gamma}(h,a),$$

and

$$q^{\tau,\gamma}(h,a)=g(a\|h)+\gamma\mathbb{E}_{o|ha}v^{\tau-1,\gamma}(hao),\ v^{\tau,\gamma}(h)=\max_a q^{\tau,\gamma}(h,a),$$

with actions chosen greedily with respect to $q^{\tau,\gamma}(h,a)$.

It is worth pointing out that although introducing the discount enables one to define the curiosity Q-value in infinite time horizon in a number of cases, as can be seen in the following section, it is still possible to construct scenarios where such method fails. Consider an infinite list of bandits. For bandit n, there are n possible outcomes with Dirichlet prior $Dir\left(\frac{1}{n},\ldots,\frac{1}{n}\right)$. The expected information gain of pulling bandit n for the first time is then given by

$$\log n - \psi(2) + \log\left(1+\frac{1}{n}\right) \sim \log n,$$

where $\psi(\cdot)$ is the digamma function. Assume at time t, only the first $e^{e^{2t}}$ bandits are available, then the curiosity Q-value in finite time horizon is always finite. However, since the largest expected information gain grows at speed e^{t^2}, for any given $\gamma > 0$, $q^{\tau,\gamma}$ goes to infinity with increasing τ. This example shows that to make the definition curiosity Q-value in infinite time horizon meaningful, the 'total information content' of the environment (or its growing speed) must be bounded.

5.3 Exploration in Finite Markovian Environment with Dirichlet Priors

In this section, we concentrate on a simple case, where the agent assumes that the environment is Markovian, with finite state and action space. In this case, the agent's belief about the environment is summarized by a collection of *Dirichlet distributions* over the transition probabilities. These assumptions enable us to write down the expected information gain in closed form and reason about the theoretical properties of various exploration strategies. In particular, we show the existence of an optimal exploration strategy assuming an infinite planning horizon with discount, and then provide an approximation of the optimal strategy using dynamic programming, with a performance guarantee.

Formally, let $\mathscr{S} = \{1,\ldots,S\}$ be the space of possible sensory inputs, to which we refer as 'states', and $\mathscr{A} = \{1,\ldots,A\}$ the space of actions. The dynamics of a Markovian environment are fully determined by the transition probability $p(s'|s,a)$, which is to be learned. Initially, the agent assumes for each $\langle s,a\rangle$ a Dirichlet prior over the random variable $\Theta_{s,a}$ corresponding to $p(\cdot|s,a)$. Through time, the agent observes the transitions when performing a at s, and updates its estimate of $\Theta_{s,a}$ through probabilistic inference. Since the Dirichlet distribution is conjugate with multinomial distributions, the posterior is still Dirichlet.

Therefore, at any time, the agent's knowledge about the environment can be fully summarized by a three-dimensional array α ($\alpha_{s,a,s'} > 0, \forall s, a, s'$), such that $Dir\left(\alpha_{s,a,1}, \ldots, \alpha_{s,a,S}\right)$ is the current (prior or posterior) density of $\Theta_{s,a}$. We write $\alpha_{s,a}$ for the vector $\left[\alpha_{s,a,1}, \ldots, \alpha_{s,a,S}\right]$, as well as for $\sum_{s'=1}^{S} \alpha_{s,a,s'}$, and the meaning should be clear from the context. The expected information gain is given by $g(a|h) = g\left(\alpha_{s,a}\right)$ for history h ending with state s. From Penny [2001], $g\left(\alpha_{s,a}\right)$ can be written in closed form:

$$g\left(\alpha_{s,a}\right) = \log \alpha_{s,a} - \psi\left(\alpha_{s,a} + 1\right) - \sum_{s'=1}^{S} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \left[\log \alpha_{s,a,s'} - \psi\left(\alpha_{s,a,s'} + 1\right)\right].$$

For fixed lookahead τ, the curiosity Q-value for a given policy π can be written recursively as

$$q_{\pi,\alpha}^{\gamma,\tau}(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \sum_{a'} \pi_{\alpha}\left(a'|s'\right) q_{\pi,\alpha \triangleleft \langle s,a,s' \rangle}^{\gamma,\tau-1}\left(s', a'\right),$$

where $\pi_{\alpha}(a'|s')$ is the probability of performing a' at s', with the current knowledge summarized by α. The operator $\triangleleft$ is defined so that $\alpha \triangleleft \langle s, a, s' \rangle$ is the same as α, except that the entry indexed by $\langle s, a, s' \rangle$ is increased by 1. Similarly, the optimal curiosity Q-value $q_{\alpha}^{\gamma,\tau}$ is given by

$$q_{\alpha}^{\gamma,\tau}(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q_{\alpha \triangleleft \langle s,a,s' \rangle}^{\gamma,\tau-1}\left(s', a'\right).$$

The following proposition ensures that we can drop τ and deal with the curiosity Q-value with infinite planning horizon, as long as a discount is applied. The proof exploits the contraction property resulting from introducing the discount factor γ, and is detailed in Section 5.6.1.

Proposition 5.2 *Let $\gamma \in [0,1)$, and $g_{\alpha} = \max_s \max_a g\left(\alpha_{s,a}\right)$. We have*

i) $q_{\pi,\alpha}^{\gamma}(s,a) = \lim_{\tau \to \infty} q_{\pi,\alpha}^{\gamma,\tau}(s,a)$ exists for any π, α, s, a. The convergence is uniform in the sense that

$$0 \leq q_{\pi,\alpha}^{\gamma}(s,a) - q_{\pi,\alpha}^{\gamma,\tau}(s,a) \leq \frac{g_{\alpha}}{1-\gamma} \gamma^{\tau}, \ \forall s, a.$$

ii) $q_{\alpha}^{\gamma}(s,a) = \lim_{\tau \to \infty} q_{\alpha}^{\gamma,\tau}(s,a)$ exists for any α, s, a, and $\gamma \in [0,1)$. The convergence is also uniform in the sense that

$$0 \leq q^{\gamma}(s,a) - q_{\alpha}^{\gamma,\tau}(s,a) \leq \frac{g_{\alpha}}{1-\gamma} \gamma^{\tau}, \ \forall s, a.$$

iii) q^{γ}_{α} *is the solution to the infinite recursion*

$$q^{\gamma}_{\alpha}(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q^{\gamma}_{\alpha \triangleleft \langle s,a,s' \rangle}\left(s',a'\right), \tag{5.5}$$

and for any other policy π, $q^{\gamma}_{\alpha}(s,a) \geq q^{\gamma}_{\pi,\alpha}(s,a)$.

5.3.1 Approximation through Dynamic Programming

Proposition 5.2 characterizes the optimal discounted curiosity Q-value assuming infinite planning horizon. However, unlike the finite lookahead case, the infinite recursion (Equation 5.5) is not constructive, hence cannot be used to compute the optimal exploration strategy directly. A natural idea is to approximate this infinite recursion by solving at each time step the following Bellman equation,

$$\tilde{q}^{\gamma}_{\alpha}(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} \tilde{q}^{\gamma}_{\alpha}\left(s',a'\right). \tag{5.6}$$

and then following the greedy policy with respect to $\tilde{q}^{\gamma}_{\alpha}(s,a)$. The intuition here is that after sufficiently long time, q^{γ}_{α} and $q^{\gamma}_{\alpha \triangleleft \langle s,a,s' \rangle}$ should be sufficiently close. Note that Equation 5.6 can be solved by dynamic programming (DP) in time polynomial in S and A.

The central result we establish is that $\tilde{q}^{\gamma}_{\alpha}$ is a good approximation of q^{γ}_{α}. As a consequence, following a greedy policy with respect to $\tilde{q}^{\gamma}_{\alpha}(s,a)$ results in near optimal exploration in finite Markovian environments. The first result bounds the difference between $q^{\gamma}_{\alpha}(s,a)$ and $\tilde{q}^{\gamma}_{\alpha}(s,a)$ in terms of the minimum number of visits over all states and actions.

Proposition 5.3 *Let* $c_{\alpha} = \min_s \min_a \alpha_{s,a}$*, then there is some* $K > 0$ *depending on* S *and* γ *only, such that*

$$\left| q^{\gamma}_{\alpha}(s,a) - \tilde{q}^{\gamma}_{\alpha}(s,a) \right| \leq \frac{K}{c^2_{\alpha}}.$$

Proof (sketch). We proceed in two steps (see detailed proof in Section 5.6.2):

(i) Prove the boundedness of both the information gain $g\left(\alpha_{s,a}\right)$ (Lemma. 5.7), as well as its expected increment (Lemma. 5.8), using properties of polygamma functions established by Alzer [1997] (Section 5.6.2).

(ii) Bound the error between $q^{\gamma}_{\alpha}(s,a)$ and $\tilde{q}^{\gamma}_{\alpha}(s,a)$ by exploiting the contraction property of Equation 5.5 and 5.6, as well as the boundedness of $g\left(\alpha_{s,a}\right)$ (Section 5.6.2).

■

Proposition 5.3 guarantees that the difference between q_α^γ and $\tilde{q}_\alpha^\gamma$ decreases at the rate of c_α^{-2}, irrespective of the environment. However, this alone is not enough to guarantee that $\tilde{q}_\alpha^\gamma$ converges to q_α^γ over time. For example, consider an environment consisting of two disconnected components. In this case, c_α is upper bounded since in one of the components $\alpha_{s,a}$ will never increase. Here we make the following assumption about the connectivity of the environment.

Assumption 5.3 *The environment is finite Markovian with dynamics $p(s'|s,a)$, and the Markov chain with transition kernel*

$$p\left(s'|s\right) = \frac{1}{A}\sum_{a\in\mathscr{A}} p\left(s'|s,a\right)$$

is irreducible.

The first half of the assumption ensures that $\frac{\alpha_{s,a,s'}}{\alpha_{s,a}}$ converges to $p(s'|s,a)$ when $\alpha_{s,a}$ goes to infinity by the Law of Large Numbers. The second half of the assumption implies that it is always possible to navigate from one state to another with positive probability of success. Therefore, if some $g\left(\alpha_{s,a}\right)$ is large, the information is guaranteed to propagate to all the states. Under this assumption, we prove in the following proposition that $\tilde{q}_\alpha^\gamma(s,a)$ converges to $q_\alpha^\gamma(s,a)$ over time.

Proposition 5.4 *Assume 5.3, and assume at time t that the agent acts greedily with respect to $\tilde{q}_{\alpha^t}^\gamma$, where α^t is the array summarizing the posterior at time t, then*

$$\lim_{t\to\infty}\left|\frac{q_{\alpha^t}^\gamma}{\tilde{q}_{\alpha^t}^\gamma} - 1\right| = 0,\ a.s.$$

More precisely, let $c_{\alpha^t} = \min_s \min_a \alpha_{s,a}^t$, then $\lim c_{\alpha^t} = \infty$ a.s., and there is some $K > 0$ depending only on the dynamics of the environment and γ, such that

$$\limsup_{t\to\infty} c_{\alpha^t}\left|\frac{q_{\alpha^t}^\gamma}{\tilde{q}_{\alpha^t}^\gamma} - 1\right| \leq K,\ a.s.$$

Proof (sketch). We proceed in two steps (see detailed proof in Section 5.6.3):

(i) Show that under Assumption 5.3, all action pairs will be visited infinitely often (Lemma 5.12), so $\lim c_{\alpha^t} = \infty$.

(ii) Prove that all $\tilde{q}_{\alpha^t}^\gamma(s,a)$ decay at a rate lower bounded by $O\left(c_{\alpha^t}^{-1}\right)$ (Lemma 5.13).

The result then follows by dividing both sides of the inequality in Proposition 5.3 by $\tilde{q}_\alpha^\gamma(s,a)$. ■

Proposition 5.4 demonstrates that through time the precision of our estimation $\tilde{q}_\alpha^\gamma(s,a)$ will improve, in the sense that the *relative* error with respect to the optimal curiosity Q-value $q_\alpha^\gamma(s,a)$ will diminish. Moreover, the speed of convergence is governed by $c_{\alpha^t}^{-1}$.

5.4 An Illustrative Example

The idea presented in the previous section is illustrated through a simple experiment. The environment is a Markovian environment consisting of two groups of densely connected states (cliques) linked by a long corridor. The agent has two actions allowing it to move along the corridor deterministically, whereas the transition probabilities inside each clique are randomly generated. The agent assumes Dirichlet priors over all transition probabilities, and the goal is to learn the transition model of the environment. In the experiment, each clique consists of 5 states, (states 1 to 5 and states 56 to 60), and the corridor is of length 50 (states 6 to 55). The prior over each transition probability is $Dir\left(\frac{1}{60},\ldots,\frac{1}{60}\right)$.

We compare four different algorithms: i) random exploration, where the agent selects each of the two actions with equal probability at each time step; ii) Q-learning with the immediate information gain $g(ao\|h)$ as the reward; iii) greedy exploration, where the agent chooses at each time step the action maximizing $g(a\|h)$; and iv) the DP approximation presented in the previous section. The discount factors in both ii) and iv) are set to 0.995.

Figure 5.2 shows the typical behavior of the four algorithms. The upper four plots show how the agent moves starting from one clique. Both greedy exploration and DP approximation move back and forth between the two cliques. Random exploration has difficulty moving between the two cliques due to the random walk behavior in the corridor. Q-learning exploration, however, gets stuck in the initial clique. The reason for is that since the jump on the corridor is deterministic, the information gain decreases to virtually zero after only several attempts, therefore the Q-value of jumping into the corridor becomes much lower than the Q-value of jumping inside the clique. The bottom plot shows how the cumulative information gain grows over time, and how the DP approximation clearly outperforms the other algorithms, particularly in the early phase of exploration.

5.5 Discussion

The main contribution of the chapter is twofold: First, we formulated information theoretic exploration, a sound framework for Bayesian model learning. Second, for the special case of finite Markovian environment with Dirichlet priors over transition probabilities, we prove that the optimal exploration strategy can be approximated well by solving a sequence of MDP planning problems.

These results, at least partially, provide theoretical justifications for a number of similar, heuristic approaches, where the information gain is regarded as the reward to be optimized by reinforcement learning to enable efficient model learning [see e.g. Schmidhuber, 2010, for a survey]. The algorithm proposed in Section 5.3.1, aside from its theoretical value, may also be of practical use, particularly when the state space is not too large and the benefit from reducing the number of interactions with the environment is significant. The algorithm may also serve as a baseline for assessing the efficiency of other model learning algorithms, thanks to its theoretical optimality.

Our work also gives rise to a number of interesting open questions. For example, it is unclear how the DP-type algorithm in Section 5.3.1 can be extended to other classes of environments, in particular non-Markovian environments, while preserving the near optimality. In addition, our algorithm resembles closely to optimistic initialization [see e.g., Strehl et al., 2006; Kolter and Ng, 2009b] used in model-free RL (with zero external reward everywhere), as both the immediate information gain in our approach and the value function used in optimistic initialization decay at the rate $O\left(n_{s,a}^{-1}\right)$, where $n_{s,a}$ is the number of times action a is performed at state s. Establishing further theoretical connections may help to deepen the understanding of both approaches.

5.6 Proofs

We provide proofs for Proposition 5.2, 5.3 and 5.4.

5.6.1 Proof of Proposition 5.2

To prove Proposition 5.2, we start with the following two technical Lemmas which bound the difference between curiosity Q-values with different look-ahead steps.

Lemma 5.1 $q_\pi^{\tau+1,\gamma}(h,a) - q_\pi^{\tau,\gamma}(h,a) = \gamma^\tau \mathbb{E}_{oa_1\cdots o_{\tau-1}a_\tau|ha} g\left(a_\tau \| h\cdots o_{\tau-1}\right)$.

Proof. Expand $q_\pi^{\tau,\gamma}$ and $q_\pi^{\tau+1,\gamma}$,

$$\begin{aligned} q_\pi^{\tau+1,\gamma} - q_\pi^{\tau,\gamma} &= (1-\gamma)\sum_{t=1}^{\tau+1}\gamma^{t-1}q_\pi^t + \gamma^{\tau+1}q_\pi^{\tau+1} \\ &\quad - (1-\gamma)\sum_{t=1}^{\tau}\gamma^{t-1}q_\pi^t - \gamma^\tau q_\pi^\tau \\ &= \gamma^\tau\left(q_\pi^{\tau+1} - q_\pi^\tau\right). \end{aligned}$$

By definition,

$$\begin{aligned} q_\pi^{\tau+1} - q_\pi^\tau &= \mathbb{E}_{o|ha}\mathbb{E}_{a_1o_1\cdots a_\tau o_\tau|hao}g\left(haoa_1o_1\cdots a_\tau o_\tau\|h\right) \\ &\quad - \mathbb{E}_{o|ha}\mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}|hao}g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|h\right) \\ &= \mathbb{E}_{o|ha}\mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}|hao} \\ &\quad \left[\mathbb{E}_{a_\tau o_\tau|h\cdots o_{\tau-1}}g\left(haoa_1o_1\cdots a_\tau o_\tau\|h\right) - g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|h\right)\right]. \end{aligned}$$

Using Equation 5.2,

$$\begin{aligned} &\mathbb{E}_{a_\tau o_\tau|h\cdots o_{\tau-1}}g\left(haoa_1o_1\cdots a_\tau o_\tau\|h\right) - g\left(haoa_1o_1\cdots a_{\tau-1}o_{\tau-1}\|h\right) \\ &= \mathbb{E}_{a_\tau|h\cdots o_{\tau-1}}g\left(a_\tau\|h\cdots o_{\tau-1}\right), \end{aligned}$$

thus

$$\begin{aligned} q_\pi^{\tau+1,\gamma} - q_\pi^{\tau,\gamma} &= \gamma^\tau\mathbb{E}_{o|ha}\mathbb{E}_{a_1o_1\cdots a_{\tau-1}o_{\tau-1}|hao}\mathbb{E}_{a_\tau|h\cdots o_{\tau-1}}g\left(a_\tau\|h\cdots o_{\tau-1}\right) \\ &= \gamma^\tau\mathbb{E}_{oa_1\cdots o_{\tau-1}a_\tau|ha}g\left(a_\tau\|h\cdots o_{\tau-1}\right). \end{aligned}$$

■

Lemma 5.2 $q^{\tau+1,\gamma}(h,a) - q^{\tau,\gamma}(h,a) \leq \gamma^\tau\mathbb{E}_{o|ha}\max_{a_1}\mathbb{E}_{o_1|haoa_1}\cdots\max_{a_\tau}g\left(a_\tau\|h\cdots o_{\tau-1}\right)$.

Proof. Expand $q^{\tau,\gamma}$ and $q^{\tau+1,\gamma}$, and note that $\max X - \max Y \leq \max|X-Y|$, then

$$\begin{aligned} &q^{\tau+1,\gamma}(h,a) - q^{\tau,\gamma}(h,a) \\ &= \mathbb{E}_{o|ha}\max_{a_1}\mathbb{E}_{o_1|haoa_1}\cdots\max_{a_\tau}\left[g\left(a\|h\right) + \gamma g\left(a_1\|hao\right) + \cdots + \gamma^\tau g\left(a_\tau\|h\cdots o_{\tau-1}\right)\right] \\ &- \mathbb{E}_{o|ha}\max_{a_1}\mathbb{E}_{o_1|haoa_1}\cdots\max_{a_{\tau-1}}\left[g\left(a\|h\right) + \gamma g\left(a_1\|hao\right) + \cdots + \gamma^{\tau-1} g\left(a_{\tau-1}\|h\cdots o_{\tau-2}\right)\right] \\ &\leq \mathbb{E}_{o|ha}\max_{a_1}\{\mathbb{E}_{o_1|haoa_1}\cdots\max_{a_\tau}\left[g\left(a\|h\right) + \gamma g\left(a_1\|hao\right) + \cdots + \gamma^\tau g\left(a_\tau\|h\cdots o_{\tau-1}\right)\right] \\ &- \mathbb{E}_{o_1|haoa_1}\cdots\max_{a_{\tau-1}}\left[g\left(a\|h\right) + \gamma g\left(a_1\|hao\right) + \cdots + \gamma^{\tau-1} g\left(a_{\tau-1}\|h\cdots o_{\tau-2}\right)\right]\} \\ &\leq \cdots \\ &\leq \gamma^\tau\mathbb{E}_{o|ha}\max_{a_1}\mathbb{E}_{o_1|haoa_1}\cdots\max_{a_\tau}g\left(a_\tau\|h\cdots o_{\tau-1}\right). \end{aligned}$$

■

It can be seen that if $\mathbb{E}_{oa_1\cdots o_{\tau-1}a_\tau|ha} g\left(a_\tau \| h\cdots o_{\tau-1}\right)$ is bounded, then both $q_\pi^{\gamma,\tau}$ and $q^{\gamma,\tau}$ are Cauchy sequences with respect to τ.

The following three Lemmas establish the three statements in Proposition 5.2.

Lemma 5.3 *$q_{\pi,\alpha}^{\gamma}(s,a) = \lim_{\tau\to\infty} q_{\pi,\alpha}^{\gamma,\tau}(s,a)$ exists for any π, α, s, a, and $\gamma \in [0,1)$. Moreover, the convergence is uniform with respect to $\langle s,a\rangle$ in the sense that*

$$0 \le q_{\pi,\alpha}^{\gamma}(s,a) - q_{\pi,\alpha}^{\gamma,\tau}(s,a) \le \frac{g_\alpha}{1-\gamma}\gamma^\tau,\ \forall s,a$$

where $g_\alpha = \max_s \max_a g\left(\alpha_{s,a}\right)$.

Proof. Rewrite the result in Lemma 5.1 in this context:

$$\begin{aligned} q_{\pi,\alpha}^{\gamma,\tau+1}(s,a) - q_{\pi,\alpha}^{\gamma,\tau}(s,a) &= \gamma^\tau \mathbb{E}_{oa_1\cdots o_{\tau-1}a_\tau|ha} g\left(a_\tau \| h\cdots o_{\tau-1}\right) \\ &= \gamma^\tau \mathbb{E}_{s_1a_2s_2\cdots s_\tau a_{\tau+1}|ha} g\left(\alpha'_{s_\tau a_{\tau+1}}\right), \end{aligned}$$

where

$$\alpha' = \alpha \triangleleft \langle s,a,s_1\rangle \triangleleft \langle s_1,a_2,s_2\rangle \triangleleft \cdots \triangleleft \langle s_{\tau-1},a_\tau,s_\tau\rangle .$$

Because $g\left(\alpha'_{s,a}\right)$ depends only on the transitions when performing a at s, and all such transitions are exchangeable since they are assumed to be i.i.d. when $\Theta_{s,a}$ is given, one can rewrite the expectation in the following form:

$$\mathbb{E}_{s_1a_2s_2\cdots s_\tau a_{\tau+1}|ha} g\left(\alpha'_{s_\tau a_{\tau+1}}\right) = \mathbb{E}_s \mathbb{E}_a \mathbb{E}_n \mathbb{E}_{x_1} \cdots \mathbb{E}_{x_n} g\left(\alpha'_{s,a}\right).$$

The first and second expectations are taken over the possible final state-action pairs $\langle s_\tau, a_{\tau+1}\rangle$, from which $g\left(\alpha'_{s_\tau a_{\tau+1}}\right)$ is computed. Once $\langle s,a\rangle$ is fixed, the third expectation is taken over the time n that $\langle s,a\rangle$-pair appears in the trajectory $sas_1a_2\cdots s_\tau$, i.e., the time that transitions starting from s with action a occurs. The rest of the expectations are over the n destinations of the transitions, denoted as $x_1,\cdots,x_n$. By definition, $Dir\left(\alpha'_{s,a}\right)$ is the posterior distribution after seeing $x_1,\cdots,x_n$, and $g\left(\alpha'_{s,a}\right)$ is the expected information gain of seeing the outcome of the $(n+1)$-th transition, which we denote x_{n+1}, thus

$$g\left(\alpha'_{s,a}\right) = I\left(\Theta_{s,a}; X_{n+1} | x_1,\ldots,x_n\right),$$

and

$$\mathbb{E}_{x_1}\cdots\mathbb{E}_{x_n} g\left(\alpha'_{s,a}\right) = I\left(\Theta_{s,a}; X_{n+1} | X_1,\ldots,X_n\right).$$

Note that $X_1, \cdots, X_{n+1}$ are i.i.d. given $\Theta_{s,a}$, therefore

$$\begin{aligned}
&I\left(\Theta_{s,a}; X_{n+1} | X_1, \ldots, X_n\right) \\
&= I\left(\Theta_{s,a}; X_1, \ldots, X_{n+1}\right) - I\left(\Theta_{s,a}; X_1, \ldots, X_n\right) \\
&= H\left(X_1, \ldots, X_{n+1}\right) - \sum_{i=1}^{n+1} H\left(X_i | \Theta\right) - H\left(X_1, \ldots, X_n\right) + \sum_{i=1}^{n} H\left(X_i | \Theta\right) \\
&= H\left(X_{n+1} | X_1, \ldots, X_n\right) - H\left(X_{n+1} | \Theta\right) \\
&\leq H\left(X_{n+1}\right) - H\left(X_{n+1} | \Theta\right) \\
&= I\left(\Theta; X_{n+1}\right) \\
&= I\left(\Theta; X_1\right).
\end{aligned}$$

This means that $I\left(\Theta_{s,a}; X_{n+1} | X_1, \ldots, X_n\right)$ is upper bounded by $I\left(\Theta; X_1\right)$, which is the expected information gain of seeing the outcome of the transition for the first time. By definition $I\left(\Theta; X_1\right) = g\left(\alpha_{s,a}\right)$, and it follows that

$$\mathbb{E}_n \mathbb{E}_{x_1} \cdots \mathbb{E}_{x_n} g\left(\alpha'_{s,a}\right) \leq g\left(\alpha_{s,a}\right).$$

Therefore,

$$\begin{aligned}
q_{\pi,\alpha}^{\gamma,\tau+1}(s,a) - q_{\pi,\alpha}^{\gamma,\tau}(s,a) &= \gamma^\tau \mathbb{E}_{s_1 a_2 s_2 \cdots s_\tau a_{\tau+1} | ha} g\left(\alpha'_{s_\tau a_{\tau+1}}\right) \\
&= \gamma^\tau \mathbb{E}_s \mathbb{E}_a \mathbb{E}_n \mathbb{E}_{x_1} \cdots \mathbb{E}_{x_n} g\left(\alpha'_{s,a}\right) \\
&\leq \gamma^\tau \mathbb{E}_s \mathbb{E}_a g\left(\alpha_{s,a}\right) \\
&\leq \gamma^\tau \max_s \max_a g\left(\alpha_{s,a}\right) \\
&\leq \gamma^\tau g_\alpha.
\end{aligned}$$

Since g_α depends on α only, for any T

$$q_{\pi,\alpha}^{\gamma,\tau+T}(s,a) - q_{\pi,\alpha}^{\gamma,\tau}(s,a) \leq \frac{g_\alpha}{1-\gamma} \gamma^\tau.$$

This means that $q_{\pi,\alpha}^{\gamma,\tau}(s,a)$ is a Cauchy sequence with respect to τ, thus $\lim_{\tau\to\infty} q_{\pi,\alpha}^{\gamma,\tau}(s,a)$ exists. Also note that the convergence is uniform since g_α does not depend on $\langle s, a \rangle$. ■

Lemma 5.4 *$q_\alpha^\gamma(s,a) = \lim_{\tau\to\infty} q_\alpha^{\gamma,\tau}(s,a)$ exists for any α, s, a, and $\gamma \in [0,1)$. Also the convergence is uniform in the sense that*

$$0 \leq q^\gamma(s,a) - q_\alpha^{\gamma,\tau}(s,a) \leq \frac{g_\alpha}{1-\gamma} \gamma^\tau.$$

Proof. Rewrite the result in Lemma 5.2,

$$q_\alpha^{\gamma,\tau+1}(s,a) - q_\alpha^{\gamma,\tau}(s,a) \leq \gamma^\tau \mathbb{E}_{s_1|sa} \max_{a_2} \mathbb{E}_{s_2|s_1 a_2, \alpha \triangleleft \langle s,a,s_1 \rangle} \\ \cdots \mathbb{E}_{s_\tau|s_{\tau-1}a_\tau, \alpha \triangleleft \langle s a s_1 \cdots s_{\tau-1} \rangle} \max_{a_{\tau+1}} g\left(\alpha'_{s_\tau, \alpha_{\tau+1}}\right).$$

Since the max operator is only over actions, the proof in the previous proposition still holds, so

$$q_\alpha^{\gamma,\tau+1}(s,a) - q_\alpha^{\gamma,\tau}(s,a) \leq \gamma^\tau g_\alpha,$$

and the result follows. ■

The next proposition shows that q_α^γ is the solution to the infinite recursion.

Lemma 5.5 *q_α^γ is the solution to the recursion*

$$q_\alpha^\gamma(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q^\gamma_{\alpha \triangleleft \langle s,a,s' \rangle}(s',a'),$$

and for any other policy π, $q_\alpha^\gamma(s,a) \geq q_{\pi,\alpha}^\gamma(s,a)$.

Proof. To see that q_α^γ is the solution, taking any $\varepsilon > 0$, one can find a τ such that $\left|q_\alpha^{\gamma,\tau+1} - q_\alpha^\gamma\right| < \frac{\varepsilon}{2}$, and $\left|q^{\gamma,\tau}_{\alpha \triangleleft \langle s,a,s' \rangle} - q^\gamma_{\alpha \triangleleft \langle s,a,s' \rangle}\right| < \frac{\varepsilon}{2}$ for any $\langle s,a,s' \rangle$, thanks to the fact that there are only finite number of $\langle s,a,s' \rangle$ triples, and the convergence from $q_\alpha^{\gamma,\tau+1}(s,a)$ to $q_\alpha^\gamma(s,a)$ is uniform. It follows that

$$\begin{aligned} &\left| g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q^\gamma_{\alpha \triangleleft \langle s,a,s' \rangle}(s',a') - q_\alpha^\gamma(s,a) \right| \\ &\leq \left|q_\alpha^{\gamma,\tau+1} - q_\alpha^\gamma\right| + \gamma \left|q^{\gamma,\tau}_{\alpha \triangleleft \langle s,a,s' \rangle} - q^\gamma_{\alpha \triangleleft \langle s,a,s' \rangle}\right| \\ &< \varepsilon. \end{aligned}$$

Since ε is chosen arbitrary, $q_\alpha^\gamma(s,a)$ must be the solution of the infinite recursion.

The fact that $q_\alpha^\gamma(s,a) \geq q_{\pi,\alpha}^\gamma(s,a)$ follows from the fact that $q_\alpha^{\gamma,\tau}$ and $q_{\pi,\alpha}^{\gamma,\tau}$ are monotonically increasing on τ (by Lemma 5.1), and $q_\alpha^{\gamma,\tau} \geq q_{\pi,\alpha}^{\gamma,\tau}$ for any given τ and π. ■

Combining Lemma 5.3, 5.4, and 5.5 gives the proof of Proposition 5.2.

5.6.2 Proof of Proposition 5.3

The proof proceeds in two steps. First, we establish necessary properties for the expected information gain of a Dirichlet distribution.

A note on notations: The operator $\triangleleft$ is defined so that $\alpha \triangleleft \langle s,a,s' \rangle$ is the same as α, except that the entry indexed by $\langle s,a,s' \rangle$ is increased by 1. We also assume that $\triangleleft$ is right associative, so one can write $\alpha \triangleleft \langle s_1,a_2,s_2 \rangle \triangleleft \langle s_2,a_3,s_3 \rangle \cdots$, or simply $\alpha \triangleleft \langle s_1 a_2 s_2 a_3 s_3 \cdots \rangle$. Also, we drop the superscript γ so that $q^{\gamma,\tau}_{\pi,\alpha}$ and $q^{\gamma,\tau}_{\alpha}$ are written simply as $q^{\tau}_{\pi,\alpha}$ and q^{τ}_{α}. This should not be confused with the no-discount case.

Properties of Expected Information Gain in Dirichlet Case

The expected information gain of a Dirichlet distribution $Dir(n)$, where $n = [n_1,\ldots,n_S]$ is given by

$$g(n) = \log(n) - \psi(n+1) - \sum_s \frac{n_s}{n} \left[\log(n_s) - \psi(n_s+1)\right].$$

Define

$$f(x) = x\left[\psi(x+1) - \log x\right] = 1 - x\left[\log x - \psi(x)\right].$$

then

$$g(n) = \frac{1}{n}\left[\sum_s f(n_s) - f(n)\right].$$

The following important properties have been proved by Alzer [1997].

Theorem 5.1 *f has the following properties[1]:*

a) $\lim_{x\to 0} f(x) = 0$, $\lim_{x\to\infty} f(x) = \frac{1}{2}$

b) f is strictly completely monotonic, in the sense that

$$(-1)^{n+1}\frac{d^n f(x)}{dx^n} > 0.$$

In particular, Theorem 5.1 shows that f is strictly monotonically increasing, and also strictly concave. The following Lemma summarizes the properties about f to be used.

Lemma 5.6 *Define $\delta_m(x) = f(x+m) - f(x)$ for $m > 0$. Then*

a) f is sub-additive, i.e., $f(x) + f(y) > f(x+y)$ for $x, y > 0$

[1] Alzer's original paper considers the function $x\left(\log x - \psi(x)\right) = 1 - f(x)$. Here the statements are modified accordingly.

b) $\delta_m(x)$ *is monotonically decreasing on* $(0,\infty)$.

c) $0 < x\delta_m(x) < \left(1-e^{-1}\right) m$ *for* $x \in (0,\infty)$.

Proof. a) Note that $g(n)$ is mutual information, and the unknown observation depends on the parameters of the distribution, therefore $g(n) > 0$, and

$$0 < g\left(\left[x,y\right]\right) = \frac{1}{x+y}\left[f(x)+f\left(y\right)-f\left(x+y\right)\right].$$

b) Note that

$$\delta_m(x) = \int_0^m f'(x+s)\,ds,$$

and the result follows from $f''(x) < 0$.

c) Clearly, $x\delta_m(x) > 0$ because f is strictly increasing. From Intermediate Value Theorem, there is some $\delta \in (0,m)$, such that

$$\begin{aligned} x\delta_m(x) &= x\left[f(x+m)-f(x)\right] \\ &= mxf'(x+\delta) \\ &= mxf'(x)+mx\left[f'(x+\delta)-f'(x)\right] \\ &< mxf'(x). \end{aligned}$$

The inequality is because f is strictly concave.

From Alzer [1997],

$$f(x) = 1 - x\int_0^\infty \phi(t)\,e^{-tx}dt,$$

where

$$\phi(t) = \frac{1}{1-e^{-t}} - \frac{1}{t}$$

is strictly increasing, with $\lim_{t\to 0}\phi(t)=\frac{1}{2}$ and $\lim_{t\to\infty}\phi(t)=1$. Therefore,

$$\begin{aligned}
xf'(x) &= x\int_0^\infty \phi(t)e^{-tx}(xt-1)\,dt \\
&= x^2\int_0^{\frac{1}{x}} \phi(t)e^{-tx}\left(t-\frac{1}{x}\right)dt + x^2\int_{\frac{1}{x}}^\infty \phi(t)e^{-tx}\left(t-\frac{1}{x}\right)dt \\
&< x^2\phi(0)\int_0^{\frac{1}{x}} e^{-tx}\left(t-\frac{1}{x}\right)dt + x^2\phi(\infty)\int_{\frac{1}{x}}^\infty e^{-tx}\left(t-\frac{1}{x}\right)dt \\
&= \frac{x^2}{2}\int_0^\infty e^{-tx}\left(t-\frac{1}{x}\right)dt + x^2\int_{\frac{1}{x}}^\infty e^{-tx}tdt - x\int_{\frac{1}{x}}^\infty e^{-tx}dt \\
&< \frac{x^2}{2}\int_0^\infty e^{-tx}\left(t-\frac{1}{x}\right)dt + x^2\int_0^\infty e^{-tx}tdt - x\int_{\frac{1}{x}}^\infty e^{-tx}dt.
\end{aligned}$$

Note that

$$\int_0^\infty e^{-tx}tdt=\frac{1}{x^2},\ \int_0^\infty e^{-tx}dt=\frac{1}{x},$$

and

$$x\int_{\frac{1}{x}}^\infty e^{-tx}dt=e^{-1},$$

it follows that

$$xf'(x)<1-\frac{1}{e}.$$

■

The properties of f guarantee that $g(n)$ decreases at the rate of $\frac{1}{n}$. The result is formulated in the following Lemma.

Lemma 5.7 *Let $Dir\left(n_1^0,\cdots,n_S^0\right)$ and $Dir\left(n_1^t,\cdots,n_S^t\right)$ be the prior and the posterior distribution, such that $n^t=n^0+t$. Let $s^*=\arg\max_s n_s^0$. Then*

$$\frac{\sum_{s\neq s^*} f\left(n_s^0\right)-f\left(\sum_{s\neq s^*} n_s^0\right)}{n^t} < g\left(n^t\right) < \frac{S-1}{2n^t}.$$

Proof. The upper bound is because $0<f(x)<\frac{1}{2}$ and f is increasing, thus

$$\begin{aligned}
\sum_s f\left(n_s^t\right)-f\left(n^t\right) &= f\left(n_1^t\right)-f\left(n^t\right)+\sum_{s\neq 1} f\left(n_s^t\right) \\
&< \frac{S-1}{2}.
\end{aligned}$$

The lower bound follows from the fact that $f(x+m)-f(x)$ is decreasing. We show that the trajectory minimizing $g(n^t)$ is the one such that all t observations equal to s^*. To see this, let m_s be the number of times observing $s \neq s^*$, then

$$\begin{aligned} f\left(n_s^0+m_s\right)+f\left(n_{s^*}^0+m_{s^*}\right) = f\left(n_s^0\right)+f\left(n_{s^*}^0+m_{s^*}+m_s\right) \\ +f\left(n_s^0+m_s\right)-f\left(n_s^0\right) \\ -\left(f\left(n_{s^*}^0+m_{s^*}+m_s\right)-f\left(n_{s^*}^0+m_{s^*}\right)\right). \end{aligned}$$

Note that $n_{s^*}^0+m_{s^*} \geq n_s^0$, so

$$f\left(n_s^0+m_s\right)+f\left(n_{s^*}^0+m_{s^*}\right) \geq f\left(n_s^0\right)+f\left(n_{s^*}^0+m_{s^*}+m_s\right).$$

Now assume the observations are all s^*, from sub-additivity,

$$\begin{aligned} &\sum_s f\left(n_s^t\right)-f\left(n^t\right) \\ &=\sum_{s\neq s^*} f\left(n_s^0\right)+f\left(n_{s^*}^0+t\right)-f\left(n^0+t\right) \\ &=\sum_{s\neq s^*} f\left(n_s^0\right)-f\left(\sum_{s\neq s^*} n_s^0\right)+\left[f\left(\sum_{s\neq s^*} n_s^0\right)+f\left(n_{s^*}^0+t\right)-f\left(n^0+t\right)\right] \\ &>\sum_{s\neq s^*} f\left(n_s^0\right)-f\left(\sum_{s\neq s^*} n_s^0\right). \end{aligned}$$

■

Remark 5.1 *The bounds hold irrespective of the data generating process, namely, it holds for* any *sequences of observations,* including sequences with zero probabilities.

The following Lemmas bound the variation of the expected information gain, when one single observation is added.

Lemma 5.8 *Let* $n=\left[n_1,\ldots,n_S\right]$ *and* $n'=\left[n_1,\ldots,n_{s-1},n_s+1,n_{s+1},\ldots,n_S\right]$, *then*

$$\left|g\left(n'\right)-g\left(n\right)\right| \leq \frac{S}{nn_s},\ \forall n_s>0.$$

Proof. Without loss of generality let $s = 1$. Note that

$$
\begin{aligned}
&\frac{n_1}{n}\left[g\left(n'\right)-g\left(n\right)\right] \\
&=\frac{n_1}{n}\left\{\frac{1}{n+1}\left[\sum_{s\neq 1} f\left(n_s\right)+f\left(n_1+1\right)-f\left(n+1\right)\right]-\frac{1}{n}\left[\sum_{s} f\left(n_s\right)-f\left(n\right)\right]\right\} \\
&=\frac{n_1}{n}\left\{\frac{\sum_s f\left(n_s\right)}{n+1}-\frac{f\left(n+1\right)}{n+1}+\frac{f\left(n_1+1\right)-f\left(n_1\right)}{n+1}+\frac{f\left(n\right)}{n}-\frac{\sum_s f\left(n_s\right)}{n}\right\} \\
&=\frac{n_1}{n}\left\{-\frac{f\left(n+1\right)}{n+1}-\frac{\sum_s f\left(n_s\right)}{n\left(n+1\right)}+\frac{f\left(n_1+1\right)-f\left(n_1\right)}{n+1}+\frac{f\left(n\right)}{n}\right\} \\
&=\frac{n_1}{n}\left\{-\frac{f\left(n+1\right)}{n+1}-\frac{ng\left(n\right)+f\left(n\right)}{n\left(n+1\right)}+\frac{f\left(n_1+1\right)-f\left(n_1\right)}{n+1}+\frac{f\left(n\right)}{n}\right\} \\
&=\frac{n_1}{n\left(n+1\right)}\left\{-\delta_1\left(n\right)+\delta_1\left(n_1\right)-g\left(n\right)\right\} \\
&=\frac{1}{n\left(n+1\right)}\left\{n_1\delta_1\left(n_1\right)-\frac{n_1}{n}\cdot n\delta_1\left(n\right)-\frac{n_1}{n}\left[\sum_s f\left(n_s\right)-f\left(n\right)\right]\right\}
\end{aligned}
$$

From the Lemma 5.6,

$$0 < x\delta_1\left(x\right) < 1,\ 0 < f\left(x\right) < \frac{1}{2},$$

so

$$-\frac{S}{n^2} < -\frac{n_1}{n}\frac{S+1}{2n^2} < \frac{n_1}{n}\left[g\left(n'\right)-g\left(n\right)\right] < \frac{1}{n^2} < \frac{S}{n^2},$$

and thus

$$\frac{n_1}{n}\left|g\left(n'\right)-g\left(n\right)\right| < \frac{S}{n^2}.$$

■

Bounding the Difference Between q_α and $\tilde{q}_\alpha$

Let $c_\alpha = \min_s \min_a \alpha_{s,a}$, we prove that

$$\left|q_\alpha\left(s,a\right)-\tilde{q}_\alpha\left(s,a\right)\right| \sim \frac{1}{c_\alpha^2}.$$

Lemma 5.9 $q_\alpha\left(s,a\right) \leq \frac{S-1}{2\left(1-\gamma\right)c_\alpha}$.

Proof. From Lemma 5.7, write $K_0 = \frac{S-1}{2}$, then

$$g\left(\alpha_{s,a}\right) < \frac{K_0}{\alpha_{s,a}} \leq \frac{K_0}{c_\alpha}, \forall s, a$$

By definition,

$$\begin{aligned} q_\alpha^{\gamma,2}(s,a) &= g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q_{\alpha \triangleleft \langle s,a,s' \rangle}^{\gamma,1}\left(s', a'\right) \\ &< K_0 \left(\frac{1}{c_\alpha} + \gamma \frac{1}{c_{\alpha \triangleleft \langle s,a,s' \rangle}} \right) \\ &\leq \frac{K_0}{c_\alpha}\left(1+\gamma\right), \end{aligned}$$

since $c_{\alpha \triangleleft \langle s,a,s' \rangle} \geq c_\alpha$. Repeat the process, it follows that for any τ,

$$q_\alpha^{\gamma,\tau}(s,a) \leq \frac{K_0}{c_\alpha}\left(1+\gamma+\cdots+\gamma^{\tau-1}\right) < \frac{K_0}{\left(1-\gamma\right)c_\alpha},$$

thus

$$q_\alpha^{\gamma}(s,a) = \lim_{\tau \to \infty} q_\alpha^{\gamma,\tau}(s,a) \leq \frac{K_0}{\left(1-\gamma\right)c_\alpha} = \frac{S-1}{2\left(1-\gamma\right)c_\alpha}.$$

■

Lemma 5.10 *Let $n = \left[n_1, \cdots, n_S\right]$, and $n' = \left[n_1+1, n_2, \cdots, n_S\right]$. Let $x_1, \cdots, x_S$ be S non-negative numbers. Define $p_s = \frac{n_s}{n}$, $p_1' = \frac{n_1+1}{n+1}$ and $p_s' = \frac{n_s}{n+1}$ for $s = 2, \cdots, S$. Then*

$$\left| \sum_s \left(p_s' - p_s\right) x_s \right| \leq \frac{1}{n_1} \sum_s p_s x_s.$$

Proof. Simple derivation gives

$$\begin{aligned} p_1 \left| \sum_s \left(p_s' - p_s\right) x_s \right| &= \left(\sum_s p_s x_s \right) \cdot p_1 \left| \frac{\sum_s \left(p_s' - p_s\right) x_s}{\sum_s p_s x_s} \right| \\ &\leq \left(\sum_s p_s x_s \right) \max_s \frac{p_1 \cdot \left|p_s' - p_s\right|}{p_s}. \end{aligned}$$

If $s = 1$,

$$\frac{p_1 \cdot \left|p_1' - p_1\right|}{p_1} = \frac{n_1}{n} \cdot \frac{\frac{n_1+1}{n+1} - \frac{n_1}{n}}{\frac{n_1}{n}} = \frac{n-n_1}{n\left(n+1\right)} \leq \frac{1}{n}.$$

If $s \neq 1$,

$$\frac{p_1 \cdot \left|p'_s - p_s\right|}{p_s} = \frac{n_1}{n} \cdot \frac{\frac{n_s}{n} - \frac{n_s}{n+1}}{\frac{n_s}{n}} = \frac{n_1}{n(n+1)} \leq \frac{1}{n}.$$

Therefore,

$$\max_s \frac{p_1 \cdot \left|p'_s - p_s\right|}{p_s} \leq \frac{1}{n},$$

and

$$p_1 \left|\sum_s \left(p'_s - p_s\right) x_s\right| \leq \frac{1}{n} \sum_s p_s x_s.$$

■

Lemma 5.11 *For any α, $s^\dagger$, $a^\dagger$, there is some constant K depending on S and γ only, such that*

$$\sum_s \frac{\alpha_{s^\dagger,a^\dagger,s}}{\alpha_{s^\dagger,a^\dagger}} \max_a \left|q_{\alpha \triangleleft \langle s^\dagger,a^\dagger,s\rangle}(s,a) - q_\alpha(s,a)\right| \leq \frac{K}{c_\alpha^2}.$$

Proof. First change the notations. Let $s_0 = s^\dagger$, $a_0 = a^\dagger$. Also let $\alpha^1 = \alpha$, $\beta^1 = \alpha \triangleleft \left\langle s^\dagger, a^\dagger, s \right\rangle$. The result to prove becomes

$$\sum_{s_1} \frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \left|q_{\beta^1}(s_1,a_1) - q_{\alpha^1}(s_1,a_1)\right| \leq \frac{K}{c_\alpha^2}.$$

Consider the finite time horizon approximations of q_{β^1} and q_{α^1}, namely $q^{\gamma,\tau}_{\beta^1}$ and $q^{\gamma,\tau}_{\alpha^1}$. With a little abuse of notation, we drop the superscript γ in this proof. Note that this shall not be confused with the finite time horizon curiosity Q-values without discount.

For $\tau = 2$, consider the following inequality:

$$\begin{aligned}
&\frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \left|q^2_{\beta^1}(s_1,a_1) - q^2_{\alpha^1}(s_1,a_1)\right| \\
&\leq \frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \left|g\left(\beta^1_{s_1,a_1}\right) - g\left(\alpha^1_{s_1,a_1}\right)\right| \\
&+ \gamma \frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \left|\sum_{s_2} \left(\frac{\beta^1_{s_1 a_1 s_2}}{\beta^1_{s_1 a_1}} - \frac{\alpha^1_{s_1 a_1 s_2}}{\alpha^1_{s_1 a_1}}\right) \max_{a_2} q^1_{\beta^2}(s_2,a_2)\right| \\
&+ \gamma \frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \sum_{s_2} \frac{\alpha^1_{s_1 a_1 s_2}}{\alpha^1_{s_1 a_1}} \max_{a_2} \left|q^1_{\beta^2}(s_2,a_2) - q^1_{\alpha^2}(s_2,a_2)\right|.
\end{aligned}$$

Here $\beta^2 = \beta^1 \triangleleft \langle s_1 a_1 s_2 \rangle$, $\alpha^2 = \alpha^1 \triangleleft \langle s_1 a_1 s_2 \rangle$. Note that the error between $q^2_{\beta^1}$ and $q^2_{\alpha^1}$ has been decomposed into three terms. The first term captures the difference between the immediate information gain, the second term captures the error between transition probabilities, and the third term is of the same form as the left side of the inequality, except τ is decreased by 1. To simplify the notation, let $\mathbb{F}^t$ be the operator

$$\mathbb{F}^t [\cdots] = \sum_{s_t} \frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}} \max_{a_t} [\cdots].$$

For fixed τ, let

$$\begin{aligned}\delta_t = &\left| g\left(\beta^t_{s_t,a_t}\right) - g\left(\alpha^t_{s_t,a_t}\right)\right| \\ &+ \gamma \left| \sum_{s_{t+1}} \left(\frac{\beta^t_{s_t a_t s_{t+1}}}{\beta^t_{s_t a_t}} - \frac{\alpha^t_{s_t a_t s_{t+1}}}{\alpha^t_{s_t a_t}} \right) \max_{a_{t+1}} q^{\tau-t}_{\beta^{t+1}}\left(s_{t+1}, a_{t+1}\right) \right|,\end{aligned}$$

and

$$\phi_t = q^{\tau+1-t}_{\beta^t}\left(s_t, a_t\right) - q^{\tau+1-t}_{\alpha^t}\left(s_t, a_t\right).$$

One can write

$$\mathbb{F}^1 \phi_1 \leq \mathbb{F}^1 \delta_1 + \gamma \mathbb{F}^1 \mathbb{F}^2 \phi_2.$$

Repeat this process for general τ, it follows that

$$\begin{aligned}\mathbb{F}^1 \phi_1 &\leq \mathbb{F}^1 \delta_1 + \gamma \mathbb{F}^1 \mathbb{F}^2 \phi_2 \\ &\leq \mathbb{F}^1 \delta_1 + \gamma \mathbb{F}^1 \left[\mathbb{F}^2 \delta_2 + \gamma \mathbb{F}^2 \mathbb{F}^3 \phi_3 \right] \\ &= \mathbb{F}^1 \delta_1 + \gamma \mathbb{F}^1 \mathbb{F}^2 \delta_2 + \gamma^2 \mathbb{F}^1 \mathbb{F}^2 \mathbb{F}^3 \phi_3 \\ &= \cdots \\ &= \mathbb{F}^1 \delta_1 + \gamma \mathbb{F}^1 \mathbb{F}^2 \delta_2 + \cdots + \gamma^{t-1} \mathbb{F}^1 \cdots \mathbb{F}^t \delta_t + \cdots + \gamma^{\tau-2} \mathbb{F}^1 \cdots \mathbb{F}^{\tau-1} \delta_{\tau-1} \\ &\quad + \gamma^{\tau-1} \mathbb{F}^1 \cdots \mathbb{F}^\tau \phi_\tau.\end{aligned}$$

Now look at a particular term in the inequality above, for example,

$$\mathbb{F}^1 \cdots \mathbb{F}^t \delta_t = \sum_{s_1} \frac{\alpha^1_{s_0 a_0 s_1}}{\alpha^1_{s_0 a_0}} \max_{a_1} \cdots \sum_{s_t} \frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}} \max_{a_t} \delta_t.$$

Note that if $\langle s_t, a_t \rangle \neq \left\langle s^\dagger, a^\dagger \right\rangle$ then $\delta_t = 0$, since β^t and α^t differ only in the entry indexed by $\left\langle s^\dagger, a^\dagger, s_1 \right\rangle$. The following discussion assumes that $\langle s_t, a_t \rangle = \left\langle s^\dagger, a^\dagger \right\rangle$. From Lemma 5.8, let $K_1 = S$, then

$$\left| g\left(\beta^t_{s_t,a_t}\right) - g\left(\alpha^t_{s_t,a_t}\right)\right| \leq \frac{K_1}{\alpha^t_{s_t,a_t} \alpha^t_{s_t,a_t,s_1}}.$$

From Lemma 5.9 and 5.10, there is some K_2 that depends only on S and γ, such that

$$\begin{aligned}
&\left|\sum_{s_{t+1}}\left(\frac{\beta^t_{s_t a_t s_{t+1}}}{\beta^t_{s_t a_t}}-\frac{\alpha^t_{s_t a_t s_{t+1}}}{\alpha^t_{s_t a_t}}\right)\max_{a_{t+1}} q^{\tau-t}_{\beta^{t+1}}\left(s_{t+1},a_{t+1}\right)\right| \\
&\leq \frac{1}{\alpha^t_{s_0,a_0,s_1}}\sum_{s_{t+1}}\frac{\alpha^t_{s_t a_t s_{t+1}}}{\alpha^t_{s_t a_t}}\max_{a_{t+1}} q^{\tau-t}_{\beta^{t+1}}\left(s_{t+1},a_{t+1}\right) \\
&\leq \frac{K_2}{\alpha^t_{s_0,a_0,s_1} c_{\alpha^t}},
\end{aligned}$$

where $c_{\alpha^t}=\min_s \min_a \alpha^t_{s,a}$. In combination, there is some K_0 such that

$$\delta_t \leq \frac{K_0}{c_{\alpha^t}\alpha^t_{s^\dagger,a^\dagger,s_1}}.$$

The next step is tricky: Assume that the policy is given, say, it is already the policy maximizing $\mathbb{F}^1\cdots\mathbb{F}^t\delta_t$, so that each a is a deterministic function of the prior α^1 and the previous history. Consider a trajectory $s_0a_0s_1a_1\cdots s_ta_t$, the predictive probability of seeing such a trajectory is given by

$$p\left(s_1a_1\cdots s_ta_t|s_0a_0\right)=\frac{\alpha^1_{s_0a_0s_1}}{\alpha^1_{s_0a_0}}\frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}.$$

Again, if $\langle s_t,a_t\rangle\neq\left\langle s^\dagger,a^\dagger\right\rangle$, then $p\left(s_1a_1\cdots s_ta_t|s_0a_0\right)\delta_t=0$. Otherwise,

$$\begin{aligned}
p\left(s_1a_1\cdots s_ta_t|s_0a_0\right)\delta_t &= \frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}\cdot\left(\frac{\alpha^1_{s_0a_0s_1}}{\alpha^1_{s_0a_0}}\delta_t\right) \\
&\leq \frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}\cdot\frac{K_0}{c_{\alpha^t}\alpha^1_{s_0a_0}}\cdot\frac{\alpha^1_{s_0a_0s_1}}{\alpha^t_{s_0,a_0,s_1}} \\
&\leq \frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}\cdot\frac{K_0}{c_{\alpha^t}\alpha^1_{s_0a_0}} \\
&\leq \frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}\cdot\frac{K_0}{c^2_{\alpha^1}}.
\end{aligned}$$

Note that

$$\frac{\alpha^2_{s_1a_1s_2}}{\alpha^2_{s_1a_1}}\cdots\frac{\alpha^{t-1}_{s_{t-1}a_{t-1}s_t}}{\alpha^{t-1}_{s_{t-1}a_{t-1}}}=p\left(s_2a_2\cdots s_ta_t|s_1a_1\right)$$

is the probability of seeing the trajectory $s_2a_2\cdots s_ta_t$, when the agent assumes prior $\alpha^1 \triangleleft \langle s_0, a_0, s_1\rangle = \alpha^2$, and follows the same policy starting from $\langle s_1, a_1\rangle$. Clearly,

$$\sum_{s_2}\cdots\sum_{s_t} p\left(s_2a_2\cdots s_ta_t|s_1a_1\right) = 1,$$

which leads to

$$\begin{aligned}\mathbb{F}^1\cdots\mathbb{F}^t\delta_t &= \sum_{s_1}\sum_{s_2}\cdots\sum_{s_t} p\left(s_1a_1\cdots s_ta_t|s_0a_0\right)\delta_t\\ &\leq \sum_{s_1}\frac{K_0}{c^2_{\alpha^1}}\sum_{s_2}\cdots\sum_{s_t} p\left(s_2a_2\cdots s_ta_t|s_1a_1\right)\\ &\leq \frac{SK_0}{c^2_{\alpha^1}}.\end{aligned}$$

Putting the equation back, and note that $c^2_{\alpha^1} = c^2_\alpha$ is a constant on α, one has

$$\begin{aligned}\mathbb{F}^1\phi_1 &\leq \mathbb{F}^1\delta_1 + \gamma\mathbb{F}^1\mathbb{F}^2\delta_2 + \cdots + \gamma^{t-1}\mathbb{F}^1\cdots\mathbb{F}^t\delta_t + \cdots + \gamma^{\tau-2}\mathbb{F}^1\cdots\mathbb{F}^{\tau-1}\delta_{\tau-1}\\ &\quad + \gamma^{\tau-1}\mathbb{F}^1\cdots\mathbb{F}^\tau\phi_\tau\\ &\leq \frac{SK_0}{c^2_\alpha}\left(1+\gamma+\cdots+\gamma^{\tau-2}\right) + \gamma^{\tau-1}\mathbb{F}^1\cdots\mathbb{F}^\tau\phi_\tau\\ &\leq \frac{SK_0}{1-\gamma}\frac{1}{c^2_\alpha} + \gamma^{\tau-1}\mathbb{F}^1\cdots\mathbb{F}^\tau\phi_\tau.\end{aligned}$$

From Lemma 5.9, since the curiosity Q-values are bounded, there is some K_3 such that

$$\begin{aligned}\phi_\tau &= \left|q^1_{\beta^t}\left(s_t,a_t\right) - q^1_{\alpha^t}\left(s_t,a_t\right)\right|\\ &\leq \left|q^1_{\beta^t}\left(s_t,a_t\right)\right| + \left|q^1_{\alpha^t}\left(s_t,a_t\right)\right|\\ &\leq \left|q_{\beta^t}\left(s_t,a_t\right)\right| + \left|q_{\alpha^t}\left(s_t,a_t\right)\right|\\ &\leq \frac{K_3}{c_\alpha},\end{aligned}$$

thus

$$\mathbb{F}^1\phi_1 \leq \frac{SK_0}{1-\gamma}\frac{1}{c^2_\alpha} + \gamma^{\tau-1}\frac{K_3}{c_\alpha}.$$

Let $\tau\to\infty$, one has

$$\sum_{s_1}\frac{\alpha^1_{s_0a_0s_1}}{\alpha^1_{s_0a_0}}\max_{a_1}\left|q_{\beta^1}\left(s_1,a_1\right) - q_{\alpha^1}\left(s_1,a_1\right)\right| \leq \frac{K}{c^2_\alpha},$$

where $K = \frac{SK_0}{1-\gamma}$ is a constant depending on S and γ only. ■

Now we are ready to prove Proposition 5.3.

Proof of Proposition 5.3. Write Equation 5.5 into the following form

$$q_\alpha(s,a) = g\left(\alpha_{s,a}\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} q_\alpha\left(s',a'\right) + \gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \left[\max_{a'} q_{\alpha \triangleleft \langle s,a,s' \rangle}\left(s',a'\right) - \max_{a'} q_\alpha\left(s',a'\right)\right].$$

The last term is bounded by

$$\begin{aligned}\delta &= \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \left|\max_{a'} q_{\alpha \triangleleft \langle s,a,s' \rangle}\left(s',a'\right) - \max_{a'} q_\alpha\left(s',a'\right)\right| \\ &\leq \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} \left|q_{\alpha \triangleleft \langle s,a,s' \rangle}\left(s',a'\right) - q_\alpha\left(s',a'\right)\right|.\end{aligned}$$

Apply Lemma 5.11, it follows that there is some constant K_0 depending on S and γ only, such that

$$\sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \max_{a'} \left|q_{\alpha \triangleleft \langle s,a,s' \rangle}\left(s',a'\right) - q_\alpha\left(s',a'\right)\right| \leq \frac{K_0}{c_\alpha^2}.$$

Therefore $\delta \leq \frac{K_0}{c_\alpha^2}$.

Now compare q_α and $\tilde{q}_\alpha$:

$$\begin{aligned}&\max_s \max_a \left|q_\alpha(s,a) - \tilde{q}_\alpha(s,a)\right| \\ &\leq \max_s \max_a \left|\gamma \sum_{s'} \frac{\alpha_{s,a,s'}}{\alpha_{s,a}} \left(\max_{a'} q_\alpha\left(s',a'\right) - \max_{s'} \tilde{q}_\alpha\left(s',a'\right)\right) + \gamma\delta\right| \\ &\leq \gamma \max_s \max_a \left|q_\alpha(s,a) - \tilde{q}_\alpha(s,a)\right| + \frac{\gamma K_0}{c_\alpha^2}.\end{aligned}$$

Therefore

$$\max_s \max_a \left|q_\alpha(s,a) - \tilde{q}_\alpha(s,a)\right| \leq \frac{\gamma K_0}{1-\gamma} \cdot \frac{1}{c_\alpha^2}.$$

Letting $K = \frac{\gamma K_0}{1-\gamma}$ completes the proof. ■

5.6.3 Proof of Proposition 5.4

The proof is unwrapped in three steps. The first step is to show that all state-action pairs will be visited infinitely many times along time.

Lemma 5.12 *Assume 5.3, and the agent chooses the action greedily with respect to $\tilde{q}_{\alpha^t}$, where α^t is the posterior after t time steps. Then for any s, a,*

$$\lim_{t\to\infty} \alpha^t_{s,a} = \infty, \text{ a.s.}$$

Proof. Note that $\alpha^t_{s,a,s'}$ is non-decreasing, and can only increase by one if increasing. Therefore, $\lim_{t\to\infty} \alpha^t_{s,a} < \infty$ implies that there is some $T_{s,a}$ and $c_{s,a}$ such that for all $t > T_{s,a}$, $\alpha^t_{s,a} = c_{s,a}$.

The complement of $\lim_{t\to\infty} \alpha^t_{s,a} = \infty$ for all $\langle s, a\rangle$ is that $\exists \Lambda \subset \mathscr{S} \times \mathscr{A}$, $\Lambda \neq \emptyset$, and $\exists T_{s,a}, c_{s,a}$ for all $\langle s, a\rangle \in \Lambda$, such that $\alpha^t_{s,a} = c_{s,a}$ for all $t > T_{s,a}$. Since there are only finitely many $\langle s, a\rangle$, this can be simplified to $\exists \Lambda \neq \emptyset$, $\exists T$, $\exists c_{s,a}$, such that $\alpha^t_{s,a} = c_{s,a}$ for all $t > T$ and $\langle s, a\rangle \in \Lambda$.

Fix $\Lambda \neq \emptyset$, T and $c_{s,a}$, we show that the event $\alpha^t_{s,a} = c_{s,a}$ for $t > T$ and $\langle s, a\rangle \in \Lambda$ is a null event. Let $\bar{\Lambda} = \mathscr{S} \times \mathscr{A} \backslash \Lambda$, by definition, $\alpha^t_{s,a} \to \infty$ for all $\langle s, a\rangle \in \bar{\Lambda}$. Clearly, $\bar{\Lambda}$ is not empty. Define

$$\mathscr{S}_I = \left\{ s \in \mathscr{S} : \exists a, a'' \text{ such that } \langle s, a\rangle \in \Lambda, \ \langle s, a''\rangle \notin \Lambda \right\}.$$

Namely, $\mathscr{S}_I$ is the 'boundary' between Λ and $\bar{\Lambda}$.

The first step is to show $\mathscr{S}_I \neq \emptyset$ if $\Lambda \neq \emptyset$, or more precisely, the event $\mathscr{S}_I = \emptyset$ and $\Lambda \neq \emptyset$ is null. Assume $\mathscr{S}_I = \emptyset$ and $\Lambda \neq \emptyset$, then Λ must satisfy that if $\langle s, a\rangle \in \Lambda$ for some a, then $\langle s, a\rangle \in \Lambda$ for all a. Let $\mathscr{S}_\Lambda \subset \mathscr{S}$ be the set of s such that $\langle s, a\rangle \in \Lambda$. Clearly, once reaching $s \in \mathscr{S}_\Lambda$, any action chosen would cause Λ be visited, which can only happen for finitely many times. This implies that for any $s \in \mathscr{S}_\Lambda$, any state action pair $\langle s', a'\rangle$ such that $p(s|s', a') > 0$ can only be visited finite number of times *almost surely*, because the probability of sampling from $p(\cdot|s', a')$ for infinitely many times but only getting finite number of s is zero. From Assumption 5.3, for any $\mathscr{S}_\Lambda \neq \mathscr{S}$, there is always some $\langle s', a'\rangle$ such that $s' \notin \mathscr{S}_\Lambda$ and $p(s|s', a') > 0$, so $\langle s', a'\rangle$ can only be visited finitely many times, by definition $\langle s', a'\rangle \in \Lambda$, which contradicts with the fact that $s' \notin \mathscr{S}_\Lambda$.

Next we show that at least for one $s \in \mathscr{S}_I$, following the optimal strategy leads to some $\langle s, a\rangle \in \Lambda$ being visited. For $t > T$, Define

$$\hat{q}(s, a) = r(s, a) + \gamma \sum_{s'} \hat{p}\left(s'|s, a\right) \max_{a'} \hat{q}\left(s', a'\right),$$

with

$$\hat{p}\left(s'|s, a\right) = \begin{cases} p(s'|s, a), \text{ if } \langle s, a\rangle \notin \Lambda \\ \frac{\alpha^t_{s,a,s'}}{\alpha^t_{s,a}}, \text{ if } \langle s, a\rangle \in \Lambda \end{cases},$$

and

$$r(s,a) = \begin{cases} 0, \text{ if } \langle s,a\rangle \notin \Lambda \\ g\left(\alpha^t_{s,a}\right), \text{ if } \langle s,a\rangle \in \Lambda \end{cases}.$$

Clearly, $\hat{p}$ and r do not depend on t, and $\hat{q}$ is the unique optimal solution. Now let $\left\langle s^\dagger, a^\dagger \right\rangle \in \Lambda$ be the pair such that $s \in \mathscr{S}_I$, and

$$\hat{q}\left(s^\dagger, a^\dagger\right) = \max_{\langle s,a\rangle \in \Lambda, s \in \mathscr{S}_I} \hat{q}(s,a).$$

It can be seen that for any a' such that $\left\langle s^\dagger, a' \right\rangle \notin \Lambda$, $\hat{q}\left(s^\dagger, a'\right) \leq \gamma \hat{q}\left(s^\dagger, a^\dagger\right)$. The reason is the following: Performing a' leads to zero immediate reward since $\left\langle s^\dagger, a' \right\rangle \notin \Lambda$. Let s'' be the result of the transition, then either $s'' \in \mathscr{S}_I$, so $\max_{a''} \hat{q}(s'', a'') \leq \hat{q}\left(s^\dagger, a^\dagger\right)$, or s'' is some other state such that $\langle s'', a''\rangle \notin \Lambda$ for all a''. (Note that s'' cannot be a state such that $\langle s'', a''\rangle \in \Lambda$ for all a''.) In the latter case, since s'' is only connected to states in Λ through $\mathscr{S}_I$, it must be that

$$\max_{a''} \hat{q}\left(s'', a''\right) \leq \gamma \hat{q}\left(s^\dagger, a^\dagger\right),$$

since at least one more step must be made to reach $\mathscr{S}_I$ first. Taking into account the discount, it follows that

$$\hat{q}\left(s^\dagger, a^\dagger\right) - \hat{q}\left(s^\dagger, a'\right) \geq (1-\gamma)\, \hat{q}\left(s^\dagger, a^\dagger\right).$$

Replace $\hat{q}$ with $\tilde{q}_{\alpha^t}$ leads to

$$\begin{aligned} \tilde{q}_{\alpha^t}\left(s^\dagger, a^\dagger\right) - \tilde{q}_{\alpha^t}\left(s^\dagger, a'\right) \geq{}& (1-\gamma)\, \hat{q}\left(s^\dagger, a^\dagger\right) \\ &+ \tilde{q}_{\alpha^t}\left(s^\dagger, a^\dagger\right) - \hat{q}\left(s^\dagger, a^\dagger\right) + \tilde{q}_{\alpha^t}\left(s^\dagger, a'\right) - \hat{q}\left(s^\dagger, a'\right) \end{aligned}$$

From the initial assumption, when $t > T$, $\left\langle s^\dagger, a^\dagger \right\rangle$ is never visited, also, the action is chosen greedily with respect to $\tilde{q}_{\alpha^t}$. This implies that at least for one a' such that $\left\langle s^\dagger, a' \right\rangle \notin \Lambda$,

$$\tilde{q}_{\alpha^t}\left(s^\dagger, a^\dagger\right) - \tilde{q}_{\alpha^t}\left(s^\dagger, a'\right) \leq 0,$$

or

$$\begin{aligned} 0 &\geq (1-\gamma)\, \hat{q}\left(s^\dagger, a^\dagger\right) + \tilde{q}_{\alpha^t}\left(s^\dagger, a^\dagger\right) - \hat{q}\left(s^\dagger, a^\dagger\right) + \tilde{q}_{\alpha^t}\left(s^\dagger, a'\right) - \hat{q}\left(s^\dagger, a'\right) \\ &\geq (1-\gamma)\, \hat{q}\left(s^\dagger, a^\dagger\right) - 2 \max_s \max_a \left|\tilde{q}_{\alpha^t}(s,a) - \hat{q}(s,a)\right|, \end{aligned}$$

which leads to

$$\max_s \max_a \left|\tilde{q}_{\alpha^t}(s,a) - \hat{q}(s,a)\right| \geq \frac{1-\gamma}{2} \hat{q}\left(s^\dagger, a^\dagger\right).$$

Note that

$$\begin{aligned}\left|\tilde{q}_{\alpha^t}(s,a)-\hat{q}(s,a)\right| \leq & \left|g\left(\alpha_{s,a}^t\right)-r(s,a)\right| \\ & +\gamma \sum_{s'}\left|\frac{\alpha_{s,a,s'}^t}{\alpha_{s,a}^t}-\hat{p}\left(s'|s,a\right)\right| \max_{a'} \tilde{q}_{\alpha^t}\left(s',a'\right) \\ & +\gamma \sum_{s'} \hat{p}\left(s'|s,a\right) \max_{a'}\left|\tilde{q}_{\alpha^t}\left(s',a'\right)-\hat{q}(s,a)\right|,\end{aligned}$$

so

$$\begin{aligned}& \max_s \max_a\left|\tilde{q}_{\alpha^t}(s,a)-\hat{q}(s,a)\right| \\ & \leq \frac{1}{1-\gamma}\left|g\left(\alpha_{s,a}^t\right)-r(s,a)\right| \\ & +\frac{\gamma}{1-\gamma} \max_s \max_a \sum_{s'}\left|\frac{\alpha_{s,a,s'}^t}{\alpha_{s,a}^t}-\hat{p}\left(s'|s,a\right)\right| \max_{a'} \tilde{q}_{\alpha^t}\left(s',a'\right).\end{aligned}$$

From Lemma 5.7,

$$\left|g\left(\alpha_{s,a}^t\right)-r(s,a)\right|=\max_{\langle s,a\rangle \notin \Lambda} g\left(\alpha_{s,a}^t\right)<\frac{S-1}{2\alpha_{s,a}^t} \rightarrow 0.$$

Therefore, there is some T' such that $\left|g\left(\alpha_{s,a}^t\right)-r(s,a)\right|<\frac{1-\gamma}{4}\hat{q}\left(s^{\dagger},a^{\dagger}\right)$ for all $\langle s,a\rangle \notin \Lambda$. Also note that

$$\tilde{q}_{\alpha^t}\left(s',a'\right) \leq \frac{S-1}{2\left(1-\gamma\right)c_\alpha},$$

where $c_\alpha=\min_{\langle s,a\rangle \in \Lambda} c_{s,a}$. Let $K=\frac{\gamma(S-1)}{2\left(1-\gamma\right)^2 c_\alpha}$, then

$$\begin{aligned}& K \max_s \max_a \sum_{s'}\left|\frac{\alpha_{s,a,s'}^t}{\alpha_{s,a}^t}-\hat{p}\left(s'|s,a\right)\right|+\frac{1-\gamma}{4}\hat{q}\left(s^{\dagger},a^{\dagger}\right) \\ & \geq \frac{\gamma}{1-\gamma} \max_s \max_a \sum_{s'}\left|\frac{\alpha_{s,a,s'}^t}{\alpha_{s,a}^t}-\hat{p}\left(s'|s,a\right)\right| \max_{a'} \tilde{q}_{\alpha^t}\left(s',a'\right) \\ & +\frac{1}{1-\gamma}\left|g\left(\alpha_{s,a}^t\right)-r(s,a)\right| \\ & \geq \max_s \max_a\left|\tilde{q}_{\alpha^t}(s,a)-\hat{q}(s,a)\right| \\ & \geq \frac{1-\gamma}{2}\hat{q}\left(s^{\dagger},a^{\dagger}\right),\end{aligned}$$

thus

$$\max_{\langle s,a\rangle\notin\Lambda}\sum_{s'}\left|\frac{\alpha^t_{s,a,s'}}{\alpha^t_{s,a}}-p\left(s'|s,a\right)\right|=\max_s\max_a\sum_{s'}\left|\frac{\alpha^t_{s,a,s'}}{\alpha^t_{s,a}}-\hat{p}\left(s'|s,a\right)\right|$$
$$\geq\frac{1-\gamma}{4K}\hat{q}\left(s^\dagger,a^\dagger\right),$$

for all $t>T'$. This implies that when $t\to\infty$, the empirical ratio $\frac{\alpha^t_{s,a,s'}}{\alpha^t_{s,a}}$ does not converge to $p\left(s'|s,a\right)$, which is a null event because it contradicts the Strong Law of Large Numbers. This in turn implies that for fixed Λ, T and $c_{s,a}$, the event $\exists\Lambda\neq\emptyset$, $\exists T$, $\exists c_{s,a}$, such that $\alpha^t_{s,a}=c_{s,a}$ for all $t>T$ and $\langle s,a\rangle\in\Lambda$ is null.

As the last step, notice that there are only countably many such events, and since the union of countably many null events is still null, one can conclude that $\lim_{t\to\infty}\alpha^t_{s,a}=\infty$ for all $\langle s,a\rangle$ holds almost surely. ■

The second step is to show that all $\tilde{q}_{\alpha^t}(s,a)$ decrease at a rate lower bounded by c_α^{-1}. Let $q_{s,a}$ be the Q-value of performing a at state s, assuming the reward is 1 at all states. Namely, $q_{s,a}$ is the solution of the following Bellman equation

$$q_{s,a}=1+\gamma\sum_{s'}p\left(s'|s,a\right)\sum_{a'}q_{s',a'}.$$

Clearly, $q_{s,a}>0$ from Assumption 5.3. Define $q=\min_{s,a}q_{s,a}$. Also, let

$$u_{s,a}=\frac{\sum_{s'\neq s^*}f\left(\alpha^0_{s,a,s'}\right)-f\left(\sum_{s'\neq s^*}\alpha^0_{s,a,s'}\right)}{2},$$

where α^0 is the initial α representing the agent's prior, and $s^*=\arg\max_{s'}\alpha^0_{s,a,s'}$ as defined in Lemma 5.7. Define $u=\min_{s,a}u_{s,a}$.

Lemma 5.13 *Assume 5.3, and let* $c_{\alpha^t}=\min_s\min_a\alpha^t_{s,a}$, *then*

$$\liminf_{t\to\infty}c_{\alpha^t}\tilde{q}_{\alpha^t}(s,a)\geq uq,\ a.s.$$

Proof. Let $\hat{q}_{\alpha^t}$ be the solution to the following Bellman equation:

$$\hat{q}_{\alpha^t}(s,a)=g\left(\alpha^t_{s,a}\right)+\gamma\sum_{s'}p\left(s'|s,a\right)\max_{a'}\hat{q}_{\alpha^t}\left(s',a'\right).$$

Clearly, for any $\langle s,a\rangle$,

$$g\left(\alpha^t_{s,a}\right)\geq\frac{u_{s,a}}{\alpha^t_{s,a}}\geq\frac{u}{c_{\alpha^t}}.$$

and because $\hat{q}_{\alpha^t}$ is optimal,

$$\hat{q}_{\alpha^t}(s,a) \geq \frac{u}{c_{\alpha^t}} q_{s,a} \geq \frac{uq}{c_{\alpha^t}},\ \forall s,a,$$

or $c_{\alpha^t}\hat{q}_{\alpha^t}(s,a) \geq uq$.

Fix an $\varepsilon > 0$, we show that

$$\liminf_{t\to\infty} c_{\alpha^t}\tilde{q}_{\alpha^t}(s,a) \leq uq(1-\varepsilon),\ \forall s,a$$

is a null event. Assuming $\liminf_{t\to\infty} c_{\alpha^t}\tilde{q}_{\alpha^t} \leq uq(1-\varepsilon)$, and following a similar procedure as in the proof of Lemma 5.12, let $K = \frac{\gamma(S-1)}{2(1-\gamma)^2}$, then

$$\begin{aligned}\max_s \max_a \sum_{s'} \left| \frac{\alpha^t_{s,a,s'}}{\alpha^t_{s,a}} - \hat{p}\left(s'|s,a\right) \right| &\geq \frac{c_\alpha}{K} \max_s \max_a \left| \tilde{q}_{\alpha^t}(s,a) - \hat{q}(s,a) \right| \\ &\geq \frac{uq\varepsilon}{K}\end{aligned}$$

holds for infinitely many t, which contradicts again with the Law of Large Numbers.

Let $\varepsilon_n = \frac{1}{n}$, then the union of the countably many events

$$\liminf_{t\to\infty} c_{\alpha^t}\tilde{q}_{\alpha^t}(s,a) \leq uq\left(1-\varepsilon_n\right),\ \forall s,a$$

is again a null event, therefore

$$\liminf_{t\to\infty} c_{\alpha^t}\tilde{q}_{\alpha^t}(s,a) \geq uq,\ \text{a.s.}$$

■

Finally, combining Lemma 5.12 and 5.13 gives the proof of Proposition 5.4.

Proof of Proposition 5.4. Note that

$$\left| \frac{q_\alpha}{\tilde{q}_\alpha} - 1 \right| \leq \frac{K}{c_\alpha} \cdot \frac{1}{c_\alpha \tilde{q}_\alpha},$$

where K is given in Proposition 5.3. Use Lemma 5.12, 5.13, the result follows trivially. ■

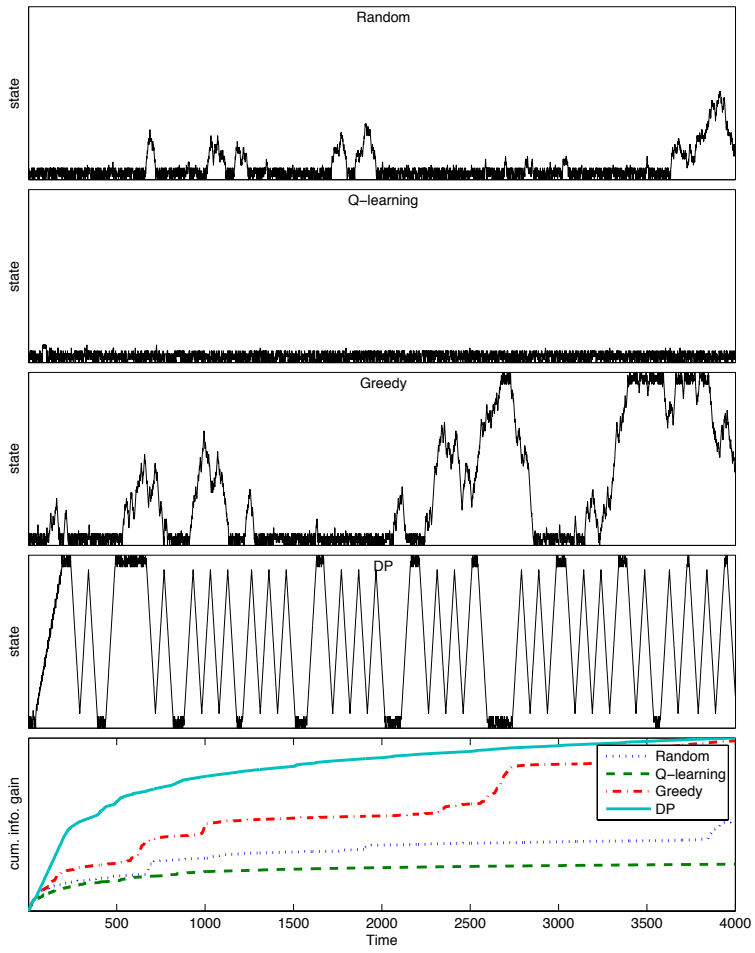

Figure 5.2. The exploration process of a typical run of 4000 steps. The upper four plots shows the position of the agent between state 1 (the lowest) and 60 (the highest). The states at the top and the bottom correspond to the two cliques, and the states in the middle correspond to the corridor. The lowest plot is the cumulative information gain with respect to the prior.

Chapter 6

Conclusion

We have presented three pieces of research concerning the representation generation problem in reinforcement learning, with the following original contributions:

- In Chapter 3, we proved that under certain technical conditions, the size of the online kernel sparsification dictionary will grow sub-linearly in the number of data points, which leads to the consistency of the kernel linear regressor from the resulting dictionary.
- In Chapter 4, we proposed V-BEBF as a principled alternative to BEBF for basis construction in the reward sensitive setting. To this end, we showed theoretically that V-BEBF may offer a significant advantage when the discount factor $\gamma \to 1$, which is partially confirmed by experiments.
- In Chapter 5, we formulated information theoretic exploration for Bayesian model learning, and then for the special case of finite Markovian environment with Dirichlet priors over transition probabilities, we proved that the optimal exploration strategy can be approximated well by solving a sequence of MDP planning problems.

The work presented is limited in many ways, and each of our findings many give rise to many more questions than answers. To list a few:

- The analysis in Chapter 3 is done assuming observations are given i.i.d., and further theoretical analysis is required to remove this assumption in order to extend the results to kernel RL. Moreover, the finite sample bounds are weak, and much improvement is needed to strengthen the bounds.

- In Chapter 4, further theoretical and experimental study is needed to better understand the effectiveness of the V-BEBF approach, especially in the non-linear case. In addition, it would be interesting to investigate IPB-V further, as our preliminary study supports its potential as a practical dimensionality reduction method for trading off computational complexity and sample complexity.

- Chapter 5 poses a number of interesting theoretical questions. In particular, it is unclear if we can achieve near optimal exploration in environment that is not finite Markovian, and whether there are strong links between the optimal initialization in model-free RL, and information theoretic exploration.

Bibliography

R. Agrawal. Sample mean based index policies with $o(\log n)$ regret for the multi-armed bandit problem. *Adv. Appl. Probab.*, 27(4):1054–1078, 1995.

N. Akhiezer and I. Glazman. *Theory of linear operators in Hilbert space*. F. Ungar Pub. Co., 1961.

H. Alzer. On some inequalities for the Gamma and Psi functions. *Math. Comput.*, 66(217):373–389, 1997.

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *J. Mach. Learn. Res.*, 3:1–48, 2002.

F. R. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *ICML'05*, 2005.

L. Baird. Residual algorithms: reinforcement learning with function approximation. In *ICML'95*, pages 30–37, 1995.

L. Baird. Reinforcement learning through gradient descent. Technical report, School of Computer Science, Carnegie Mellon University, CMU-CS-99-132, 1999.

L. Barrett and S. Narayanan. Learning all optimal policies with multiple criteria. In *ICML'08*, 2008.

J. Baxter and P. L. Barlett. Infinite-horizon policy-gradient estimation. *J. Artif. Intell. Res.*, 15:319–350, 2001.

R. E. Bellman. On the theory of dynamic programming. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 38, pages 716–719, 1952.

D. P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, iii edition, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1st edition, 1996. ISBN 1886529108.

R. Bhatia and L. Elsner. The Hoffman-Wielandt inequality in infinite dimensions. *P. Indian. As. - Math. Sci.*, 104(3):483–494, 1994.

G. Blanchard, O. Bousquet, and L. Zwald. Statistical properties of kernel principal component analysis. *Mach. Learn.*, 66:259–294, 2007.

B. Boots and G. J. Gordon. Predictive state temporal difference learning. In *NIPS'10*, pages 271–279, 2010.

B. Boots, S. M. Siddiqi, and G. J. Gordon. Closing the learning-planning loop with predictive state representations. Technical report, arXiv:0912.2385, 2009.

S. Boucheron, G. Lugosi, and P. Massart. A sharp concentration inequality with applications. Technical report, Department of Economics and Business, Universitat Pompeu Fabra, 376, 1999. URL `http://ideas.repec.org/p/upf/upfgen/376.html`.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

J. A. Boyan. Technical update: least-squares temporal difference learning. *Mach. Learn.*, 49:233–246, 2002.

S. J. Bradtke, A. G. Barto, and L. P. Kaelbling. Linear least-squares algorithms for temporal difference learning. *Mach. Learn.*, 22:33–57, 1996.

M. L. Braun. *Spectral properties of the kernel matrix and their relation to kernel methods in machine learning*. PhD thesis, University of Bonn, 2005.

L. Busoniu, R. Babuska, and B. D. Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, 38(2):156–172, 2008.

L. Busoniu, D. Ernst, B. De Schutter, and R. Babuska. Online least-squares policy iteration for reinforcement learning control. In *ACC'10*, pages 486–491, 2010.

D. Chakraborty and P. Stone. Structure learning in ergodic factored MDPs without knowledge of the transition function's in-degree. In *ICML'11*, 2011.

K. Chaloner and I. Verdinelli. Bayesian experimental design: a review. *Stat. Sci.*, 10:273–304, 1995.

T. M. Cover and J. A. Thomas. Determinant inequalities via information theory. *SIAM J. Matrix Anal. A.*, 9(3):384–392, 1988.

F. Creutzig, A. Globerson, and N. Tishby. Past-future information bottleneck in dynamical systems. *Phys. Rev. E*, 79(4):041925, 2009.

R. H. Crites and A. G. Barto. Improving elevator performance using reinforcement learning. In *NIPS'96*, 1996.

S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Struct. Algor.*, 22(1):60–65, 2003.

P. Dayan and T. J. Sejnowski. TD(λ) converges with probability 1. *Mach. Learn.*, 14(1):295–301, 1994.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Statistical Society, Series B*, 39(1): 1–38, 1977.

F. d'Epenoux. A probabilistic production and inventory problem. *Manage. Sci.*, 10:98–108, 1963.

P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *J. Mach. Learn. Res.*, 6: 2153–2175, 2005.

N. Duy and J. Peters. Incremental sparsification for real-time online model learning. In *AISTAT'10*, pages 557–564, 2010.

Y. Engel. *Algorithms and representations for reinforcement learning*. PhD thesis, Hebrew University, 2005.

Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: the Gaussian process approach to temporal difference learning. In *ICML'03*, pages 154–161, 2003.

Y. Engel, S. Mannor, and R. Meir. The kernel recursive least-squares algorithm. *IEEE Trans. Signal Process.*, 52(8):2275–2285, 2004.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *ICML'05*, pages 201–208, 2005.

V. V. Fedorov. *Theory of optimal experiments*. Academic Press, 1972.

M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recogn.*, 41(1):176–190, 2008.

C. Fleck. A comparison of POMDP algorithms, 2004.

A. Garivier and O. Cappè. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *COLT'11*, 2011.

A. Geramifard, M. Bowling, and R. S. Sutton. Incremental least-squares temporal difference learning. In *AAAI'06*, pages 356–361, 2006.

M. Ghavamzadeh, A. Lazaric, O. Maillard, and R. Munos. LSTD with random projections. In *NIPS'10*, pages 712–720, 2010.

T. Glasmachers, T. Schaul, Y. Sun, and J. Schmidhuber. Exponential natural evolution strategies. In *GECCO'10*, pages 393–400, 2010.

A. Gretton, K. Fukumizu, Z. Harchaoui, and B. K. Sriperumbudur. A fast, consistent kernel two-sample test. In *NIPS'09*, 2009.

C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solutions algorithms for factored MDPs. *J. Artif. Intell. Res.*, 19:399–468, 2003.

X. Guo and O. Hernández-Lerma. *Continuous-time Markov decision processes*. Springer, 2009.

L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer, 2004.

Z. Harchaoui, F. R. Bach, and Éric Moulines. Testing for homogeneity with kernel Fisher discriminant analysis. In *NIPS'08*, 2008.

T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Ann. Stat.*, 32 (2):407–499, 2004.

W. Hoeffding. The strong law of large numbers for U-statistics. Technical report, Department of statistics, University of North Carolina, 302, 1961.

G.-B. Huang, L. Chen, and C.-K. Siew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Netw.*, 17(4):879–892, 2006.

M. Hutter. *Universal artificial intelligence: sequential decisions based on algorithmic probability*. Texts in Theoretical Computer Science. Springer, 2005. ISBN 9783540221395.

M. Hutter. Feature dynamic Bayesian networks. In *AGI'09*, volume 8, pages 67–73, 2009a.

M. Hutter. Feature reinforcement learning: Part I: Unstructured MDPs. *Journal of Artificial General Intelligence*, 1:3–24, 2009b.

L. Itti and P. F. Baldi. Bayesian surprise attracts human attention. In *NIPS'05*, pages 547–554, 2006.

M. T. Izadi and D. Precup. A planning algorithm for predictive state representations. In *IJCAI'03*, pages 1520–1521, 2003.

T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11:1563–1600, 2010.

M. R. James and S. Singh. Learning and discovery of predictive state representations in dynamical systems with reset. In *ICML'04*, pages 417–424, 2004.

R. Jin, T.-B. Yang, M. Mahdavi, Y.-F. Li, and Z.-H. Zhou. Improved bound for the Nyström method and its application to kernel classification. Technical report, arXiv:1111.2262, 2012.

A. Jonsson and A. G. Barto. Active learning of dynamic Bayesian networks in Markov decision processes. In *SARA'07*, 2007.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134, 1998.

M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. *Mach. Learn.*, 49(2-3):209–232, 2002.

P. W. Keller, S. Mannor, and D. Precup. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *ICML'06*, pages 449–456, 2006.

V. Koltchinskii and E. Giné. Random matrix approximation of spectra of integral operators. *Bernoulli*, 6(1):113–167, 2000.

Z. J. Kolter and A. Y. Ng. Regularization and feature selection in least-squares temporal difference learning. In *ICML'09*, pages 521–528, 2009a.

Z. J. Kolter and A. Y. Ng. Near-Bayesian exploration in polynomial time. In *ICML'09*, pages 513–520, 2009b.

R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *ICML'02*, pages 315–322, 2002.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *J. Mach. Learn. Res.*, 4:1107–1149, 2003.

R. Legenstein and W. Maass. Edge of chaos and prediction of computational performance for neural circuit models. Technical report, Technische Universitaet Graz, 2007.

S. Legg. *Machine super intelligence*. PhD thesis, University of Lugano, 2008.

L.-J. Lin and T. M. Mitchell. Reinforcement learning with hidden states. In *From animals to animats 2: simulation of adaptive behavior*, pages 271–280, 1993.

D. V. Lindley. On a measure of the information provided by an experiment. *Ann. Math. Statist.*, 27(4):986–1005, 1956.

M. L. Littman, T. L. Dean, and L. P. Kaelbling. On the complexity of solving Markov decision problems. In *UAI'95*, 1995.

M. L. Littman, R. S. Sutton, and S. Singh. Predictive representations of state. In *NIPS'01*, pages 1555–1561, 2001.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Test classification using string kernels. *Journal of Machine Learning Research*, 2: 419–444, 2002.

H. R. Maei, C. Szepesvári, S. Bhatnagar, D. Precup, D. Silver, and R. S. Sutton. Convergent temporal-difference learning with arbitrary smooth function approximation. In *NIPS'99*, pages 1204–1212, 2009.

S. Mahadevan. Average reward reinforcement learning: foundations, algorithms, and empirical results. *Mach. Learn.*, 22(1-3):159–195, 1996.

S. Mahadevan and B. Liu. Basis construction from power series expansions of value functions. In *NIPS'10*, pages 1531–1539, 2010.

S. Mahadevan, M. Maggioni, and C. Guestrin. Proto-value functions: a Laplacian framework for learning representation and control in Markov decision processes. *J. Mach. Learn. Res.*, 8:2169–2231, 2007.

S. Mannor and J. Tsitsiklis. Mean-variance optimization in Markov decision processes. In *ICML'11*, 2011.

J. Martin. The expected determinant of the random Gram matrix and its application to information retrieval systems, 2007. URL `http://dydan.rutgers.edu/Seminars/Slides/martin2.pdf`.

D. D. Mauá and C. P. de Campos. Solving decision problems with limited information. In *NIPS'11*, 2011.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *J. Artif. Intell. Res.*, 44:97–140, 2012.

A. K. McCallum. *Reinforcement learning with selective perception and hidden state*. PhD thesis, Department of Computer Science, University of Rochester, 1996.

P. McCracken and M. Bowling. Online discovery and learning of predictive state representations. In *NIPS'05*, pages 875–882, 2005.

I. Menache, S. Mannor, and N. Shimkin. Basis function adaptation in temporal difference reinforcement learning. *Ann. Oper. Res.*, 134:215–238, 2005.

C. D. Meyer. *Matrix analysis and applied linear algebra*. SIAM: Society for Industrial and Applied Mathematics, 2001.

O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. *Mach. Learn.*, 49(2-3):267–290, Nov. 2002.

G. E. Monahan. A survey of partially observable Markov decision processes: theory, models, and algorithms. *Manage. Sci.*, 28(1):1–16, 1982.

K. P. Murphy. A survey of POMDP solution techniques. Technical report, 2000.

A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *ICML'00*, 2000.

C. P. Niculescu. A new look at Newton's inequalities. *JIPAM*, 1(2), 2000.

D. Ormoneit and S. Sen. Kernel-based reinforcement learning. *Mach. Learn.*, 49 (2-3):161–178, 2002.

L. Orseau. Universal knowledge seeking agents. In *ALT'11*, volume 6925 of LNAI, pages 353–367, 2011.

P. A. Ortega and D. A. Braun. A minimum relative entropy principle for learning and acting. *J. Artif. Intell. Res.*, 38:475–511, 2010.

Özgür Şimşek and A. G. Barto. An intrinsic reward mechanism for efficient exploration. In *ICML'06*, pages 833–840, 2006.

R. Parr, C. Painter-Wakefield, L.-H. Li, and M. Littman. Analyzing feature generation for value-function approximation. In *ICML'07*, pages 737–744, 2007.

W. Penny. Kullback-Liebler divergences of normal, Gamma, Dirichlet and Wishart densities. Technical report, Wellcome Department of Cognitive Neurology, University College London, 2001.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21:682–697, 2008.

K. B. Petersen and M. S. Pedersen. The matrix cookbook. Technical report, Technical University of Denmark, 2008.

T. K. Philips and R. Nelson. The moment bound is tighter than Chernoff's bound for positive tail probabilities. *Am. Stat.*, 49(2):175–178, 1995.

D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *IJCAI'07*, 2007.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

M. Rosencrantz, G. Gordon, and S. Thrun. Learning low dimensional predictive representations. In *ICML'04*, pages 695–702, 2004.

S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *J. Artif. Intell. Res.*, 32:663–704, 2008.

S. Ross, J. Pineau, B. Chaib-draa, and P. Krietmann. A Bayesian approach for learning and planning in partially observable Markov decision processes. *J. Mach. Learn. Res.*, 12:1655–1696, 2011.

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

G. A. Rummery. *Problem solving with reinforcement learning*. PhD thesis, Cambridge University Engineering Department, 1994.

G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical report, Cambridge University Engineering Department, CUED/F-INFENG/TR 166, 1994.

H. A. M. Schäfer. *Reinforcement learning with recurrent neural networ*. PhD thesis, Universität Osnabrück, 2008.

J. Schmidhuber. Making the world differentiable: on using fully recurrent self-supervised neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical Report FKI-126-90, Institut für Informatik, Technische Universität München, 1990.

J. Schmidhuber. Curious model-building control systems. In *IJCNN'91*, volume 2, pages 1458–1463, 1991.

J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990-2010). *IEEE Trans. Auton. Mental Develop.*, 2(3):230–247, 2010.

B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

B. Schölkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

R. J. Serfling. *Approximation theorems of mathematical statistics*. Wiley, 1980. ISBN 9780471024033.

W. Shi and Y.-F. Guo. Incomplete Cholesky decomposition based kernel principal component analysis for large-scale data set. In *IJCNN'10*, pages 1–6, 2010.

D. Silver, R. S. Sutton, and M. Müller. Temporal-difference search in computer Go. *Mach. Learn.*, 87(2):183–219, 2012.

S. Singh, A. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. In *NIPS'04*, pages 1281–1288, 2004.

K. Slavakis, S. Theodoridis, and I. Yamada. Online kernel-based classification using adaptive projection algorithms. *IEEE Trans. Signal Process.*, 56(7):2781–2796, 2008.

I. Steinwart, D. R. Hush, and C. Scovel. An explicit description of the reproducing kernel Hilbert spaces of Gaussian RBF kernels. *IEEE Trans. Inf. Theory*, 52: 4635–4643, 2006.

J. Storck, S. Hochreiter, and J. Schmidhuber. Reinforcement driven information acquisition in non-deterministic environments. In *ICANN'95*, pages 159–164, 1995.

A. Stout, G. Konidaris, and A. G. Barto. Intrinsically motivated reinforcement learning: a promising framework for developmental robot learning. In *AAAI Spring Symposium on Developmental Robotics*, 2005.

A. L. Strehl and M. L. Littman. A theoretical analysis of model-based interval estimation. In *ICML'05*, pages 856–863, 2005.

A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. PAC model-free reinforcement learning. In *ICML'06*, 2006.

Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Efficient natural evolution strategies. In *GECCO'09*, pages 539–546, 2009a.

Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Stochastic search using the natural gradient. In *ICML'09*, pages 1161–1168, 2009b.

Y. Sun, F. Gomez, M. Ring, and J. Schmidhuber. Incremental basis construction from temporal difference error. In *ICML'11*, pages 481–488, 2011a.

Y. Sun, F. Gomez, and J. Schmidhuber. Planning to be surprised: optimal Bayesian exploration in dynamic environments. In *AGI'11*, pages 41–51, 2011b.

Y. Sun, F. Gomez, and J. Schmidhuber. On the size of the online kernel sparsification. In *ICML'12*, 2012.

R. S. Sutton. Learning to predict by the methods of temporal differences. *Mach. Learn.*, 3:9–44, 1988.

R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *ICML'90*, pages 216–224, 1990.

R. S. Sutton and A. G. Barto. *Reinforcement learning : an introduction*. MIT Press, 1998.

R. S. Sutton, C. Szepesvári, and H. R. Maei. A convergent $O(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In *NIPS'08*, pages 1–8, 2008.

R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *ICML'09*, pages 993–1000, 2009.

C. Szepesvári. *Algorithms for reinforcement learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010. ISBN 9781608454921.

I. Szita and A. Lörincz. Optimistic initialization and greediness lead to polynomial time learning in factored MDPs. In *ICML'09*, 2009.

I. Szita, V. Gyenes, and A. Lőrincz. Reinforcement learning with echo state networks. In *ICANN'06*, pages 830–839, 2006.

G. Taylor and R. Parr. Kernelized value function approximation for reinforcement learning. In *ICML'09*, 2009.

G. Tesauro. Temporal difference learning and TD-Gammon. *Commun. ACM*, 38 (3):58–68, 1995.

S. B. Thrun and K. Möller. Active exploration in dynamic environments. In *NIPS'91*, pages 531–538, 1991.

M. Toussaint, S. Harmeling, and A. Storkey. Probabilistic inference for solving (PO)MDPs. In *ICML'06*, 2006.

J. Unkelbach, Y. Sun, and J. Schmidhuber. An EM based training algorithm for recurrent neural networks. In *ICANN'09*, pages 964–974, 2009.

H. van Hasselt and M. A. Wiering. Reinforcement learning in continuous action spaces. In *Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on*, pages 272–279, 2007.

J. Veness, K. S. Ng, M. Hutter, W. Uther, and D. Silver. A Monte Carlo AIXI approximation. *J. Artif. Intell. Res.*, 40:95–142, 2011.

N. Vlassis, M. Toussaint, G. Kontes, and S. Piperidis. Learning model-free robot control by a Monte Carlo EM algorithm. *Auton. Robot.*, 27(2):123–130, 2009.

U. von Luxburg. A tutorial on spectral clustering. Technical report, Max Planck Institute for Biological Cybernetics, TR-149, 2006.

C. J. Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge, 1989.

C. J. Watkins and P. Dayan. Technical note: Q-learning. *Mach. Learn.*, 8:279–292, 1992.

S. D. Whitehead and L.-J. Lin. Reinforcement learning of non-Markov decision processes. *Artif. Intell.*, 73(1-2):271–306, 1995.

B. Widrow and S. D. Stearns. *Adaptive signal processing*. Prentice-Hall, 1985.

D. Wierstra and J. Schmidhuber. Policy gradient critics. In *ECML'07*, pages 466–477, 2007.

D. Wierstra, A. Foerster, J. Peters, and J. Schmidhuber. Solving deep memory POMDPs with recurrent policy gradients. In *ICANN'07*, pages 697–706, 2007.

C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *NIPS'00*, pages 682–688, 2000.

B. Wolfe, M. R. James, and S. Singh. Learning predictive state representations in dynamical systems without reset. In *ICML'05*, pages 980–987, 2005.

J.-H. Wu and R. Givan. Feature-discovering approximate value iteration methods. In *SARA'05*, pages 321–331, 2005.

J. L. Wyatt, P. Dayan, A. Leonardis, and J. Peters. Exploration and curiosity in robot learning and inference. *Dagstuhl Reports*, 1(3):67–95, 2011.

X. Xu. A sparse kernel-based least-squares temporal difference algorithm for reinforcement learning. In *Advances in Natural Computation*, volume 4221 of *Lecture Notes in Computer Science*, pages 47–56. Springer, 2006.

X. Xu, T. Xie, D.-W. Hu, and X.-C. Lu. Kernel least-squares temporal difference learning. *International Journal of Information Technology*, 11(9):55–63, 2005.

X. Xu, D. Hu, and X. Lu. Kernel-based least squares policy iteration for reinforcement learning. *IEEE Trans. Neural Netw.*, 18(4):973–992, 2007.

X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

W.-H. Zhang. *Algorithms for partially observable Markov decision processes*. PhD thesis, Hong Kong University of Science and Technology, 2001.

B. D. Ziebart, A. Maas, A. J. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI'08*, 2008.