Decision Making Under Uncertainty

Robert Edward McInerney

New College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity Term 2014

Robert Edward McInerney
New College

Doctor of Philosophy
Trinity Term 2014

Decision Making Under Uncertainty

Summary

Operating and interacting in an environment requires the ability to manage uncertainty and to choose definite courses of action. In this thesis we look to Bayesian probability theory as the means to achieve the former, and find that through rigorous application of the rules it prescribes we can, in theory, solve problems of decision making under uncertainty. Unfortunately such methodology is intractable in real-world problems, and thus approximation of one form or another is inevitable.

Many techniques make use of heuristic procedures for managing uncertainty. We note that such methods suffer unreliable performance and rely on the specification of ad-hoc variables. Performance is often judged according to long-term asymptotic performance measures which we also believe ignores the most complex and relevant parts of the problem domain. We therefore look to develop principled approximate methods that preserve the meaning of Bayesian theory but operate with the scalability of heuristics.

We start doing this by looking at function approximation in continuous state and action spaces using Gaussian Processes. We develop a novel family of covariance functions which allow tractable inference methods to accommodate some of the uncertainty lost by not following full Bayesian inference. We also investigate the exploration versus exploitation tradeoff in the context of the Multi-Armed Bandit, and demonstrate that principled approximations behave close to optimal behaviour and perform significantly better than heuristics on a range of experimental test beds.

This thesis is dedicated to my grandfathers

Dr. Leslie Singleton
CHEMIST

Maurice John McInerney
ENGINEER

Growing up in a family of lawyers,
their blood runs thickest in my veins

Acknowledgements

My research was funded by the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Systems) Project, made possible by an EPSRC (Engineering and Physical Sciences Research Council) and BAE Systems strategic partnership. I thank all those involved in the project for creating the opportunity and for the enriching environment it provided.

I would like to thank the many members of the Machine Learning Research Group for endless cerebral discussions and for making the lab an entertaining place to be. I would particularly like to thank Iead Rezek, who heavily influenced the early direction of my research; Steve Reece, for his many academic contributions and reassuring cynicism; and my examiners, Michael Osborne and David Leslie, who provided valuable input and advice throughout my DPhil.

Of course, I am hugely indebted to my supervisor, Professor Stephen Roberts, whose monumental patience and untold support throughout the whole process was both astonishing and well beyond the call of duty... we got there in the end!

I could not possibly have completed this without the welcome distraction from my incredible friends at New College; the musicians I've been fortunate enough to share a stage with, particularly Andy Smith who would do anything for anybody; the rugby boys, especially fellow 'old man' Mark Chadwick; and everyone from the Summertown Wine Cafe. Collectively you may have added a year to the write up, but we had fun.

Special thanks must go to those disadvantaged individuals who had me as a flatmate: Zita, who imbued me with a deep appreciation of the human brain and whose scientific integrity is the greatest of anyone I know; JP, my sous chef and broer

vir altyd; and Paul, who somehow always made things OK... even if it did mean watching 20 seasons of MasterChef.

Above all, I thank my family who somehow continue to think the very best of me. I must particularly thank: Dr. Leslie Singleton, Grandpa, who spent weeks with me in the Science Museum as a child and never doubted my ability to get this thing done; my sister Josie, whose resilience and tenacity has given me inspiration far beyond that which she knows; and finally my parents, Peter and Jennifer, whose unconditional support continues to give me every opportunity and who instilled in me the intellectualism that made this possible.

Contents

Chapter 1

Introduction

> "The message is that there are *knowns*. There are things we know that we know. There are *known unknowns*. That is to say there are things that we now know we don't know. But there are also *unknown unknowns*. There are things we don't know we don't know."
>
> Donald H. Rumsfeld, *NATO Headquarters* (2002)

I was recently talking to a friend of mine who was concerned that, as technology becomes more and more personalised, his online profile will become pigeon holed to such an extent that he will only ever be presented with content completely pursuant to some sort of idealised stereotype of himself. Through this, he felt that he would lose the charm of discovering new interests outside his own experience and potentially inconsistent with his browsing habits. He likened this to his visits to independent record stores and booksellers like *Shakespeare and Company* in Paris, whose abstractly organised collections always yielded something unexpected. Surely such pioneering would be consigned to the past once our digital avatar completely dictates what we do and don't get to see?

We are finite beings, and our knowledge about the world around us is similarly finite. There are things we don't know. Fortunately, as humans we are remarkably good at operating under such uncertainty, and do so almost subconsciously. However, as much as our knowledge is bounded, so too is our domain of influence and mental resources. I asked my friend whether he would be happier if a person was directly

responsible for the online traffic sent his way, but the incontrovertible reality was that no real team of people could operate on the scale of the web to achieve close to the same level of personalisation. There is simply too much data, and too many users.

Autonomous systems fulfil a need. There are things that as humans we are bad at, incapable of, or simply do not want to do. Whilst the requirement for computational frameworks that can sift through petabytes of online data is patently obvious, autonomous systems extend far beyond that domain. As Curiosity touched down on the sands of Mars, the requirement wasn't one of arduous number crunching, rather the fact that human instruction would always be 13 minutes away; 'fly by wire' was not an option. Interplanetary rovers are perhaps an extreme end of the spectrum, but here on Earth there are almost illimitable examples in which autonomous systems can and will be deployed to work alongside and enhance the capabilities of humans.

In order to allow autonomous systems to operate independently whilst faced with all the facets of existence including limited resources and uncertainty, we must first provide it with some form of artificial intelligence. Intelligence itself is defined by Gottfredson as:

> "A very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience. It is not merely book learning, a narrow academic skill, or test-taking smarts. Rather, it reflects a broader and deeper capability for comprehending our surroundings - 'catching on', 'making sense' of things, or 'figuring out' what to do." (Gottfredson, 1997)

In the context of an artificial agent, rather more compactly we look to give it the capability to act in a way that would be consistent with a *rational* thinking human mind. With regards to my friend's consternation, we could think of a system which can operate on the scale of the web, but with the apparent judgement of a person.

After that he can direct his dismay towards the commercial interests of the content aggregator which would almost certainty not be aligned with his own; however that is a whole different problem.

1.1 Definite Courses of Action

The central object of interest in this thesis is the brain of an artificial being, residing in the core of some computational framework. Jaynes refers to this entity as *the robot* (Jaynes, 2003), although we also use the name *agent* or *decision maker* where custom and context dictates. Just as information is limited for humans, so too it is for the robot. Therefore a primary concern of its design is the ability to reason under uncertainty. We will find that, by utilising a principled approach to uncertainty specified by Bayesian probability theory, many of the other concerns of the robot will fall quickly into place.

In a way we approach this thesis without context, meaning the solutions should form part of the core functionality of the robot. By necessity, we approach one particular component of the robot's design: the ability to choose definite courses of action through decision making. Clearly acting in an environment is a principle behaviour; however one should not view it as the last step in a complicated thought process. Rather, it is a prerequisite of interaction, and only one part of a cyclical operation which also includes sensing and thinking.

Even considering decision making alone is no small task, and whilst we attempt to provide a clear working through of fundamental principles, we do not have all the answers. There are a great many bottlenecks, from computational to mathematical, only a few of which we could possibly consider here. We concentrate on the application of Bayesian theory because we find it to be correct, although we are quick to acknowledge its practical limitations. What we hope this thesis provides is a sound

analysis of practical Bayesian theory in the context of decision making, from which further development, to which we contribute, can be made.

1.2 The Role of Humans in Autonomous Systems

Whilst the knowledge and skills applied here are removed from those available in the cognitive sciences, we feel it necessary to make a statement on the role humans play in the design of autonomous systems. To our knowledge human beings possess the single greatest intellect, the human brain being the most adept learning tool on our planet. From conception to adulthood our cognitive development is astounding. The psychologist and philosopher William James described the early stages of human life:

> "The baby, assailed by eyes, ears, nose, skin, and entrails at once, feels it all as one great blooming, buzzing confusion." (James, 1890)

Yet through this confusion we learn to interact with the nerves, flesh and bones that form our body, and through those the world around us. The brain also has extraordinary adaptive abilities. Take, for example, research which demonstrates how part of the brain of stroke patients will compensate for a damaged area by fulfilling certain missing functions not normally associated with it (Pascual-Leone et al., 2005). As designers of autonomous systems, intuition would suggest we should at least attempt to understand the processes through which humans operate. After all, the field of artificial intelligence was itself forged on the conjecture that "every aspect of learning can in principle be so precisely described that a machine can be made to simulate it" (McCarthy et al., 1955).

Although building a machine that could match the achievements of humans may be desirable in some senses, this is not the purpose we align ourselves to. As already stated, autonomous systems are needed to accomplish tasks for which human performance is inadequate. After all, human behaviour is heavily influenced by

emotions, values, culture and genetics. When we talk about a rational thinking mind or 'common sense', we refer to a carefully considered, almost mathematical operation, outside the realms of emotional biasing.

Regardless of whether we recreate elements of the brain or not, there is a definite need to consider human behaviour when designing autonomous systems. More often than not, these systems rely on a fundamental interaction with human beings; the very foundation on which problems of this sort are built implies that autonomous agents should be able to model the behaviour of and act symbiotically with humans. Whilst here we concentrate on other parts of the problem, eventually it is critical to both understand human behaviour, as well as to understand how this behaviour can be influenced by the actions of autonomous agents.

1.3 Thesis Overview

We begin the thesis proper in Chapter 2 by describing the fundamental principles by which the robot's brain must operate. This begins with a discussion on the form of reasoning needed for operation in the real-world, and leads on to the introduction of *Bayesian probability theory* as the means to characterise uncertainty. From there we move towards inference processes and an initial study of decision making, emphasising the importance of utility and its specification. We conclude this chapter by highlighting the need for principled approximations due to computational and physical limitations.

The majority of this thesis is devoted to the study of decision making in which actions are taken sequentially with the objective of maximising some notion of reward. In Chapter 3, we introduce *Reinforcement Learning*, and through that *Markov Decision Processes*, as the means to formulate and describe these type of problems. In Chapter 4, we demonstrate how to solve such decision problems under the auspices

of Bayesian theory. After this we discuss *Dynamic Programming* as an efficient algorithm for calculating the solution, as well as discuss the practical challenges real-world implementation confronts us with.

In Chapter 5, we approach the problem of applying Bayesian decision theory in continuous spaces, requiring some form of approximation. We attack the problem from a function modelling point of view and introduce *Gaussian Processes* as a robust means to do this. Understanding the need to derive computationally tractable methodology, we develop a novel approximation for performing inference, using a Gaussian process[1]. Through this we generate a new general class of covariance functions that can be used interchangeably with a standard covariance function. We demonstrate the favourable performance of this approach across a suite of test problems, including a novel form of value iteration.

Dealing with uncertainty is fundamental to all problems involving interaction with the real world. For a decision maker, this presents a difficult problem in which learning about the world must be accomplished at the same time as achieving objectives. This challenge is known in the literature as the *exploration versus exploitation* tradeoff. In Chapter 6, we develop the theory already presented so as to solve the tradeoff in a Bayesian optimal manner. We do this in the context of *Multi-Armed Bandits* and discuss a particular solution to this known as the *Gittins Index*.

The main part of the thesis concludes in Chapter 7 with an exploration into approximate decision methods. For this, we begin by defining a set of core behavioural features which an optimal decision process should exhibit, and demonstrate that these naturally occur when following the principles laid out in the rest of the thesis. This is achieved by dissecting the decision problem to the point where the balance between exploration and exploitation can be clearly observed. In order to provide a comparison with other methodology, we describe a set of heuristic methods, and provide a critique

[1]Note that the derivation of this was joint work with two other people in the research group.

of stochastic decision making. Finally we present a range of experimental test beds and demonstrate the performance advantages of using optimal decision making, at the same time as deriving a variety of principled approximations with similarly favourable properties.

In Chapter 8, we conclude the thesis with a discussion of results and the core themes raised, as well as providing direction for future research.

We opened with a quote from Donald Rumsfeld referencing the perceived presence of weapons of mass destruction in Iraq. The statement beautifully captures the idea that we can be unaware of things we are unaware of: unknown unknowns. To believe one knows everything is clearly ignorant. However, to us it seems equally ignorant for a person to believe they know precisely what they do not know. We spend much of our time in this field generating models of the world that provide us with the means to characterise uncertainty about things we know we should be uncertain about. Nevertheless, just as a physicist would accept that there are parts of the universe yet to be observed, so too can we characterise uncertainty at a level beyond our experience. Given careful enough thought, there is no reason for us to be ignorant, even about the things that we don't know we don't know.

Chapter 2

Probability Theory

> "We are so far from knowing all the agents of nature and their divers modes of action that it would not be philosophical to deny phenomena solely because they are inexplicable in the actual state of our knowledge. But we ought to examine them with an attention all the more scrupulous as it appears more difficult to admit them."
>
> *Essai philosophique sur les probabilités*, Laplace (1814)

The objective of this chapter is to precisely lay out the set of principles by which the brain of our robot operates. Faced with the problem of designing a 'thinking' machine, it is tempting to get drawn towards centuries of deliberation and discussion on the subject of intelligence and existence; from the great philosophers through to the modern day practitioners, the likes of which can be found in Turing and von Neumann. We could easily find ourselves drawn into deep rumination on what constitutes consciousness, or the relationship between understanding and experience. Fortunately, our objective here is somewhat more constrained. We place those questions to the side, for they are the meditations of greater men and women, and firmly don our engineer's cap instead. Whilst our understanding of what exactly 'thinking' amounts to may still be vague, we will find that by constructing mathematical models with definite objectives we are able to reproduce just the sort of planning and reasoning

we might expect from a rational human mind.

2.1 Plausible Reasoning

We start our hardheaded investigation by considering the manner by which the robot is able to draw conclusions about something from knowledge of something else: the process of inference. One could think of this as the understanding of cause and effect, although we are most interested in the development of meaningful relationships between events and their truth or falsehood, whether causal or not. In essence, we wish to be able to gain insight based on the observation or knowing of some other piece of information.

Taking a symbolic view to begin with, we in general want to be able to reason about a proposition B given knowledge of A. Deductive reasoning would allow us to reach a logically certain conclusion about B given A, as summarised in the following strong syllogisms:

$$\begin{array}{c} \textbf{if } A \textbf{ is true, then } B \textbf{ is true} \\ \hline A \text{ is true, therefore } B \text{ is true} \\ B \text{ is false, therefore } A \text{ is false} \end{array} \tag{2.1}$$

As an example of the first of these, let me express the fact that I always go running on a Tuesday, and today is a Tuesday. Now given that '$A =$ *today is a Tuesday*' is true, the certitude of '$B =$ *I go running today*' is the only logical conclusion. Unfortunately, we should be quick to subsume that the fact I go running today *does not* allow us to conclude with absolute certainty that today is indeed a Tuesday. We can, however,

fall back on a weaker syllogism:

$$\frac{\textbf{if } A \textbf{ is true, then } B \textbf{ is true}}{\begin{array}{c} B \text{ is true, therefore } A \text{ becomes more plausible} \\ A \text{ is false, therefore } B \text{ becomes less plausible} \end{array}} \tag{2.2}$$

By observing the fact that I am running we are able to increase our confidence in the original proposition, despite there being no direct line of causality to allow us to say so. In reality, it is this kind of *plausible* reasoning that we are faced with in the majority of our day to day lives. Given further consideration of our example, despite my best intentions I could hardly guarantee that I will go running so consistently, meaning that we really need to make use of an even weaker syllogism:

$$\frac{\textbf{if } A \textbf{ is true, then } B \textbf{ becomes more plausible}}{B \text{ is true, therefore } A \text{ becomes more plausible}} \tag{2.3}$$

In performing this thought experiment, we should hopefully be aware that the human brain is not only able to reason whether something is more or less plausible, but also to evaluate the *degree* of plausibility. Jaynes (2003), from which we unashamedly draw much inspiration for this section, provides the example of predicting the chance of rain given the formation of clouds. If given the opportunity to forecast the likelihood of rain, a healthy majority would reply with some form of percentage. It seems, therefore, that plausibility and its quantitative characterisation are encoded into the human brain. Perhaps more interesting is that, in evaluating the plausibility of new problems, we make use of *prior knowledge*. This is all the more remarkable when we realise that new problems can be a considerable abstraction from that which we have already seen. This complicated process happens almost instantaneously, yet, as Jaynes remarks, we frequently conceal it as merely *common*

sense.

The need for plausible reasoning arises from the fact that *uncertainty* is endemic in the world we, and certainly our robot, will encounter. Whether the environment itself can deliver uncertainty is potentially a source of debate, although modern physics would suggest that it is *at least* a feature of the universe at nanoscale. For our purposes, however, uncertainty is epistemic and arises through incomplete or imperfect observation and cognitive limitations, as well as shortsighted analysis and ignorance of a problem domain. Given this, we acknowledge that the state of uncertainty, like knowledge, is peculiar to the individual and his or her experience. Of course, our robot should be able to take steps to reduce uncertainty through directed observation from which it can, or at least ought, to learn. This objective is the primary aim of the field of Machine Learning, to which we associate the content of this thesis.

The pressing need here is to possess a consistent framework for the quantification and manipulation of uncertainty. Once we have this, we can begin enabling the brain of our robot to reason, formulate predictions, and select courses of actions which in turn may affect the outcome of future events. Fortunately, the very set of tools we seek are laid out in the language of probability, specifically the *Bayesian* approach in which probabilities represent *degrees of belief* in propositions. Probability theory has evolved over many epochs to arrive at this particular formulation, and whilst here we extol the virtues of Bayesianism, even now there are still plenty who do not agree with its fundamentals.[1] Nevertheless, in the next section we introduce Bayesian thinking, and hope that it is found by the reader to be entirely consistent with the description of plausible reasoning just provided. In truth, that is a prerequisite of its invention.

[1] As noted in Cox (1946), the true number of schools of thought when including minor differences is somewhere between two and the number of authors writing on the subject.

2.2 Bayesian Probability Theory

It is almost inevitable that introductions to probability start with discussions of clay pots containing different numbers of coloured balls. Unfortunately, after initially attempting complete avoidance, we quickly realised that any re-characterisation in terms of modern day artefacts[2] was at best painfully contrived. So, let us take our urn and in it place a selection of black and white balls, indistinguishable except by colour. For the following, we assume that the reader has a certain basic understanding of probability theory, as our main objective here is to assert both notation and interpretation.

A blind-folded child enters the room containing the urn and is asked to pick out balls one after the other, sometimes returning the selected ball to the urn and sometimes not. Let us assume, as the problem dictates, that we are interested in assessing the validity of various propositions about the colour of the chosen balls: A - 'the first ball is black' ; B - 'the second two are white'; and so forth. It turns out that we can answer complex questions about such propositions through the specification of two fundamental laws, namely the *sum* and *product* rules:

$$P(A) + P(\neg A) = 1 \tag{2.4}$$

$$P(A, B) = P(A)P(B|A) = P(B)P(A|B) \tag{2.5}$$

where $\neg A$ implies the negation of the proposition A, and $P(A, B)$ refers to the probability of the conjunction $A \wedge B$. Importantly, conjunction and negation form a sufficient set of operations to create all functions of propositions. As a word on notation, we use $P(A)$ to refer to the probability of A, and $P(A|B)$ the probability of A given proposition B is known to be true.

Through rearrangement of the sum and product rules as well as symmetry,

[2]We considered picking Apple or Samsung phones out of a rucksack...

$P(A, B) = P(B, A)$, we arrive at the following:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{2.6}$$

which is referred to as *Bayes' rule*. As shall become clear, this simple rule will form the backbone of our robot's reasoning process. Using the sum rule we are free to redefine the denominator in Bayes' rule in terms of quantities appearing in the numerator:

$$P(A) = \sum_{B} P(A|B)P(B) \tag{2.7}$$

This expression means the denominator can hence be seen as a normalisation constant, required to ensure the conditional probability on the left hand side sums to one over all values of A.

Let us consider the urn in more detail. Imagine that at the start of the problem we inserted one black and two white balls into the urn. We would not argue that the probability of producing a white ball on the first draw would be $\frac{2}{3}$, which through our fundamental laws means the probability of a black ball must be $\frac{1}{3}$. The question we now pose is what the 'probability' of that event represents. Two schools of thought have abjectly different views on the answer. One view (Fisher, 1922), variously called frequentist, orthodox, Fisherian or piscatorial probability, begins by imagining the indefinite repetition of the exact same experiment. Doing this, one would observe a certain proportion of the experiments resulting in a white ball and the rest in a black ball. The proportion, which barring any unforeseen bias would tend towards that which we have calculated, forms the frequentist definition of probability. On the other hand, Bayesians treat the probability as a *belief*, or *reasonable expectation*, in the outcome of that single experiment. That is to say, the probabilities just calculated imply a white ball is a more likely result of a single trial than that of a black ball, where the numbers $\frac{2}{3}$ and $\frac{1}{3}$ enable comparison of the likelihoods.

The frequentist understanding of probability is, by definition, a characteristic of the indefinite set of repeated experiments, which we will call the *ensemble*. Cox states that without the ensemble this characteristic cannot be said to exist (Cox, 1946). In the large majority of singleton events we would like to reason about, the notion of an ensemble would be, again as put forward by Cox, no more than a convenient and sometimes barely tenable mental artifice. Our belief in Bayesian probability is at least partly motivated by the needs of our robot brain to manage non-repeatable situations. It would however be improvident to favour this approach simply on that nuance.

At this point, we must express more precisely what we wish the extended logic of our robot brain to look like. Jaynes (2003) provides the following three *desiderata*[3]:

(**i**) Degrees of plausibility are represented by real numbers.

(**ii**) Plausibilities have qualitative correspondence with common sense.

(**iii**) Reasoning is consistent.

A fuller description of these is made in the quoted text, although we will briefly summarise. The first point is to all intents and purposes forced on the robot brain by virtue of having to define a physical process. The natural convention, which we adopt here, is for a higher degree of plausibility to be represented by a higher number. The interpretation of the second point is potentially subjective, however we comprehend it in the context of the goals set out in the last section. By this, we mean that our robot should reason in a way analogous to that of a rational thinking human. The last point better elucidates what 'rationality' means in the context of the robot, and includes assertions like "if the plausibility of a statement can be derived in two ways,

[3]The author notes a somewhat pleasing similarity between several fictional ideas of a thinking machine that also make reference to a three point directive, not least of which being Asimov's 'Three Laws of Robotics' (Asimov, 1942).

the two results must be equal" (Arnborg and Sjodin, 2001).

We do not wish to deprive the reader of greater explication; however the results we now reference are significant and require more space to present than we could commit here. Vastly paraphrasing, it turns out that by adopting a reasonable set of postulates about how degrees of belief should behave, set out in Cox (1946) amongst others, it is possible to both satisfy the desiderata just laid out as well as derive the fundamental sum and product rules. Bayesian theory, which encapsulates these results, therefore gives us the consistent framework for quantifying uncertainty that we need to proceed.

We should note that both Bayesian and frequentist views have their respective conceptual advantages and practical shortcomings. A Bayesian 'subjective' probability can be stated for an event that escapes repetition, yet requires a person to state consistent beliefs about arbitrarily complex systems of hypotheses (Kuss, 2006). Frequentists argue that such assignments are inherently arbitrary and can lead to incomparableness of scientific conclusions. We must also state that asymptotically both approaches tend towards the same conclusions, and the mathematical laws of probability calculus apply equally to both notions.

Apart from those already mentioned, for our purposes the biggest difference between each interpretation is the manner in which randomness is attributed in a statistical model. The frequentist view states that only data is random, or partially random, and that the values of parameters are fixed and should not be considered as random variables. Bayesians argue that probabilities can be used to represent uncertainty about quantities including those which cannot be directly observed, such as the parameters in a model. This leads to substantially different processes of inference.

2.2.1 Bayesian Inference

Now we have explained our reasoning behind using Bayesian analysis, let us formalise the process of inference this prescribes. In order to do this we first provide some definitions and notational details. In general, we will be interested in the value of some (unknown) variable and will make statements about this value in terms probabilities. If x is the *random variable* of interest then $p(x)$ is a probability statement about x, whilst $p(x|y)$ implies a probability statement about x conditioned on another variable y. To be more specific, we use the upper-case $P(\cdot)$ to refer to the probability of an event, for example 'the ball is white', and the lower-case $p(\cdot)$ to denote a *probability density function* or pdf:

$$P(a \leqslant x \leqslant b) = \int_a^b p(x)\,\mathrm{d}x \tag{2.8}$$

for which the following must also be true:

$$p(x) \geqslant 0 \tag{2.9}$$

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1 \tag{2.10}$$

In this case x is implicitly continuous, although sometimes the random variable may be discrete. In that case we use $p(\cdot)$ to define a *probability mass function*, although we maintain the same notation for both (as per Gelman et al. (2013)).

Using this notation, the starting point for our analysis will always be the specification of a *probabilistic model* $\mathcal{M}$, which provides a hypothesis about how observable data $\mathcal{D}$ is generated. The model details how different elements are dependent on one another in terms of probabilities, and is constructed around parameters θ. As described in the last section, we treat θ as a collection of random variables, since we do not know their true value.

Before we have access to observations, the model allows us to form a distribution over $\mathcal{D}$ as a function of the parameters $p(\mathcal{D}|\theta, \mathcal{M})$. This distribution is referred to as

the *sampling distribution*, and enables the generation of synthetic data for particular values of θ. Once we observe $\mathcal{D}$, we can insert it into the sampling distribution and use it as a function of θ, in which case $p(\mathcal{D}|\theta, \mathcal{M})$ becomes the *likelihood* of the parameters under observations.

The next part of our analysis will be the specification of a *prior* distribution $p(\theta|\mathcal{M})$, which describes any beliefs and uncertainties we have about the true values of θ before engaging with any data. The objective of the inference process is then to form a belief about the parameters conditioned on the data, $p(\theta|\mathcal{D}, \mathcal{M})$. We can arrive at an expression for this through application of Bayes' theorem[4]:

$$p(\theta|\mathcal{D}, \mathcal{M}) = \frac{p(\mathcal{D}|\theta, \mathcal{M})\, p(\theta|\mathcal{M})}{p(\mathcal{D}|\mathcal{M})} \tag{2.11}$$

As we did in the last section, we can also reformulate the denominator as a normalising constant:

$$p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|\theta, \mathcal{M})\, p(\theta|\mathcal{M})\, \mathrm{d}\theta \tag{2.12}$$

which is referred to as the model *evidence* or *marginal likelihood*.

Once we have a distribution over the parameters in our model, quite often we will want to take the inference further and make predictions about variables dependent on these parameters. For example, imagine a linear model which specifies the output of a function $f(x, \theta)$ evaluated at input locations x, and where through application of Equation (2.11) we have a posterior distribution over the model weights $p(\theta|\mathcal{D}, \mathcal{M})$. This yields a joint distribution over both the function output and the model parameters, given by:

$$p(f, \theta|\mathcal{D}, \mathcal{M}) = p(f|\theta, \mathcal{M})\, p(\theta|\mathcal{D}, \mathcal{M}) \tag{2.13}$$

[4]As noted in Osborne (2010), proceeding as though the laws of probability must necessarily apply to continuous variables can lead to error. However, given certain restrictions, pdfs can be used almost exactly as if they were probabilities (Bretthorst, 1999).

In order to calculate the distribution over the function output alone, where the frequentist approach would only allow us to consider one value for θ, the Bayesian approach allows us to consider all values of θ weighted by their relative probabilies. This process of *marginalisation* is achieved through judicious application of the sum and product rules:

$$p(f|\mathcal{D}, \mathcal{M}) = \int p(f|\theta, \mathcal{M})\, p(\theta|\mathcal{D}, \mathcal{M})\, \mathrm{d}\theta \tag{2.14}$$

To perform Bayesian analysis we only need to 'crank the handle' defined by the fundamental rules already described, without making further assumptions or dismissing relevant data.

2.2.2 Priors

In applying Bayesian inference we must acknowledge that the model, and through that the likelihood and prior, are specified not as a function of data, but through the assumptions of the individual implementing the inference. Practically speaking, the likelihood and prior are usually chosen more or less as convenient approximations. This raises concerns with the prior particularly, which should ultimately convey all information available about θ *before* $\mathcal{D}$ is taken into account. The influence of the prior does reduce as data that is informative about the parameters is added; however it should be stressed that, in complex models with few observations, inaccurate specification of the prior can result in severe consequences.

Perhaps the simplest principle to appeal to when setting the prior is that of *indifference*; that is, an allocation of equal probabilities across all possible outcomes, i.e., a non-informative prior. We could think of this in the context of the roll of a die. By assigning $P(\cdot) = \frac{1}{6}$ for all numbers between 1 and 6 we express maximal uncertainty about the outcome of the experiment. Of course, this is sensible if we

want to convey our total uncertainty about the outcome of that experiment.

In expressing indifference over continuous variables, there are two problems which we potentially face. The first is in the specification of an *improper* prior, e.g., a uniform distribution of infinite width. This is, however, usually avoidable by careful verification that the posterior is a proper distribution. The second regards our intuition which dictates that a uniform prior over a continuous variable would convey maximal uncertainty in the same manner as the discrete case. Unfortunately, re-parameterisation results in the beliefs expressed by the prior over the new variable no longer being uniform. One solution to this is found by Jeffreys (1946), who introduces a likelihood dependent prior that remains invariant under such changes of variable.

Whilst the specification of the prior is a recurrent source of concern for Bayesians and frequentists alike (albeit for different reasons), quite often certain concessions are made in order to provide a reasonable compromise between accurate representation and tractability. In much of the work presented in this thesis we will make use of priors which are of a parametric form *conjugate* to the likelihood, meaning the resulting posterior is of the same family of distributions as the prior. Once we have a adopted this form, it is necessary to alter the parameters to adequately represent our uncertainty about θ, which we often do by visualisation of the resulting distribution or through generating data *a priori*.

We note one last feature of the prior that provides a significant justification for the Bayesian approach in general, the result of which is summarised in the chapter epigraph. Remembering that through the frequentist interpretation only single parameter values can be considered, a common means of estimating this parameter is through *maximum likelihood*, i.e., choosing the value of θ that maximises $p(\mathcal{D}|\theta, \mathcal{M})$. Now let us imagine that we flip a fair-looking coin three times, and it happens to land on heads each time. The maximum likelihood approach would predict the probability of heads to be one on the next flip, such that all future flips are predicted as heads.

By contrast, a Bayesian approach with any reasonable prior will lead to a much less extreme conclusion (Bishop, 2006).

2.2.3 Model Selection

In this section we have so far assumed that the model itself is pre-specified, such that all inference takes place over the parameters it stipulates. We can ultimately apply the mechanics of Bayesian inference at a higher level on the model itself, such that we interpret $\mathcal{M}$ as another hypothesis to be evaluated. In this case we may have a set of L models $\{\mathcal{M}_i\}$ where $i = 1, \ldots, L$, each with an appropriate prior distribution $p(\mathcal{M}_i)$. Given a set of observations $\mathcal{D}$ we can evaluate the posterior probability of each model as we did when dealing with θ:

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{M}_i)p(\mathcal{M}_i) \tag{2.15}$$

where we should note that the parameters have been marginalised. Our discussion of model comparison stops there, although we state that the process is all part of the consistent application of Bayesian analysis under uncertainty. Further elaboration can be found in Jaynes (2003). From now on, we will drop the dependence on $\mathcal{M}$ when conditioning on a specific model is implicit.

2.2.4 Expectation, Covariance and Quartiles

When dealing with probabilistic data, it is almost always the case that it is better to operate directly with the distributions themselves as they provide a full characterisation of uncertainty. Nevertheless, we will sometimes want to produce a low-dimensional characterisation. In certain situations the following summary statistics are appropriate:

$$\mathbb{E}[x] \triangleq \int x\, p(x)\, \mathrm{d}x \tag{2.16}$$

known as the *expectation* or *mean*, and

$$\begin{aligned} Var[x] &\triangleq \mathbb{E}[(x - \mathbb{E}[x])^2] \\ &= \int (x - \mathbb{E}[x])^2 \, p(x) \, \mathrm{d}x \end{aligned} \tag{2.17}$$

the *variance*. The square root of the variance is known as the *standard deviation*.

In cases where we have a multivariate distribution, we may also be interested in an extension to variance, known as *covariance*:

$$\begin{aligned} Cov[x, y] &\triangleq \mathbb{E}[(x - \mathbb{E}[x])(y - \mathbb{E}[y])] \\ &= \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y] \end{aligned} \tag{2.18}$$

The covariance of a single variable is simply the variance. As well as giving an indication of the spread of the data like the variance, crucially the covariance indicates how much the variables under consideration *change together*.

The moments of a distribution, the first two of which are the expectation and covariance, are not always an appropriate representation of a distribution. This is particularly so in skewed distributions where the mean typically resides in an area of low density. At several points in this thesis we perform analysis on results that behave in this manner, and for these we look to a different set of summary statistics called the *quartiles*. These are defined through the following equation:

$$\int_{-\infty}^{q} p(x) \, \mathrm{d}x = Q \tag{2.19}$$

where for $Q = \frac{1}{4}$, $\frac{1}{2}$ and $\frac{3}{4}$, q equals the lower quartile, median and upper quartile respectively.

2.2.5 The Gaussian Distribution

We will make use of a variety of distribution functions throughout this thesis, not least of which being the Gaussian Distribution:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{|\boldsymbol{\Sigma}|^{1/2}(2\pi)^{D/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\} \tag{2.20}$$

where D is the number of dimensions, and μ and Σ are the mean and covariance respectively. Note that for the Gaussian, like other popular distributions, we use an alternative notation for the pdf, such that here $p(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We mention the Gaussian as it holds a unique position of popularity in the community. There appear to be two main reasons for this, the first of which is undoubtedly due to the favourable mathematical properties of the distribution which enable the tractable analysis of inference problems. These include, for example: the Gaussian is completely specified by mean and covariance; it is symmetric and infinitely differentiable; linear combinations of Gaussian distributed variables are Gaussian distributed; and, as will become useful in Chapter 5, the marginal distribution of a subset of multivariate Gaussian distributed variables is also Gaussian.

The second reason for the Gaussian's popularity stems from the idea that many physical processes are well described by it. This principle can be related to the *central limit theorem*, which says that the sample mean of a large number of independently distributed random variables with well defined means and variances will be approximately Gaussian distributed. Given this, Jazwinski (2007) notes that a Gaussian component is often a good model for noise in physical devices, due to the effect of a large number of small independent random effects being superimposed.[5] We emphasise, however, that our main motivation for using the Gaussian will almost

[5]We stress that by 'random' we are referring to the epistemic probabilities characterising many small physical processes of which we have little knowledge. Random noise should not be misinterpreted as necessarily being a product of any 'physical' stochasticity.

certainly be due to a combination of the earlier mentioned properties, as well as it being an appropriate characterisation of prior belief (as discussed earlier), rather than a strongly held belief in the relevance of the central limit theorem. Indeed the reader would be right in equating the notion of central limit to frequentist interpretations of probability, which only coincide asymptotically with Bayesian views.

It is worth mentioning at this point that we will often want to assess the likelihood of a dataset, either as a means of testing the ability of a model to generalise, or occasionally to choose parameters. The latter use may come as a small surprise given our presentation so far; however, as we will see throughout this thesis, we often have to make concessions either because of tractability or practicality. When evaluating the likelihood and related quantities of the Gaussian, and ultimately a variety of other exponential forms, it is usually more convenient to operate in the log domain. For example, the likelihood of a set of observations $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of a Gaussian distributed variable under parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ is:

$$p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{2.21}$$

which after taking logarithms becomes:

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{N}{2} \ln |\boldsymbol{\Sigma}| - \frac{ND}{2} \ln(2\pi) - \frac{1}{2} \sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}) \tag{2.22}$$

Because the logarithm is a monotonically increasing function of its argument, maximisation of Equation (2.21) with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ is equivalent to maximising Equation (2.22). This has the effect of smoothing the function, which helps with numerical optimisation, both in terms of the shape of the function and numerical precision, as well as making performance analyses easier to interpret.

2.3 Bayesian Decision Theory

So far we have seen how Bayesian theory provides us with a consistent framework for quantifying uncertainty about the environment. What we have not yet seen is how one should use this in order to best act and influence that environment, which is the problem that forms the main context for this thesis. Specifically, where up until now we have inferred distributions about the world, interaction essentially constrains us to choose *definite* courses of action or predictions. Decision theory is concerned with ensuring that the courses of action chosen are in some way 'optimal' under inferred beliefs.

We will start the discussion on decision theory by considering what value our robot should return when asked for a single estimation of the parameters, given its belief $p(\theta|\mathcal{D})$.[6] There seems to be a number of legitimate hypotheses as to what should be returned, including the mean, median, or maximum, and so forth. The question that has to be asked is: what makes one of these estimates better than another?

In order to answer the last question, let us assume that there is some sort of penalty which is a function of both the estimate made $\hat{\theta}$ and the unknown true value, which we call the *loss function* $\mathcal{L}(\hat{\theta}, \theta)$ (Bernardo and Smith, 2009). Given this assumption, it turns out that Bayesian theory has already provided the solution to the robot's problem through continued application of the sum and product rules. The nuisance parameter θ can simply be marginalised, which gives the *expected loss* for the prediction as:

$$\mathbb{E}_\theta[\mathcal{L}(\hat{\theta}, \theta)] = \int \mathcal{L}(\hat{\theta}, \theta)\, p(\theta|\mathcal{D})\, \mathrm{d}\theta \tag{2.23}$$

The optimal decision is the choice of $\hat{\theta}$ which minimises this expectation:

$$\theta^* = \underset{\hat{\theta}}{\operatorname{argmin}}\, \mathbb{E}_\theta[\mathcal{L}(\hat{\theta}, \theta)] \tag{2.24}$$

[6] A point estimate such as this could only be of relevance to interaction, as it should certainly not form a part of the reasoning process within the robots brain.

Note that optimality is defined with respect to the uncertainty of the decision maker, *not* from the perspective of an omnipotent observer who is able to determine whether the prediction is 'right' or not.

The question therefore returns to the nature of $\mathcal{L}(\cdot)$, for which we will test a few possibilities. If we set the loss to be the absolute error $\mathcal{L}(\hat{\theta}, \theta) = |\hat{\theta} - \theta|$, then the parameter returned will be the median of the posterior distribution. Likewise, if loss is specified as the squared error $\mathcal{L}(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2$, we will get back the mean. Both of these estimates are 'optimal' under one form of the loss function, and therefore it seems we are not any further forward than we were when the question was first posed.

As further motivation, let us look at a medical diagnosis problem in which we are concerned with determining whether a patient has a certain genetic disease through karotyping (chromosome analysis). In this case we will have an input vector $\mathbf{x}$ representing the karotype, and our task is to predict whether the patient has the disease, which will denote as C_1, or is healthy, denoted as C_2. We will assume we have already gone through the appropriate inference stage, which leaves us with a joint distribution of the form $p(\boldsymbol{x}, C_k)$. From this we can make a statement about the diagnoses under the observation $\mathbf{x}$ using Bayes' rule:

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})} \tag{2.25}$$

The prior over the input $p(\mathbf{x})$ is the same for both classes, so for our purposes we can just consider:

$$p(C_k|\mathbf{x}) \propto p(\mathbf{x}|C_k)p(C_k) \tag{2.26}$$

Our task is now to use the above posterior probability to provide a prediction as to whether the patient does or does not have the disease. Intuitively, if we simply want to minimise the chance of assigning a patient to a wrong diagnosis, the *misclassification rate*, then the correct response is to output the class C_k with the highest

posterior probability. For this we need not proceed further. However let us consider the full scale of the problem and decide, based on the model, whether or not to medicate the patient. This puts the problem in a very different context as the decision to not medicate a sick patient could lead to premature death, whilst the penalty for medicating a healthy patient may just be wasted resources.

This leaves us with the problem of choosing a *decision boundary* that divides the input space into two regions, $\mathcal{B}_1$ and $\mathcal{B}_2$, the assignment of which determines whether or not to medicate the patient. As we did with parameter estimation, we can define a loss function $\mathcal{L}(j, k)$, which defines the loss associated with the prescription given and the true state of the patients health. Once again, under this loss function the Bayesian framework essentially solves the problem for us, and we are left to choose the boundary that minimises:

$$\mathbb{E}_\theta[\mathcal{L}(j,k)|\mathcal{B}] = \sum_k \sum_j \int_{\mathcal{B}_j} \mathcal{L}(j,k)\, p(\boldsymbol{x}, C_k)\, \mathrm{d}\boldsymbol{x} \tag{2.27}$$

Unfortunately we are still left with the problem of how to define $\mathcal{L}(\cdot)$. Bishop (2006) suggests a loss function for a similar problem that looks like this:

	healthy	sick
medicate	1	0
don't medicate	0	1000

(2.28)

however this only represents one of a continuum of possible choices.

It is at this point that we concede there is no fundamental way for us to derive appropriate loss functions from the framework of Bayesian analysis. As noted in Jaynes (2003), whilst we have principles to remove the arbitrariness of prior probabilities, we have no such principles for determining loss functions. Any choice for $\mathcal{L}(\cdot)$ is by necessity a subjective choice of the designer of the decision problem. This may

seem unfortunate, but we argue that any decision problem is by definition a consequence of a choice made by the problem setter, and the loss function is simply the characterisation of that choice. The only consideration we stress now is that, when specifying the loss function, we must ensure that it determines *precisely* the decision problem we want solved.

2.3.1 Utility and the St. Petersburg Paradox

The majority of this thesis is concerned with a problem domain, that of *Reinforcement Learning*, in which an explicit 'reward' signal is received by the decision maker. We will go further into exactly what constitutes a reward signal in Chapter 3, although our statement regarding how important it is to carefully specify the loss function prompts some discussion now.

When the environment returns some form of rewarding feedback to an individual we must be careful to notice that, as with specification of the loss function, there is no fundamental rule provided by the universe that dictates exactly how *valuable* that feedback is to an individual. This can be evidenced in no greater way than the contrasting manner in which human beings value the completion of different personal objectives. This principle, however, may not seem so obvious when dealing with feedback that has a well defined social value, in particular that of money. Historically this point has been of substantial concern in a number of fields, and to partially illustrate this we discuss a theoretical problem titled the St. Petersburg Paradox, introduced by Nicolas Bernoulli in Bernoulli (1713b).

The problem concerns a theoretical lottery, and begins with a player paying a fixed entry fee F. Once entered into the lottery, a fair coin is repeatedly tossed until a tail appears signalling the end of the game. Initially the player starts with 1 franc in the bank, and this is doubled every time a head appears. At the end of the game the player wins whatever is in the bank. So if a tail first appears on the nth toss the

player wins 2^{n-1} francs. The question that is posed is how much a rational thinking person should pay to enter.

Considering our discussion of decision making so far, let us propose that our loss function is trivially the amount we pay to enter minus that which we get back, such that $\mathcal{L}(F, n) = F - 2^{n-1}$. If the return exceeds F then we receive a negative loss (a profit). A reasonable argument would be to set the fair price so that the expected loss was zero, i.e., the break even point, and assume that a sensible gambler will only enter if the entrance fee is less than the value of F this equates to. Evaluating the expected loss we find the following:

$$\begin{aligned}
\mathbb{E}_n[\mathcal{L}(F, n)] &= \sum_{n=1}^{\infty} (F - 2^{n-1})\, p(n) \\
&= F - \sum_{n=1}^{\infty} 2^{n-1}(1/2)^n \qquad (2.29) \\
&= F - \sum_{n=1}^{\infty} \frac{1}{2} \\
&= F - \infty \qquad (2.30)
\end{aligned}$$

Accordingly our sensible gambler would play the lottery no matter how large the entrance fee was as the expected profit is infinite regardless. However, on closer examination we find that the probability of actually receiving a high profit is very small. For example a run of 10 heads would payout over 1000 francs but the probability of that occurring is less than 0.001. In fact there is a 50% chance the gambler would receive 2 francs from the game, and a 75% chance of receiving 4 francs or less. Hacking suggested that "few of us would pay even 25 (dollars) to enter such a game" (Hacking, 1980).

Daniel Bernoulli, cousin of Nicolas, proposed a solution to the paradox through a principle of *moral expectation*, or what we now refer to as *utility*. In Bernoulli (1738) he recognises that the true value to a person of receiving a certain amount of

money is not simply a function of the amount, but of both the amount and what that person has already. Bernoulli suggested that a realistic measure of a gambler's utility would be a function of the gambler's wealth W and the logarithm of money. By this approach the utility for the above lottery would be defined as the log difference between the gambler's wealth before and after the event. Substituting the problem of loss minimisation for that of utility maximisation we arrive at the following:

$$\mathbb{E}_n[U(W, F, n)] = \sum_{n=1}^{\infty} \frac{\log(W + 2^{n-1} - F) - \log(W)}{2^n} < \infty \tag{2.31}$$

which provides a more reasonable solution.

Jaynes warns that we should not take Bernoulli's logarithmic assignment of utility literally (Jaynes, 2003). There are plenty of scenarios, see for example Savage (1954), when unaltered application of the approach would yield 'distinctly disadvantageous' decisions. We should instead pay attention to the message of Bernoulli's work, which tells us that a decision maker should not maximise profit itself (or whatever signal the environment yields), but some function of the profit. It is through this function that the whole array of more 'human' judgements on risk, personal value and so forth can be accommodated.

When we come to formally introduce rewards in the next chapter, we will discuss them as if they already represent the output of whatever utility function the decision maker operates under; the functionality for this is encoded in a 'critic' assessing environmental feedback. With regards to the relationship between loss and utility and the process of decision making developed thus far, utility functions can be completely captured within the loss function as already specified. In fact, the two quantities can specify the same objective; whether we choose to minimise a loss, or maximise a utility is mostly down to preference or context. For the most part, we take the less pessimistic avenue and seek to maximise a utility, which in keeping with convention

will be in the form of reward.

2.4 Practical Limitations and Principled Approximation

It may seem that, provided we follow the Bayesian framework, there is little else to do other than follow the principles set forth and provide data for our robot where needed. Indeed a colleague of mine, who shall remain nameless, once came to me in a rather melancholy mood proclaiming that he would shortly be out of a job because Bayes' rule had already solved everything. The reality is of course quite different, and we have already identified a great deal of flexibility in choices of model, priors and utility functions. There is, however, another pressing problem, which revolves around the practical and scalable application of Bayesian methodology.

When developing algorithms either for inference, prediction, or decision making, it is quite often the case that prototyping takes place on synthetically generated datasets operating in low dimensions, in which the application of Bayesian methodology is straightforward. However, when applying these methods to real-world problems we find that quite quickly solutions can become unmanageable. Bellman (1961) refers to this as the *curse of dimensionality*, which describes how the space in which a problem operates can grow, often exponentially. We simply have to accept that the computational resources on which we operate are limited, now and for the foreseeable future. Even without the burden of high dimensions, we frequently find situations in which analytic solutions are evasive and we have to rely on, for example, numerical or variational approximations. Being 'Bayesian' in the ideal sense is not something that can be *wholly* achieved in practice.

Stepping back from this, we believe there are two main ways in which problems of the sort we are interested in here can be approached. The first is to choose a

set of principles that form a cogent framework for analysis, from which we must rigorously follow the theory laid out. It goes without saying that our choice for this will always be that prescribed by Bayesian analysis; however, as just stated, this is not necessarily something that can be achieved in practice. The second possibility is to try and understand intuition about what sort of behaviour would be desirable, and then develop algorithms that in some way appear to do the 'right thing'. The latter approach is what we will generally refer to as *heuristic* methodology. As we will see in Chapter 7, a large proportion of the literature on reinforcement learning is based on sets of heuristics which attempt to mimic a superficial understanding of the problem: trial and error, learning from mistakes, and so forth.

The difficulty in fulfilling Bayesian aims and the patent underperformance of heuristic methods form the main driver of the methodology we present in this thesis. We therefore highlight the need for what we call 'principled approximation'. It is our ardent belief that approximation is a necessity, which in principle we don't find at odds with Bayesian philosophy, although there is quite clearly 'good' and 'bad' approximation. By principled approximation, we mean to understand the underlying mechanisms at play when the Bayesian ideal is followed, so that when we find ourselves unable to fully enact these methods our approximations are based on principles and not rudimentary intuition. It is for this reason that we devote much of this thesis to the understanding and study of what Bayesian theory prescribes. What we suggest is by no means unusual; however we feel that, given our proclamation on the virtues of Bayesian analysis, we are obliged to make this clear.

Chapter 3

Learning from Interaction

> "Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond."
>
> *Animal Intelligence*, Thorndike (1911)

Interacting with the world requires that we choose definite courses of action. We have seen that, given some probabilistic model of the environment, an optimal decision is made by minimising the expected loss under that model. To remind ourselves, the optimal decision a^* is given by

$$a^* = \underset{a}{\operatorname{argmin}} \int \mathcal{L}(a|\theta)p(\theta|\mathcal{D})\mathrm{d}\theta$$

where $\mathcal{L}(\cdot)$ is a loss function, and θ represents the parameters of the model. Now what happens if instead of a single decision, we have a whole chain of decisions to enact? Could we just treat a^* as representing all the decisions in the chain or would that be missing something crucial in the formulation?

This is a fundamental question to address, as the idea of making isolated

decisions is in many ways an artificial construct. Life consists of a whole series of decisions, each one of which affects both the outcome and indeed our *judgement* of the outcome of future decisions. Now it would be hard to argue that *all* the decisions we make are considered with great forethought; your choice to have orange juice instead of coffee at breakfast probability didn't consider how you might get a promotion in a years time, although we should at least admit that it will have some small effect on a chain of consequences that propagates all the way to the end of year review. If life does consist of such a chain of decisions then one option we have is to identify all the goals we intend to achieve, formulate those as $\mathcal{L}(\cdot)$, and choose the set of 'life' actions a^* so as to maximise its expectation. Why then do we not do this?

The first missing component in this reasoning can be identified in the construction of the original decision objective. The optimality of decisions is calculated with respect to a *current* belief in the world $p(\theta|\mathcal{D})$. We emphasise the word 'current' as this will surely change as the result of every decision is observed: we *learn from interaction*. Added to this, our concept of the action space we wish to optimise may very well change as well. Choosing a definite course of action would disregard how new information will affect our future judgement on which decisions are best. Of course, this is different to planning how one *may* act if certain situations present themselves.

The second component is surprisingly well illustrated in an article for Vanity Fair (Lewis, 2012), in which the author describes following US President Barack Obama for six months. Quoting Obama he writes:

> "You'll see I wear only gray or blue suits ... I'm trying to pare down decisions. I don't want to make decisions about what I'm eating or wearing. Because I have too many other decisions to make."

As humans we are limited both in terms of the information we have, but also in our cognitive capacity. We have to prioritise objectives both physically and mentally, and it is almost inevitable that a principle of 'decision management' is adopted by many

successful individuals. This translates directly to our robot, who will certainly be limited in terms of computational ability. If we don't derive decision mechanics that can plan and act under bounded resources, our robot will be stuck choosing what to have for breakfast for the foreseeable future.

3.1 A Reinforcement Learning Problem

Reinforcement Learning (RL) formulates the sequential decision problem just described. It is characterised by a learner and decision maker, the *agent*, interacting with and learning from an *environment* in order to achieve a certain objective. The term 'reinforcement' derives from studies of animal learning in experimental psychology, where it refers to the occurrence of an event coinciding with a response or action that increases the probability of that response occurring again in the same situation (Barto and Dietterich, 2004). RL problems do not provide the agent with examples of desirable behaviour; instead actions are scored against some performance measure which is given to the agent in the form of a *reward* signal.

As an example, and as a break from more traditional illustrations, let us consider the situation faced by a golfer. For the benefit of the reader, the object of golf is to hit a small ball using a selection of clubs into a series of holes. A course is divided into 18 'holes', each one consisting of a tee, from where the ball is first hit, a fairway, down which the ball is played, and a green on which the physical hole is located (see Figure 3.1).

There are various ways in which scoring can be executed, but for simplicity we shall just consider that for each hit of the ball a player accrues a single point. The winner is the golfer who has the fewest points, so we can formalise the problem as one of minimising a *loss function*[1] $\mathcal{L}(s)$, where s is the location of the ball. $\mathcal{L}(s)$ is

[1] As we said at the end of the last chapter, we could equally well look at this problem as one of maximising a *reward function*, $\mathcal{R}$, where $\mathcal{R} \propto -\mathcal{L}$, although loss minimisation is more natural here.

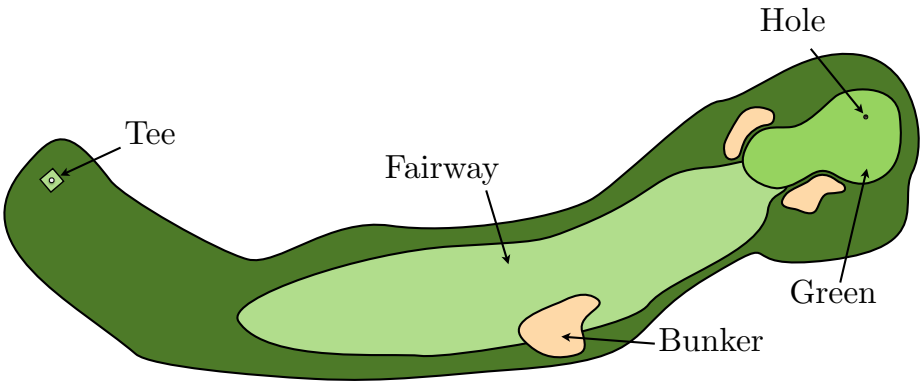


Figure 3.1: Plan view of a golf hole consisting of a tee, a fairway, a green and a number of bunkers (sand hazards).

equal to 0 at the hole, and +1 everywhere else.

The golfer chooses actions in order to place the ball at the location $s = s^*$ which minimises $\mathcal{L}(s)$ (i.e., s^* is the hole) in the least number of strokes. If we assumed she had the ability to perfectly control and predict where the ball landed after each stroke, then whilst standing at the tee she could evaluate all the possible paths that could be taken to the hole, as in Figure 3.2, enacting one of the subset of strokes that minimises $\mathcal{L}(\cdot)$. This is akin to the idea of definite action chains we raised at the beginning of the chapter. The solution to this can certainly be considered within the RL framework, however as should be obvious the problem is more complex.

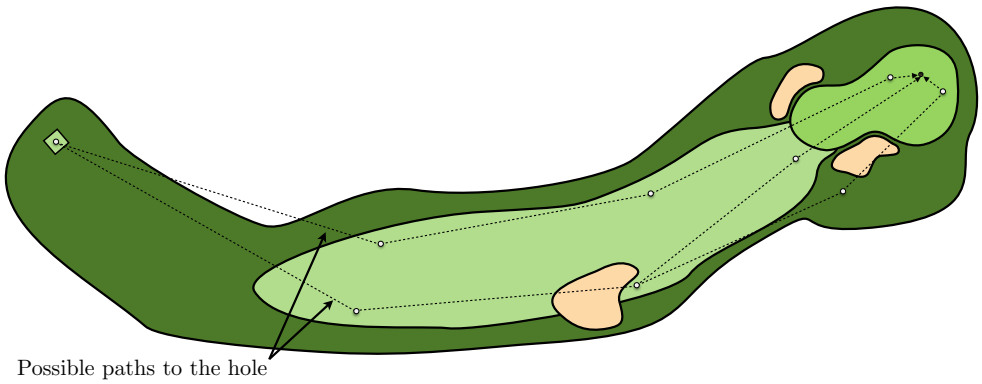


Figure 3.2: A diagram showing possible paths the golfer could take to the hole. The golfer should choose one of the subset of all the possible paths that minimises $\mathcal{L}$.

There are two problems which make planning a path to the hole difficult. To

begin with, even if we just consider the strength and direction of a shot, there are infinitely many decisions and therefore paths than can be chosen from. Without a perfect mental model of the course, the golfer cannot possibly know the result of every stroke that could be made. As such, she is inevitably faced with uncertainty about which path is optimal. In addition to this, uncertainty arises from the many factors which govern the landing position of the ball such as the prevailing wind or roughness of the exact piece of grass the ball lands on. Even variables which the golfer does have some control over, such as the strength and direction of the swing, can also introduce uncertainty due to the sheer number of muscle activations involved. This later uncertainty tends to grow the further the ball is hit. Given these factors, the problem transitions to one in which the golfer must plan where to hit her ball whilst considering uncertainty in both where it will land and the path that should be taken. We illustrate how this may be visualised in Figure 3.3.

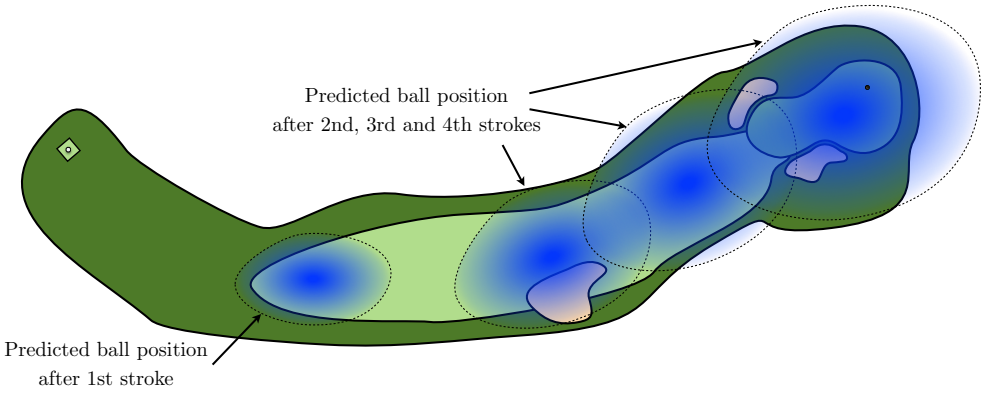


Figure 3.3: A diagram showing the golfer's belief in where the ball will land after each shot, indicated by probability distributions. As later shots are dependent on the position of the first shot, which is itself uncertain, the uncertainty in their prediction is larger the later the shot is (indicated by increasingly sparser distributions).

To understand how a golfer begins to solve these problems, let us consider what happens when a novice golfer approaches the game for the first time. Imagine the novice stands at the first tee and is given the simplest of instructions on how to hit the ball. As she swings the club back for the first time she does not have a very good idea

of where the ball will go, nor will her first shot necessarily be 'good'. However, as she plays more shots her game improves, even without further instruction being given. She engages in a process of learning in which she tries novel strokes, seeing that some increase her loss by making her take longer to get to the hole (negative reinforcement) whilst others help reduce her loss by helping to get to the hole quicker (positive reinforcement). Over time she attempts to maintain the shots which resulted in a lower score, whilst continuing to search for shots that improve her score even further. One can think of the learning just described as a process of trial-and-error, although this implies reactive decision-making rather than deliberative planning. Either way, over time she will both reduce her uncertainty about where the ball will land by improving her accuracy, and also learn which shots help her get to the hole in the least number of strokes.

Of course, a golfer needn't have played a hole numerous times to be able to make good decisions about where to take a shot. As we highlighted in Chapter 2, part of the beauty of the human mind is its ability to generalise previous experience and prior knowledge to novel situations. Imagine you were given a ball to throw and asked to guess where it would land; you would be able to give a reasonably good guess even if you had never thrown that exact ball, just by comparing its shape and weight to other balls that you had thrown before. The golfer can similarly form a belief over where the golf ball is likely to land after any given shot, whether she has been there before or not.

Summarising, the objective of RL is to formulate multi-stage decision making under a global goal, in which the outcome of each action can be considered in terms of the actions that follow. As with many problems, there are multiple ways in which the RL problem can be solved, the biggest distinction being whether a model of the environment is sustained. There are arguments to say that *model-free* methods are more closely aligned with animal decision making and avoid biasing (Sutton and Barto,

1998); however, these are statistically inefficient, as information from the environment is combined with previous and possibly erroneous estimates or beliefs, rather than being used directly (Dayan and Niv, 2008). There is research on 'Bayesian' model-free solutions, for example *Gaussian Process Temporal Difference learning* (Engel et al., 2003, 2005), and work by Ghavamzadeh and Engel on *policy gradient* and *actor-critic* methods (Ghavamzadeh and Engel, 2007a,b). However, given the fundamental statistical inefficiencies we only consider *model-based* methods fully capable of consistency with Bayesian theory and the principled management of uncertainty. Such solutions are in effect an extension to the original loss minimisation problem to which we have a theoretical solution.

3.2 A Reinforcement Learning Framework

In order to generate solutions we must begin with a model of the problem. The manner in which we formulate RL problems here, as *Markov Decision Processes* (MDPs), is the approach most prevalent in machine learning literature. The MDP framework is a good choice as it is abstract and flexible, and represents the main sub-elements of the problem: an environment represented by a *state* variable, a decision maker taking *actions* which cause the state variable to change, and a *reward* function which is a function of both actions and state transitions.

3.2.1 The Agent-Environment Interaction

In our RL framework, the agent interacts with its environment by taking actions. The environment responds to an agent's action by transitioning into a new *state*, and returning a numerical reward to the agent. The agent chooses actions in order to maximise the returned reward values over time.

The interaction between agent and environment occurs repeatedly in discrete

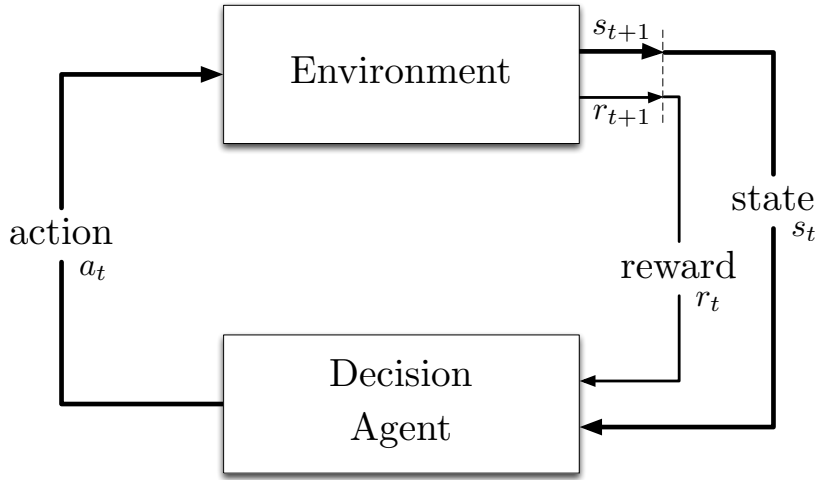


Figure 3.4: The agent-environment interaction in the standard RL model. At time t, the agent senses the state s_t and chooses an action a_t, earning a reward r_{t+1}. Adapted from Sutton and Barto (1998).

time steps, $t = 0, 1, 2 \ldots$ [2]. After t time steps the agent receives a sensory input representing the current state of the environment, $s_t \in \mathcal{S}$, where $\mathcal{S}$ is a set containing all the possible states the environment could be in, and selects an action, $a_t \in \mathcal{A}(s_t)$, where $\mathcal{A}(s_t)$ is the set of actions the agent is able to take at the current state s_t. After one time step, the environment transitions into a new state s_{t+1} and the agent receives a numerical reward r_{t+1}. Figure 3.4 diagrams this interaction.

The boundary between decision agent and environment is not necessarily the same as the physical boundary between an animal's or robot's body and its surroundings. As stated in Sutton and Barto (1998), the agent-environment boundary represents the limit of the agent's *absolute control*. This means that anything that cannot be arbitrarily changed by the agent, such as its physical hardware, limbs, motors, and sensing apparatus, should be regarded as part of the environment. Figure 3.5, adapted from Barto et al. (2004), gives an elaboration of the agent-environment interaction in which we can see the separation of decision agent from the rest of its

[2]Time steps need not refer to fixed intervals of real time; they can refer to arbitrary successive stages of decision-making and acting (Sutton and Barto, 1998), like the interval between strokes in the golf scenario. Additionally, although it is much simpler to operate in a discrete-time framework, it is also possible to approach the problem from a continuous-time setting (e.g., see Hennig (2011)).

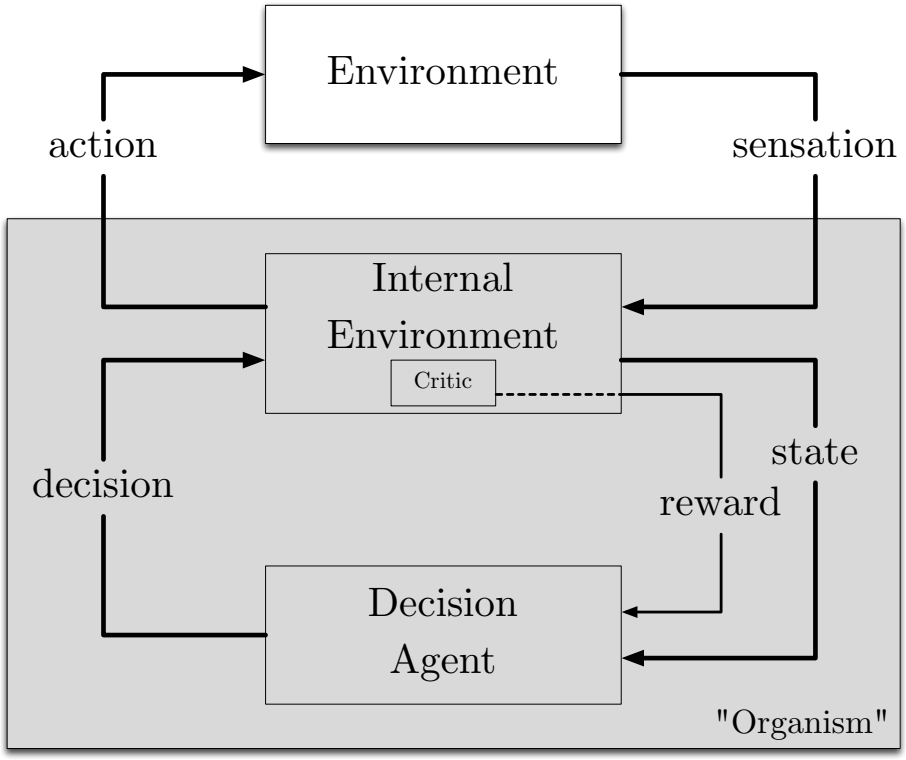


Figure 3.5: An elaboration of the original agent-environment interaction in Figure 3.4. The grey box indicates what we think of as the organism, or body. Inside the body, sensations of the environment are translated into a state signal, and decisions are turned into actions. A critic component analyses the state signal and produces a reward signal for the decision agent. Adapted from Barto et al. (2004).

body, which we call the *internal environment*. Here the (external) environment receives actions, but gives back sensations including observations, which must be turned into a state signal by the internal environment. As the reward function defines the agent's decision task, and thus also cannot be changed arbitrarily, it is also calculated outside the decision agent by a 'critic' component evaluating the agent's behaviour. We will discuss in more detail exactly what the critic does later in this chapter.

3.2.2 States and Markov Chains

To begin our understanding of the RL framework we look at the state signal, which tells the agent about the condition of the environment. The state signal emerges out of a broader need to model how the world changes, both as a function of actions but

also independently of the agent. Probability theory is rooted in games of chance, where experiments, such as the toss of a coin, are independent. However, much of the real-world does not conform to this principle of independence; events that happen in the future are frequently determined or affected by things that are happening now. In order to model this, we look at systems as chains of linked events.

The Markov chain is a mathematical system that transitions probabilistically from one state to another. The chain evolves in steps, changing between states from a set $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, which may or may not be finite. If the system is currently in state s_i, then the probability that the chain moves at the next step to state s_j is denoted by p_{ij}. The individual probabilities, which are called transition probabilities, can be aggregated into a transition matrix, $\mathbf{T}$, indexed by i and j. A feature of the Markov chain is that the transition probabilities are dependent *only* on the current state, not on any of the previous states the system was in. This property, known as the *Markov* property, means that it does not matter how the system got to a certain state, only that it is in that state now. Hence Markov chains are sometimes said to have *independence of path*.

For example, let us consider a simple model of the weather over Oxford. Suppose that the weather on any given day can either be sunny, cloudy or rainy. The probability that a particular day is rainy may be $\frac{1}{4}$, however it does not follow that the probability of two days of rain is equal to $\frac{1}{4} \times \frac{1}{4} = \frac{1}{16}$: wet weather systems frequently last for periods longer than a day, so we would expect it more likely for a rainy day to follow a rainy day than for a sunny one to follow. Figure 3.6 shows how we could model the system as a Markov chain, in which there are three states representing sunny (s), cloudy (c), and rainy (r) weather, represented by *nodes*. Transitions between states are indicated by *arrows*, each of which has an associated probability of occurrence, summarised in $\mathbf{T}$. We can see clearly from $\mathbf{T}$ that the probability of rain tomorrow is higher if it has rained today than if it was cloudy or sunny.

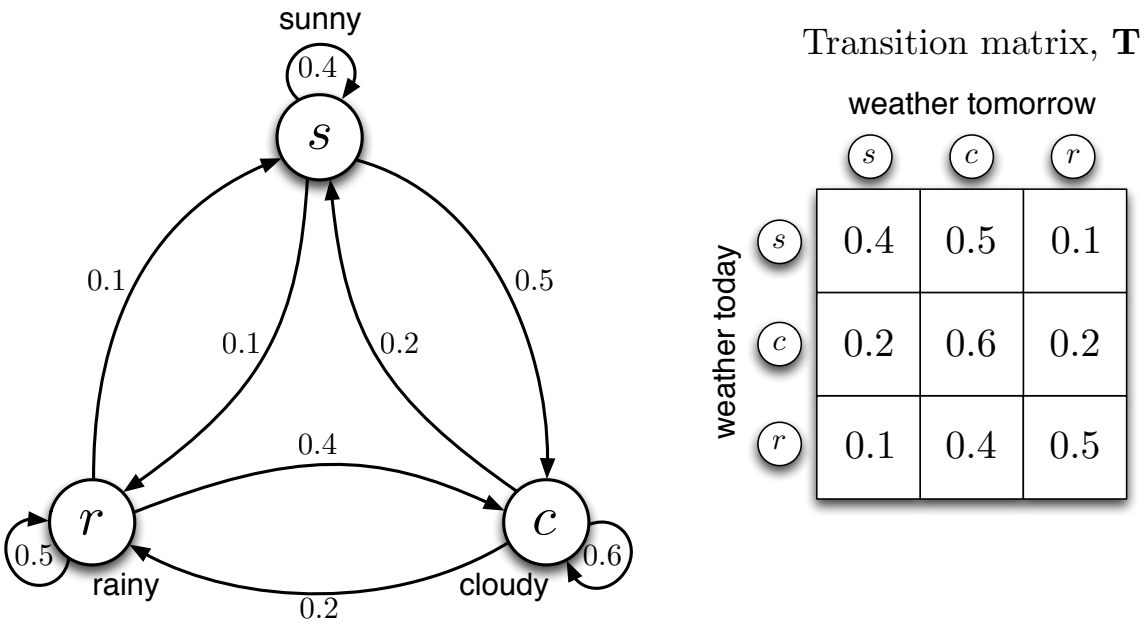


Figure 3.6: A classic Markov chain representing a simple weather model. The weather can take one of three states: sunny (s), cloudy(c), or rainy(r), illustrated as nodes. Transitions between states are indicated by arrows, each of which has an associated probability of occurrence. These probabilities can be summarised in a transition matrix, $\mathbf{T}$, in which rows indicate the weather today, and columns the weather tomorrow.

The Markov property states that the *conditional probability* of the weather tomorrow is *only* dependent on the weather today. This allows for a compact representation of predictions such that when considering the next weather state, w_{t+1}, instead of having to consider the full probability distribution:

$$p(w_{t+1}|w_t, w_{t-1}, \dots w_0) \tag{3.1}$$

we only need consider the following:

$$p(w_{t+1}|w_t) \tag{3.2}$$

One should not get confused into thinking this means that there is no causal relationship between weather more than one day apart. In fact, the weather today surely has influence over the weather in two days, due to its influence over the weather

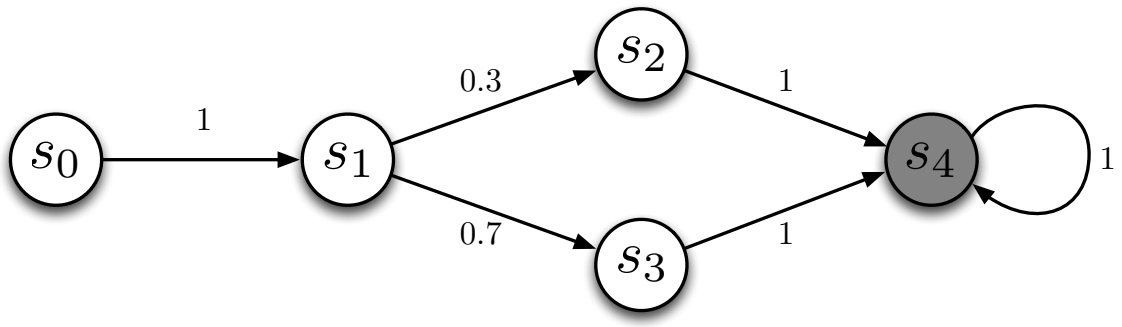


Figure 3.7: A Markov chain representing an episodic task, with several 'deterministic' transitions, and an absorbing state, s_4, illustrated in grey. The chain stays in the absorbing state once reached.

tomorrow. The Markov property simply says that once a state has been observed, all previous states become irrelevant. We can think of this as the past always moving to the future through the present.

If we want to forecast further into the future, we have to consider all the paths the system could take through the chain alongside their probability of occurrence. For example, if we wanted to know the probability of sun in two days given cloud today, we would consider the paths: $c-c-s$, $c-r-s$, and $c-s-s$, summing their probabilities of occurrence, which is the same as marginalising over the weather in one day's time, w_1:

$$p(w_2 = s|w_0 = c) = \sum_{w_1} p(w_2 = s|w_1)p(w_1|w_0 = c) \tag{3.3}$$

This can also be achieved via vector multiplication, using elements of $\mathbf{T}$.

Whilst the weather example defines a continuous and cyclical model, the framework can also represent episodic tasks. Figure 3.7 shows a Markov chain which begins in state s_0, and finishes in s_4. We define s_4 as an *absorbing state* as this allows us to unify notation with continuous tasks, such that the chain effectively comes to rest in the absorbing state. Notice also that transitions can be deterministic, although for generality we will always treat such transitions as stochastic with

transition probability equal to 1.

The State Variable

So far we have referred to 'state' as if it might be a ground truth of the real-world. That is to say that, if we wanted to, we could fully represent the world as a Markov chain, stepping from one state to another. For this to be true, one would have to be able to at least define (if not evaluate) a real-world state which contained all the positions, momentums and forces of every particle in the universe. We are drawn to an idea raised by Pierre-Simon Laplace in his *'Essai philosophique sur les probabilités'*:

> "We ought then to regard the present state of the universe as the effect of its anterior state and as the cause of the one which is to follow. Given for one instant an intelligence which could comprehend all the forces by which nature is animated and the respective situation of the beings who compose it - an intelligence sufficiently vast to submit these data to analysis - it would embrace in the same formula the movements of the greatest bodies of the universe and those of the lightest atom; for it, nothing would be uncertain and the future, as the past, would be present to its eyes."
> Laplace (1814)

This passage is possibly the first articulation of scientific determinism, in which a universe might be well described by classical mechanics. The advent of quantum mechanics and Heisenberg's uncertainty principle largely undermines this idea, but the Markov chain can accommodate randomness, so let us assume for a moment the real-world can in fact be well represented in this way. The real problem from our point of view is that, whatever decision agent we design, it could never fulfil the role of Laplace's Demon[3], the intellect referred to in the passage. Our decision maker will

[3]Laplace never actually used the word *'demon'* in his description of the intellect

most certainly be limited in the computational resources available to it, and the state variable will always be limited by what is observable to the agent.

Given that the state variable will always be a considerable abstraction over the exact state of the world, we must highlight the importance of retaining information pertinent to decision making. For example, in our simple weather model we only look at the previous day to forecast the next day's weather. Our state variable of either sunny, cloudy, or rainy, is clearly a huge simplification over the vast array of variables that actually alter the weather, and as such it may be that our predictions can be improved by including weather from further into the past. This is because information that the previous day's weather gives us with regards to predicting the future may not be fully preserved in our simplified single day state variable. To avoid losing information, we can redefine our current state to represent the weather today and yesterday, without violating the Markov property. We should therefore not think of the state variable as necessarily representing everything that is happening *right now*. State can be highly generalised, representing anything from the physical location and momentum of objects within the environment, to knowledge or memory of past sensations like "it rained earlier". In general, the Markov property requires state to represent more than immediate sensations, but not more than the history of past sensations (Sutton and Barto, 1998). We are concerned here with the general principles of learning how to behave once a state variable has been generated, and therefore we will not discuss further how a state variable is formed, which is a problem in its own right.

An Aside on Mathematical Theology, and the Development of the Markov Chain

In 1713, Jacob Bernoulli stated the 'Law of Large Numbers' (LLN) in his *Ars Conjectandi* (Bernoulli, 1713a), which says that the average result of performing the same

random experiment a large number of times will become closer to an expected value as more trials are performed. For example, if we flipped an unbiased coin repeatedly, as the number of flips went to infinity, the law of large numbers says that the proportion of heads observed would approach $\frac{1}{2}$. Intuitively the LLN makes sense, although a rigorous proof is non-trivial and was not immediately forthcoming.

One might not think an insight such as the LLN could prove overly controversial; simple situations such as the repeated drawing of balls with replacement from an urn (the setting of Bernoulli's first mathematical deduction of a LLN) do not provoke intense discussion of causality (Seneta, 2003). However, when considering the empirically observed stability of averages encompassing human behaviour, a question arises regarding the notion of *free-will* and the ability of man to be responsible for his own actions.

In the early 19th century the Belgian statistician and sociologist L.A.J. Quetelet introduced the study of what he called 'Social Physics', with the intention of understanding statistical laws underlying such phenomena as crime rates, and marriage rates. In his most influential work (Quetelet, 1835), Quetelet presents observations of various kinds of data, noting that there is a stability in the outcomes of certain social behaviours from year to year. His observation that "moral phenomena, when observed on a great scale, are found to resemble physical phenomena", led to him being labelled a fatalist. Although he had originally skirted the issue of free-will, he explicitly refuted accusations of blind fatalism in a later English translation of his work (Quetelet, 1842). He goes on to explain in Quetelet (1847) that "the effect of free-will declines and disappears when observations involve a large number of people"[4]. The most notable result in this paper was a study regarding the number of marriages, by age of the bride, to a bridegroom of age 30 or under, partially reproduced in Table 3.1. Here he notes the stability of the data across a number of years, but argues that

[4]Translation taken from Seneta (2003)

		Year				
		1841	1842	1843	1844	1845
Age of Bride	$\leqslant 30$	12,788	12,422	12,368	13,024	13,157
	$(30, 45]$	2,630	2,626	2,406	2,375	2,438
	$(45, 60]$	93	121	125	129	102
	> 60	7	6	8	5	5
Total Marriages (Brides and Grooms of all ages)		29,876	29,023	28,220	29,326	29,210

Table 3.1: Numbers of Belgian Marriages Classified by Age of Bride and Year of Marriage, for Bridegrooms of Age 30 and Under. Portion of table from Quetelet (1847), also reproduced in Seneta (2003).

one could hardly reason the marriage of a man of less than 30 to a woman of more than 60 could be anything other than a full exercise of free-will (Seneta, 2003).

During the late 19th century to early 20th century the Russian mathematician P. A. Nekrasov, a deeply conservative monarchist, attempted to use the empirical results of Quetelet's social physics to affirm the fundamental Judeo-Christian doctrine of free-will as a necessary condition for the existence of statistical regularity in the social sciences. Nekrasov was a member of the faculty of Moscow University, which at that time was a stronghold of the Russian Orthodox Church. In Nekrasov (1902) he injects the LLN into the discussion, arguing that the LLN *only* applies when events are independent. He says that expressions of free-will are like the independent events of probability theory, and given that data gathered by social scientists conform to the LLN, individuals must act in a voluntary and independent manner. In his proof, he relies on a result by the Russian mathematician P. Chebyshev which says that no more than $\frac{1}{k^2}$ of a distribution's values can be more than k standard deviations away from the mean:

$$P(|X - \mu| \geqslant k\sigma) \leqslant \frac{1}{k^2} \tag{3.4}$$

known as Chebyshev's inequality.

In contrast to the Moscow Mathematical School, and without the religious underpinnings, was the St. Petersburg 'School', founded by Chebyshev, of which A. A. Markov was a member. Markov had a long history of vitriol regarding Nekrasov, he at one point referred to Nekrasov's work as representing "an abuse of mathematics" (Seneta, 1996). Markov took particular exception to Nekrasov's use of mathematics in support of free-will, but rather than attack the validity of the philosophical position, he directed his attention to the underlying mathematics. In Markov (1906) he proved that the LLN also applies to systems of dependent variables that meet certain criteria. The scheme of dependent variables he constructed to prove this is in fact a finite Markov chain. Perhaps the greatest renunciation of Nekrasov in the paper is his lack of reference to the original Nekrasov work, simply concluding with the sentence:

> "Thus, independence of quantities does not constitute a necessary condition for the existence of the law of large numbers."

Although Markov's initial work on chains began in a theoretical domain, in Markov (1913) he explored a practical application in his analysis of Alexander Pushkin's novel in verse 'Eugene Onegin'. Markov took the first 20,000 letters of the poem and split them into two classes: vowels and consonants. He then looked at pairs of successive letters and further classed these into either double vowel, double consonant, or vowel-consonant pairs. Markov's objective was to see how far the poem's text violated the principle of independence. He did this by first calculating the probability that a randomly chosen letter was a vowel to be about 0.43, from which he could calculate the probability of observing two vowels in succession if adjacent letters were independent (0.43^2). The actual number of double vowels observed was three times less than the number that would be expected if the principle of independence held. In fact there was a much greater tendency for vowels and consonants to alternate. Markov's analysis was superficial, but he demonstrated dependence between real-world variables, and thus established chain theory as a tool of probability.

We discuss the process that led to the introduction of the Markov chain partly out of deep admiration for the mathematicians responsible, but also because the narrative is grounded in real-world decision making. Whether or not one gives credence to the notion of free will, the actions of individuals has a direct effect on the environment, which Markov chains allow us to model. Long term stability of Markov chains is not really the focus of our work, although such behaviour does have relevance to our computations.

3.2.3 Actions

We now address the remaining components of the RL framework, starting with *actions* which ultimately describe whatever the agent can 'do' in the environment, for example: the bid-ask pricing of a financial instrument in a market making algorithm; the temperature setting of an intelligent thermostat; or the direction and acceleration of an autonomous vehicle. Actions can be physical, such as the direction of the golfer's swing, or mental and information gathering, such as the golfer deciding to observe the wind direction before her shot. We can even define a 'null' action in which an agent chooses to do nothing; sometimes taking any other action would be detrimental to the goal.

Depending on the decision task of interest, the action space could consist of high-level decisions, such as which town to visit next or which commodity to invest in, leaving the enactment of those decisions to other system components; or low-level controls, such as the voltage used to operate the control surfaces on an UAV[5]. One can imagine a number of high-level and low-level decision agents operating in the same artificial system, each with its own agent-environment boundary. The actions of a high-level agent would become part of the states of a lower-level agent, whose task it is to implement the higher-level decisions.

[5]Unmanned Aerial Vehicle

3.2.4 Rewards

In the problem frameworks we are concerned with, the notion of *reward* is introduced as an agent's sole motivator. RL theory is completely insensitive to the source of rewards, viewing it as a signal to be maximised. This confers the advantage of generality, but defers key questions about the nature of reward functions (Singh et al., 2009). In some systems an agent will receive a direct numerical payoff such as the monetary return on an investment; however these sort of problems are not typical and such payoffs are not what we mean when referring to reward in a RL sense. Rather, we take the view expressed in both Singh et al. (2009) and Oudeyer and Kaplan (2007): that all rewards are *internal*, calculated in the agent's internal environment.

We introduced the particular component that provides rewards to the decision maker as the 'critic' in Figure 3.5. The critic conveniently encapsulates what we described as the utility function in Section 2.3.1, and represents the agent's idiosyncratic *evaluation* of perceived behaviour, the current state of the environment, and any real payoffs. Of course, it is perfectly legitimate for a real numerical reward to have a one-to-one mapping with the reward signal given to the decision maker, but we maintain the internal model for generality. By taking this approach, we are also able to use the same decision mechanics to reach goals whether those goals form a natural reward signal or not.

Reward, as we have defined it, is given to the agent on a step by step basis. That is to say, once a time step has passed, the critic performs its analysis and provides a numerical reward signal to the decision agent. The overall goal of the decision agent when choosing its next action is not just to maximise the reward at the next step, but to maximise the sum of the returned rewards throughout its decision task; we call this sum the agent's *return*, although it is sometimes referred to as *long term reward*.

3.3 Markov Decision Processes

Now that we have described the individual components of our framework, namely states, actions and rewards, we can unify the elements into the Markov Decision Process (Bellman, 1954, 1957b; Howard, 1970). We begin with a Markov chain, but where before state transitions were uncontrolled (i.e., there were no actions involved), the transitions between states of a MDP are dependent on both the current state and an action a chosen by the decision agent. Additionally, every state transition activates a reward process; we refer to these two elements (transition and reward) collectively as the *dynamics* of the MDP.

Formally, our MDP formulation consists of three elements:

- A set of states: $\mathcal{S}$
- A set of actions: $\mathcal{A}$
- The dynamics of the environment: $\mathcal{P}(s', r|s, a) = P(s_{t+1} = s', r_{t+1} = r|s_t = s, a_t = a)$ is the probability of the next state and reward pair (s', r), when starting in state s and taking action a.

Figure 3.8 shows an example of a MDP with two states, up to three actions, and a mixture of deterministic and stochastic transitions resulting in various rewards.

It is important to note that in many MDP formulations, for example that in Sutton and Barto (1998), the transition and reward processes are explicitly separated. Ultimately, however, both are a product of a change in the environment which, as we discuss in section 3.4, means there is a duality between stochastic rewards and transitions that makes the grouping into a single 'dynamics' element sensible. This formulation also simplifies notation without preventing access to any quantities we need to know. For instance, for the state-action pair (s, a), we can find the state

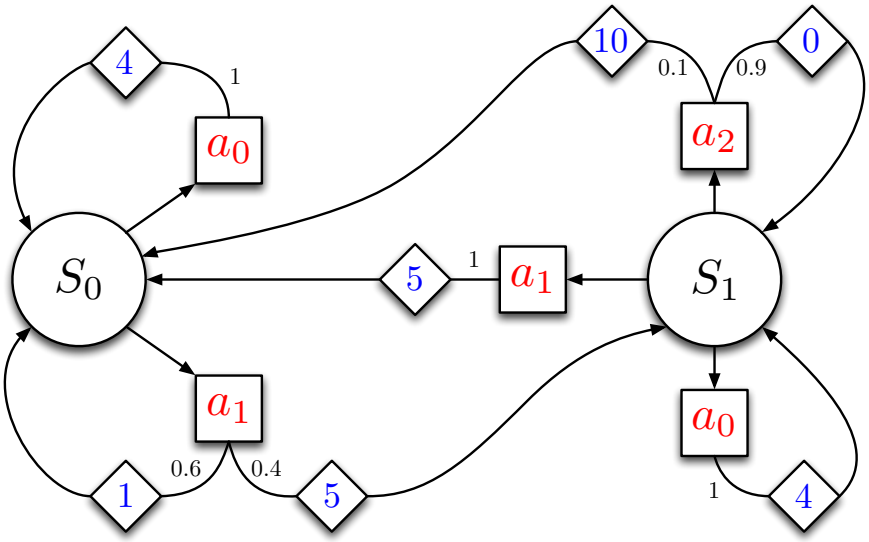


Figure 3.8: An example Markov decision process with two states. State nodes are illustrated as circles with black text, action nodes as squares with red text, and reward nodes as diamonds with blue text. State transitions are dependent on the current state and the action chosen; depending on the transition, a reward is given to the agent. In general, rewards, as well as transitions, may be stochastic.

transition distribution:

$$\mathcal{T}(s'|s,a) = P(s_{t+1} = s'|s_t = s, a_t = a) = \sum_r \mathcal{P}(s', r|s, a) \tag{3.5}$$

and the reward distribution:

$$\mathcal{R}(r|s,a,s') = P(r_{t+1} = r|s_t = s, a_t = a, s_{t+1} = s') = \frac{\mathcal{P}(s', r|s, a)}{\mathcal{T}(s'|s, a)} \tag{3.6}$$

The following relationships are also true:

$$\mathcal{P}(s', r|s, a) = \mathcal{T}(s'|s, a)\mathcal{R}(r|s, a, s') \tag{3.7}$$

$$= \mathcal{T}(s'|s, a, r)\mathcal{R}(r|s, a) \tag{3.8}$$

which is useful as we will often wish to specify $\mathcal{T}$ and $\mathcal{R}$ directly. Note that we may occasionally treat $\mathcal{T}$ and $\mathcal{R}$ as deterministic functions, in which case our notation will

take a form equivalent to $s' = \mathcal{T}(s, a)$ or $r = \mathcal{R}(s, a, s')$.

3.3.1 Objective of Decision Maker

The agent takes actions sequentially, earning a reward after each action and state transition. The goal of the agent is to finish its task with the maximum cumulative reward, which we call the agent's return, R:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots + r_{\mathcal{H}} \tag{3.9}$$

where r_t is the reward received after taking an action at time t. The time horizon $\mathcal{H}$ represents the total number of actions the decision maker can take and thus the number of rewards that can be received.

Rather than select actions directly, the agent finds a mapping from states to actions called the policy, π:

$$a_t = \pi(s_t) \tag{3.10}$$

The policy prescribes the action to be taken both in the current state and any possible future state, and therefore forms a sequence of decision rules (Puterman, 1994). Implementation of a particular policy yields a sequence of rewards. The policy that maximises *expected* return, for all $s_t \in \mathcal{S}$, is called the optimal policy:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}\left[R|\pi\right] \tag{3.11}$$

The optimal policy may not be unique, in which case π represents the set of all optimal policies.

It it is important to note that a policy is not the only way to act in a MDP. Indeed, the framework itself does not define the objective of the agent, and therefore

actions *could* be chosen however we please. We operate in the domain of the policy as it provides a mechanism for the decision maker to plan and consider the likelihood of future actions, transitions and rewards. This is clearly necessary if the decision maker is looking to directly maximise the expected return.

Most introductions to MDPs (including Puterman (1994) and Sutton and Barto (1998)) allow policy to be stochastic, such that $\pi(a|s)$ is defined as the probability of taking action a when in state s. For generality we accommodate this definition here; however, as we shall discuss later in the thesis, one must be very careful when dealing with stochastic decision making. We could imagine a stochastic policy as being the result of the imperfect and noisy execution of decisions; however it is more common for stochastic decisions to be used as a sub-optimal heuristic method for gaining information. If we are not careful, this can result in the agent taking actions which it would, given proper consideration, deem to be contrary to its goals.

The approach of finding the optimal policy defined in Equation (3.11) is acceptable for episodic tasks; however in many cases the interaction between agent and environment forms a continuous decision problem, with no discernible end to the task. In such cases the horizon would be $\mathcal{H} = \infty$ and the expected return, which is what we are trying to maximise, could also be infinite. To avoid this problem, it is common to introduce a discount parameter, $\gamma \in [0, 1]$, into the expression for return. The expression for expected *discounted* return is as follows:

$$\begin{aligned} \mathbb{E}\left[R_t\right] &= \mathbb{E}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots\right] \\ &= \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1}\right] \end{aligned} \tag{3.12}$$

Whether R_t is discounted or not should be obvious from the context. Using this approach, more immediate rewards are valued higher than rewards further away.

3.4 Uncertainty in MDPs

Uncertainty can arise in a number of places in the RL framework. One form of uncertainty can be characterised by a belief over some element of the model. It is a product of inference that, as new information becomes available, such beliefs change. The optimality of actions is clearly entangled with currently held beliefs, and as such we have what appears on the surface to be two conflicting aims. In the first instance we want to maximise the expected return, but at the same time we understand that the evaluation of this can change under new information. The second aim would therefore appear to be the need to reduce uncertainty, so as to make 'better' decisions. The need to balance these two aims is known in the literature as the *exploration versus exploitation tradeoff*, and features as a central theme across much of machine learning, artificial intelligence, and the behavioural sciences.

In Chapter 6 we explore the notion of a tradeoff between exploration and exploitation in detail, but we must make an initial statement on the subject now before moving further into the thesis. The reader should be very clear that, if we follow the mechanics of Bayesian analysis in the framework of an MDP, the tradeoff does not exist. Information gain is efficiently incorporated into the main objective of return maximisation, such that it does not need to be considered separately. It is very important to understand this now, as a large proportion of research in RL separates learning and reward seeking as two competing aims. Quite often this is due to the use of heuristic methodology, although that being the reason for doing so is very rarely acknowledged. It is also undoubtedly related to perceptions of animal behaviour where the distinction is highly prevalent, see for example Daw et al. (2006) and Gold and Shadlen (2007).

Uncertainty which can be reduced through observation specifically refers to that which the decision maker has about elements of the model, most typically the part of

the model that describes the dynamics[6] $\mathcal{P}$. Throughout much of this thesis we will treat both rewards and transitions as stochastic in their own right. This introduces a different form of uncertainty, one that cannot be reduced through observation. We must be clinical in differentiating between 'real-world' randomness and an agent's uncertainty about a model, as both potentially involve some form of probability distribution.

Bayesian inference can be used to form a belief about unknown quantities. By modelling such quantities, we implicitly act as if they are a 'ground truth' of the environment. That is to say: under perfect information, our belief distributions would collapse to the true value. In contrast, when we discuss stochastic dynamics we model the world as if there is something inherently *unpredictable* about it. In this case, under perfect information in the model we would still have uncertainty about the real world variable of interest. Typically we will model this stochasticity such that our inference procedure learns the parameters of, for example, a particular distribution function. Given this, it is possible to have *confidence in uncertainty*, where beliefs characterising the parameters of the function are concentrated, whilst predictions about the random variable remain broad.

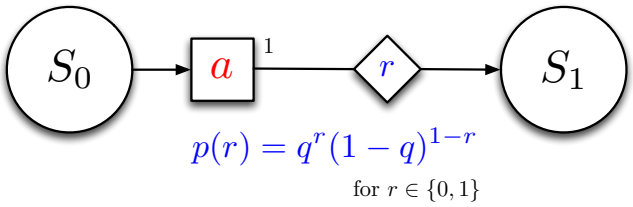
We emphasise that the consideration of stochastic dynamics is a product of modelling; one should not get confused into thinking that such randomness must come from the environment itself. The main reason for explicitly describing the world in this way is due to limitations in both the complexity of a model and the extent of observations. We mentioned earlier that the state variable is nearly always a considerable abstraction over the true state of the environment; in this case, the environment will not always respond in exactly the same way, even when similar actions are made. This is due to minor (or even major) perturbations in ground truth variables which the agent either cannot detect, or is not modelling. To appreciate

[6]Although we don't consider it here, uncertainty can also be expressed over the state variable, in which case the model becomes a Partially-Observable Markov Decision Process (POMDP).

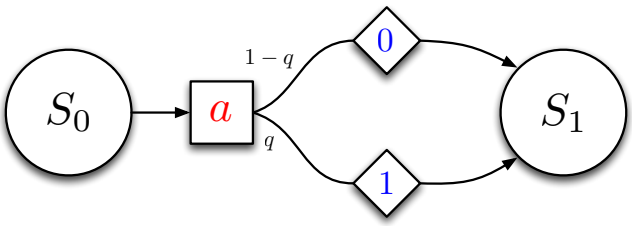
what this means, imagine repeatedly throwing a tennis ball against a wall, aiming for the same spot each time. The ball will almost certainly hit the wall in a grouping around the point being aimed for. This result is clearly not to do with some strange quantum randomness or other worldly stochastic effect; rather it arises as there are many elements that we cannot *precisely* control or observe, such as the air density, roughness of the ball, or exact muscle activation. Probability theory can characterise this inherent uncertainty, such that it can be incorporated into the MDP model as if it were a ground truth.

With regards to stochastic rewards, stochastic transitions, or even stochastic actions, there is a certain duality in the concepts that could allow us to describe one in terms of the other. Figure 3.9(a) shows a simple two state, single action MDP in which reward is drawn from a Bernoulli distribution (given in the figure). The expected reward when taking action a is equal to q. We could re-define this example, as in Figure 3.9(b), such that instead of reward being stochastic, two possible paths are followed after taking action a, each giving back a deterministic reward of 1 or 0. From the point of view of the decision maker, there is no difference in the perceived reward distribution from this arrangement. Lastly, we could look at the same situation as one of stochastic action taking. In Figure 3.9(c) we see the agent selects an action a, indicated by the shaded box, and either a_1 or a_2 is actually executed. Each action leads on to a deterministic reward. Once again, the decision maker sees no difference in the reward distribution.

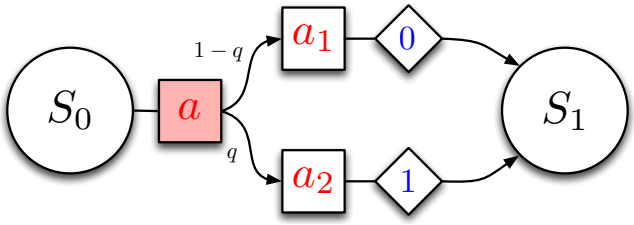
Given our definition of an MDP, only the first approach is valid: $\mathcal{P}$ does not differentiate between multiple paths between the same two states, and there is no mechanism for actions to be selected stochastically (as a product of the MDP rather than a function of the policy). Although this is the case, the latter two approaches could actually be better grounded in real-world decision making. If we return to thinking about the tennis ball against a wall scenario, stochastic transitions are analogous



(a) Stochastic reward



(b) Stochastic transition



(c) Stochastic action

Figure 3.9: Diagram demonstrating the duality of stochasticity in rewards, transitions and actions. From the point of view of the decision maker, the reward distribution the arises from the one possible action is the same in all three examples. a) Reward is stochastic with $P(r = 1) = q$ and $P(r = 0) = 1 - q$, b) transition is stochastic such that paths with different reward functions are followed, and c) actions are stochastic such that the agent chooses an action a, and either a_1 or a_2 is enacted.

to our inability to completely predict all the external factors that could effect the ball's path to the wall; we also cannot perfectly control the movement of our arm, which could be seen as the enactment of stochastic actions conditioned on a selected action. However, the notion of stochastic reward neatly encapsulates *both* forms of stochasticity, so we usually remain with that approach.

Chapter 4

Bayesian Optimal Decision Making

"There is a valid defence of using non-Bayesian methods, namely incompetence"

Skilling (1991)

It may come as no surprise that the solution to the full MDP lies in the principled application of Bayesian analysis as already laid out. In order to aid understanding, we present the solution in two key steps. In this chapter we present the first step and reformulate the mathematics of MDPs in terms of a recursive set of equations. This allows us to define an algorithm from which an analytic solution to the sequential decision problem can be derived. Crucially, we do this without the burden of uncertainty in the model parameters. As such, all questions related to the exploration vs. exploitation tradeoff are deferred until Chapter 6. The reason for doing this is very simple: once we have the ability to solve MDPs under known model dynamics, the general case follows only by redefinition of the state variable to accommodate *states of knowledge*.

4.1 Value Functions

Almost all methods designed to solve RL problems are based on estimating *how good* it is for an agent to be in a particular state, or to take a particular action, in the context of the goal of maximising cumulative reward. We can formally define this as finding a *value function* over the state or state-action space. Although value functions help generate a solution, they should be seen simply as a restatement of the mathematics of MDPs. Before we present the mathematics, we ground the idea of value in a real-world problem, and return to the golf example from the beginning of Chapter 3.

Imagine the golfer is just about to take a shot. Her ball is resting on the edge of a green at the foot of a gentle incline that slopes up to the hole before dropping steeply away, as in Figure 4.1. Let us assume that the golfer can choose to hit the ball with either a hard or soft putt. Before choosing which putt to take, she can predict where the ball is likely to end up after each one. In Figure 4.1, we superimpose two probability distributions, which can now be thought of as being defined by $\mathcal{T}(s'|s,a)$,

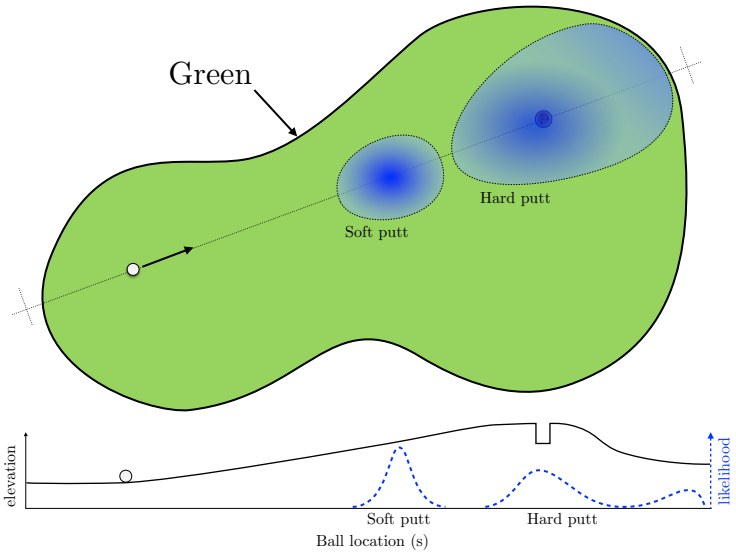


Figure 4.1: A plan view of the problem faced by the golfer on the green, below which is an elevation plot of the green taken along the dotted line between the ball and the hole. Superimposed are probability distributions, defined by $\mathcal{T}(s'|s,a)$, representing the probability of the finishing position of the ball after either a hard or soft putt in the direction of the hole; darker blues represent greater likelihood.

representing the probability of the finishing position of the ball given each of the strokes. We can see from this diagram that a hard putt is more likely to result in the ball entering the hole than the soft putt, however there is also significant risk of the ball rolling to the bottom of the hill when taking that shot.

Once the golfer has predicted where the ball might end up after the first putt, given it is not certain she can hole the ball, she will inevitably evaluate the 'goodness' of other ball positions across the green. For example, if we look at the middle of the green versus the bottom of the slope, even though the actual loss $\mathcal{L}(s)$ is the same at both locations, the middle of the green seems a more desirable place to be as it provides an easier stroke to the hole; how much more desirable it is will determine whether she risks a hard putt which may overshoot the hole. The assessment is equivalent to finding a value function over the green, an example of which is shown in Figure 4.2. Here, higher value is given to locations which make for easier and prompter holing of the ball. The value function ultimately allows the golfer to consider future actions when choosing the next action, allowing locations to be evaluated in

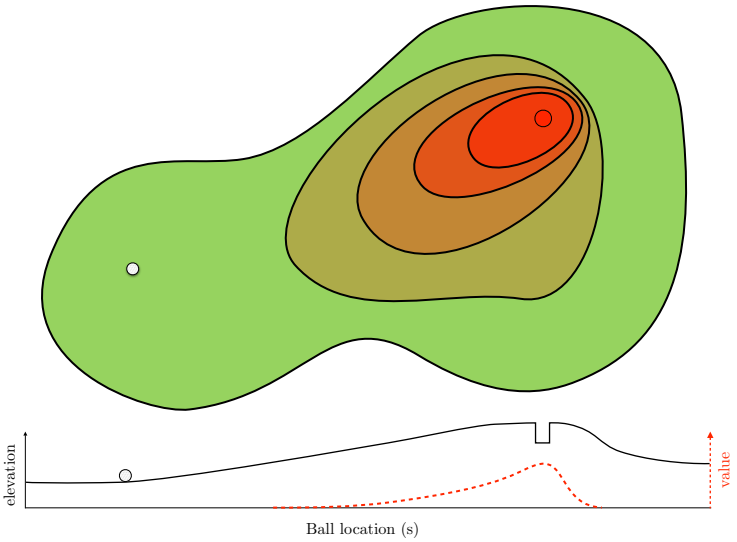


Figure 4.2: A diagram indicating value contours across locations on the green. Higher value, shown in red, is given to locations which make for easier and prompter holing of the ball, whilst lower value, shown in green, is given to more difficult shots.

the context of their relationship to the goal, whether they are rewarding themselves or not. If the golfer didn't value locations in this way, she would run the risk of myopically shooting straight for the goal state (the hole). This may occasionally pay off as the ball will sometimes go in, but in the long run her score will be damaged.

4.1.1 Value Defined

Value is a measure of how much return an agent can expect to receive when in a given state, or when taking a certain action in a given state. As we have seen, expected return is a function of what actions the agent takes in the future, which is defined by the policy. Value is therefore intimately related to the agent's policy, and is calculated with respect to it. We can define the value of a state s_t under a policy π as the expected return when starting in s_t, following π from then on:

$$\begin{aligned} V^{\pi}(s_t) &= \mathbb{E}_{\pi}\left[R_t | s_t\right] \\ &= \mathbb{E}_{\pi}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \middle| s_t\right] \end{aligned} \tag{4.1}$$

where $\mathbb{E}_{\pi}$ is the expected return given policy π is being followed. V^{π} is called the *state-value function*.

As well as the state-value function, we can define an *action-value function*, $Q^{\pi}(s_t, a_t)$, which tells us the expected return when starting in state s_t, taking the action a_t, and from then on following the policy π:

$$\begin{aligned} Q^{\pi}(s_t, a_t) &= \mathbb{E}_{\pi}\left[R_t | s_t, a_t\right] \\ &= \mathbb{E}_{\pi}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \middle| s_t, a_t\right] \end{aligned} \tag{4.2}$$

The policy normally defines a_t, so in using this expression for Q^{π} we are excluding the first action from π.

In our golf example we imagined a problem that could reasonably be expected to be finished within one or two actions, so were not concerned with the value of states beyond the second putt. In longer action chains, one may not be surprised to hear that the value of one state is related to the value of the states that follow on from it. In fact, we can generate a recursive relationship between both state-values and action-values and the value of the states that follow them. By extracting the immediate reward from Equations (4.1) and (4.2), we get the following expressions for state-values:

$$\begin{aligned} V^{\pi}(s_t) &= \mathbb{E}_{\pi}\left[r_{t+1} + \gamma \sum_{i=0}^{\infty} \gamma^i r_{t+i+2} \,\middle|\, s_t \right] \\ &= \mathbb{E}_{\pi}\left[r_{t+1} + \gamma\, V^{\pi}(s_{t+1}) \,|\, s_t \right] \end{aligned} \tag{4.3}$$

and action-values:

$$\begin{aligned} Q^{\pi}(s_t.a_t) &= \mathbb{E}_{\pi}\left[r_{t+1} + \gamma \sum_{i=0}^{\infty} \gamma^i r_{t+i+2} \,\middle|\, s_t, a_t \right] \\ &= \mathbb{E}\left[r_{t+1} + \gamma\, V^{\pi}(s_{t+1}) \,|\, s_t, a_t \right] \end{aligned} \tag{4.4}$$

Note that in Equation (4.4), the expectation operator is no longer conditioned on the policy due to its inclusion in the expression for V^{π}. Both forms of value are a sum of the expected immediate reward and the *discounted* expected value of the next state (conditioned either on the policy or the immediate action, depending on the value function). Expanding the expectation operator in Equation (4.3) gives the following:

$$V^{\pi}(s_t) = \sum_{a_t} \pi(a_t|s_t) \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1}|s_t, a_t) \Big[r_{t+1} + \gamma V^{\pi}(s_{t+1}) \Big] \tag{4.5}$$

and doing the same for Equation (4.4) gives:

$$Q^{\pi}(s_t.a_t) = \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1}|s_t, a_t)\Big[r_{t+1} + \gamma V^{\pi}(s_{t+1})\Big] \tag{4.6}$$

The recursive relationship given in Equations (4.3) and (4.5) is called the *Bellman equation* for V^{π}, named after the American applied mathematician Richard Bellman for work presented in Bellman (1957a). The Bellman equation averages over the value of all the states which could succeed state s_t, weighted according to their probability of occurrence, to arrive at the value for s_t. Equations (4.4) and (4.6) represent the Bellman equation for Q^{π}, which also averages over the value of all possible successor states to arrive at the value for (s_t, a_t). In both forms, information is essentially propagated from successor states *back* to starting state.

The Bellman equation allows us to see the natural tree-like structure that arises from decision problems. Figure 4.3 shows a single 'branch' on the tree. The tree continues splitting from state nodes into action nodes, through the policy, and from action nodes to state nodes, through both the transition and reward functions, up until the horizon state $s_{\mathcal{H}}$ is reached. The representation of the decision problem in Figure 4.3 differs from the representation in Figure 3.8 as it shows the progress of state through time, from left to right, rather than a distinct layout of states. As such, it is possible for nodes at s_{t+1} and s_t to be realisations of the same state $s \in \mathcal{S}$, or in other words for a state to be its own successor. In Sutton and Barto (1998) this representation is referred to as a 'backup diagram', because of the way in which value propagates between nodes. In Section 4.2 we shall see in greater detail how individual decision branches form a full decision tree.

Due to the recursive structure of the Bellman equation, we can define any node on the tree in terms of its dependants. We have already seen in Equations (4.4) and (4.6) how Q^{π} can be defined in terms of V^{π}. We can extend this and define V^{π} in

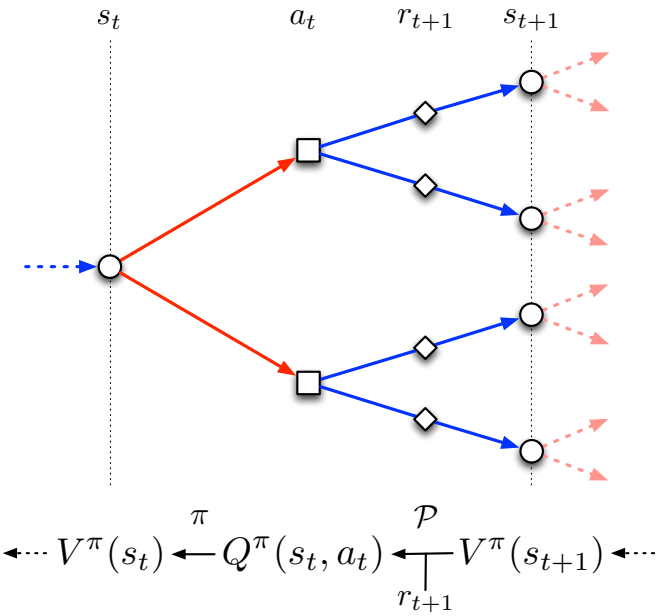


Figure 4.3: One branch of a decision tree giving an example of the relationship between value functions of future states $s_{t+1} = s'$ and the current state s_t for V^π. Here, two actions yield four possible successor states, from which value can be back propagated using the Bellman equation. Nodes are illustrated in the same way as in Figure 3.8, however this representation shows the progress of the MDP through time, from left to right; this means that it is possible for state nodes at s_t and s_{t+1} to be realisations of the same state $s \in \mathcal{S}$.

terms of Q^π as follows:

$$\begin{aligned} V^\pi(s_t) &= \mathbb{E}_\pi \left[Q^\pi(s_t.a_t) | s_t \right] \\ &= \sum_{a_t} \pi(a_t|s_t)\, Q^\pi(s_t.a_t) \end{aligned} \tag{4.7}$$

by using the policy to weight the Q^π values through the expectation operator. We can also define the Q^π at one node in terms of Q^π values later in the tree by using Equations (4.6) and (4.7):

$$Q^\pi(s_t.a_t) = \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1}|s_t, a_t) \left[r_{t+1} + \gamma \sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1}) Q^\pi(s_{t+1}, a_{t+1}) \right] \tag{4.8}$$

This equation is particularly relevant to an estimation method called *Q-learning*, which uses the (estimated) value of later state-action pairs to recursively update the

estimated value of the current state-action pair.

4.1.2 Closed Form Solution

The Bellman equation defines a system of linear equations, which we could in theory solve directly. If, under policy π, $\mathbf{V}^\pi$ is a column vector of individual state values (one for each state $s_t \in \mathcal{S}$), $\mathbf{R}^\pi$ is the equivalent vector of expected immediate rewards, and $\mathbf{T}^\pi$ the transition matrix, then we can express the Bellman equation as:

$$\mathbf{V}^\pi = \mathbf{R}^\pi + \gamma \mathbf{T}^\pi \mathbf{V}^\pi \tag{4.9}$$

Rearranging, the value of $\mathbf{V}^\pi$ can be found:

$$\mathbf{V}^\pi = (1 - \gamma \mathbf{T}^\pi)^{-1} \mathbf{R}^\pi \tag{4.10}$$

Unfortunately this solution offers little in terms of real-world application as it is tractable only for a subset of very small, finite state MDPs. Further to this, in many problems, for example infinite horizon problems or those in which the dynamics are unknown, no closed form solution will be available. In Section 4.2 we will introduce a powerful set of techniques which have been associated with MDPs since their formation, and provide a realistic alternative to the closed form solution.

4.1.3 Optimal Value

In section 3.3.1 we introduced the concept of optimal policy, which is the policy, or set of policies, that gives the largest expected return. We can also look at optimality in the context of value functions. The optimal state-value function, V^*, is the value

function under the optimal policy and is given by:

$$V^*(s_t) = V^{\pi^*}(s_t) = \max_{\pi} V^{\pi}(s_t) \tag{4.11}$$

for all $s_t \in \mathcal{S}$, where π^* is defined in Equation (3.11). For this to be true, the expected return from the optimal policy must be at least as good as that from any other policy. The equivalent optimal action-value function is given by:

$$Q^*(s_t, a_t) = Q^{\pi^*}(s_t, a_t) = \max_{\pi} Q^{\pi}(s_t, a_t) \tag{4.12}$$

Optimal value functions fulfil the recursive relationship defined earlier, which are expressed as *Bellman optimality equations*. The Bellman optimality equation for $Q^*(s_t, a_t)$ can be written in terms of $V^*(s_{t+1})$ by substituting Equation (4.4) into (4.12):

$$\begin{aligned} Q^*(s_t, a_t) &= \mathbb{E}\left[r_{t+1} + \gamma \max_{\pi} V^{\pi}(s_{t+1}) \,\middle|\, s_t, a_t\right] \\ &= \mathbb{E}\left[r_{t+1} + \gamma\, V^*(s_{t+1}) | s_t, a_t\right] \end{aligned} \tag{4.13}$$

This function states that $Q^*(s_t, a_t)$ is equal to the expected return when taking action a_t in s_t, and following an optimal policy from then on. To find the Bellman optimality equation for V^* we first note, by examining Equation (4.7), that the value of a state is maximised if the policy π selects *only* the action with the greatest expected return:

$$\begin{aligned} \max\left\{V^{\pi}(s_t)\right\} &= \max_{\pi} \sum_{a_t} \pi(a_t|s_t)\, Q^{\pi}(s_t.a_t) \\ &= \max_{a_t \in \mathcal{A}} Q^{\pi}(s_t.a_t) \end{aligned} \tag{4.14}$$

Given that $Q^{\pi}(s_t, a_t)$ is maximised under the optimal policy ($\pi = \pi^*$), and using

Equation (4.13), the optimal state value can be expressed as follows:

$$\begin{aligned} V^*(s_t) &= \max_{a_t \in \mathcal{A}} Q^*(s_t, a_t) \\ &= \max_{a_t} \mathbb{E}\left[r_{t+1} + \gamma\, V^*(s_{t+1}) \,|\, s_t, a_t\right] \\ &= \max_{a_t} \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1} | s_t, a_t) \left[r_{t+1} + \gamma\, V^*(s_{t+1})\right] \end{aligned} \tag{4.15}$$

If more than one action maximises $V^*(s_t)$, then the optimal policy is indifferent between them. Lastly, we can re-express the Bellman optimality equation for Q^*, Equation (4.13), in terms of future action-values by again using the result from Equation (4.14):

$$\begin{aligned} Q^*(s_t, a_t) &= \mathbb{E}\left[r_{t+1} + \gamma\, V^*(s_{t+1}) | s_t, a_t\right] \\ &= \mathbb{E}\left[r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \,\middle|\, s_t, a_t\right] \\ &= \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1} | s_t, a_t) \left[r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})\right] \end{aligned} \tag{4.16}$$

4.1.4 Finding the Optimal Value Function

In the next section we will introduce a method which, given knowledge of the dynamics $\mathcal{P}$, can be used to calculate the value functions V^π and Q^π for a certain policy π. Before we do this, however, imagine for a moment that rather than having access to a policy, an agent instead only knew the optimal value function, V^*, along with $\mathcal{P}$. Once the agent has V^*, it is relatively easy to extract the optimal policy π^*. For each state s_t, at least one action will maximise the Bellman optimality Equation (4.15).

Any policy that assigns nonzero probability to *only* such actions, is an optimal policy:

$$\begin{aligned}\pi^*(s_t) &= \underset{a_t}{\operatorname{argmax}}\, \mathbb{E}\left[r_{t+1} + \gamma\, V^*(s_{t+1}) \mid s_t, a_t\right] \\ &= \underset{a_t}{\operatorname{argmax}} \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1} | s_t, a_t)\left[r_{t+1} + \gamma\, V^*(s_{t+1})\right] \end{aligned} \tag{4.17}$$

If instead of V^* the agent has Q^*, then the action a_t with the highest value $Q^*(s_t, a_t)$ can be chosen directly, without search.

It might seem a strange notion for an agent to know V^* but not π, given that so far value has been defined with respect to a known policy. However the Bellman optimality equation defines a set of equations, one for each state, that in finite MDPs have a unique solution that is *independent of policy*. Theoretically this means an agent can solve the system of equations to get the optimal value function V^* or Q^*, and from that derive the optimal policy. This is ideal, as by defining the Bellman equation we have simply developed the mathematics of the MDP: if V^* can be found, so can π^*, which provides the agent with the exact set of state dependent actions it must take in order to maximise its expected return, and the solves the decision problem. To find V^* and Q^*, or indeed V^π and Q^π, we look towards a technique known as Dynamic Programming.

4.2 Dynamic Programming

Dynamic programming (DP), introduced in Bellman (1957a), refers to a collection of algorithms used to solve complex problems by breaking them down into simpler subproblems. DP can be used to find optimal policies in MDPs, as well as value functions under arbitrary policies. As we will discuss at the end of the chapter, the computational requirements of DP can be demanding and, if we are not careful, intractable; however this is primarily a function of the problem space. In fact, DP

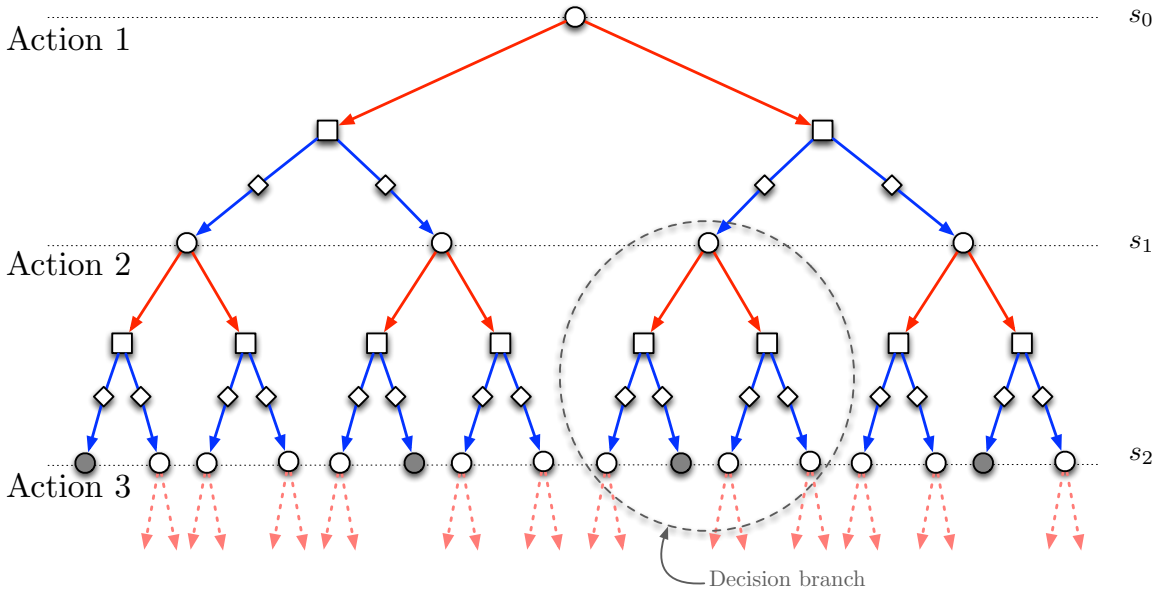


Figure 4.4: An example decision tree with discrete states and actions. The tree is illustrated in the same manner as Figure 4.3, however here the initial state, s_0, is pictured at the top with time flowing downwards, rather than on the left with time flowing to the right. Individual 'decision branches', one of which is highlighted by the dotted circle, stack on top of each other to form the tree. Grey state nodes are termination (absorbing) states in which the decision task finishes when reached. Although only the first three actions are shown, the decision tree would continue expanding until the horizon is reached in the case of finite time, or indefinitely in the case of infinite time.

is relatively efficient in calculating optimal value functions as it iterates only once across the whole decision tree structure where other methods may do so multiple times. Importantly, DP provides the foundation from which the rest of the methods in this thesis are built on.

There are various ways DP can be applied to MDPs, although all rely on a similar partitioning of the problem as defined by the Bellman equation. The equation breaks the problem down into single decisions to be solved, each forming a 'branch' on the entire decision tree, as shown in Figure 4.3. In order to understand this partitioning, let us look at how a decision problem evolves as actions are made. Figure 4.4 shows how a decision tree forms as individual decision branches are stacked together. Each decision branch represents a single state transition with associated reward, under a

single action. In the figure, only two actions are available at any point, and each action can cause a transition to one of two possible successive states. In general there may be any number of actions and possible state transitions available in a particular state. In the example tree actions and states are discrete, as this is easier to represent diagrammatically; however the concept is the same for continuous states and actions.

Now that we see how a single branch fits into the entire decision problem, we will examine in greater detail how optimal value should direct a decision across each branch. Imagine that we are in a certain state, s_t, and we happen to know the optimal state values, $V^*(s_{t+1})$, for all possible successive states s_{t+1}, as well as the dynamics, $\mathcal{T}$ and $\mathcal{R}$. Using Equation (4.13), the Bellman optimality equation for Q^*, we can calculate the action values for all actions $a_t \in \mathcal{A}(s_t)$ available in state s_t. Figure 4.5 shows the part of the branch that represents a single $Q^*(s_t, a_t)$ calculation.

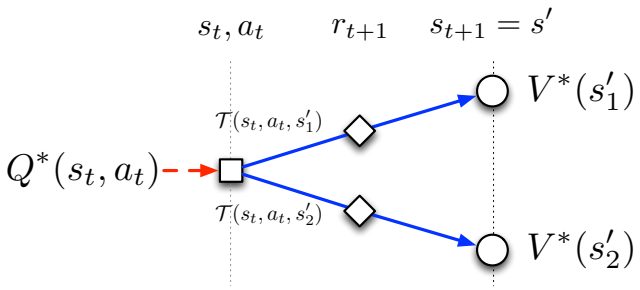


Figure 4.5: Diagram showing how an immediate action value, $Q^*(s_t, a_t)$, relates to the state value of successive states, $V^*(s_{t+1})$, in a decision tree.

The elements represented in Figure 4.5 compile into the full branch as shown in Figure 4.6(a). Equation (4.14) tells us that the optimum state value, $V^*(s_t)$, is equal to the action with the greatest optimum action-value, $Q^*(s_t, a_t)$. This is because if we were in state s_t, and we had access to the optimum action values for all available actions, $a_t \in \mathcal{A}(s_t)$, we would *always* take the action or actions a_t^* with the greatest optimal value, because any other action would yield a lesser expected return. This means we can remove any action for which $Q^*(a_t, s_t) < Q^*(a_t^*, s_t)$, as is shown in Figure 4.6(b). To continue the tree analogy, this process of removing actions from

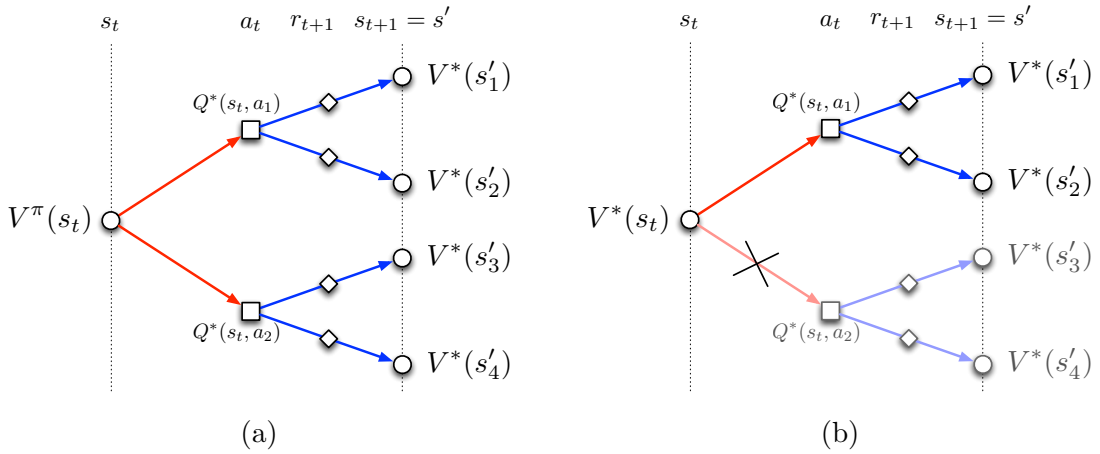


Figure 4.6: Diagram showing a single decision branch, a) before $Q^*(s_t, a_t)$ values have been compared, and b) once action a_2 has been 'pruned' because $Q^*(s_t, a_1) > Q^*(s_t, a_2)$.

the decision branch is called 'pruning'.

By pruning an action from a decision branch, it is no longer possible for us to reach any of the states and rewards that could have succeeded that action (at least through that action path). This also applies to anything that could succeed those states. As such, when we return to the decision tree, we find any node that is dependent on the action we pruned will also be pruned, as pictured in Figure 4.7. Although this appears to simplify the tree somewhat, note that in order to prune parts of the tree, we must have already calculated the value of those states and actions beforehand. Further, given that state transitions are still stochastic, the pruned tree simply allows us to see the remaining possible paths we *could* end up following given optimal action taking.

Being able to prune an action relies explicitly on our ability to evaluate comparable state and action values. We have shown how we can calculate the optimal value of a state, given knowledge of the following state values. A comparable calculation can be done for those states, and those after that. It would seem that, in order to find a solution, this recursion must stop at some point. If we consider the finite horizon task, this is certainly possible: if we imagine we are in a state $s_{\mathcal{H}-1}$, such that there

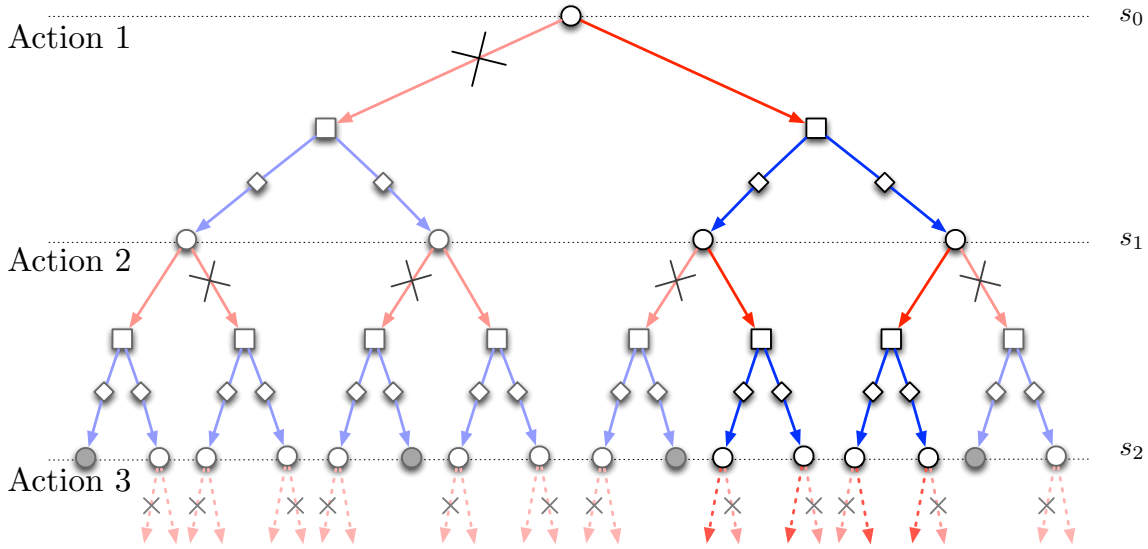


Figure 4.7: Diagram showing a decision tree after decision branches have been pruned. State and action nodes which have been pruned can no longer be reached through the pruned paths. It may be possible for states that have been pruned through one action path, to be reached via a non-pruned action path. Actions can only be pruned once state and action values succeeding them have been calculated.

is only one action remaining, the value of each action is equal to the expected reward for that action. This is because only one more reward can possibly be earned, and the value of terminal states is by definition equal to zero, as shown in Figure 4.8. By working back from terminal states, we can propagate value information back through the tree to the starting states, and then prune branches to find the optimal policy. The method is sometimes referred to as backwards induction.

The backwards induction method enables us to find the optimal value function, and can also be applied to find the value of an arbitrary policy π. Instead of the pruning operation removing all non-optimal actions, we could have instead pruned all actions that were not taken by the policy. In this way, the value of a state s_t is defined by Equation (4.5), and the value of an action by Equation (4.6): the Bellman equations for policy π.

Solving the tree by working backwards from terminal states is in a way a 'best case' scenario; this approach relies on both the problem having a definite end and the

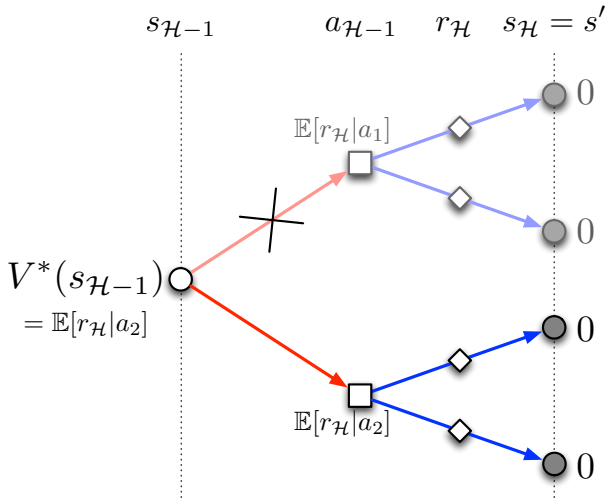


Figure 4.8: Diagram showing how the terminal decision branch is pruned. Only the action with the highest immediate expected reward is retained by the optimal policy, as terminal nodes have no inherent value.

computational burden being manageable in regard to the number of states that must be evaluated. In situations where no terminal states exist, the recursion continues indefinitely, so a different approach must be taken. Fortunately, as long as we use a discounting parameter, $\gamma < 1$, DP provides us with a set of iterative methods to accomplish this (if we don't use discounting, the value functions would tend to infinity as the return would be unbounded). The methods rely on using the Bellman equations as update rules for improving approximations to the value functions.

4.2.1 Policy Iteration

The first iterative method we introduce focuses on finding the value function for a particular policy and then improving it. The iteration consists of two steps which are repeated until the policy stabilises, *policy evaluation* and *policy improvement*, both of which are iterative schemes themselves.

The first part of the policy iteration method requires us to calculate the state value function, V^π, for an initial policy, π. Policy evaluation does this by using the Bellman equation for V^π, Equation (4.5), as an update rule. The process begins by

initialising all states to arbitrary values, $V_0(s_t)$, and then updating these values using the following equation:

$$\begin{aligned} V_{k+1}(s_t) &= \mathbb{E}_\pi \left[r_{t+1} + \gamma \, V_k(s_{t+1}) \mid s_t \right] \\ &= \sum_{a_t} \pi(a_t|s_t) \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1}|s_t, a_t) \Big[r_{t+1} + \gamma V_k(s_{t+1}) \Big] \end{aligned} \tag{4.18}$$

for $k = 0, 1, 2, \ldots$. This method is equivalent to repeatedly performing the backwards induction process described in the last section, except that the value function, V_k, is an ever improving estimate of the true value function V^π. As more sweeps of the state space are performed, the estimated values tend towards V^π and, as shown in Bertsekas (2007), converge in the limit as $k \to \infty$. In practice it is common for the iteration to terminate once the difference between consecutive estimations, $|V_{k+1}(s_t) - V_k(s_t)|$, meets a certain threshold across the state space. If there is no difference between consecutive estimates, then the value function is equal to the true value function.

In the last section we discussed how the value of a state is maximised if the policy selects *only* the action with the greatest return, as was shown in Equation (4.14). We can use this approach in an attempt to improve our initial policy, π, to a new deterministic policy, π', with a greater expected return. To understand this process, let us examine the result of improving the policy at a particular state s_t. Figure 4.9(a) shows a single decision branch from state s_t, for which the value function V^π has been calculated using the iterative policy evaluation method. If there exists an action, $a_t = \pi'(s_t)$, for which $Q^\pi(s_t, \pi'(s_t)) \geqslant V^\pi(s_t)$, then the expected return for the new policy is *at least* as good as the expected return from the original policy in that state. This can still happen even if the original policy was already deterministic, as it may be more rewarding to select a different action to the one specified. Policy improvement is therefore accomplished by looking across the entire state space, and

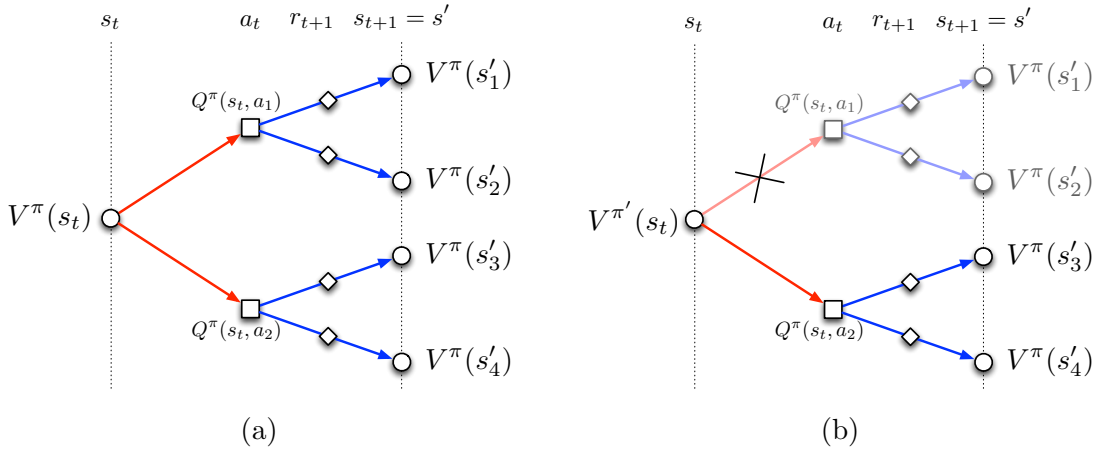


Figure 4.9: Diagram showing the policy improvement step of policy iteration. The new policy π' updates to select actions with the largest $Q^\pi(s_t, a_t)$ value.

defining a new policy as follows:

$$\begin{aligned}\pi'(s_t) &= \underset{a_t}{\operatorname{argmax}}\, Q^\pi(s_t, a_t) \\ &= \underset{a_t}{\operatorname{argmax}}\, \mathbb{E}\left[r_{t+1} + \gamma\, V^\pi(s_{t+1}) \,|\, s_t, a_t\right] \\ &= \underset{a_t}{\operatorname{argmax}} \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1} | s_t, a_t) \Big[r_{t+1} + \gamma V^\pi(s_{t+1}) \Big]\end{aligned} \tag{4.19}$$

This policy prunes actions from the original decision tree, as shown in 4.9(b).

By the above approach, the new value of state s_t, $V^{\pi'}(s_t)$, is guaranteed to be at least as good as the value under the original policy; from this it follows that the value of all states will be at least as good as their original values. Given that the value of a state is a function of the value of successive states, which will themselves change under the new policy, $V^{\pi'}$ is not directly calculable from V^π. In order to find $V^{\pi'}$ we must return to the iterative policy evaluation step. Policy iteration therefore rotates between the policy evaluation step and policy improvement step, until the policy becomes stable. Once there is no difference between consecutive policies, i.e., $\pi = \pi'$, then the policy is in fact optimal.

4.2.2 Value Iteration

The second iterative method we discuss is rather more compact than policy iteration, and in effect combines the policy evaluation and improvement steps into a single iterative scheme, from which a policy can be extracted. It relies on propagating state-value estimates back through the tree using the Bellman optimality equation for V^*, Equation (4.15), as an update rule. Value iteration begins by initialising all states to an arbitrary value, and then updating these values using the following equation:

$$\begin{aligned} V_{k+1}(s_t) &= \max_{a_t \in \mathcal{A}} \mathbb{E}\left[r_{t+1} + \gamma\, V_k(s_{t+1}) \,|\, s_t, a_t\right] \\ &= \max_{a_t} \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1} | s_t, a_t)\left[r_{t+1} + \gamma\, V_k(s_{t+1})\right] \end{aligned} \tag{4.20}$$

where $V_k(s_t)$ represents the kth estimate of the value of state s_t.

Value iteration works by selecting, for each state, whichever action yields the greatest estimated value, and using that value in the Bellman equation. This is analogous to the policy improvement step in policy iteration. The estimated value of the successive states are used as if they were the optimal values, and actions that do not correspond with the maximal value are pruned. Through repeated iterations, the estimated value function is guaranteed to converge towards the optimal value function, although this would formally require an infinite number of sweeps (Sutton and Barto, 1998). One solution to this is to evaluate the difference between consecutive value function estimates, and terminate the algorithm once a certain threshold has been met. We have already seen how an optimal policy can be derived from the optimal value function using Equation (4.17); this approach can also be applied here to find an approximate optimal policy.

4.3 Practical Challenges

The two main challenges we have in applying DP and its related iterative techniques are tractability and non-analytic solutions. We emphasise that both of these concerns are inherent in the problem domain itself, rather than a particular downside of DP. As already mentioned, DP is relatively efficient amongst competing algorithms; for instance a DP method is guaranteed to find an optimal policy in polynomial time even though the total number of deterministic policies is m^n, where m and n denote the number of states and actions (Sutton and Barto, 1998). This makes it exponentially faster than a direct search of the policy space which would have to exhaustively scrutinise each policy.

Unfortunately, many real-world problems consist of such large state spaces that even with the most efficient methodology the problem can become intractable: Bellman's so called *curse of dimensionality*. In Chapter 6 we will also see that, once the state variable is considered to encompass knowledge, the state space can be unmanageable even for very small problems. The need for effective and principled approximations is very much required. In the next chapter we take a step towards such approximations and examine problems with continuous state and action spaces. Problems of this form suffer from both tractability issues and the fact that the solution is non-analytic, meaning approximation is inevitable.

Chapter 5

Bayesian Function Modelling in Continuous State and Action Spaces

> "It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience."
>
> *On the method of theoretical physics*, Einstein (1934)

In the last chapter we discussed the relationship between state and value, which culminated with the presentation of several iterative methods from which optimal policies can be extracted. The most compact of these was value iteration, which at its heart uses the Bellman Equation as an update rule. The update rule, originally given in Equation (4.20), is:

$$V_{k+1}(s_t) = \max_{a_t \in \mathcal{A}} \mathbb{E}\left[r_{t+1} + \gamma V_k(s_{t+1}) \,|\, s_t, a_t\right]$$

Whilst the solution to this is theoretically calculable in the discrete domain, this is not so in continuous spaces for which an infinite number of operations would be required.

In order to use the tools developed for the MDP framework, including value iteration, we require a means to efficiently represent such spaces.

As a motivating example, let us consider the well-known mountain car problem (Moore, 1991), shown in Figure 5.1, which describes the scenario faced by a car that must escape a valley by driving up a steep slope. The car cannot generate the requisite power to drive up the slope from a standing start at the base of the valley, and so must utilise the landscape in order to generate momentum to help carry it up and out. As I witnessed on a recent trip to the Tuscan hill-top village of Montepulciano, this can certainly be grounded in the real world; on that occasion it took one frustrated tourist a good 10 minutes to find the solution!

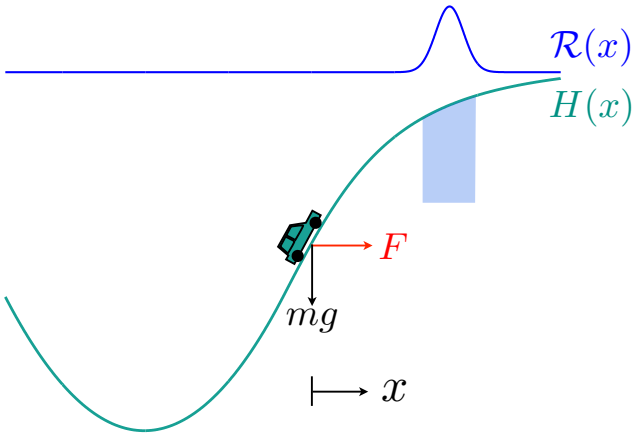


Figure 5.1: Illustration of the mountain car problem.

Taking a simplified view of reality and using the parameterisation given in Moore and Atkeson (1995), the state of the car can be modelled using just its position and velocity, such that $\boldsymbol{s} = [x, \dot{x}]^\top$, which is constrained to $-1 \leqslant x \leqslant 1$ and $-2 \leqslant \dot{x} \leqslant 2$. In order to control the car, the driver is able to apply a horizontally acting force in the range $-4 \leqslant F \leqslant 4$. The dynamics of the problem $\mathcal{T}$ which are derived from the shape of the valley $H(x)$ are given in Appendix A.1. In the Moore and Atkeson (1995) formulation, a reward of 1 is outputted if the car is in a target region, pictured as a shaded box in Figure 5.1. We use the approach taken by Kuss (2006) of redefining

this reward function as a Gaussian proportional to $\mathcal{N}([0.6, 0]^\top, 0.05^2\mathbf{I})$ with maximum reward of 1, as indicated in the Figure. This means the car must be stationary within the goal region in order achieve maximum reward.

Referring back to the value iteration update in the context of the mountain car problem, V and $\mathcal{P}$ are both defined over a continuous domain. Expanding the expectation operators we get:

$$V_{k+1}(\boldsymbol{s}_t) = \max_{a_t} \int_{\boldsymbol{s}_{t+1}} \int_{r_{t+1}} \mathcal{P}(\boldsymbol{s}_{t+1}, r_{t+1} | \boldsymbol{s}_t, a_t) \Big[r_{t+1} + \gamma\, V_k(\boldsymbol{s}_{t+1}) \Big] \mathrm{d}r_{t+1} \mathrm{d}\boldsymbol{s}_{t+1} \tag{5.1}$$

So that we can focus on the problem of representation, let us assume that the system is fully deterministic and that the decision maker has no epistemic uncertainty in the dynamics. This leaves us with the following system of equations to solve:

$$V_{k+1}(\boldsymbol{s}_t) = \max_{a_t} \left[\mathcal{R}(\boldsymbol{s}_t) + \gamma\, V_k(\boldsymbol{s}_{t+1}) \right] \tag{5.2}$$

where $\boldsymbol{s}_{t+1} = \mathcal{T}(\boldsymbol{s}_t, a_t)$ is known, and $a_t = F_t$. Note that the mountain car problem is defined such that reward is only a function of the current state $\boldsymbol{s}$, rather than the tuple $(\boldsymbol{s}, a, \boldsymbol{s}_{t+1})$. The problem now is to provide an adequate means of representing $V(\boldsymbol{s})$ and $\mathcal{R}(\boldsymbol{s})$.

A commonly used approach to deal with continuous state and action spaces is discretisation, although the computation associated with this is not necessarily tractable. As pointed out in Deisenroth et al. (2009), the number of cells in a discretised space will depend not only on the dimensionality and difficulty of the problem, but also on the time-sampling frequency. As this becomes smaller the number of cells increases, which can lead to infeasible computation even for low-dimensional problems.

An alternative to discretisation is to use effective *function approximation*, where the value functions are modelled in a function space rather than as a discrete table

of values, which bypasses the curse of dimensionality. This is not a new approach, see for example Bertsekas and Tsitsiklis (1996), although the landscape of applicable models is constantly evolving. To bring us up to the state-of-the-art we begin in the next section by briefly reviewing simple parametric function approximators, before showing how these can be reformulated into a far more powerful and flexible technique, namely *Gaussian Processes*, which belong to a larger group of *Bayesian non-parametric* methods. It should be noted that Gaussian Process prediction is by no means a recent topic; as noted in Rasmussen and Williams (2006), the basic theory goes back at least as far as the work of Kolmogorov (1941) and Wiener (1949), and it has found application in a variety of fields including, for example, geostatistics where it is known as *kriging* (Matheron, 1973; Journel and Huijbregts, 1978).

5.1 Linear Function Approximation

The Standard Linear Model is perhaps the simplest place to start when looking at parametric function approximators. This defines a function which is a linear combination of the inputs:

$$f(\boldsymbol{s}) = \boldsymbol{s}^\top \mathbf{w}, \qquad y = f(\boldsymbol{s}) + \epsilon \tag{5.3}$$

where $\boldsymbol{s}$ is an input vector, and $\mathbf{w}$ is a vector of weights which form the parameters of the model. The difference between f, the function value, and y, the observed target value, is additive noise given by ϵ. A common assumption is that the noise process is i.i.d. Gaussian with zero mean and variance σ_n^2:

$$\epsilon \sim \mathcal{N}(0, \sigma_n^2) \tag{5.4}$$

In order to train the model it is necessary to gain some knowledge of the true function, which for the value iteration problem will either be $V(\boldsymbol{s})$ or $\mathcal{R}(\boldsymbol{s})$. Let us

assume for the time being that we are able to take a finite set of observations $\mathcal{D} = \{(\mathbf{s}_i, y_i) \,|\, i = 1, \ldots, n\}$ of the function we are interested in modelling. We aggregate all the n input cases in a $D \times n$ *design matrix* $\mathbf{S}$, where D is the dimensionality of $\mathbf{s}$, along with the observed values in the vector $\mathbf{y}$, such that $\mathcal{D} = (\mathbf{S}, \mathbf{y})$. Together with the noise assumption, we can calculate the likelihood of the data given $\mathbf{w}$ as:

$$\begin{aligned} p(\mathbf{y}|\mathbf{S}, \mathbf{w}) &= \prod_{i=1}^{n} p(y_i|\mathbf{s}_i, \mathbf{w}) \\ &= \mathcal{N}(\mathbf{S}^\top \mathbf{w}, \sigma_n^2 \mathbf{I}) \end{aligned} \tag{5.5}$$

the derivation of which can be found in Appendix B.1.

The likelihood given in Equation (5.5) can be maximised in order to produce a point estimate for the parameters, which given the Gaussian noise assumption is equivalent to minimising the sum of squared errors (Bishop, 2006). Such a point estimate will likely suffer from overfitting; however before we attempt to improve this estimate we should first acknowledge that the target functions will be highly nonlinear, and thus not well represented by the model as it stands. We therefore look to extend the class of models by considering linear combinations of fixed nonlinear functions, or *basis functions*, before improving our method of inference.

5.1.1 Linear Basis Function Models

We now look to project the inputs into a high dimensional 'feature' space using a set of basis functions, and then apply the linear model in that space instead of directly on the inputs. Our model becomes:

$$f(\boldsymbol{s}) = \boldsymbol{\phi}(\boldsymbol{s})^\top \mathbf{w} \tag{5.6}$$

where $\boldsymbol{\phi}(\boldsymbol{s})$ is the set of basis functions we wish to use. Once again we can use this to calculate the likelihood of a set of observations $\mathbf{S}$, which is analogous to that of the standard linear model:

$$\begin{aligned} p(\mathbf{y}|\mathbf{S}, \mathbf{w}) &= \prod_{i=1}^{n} p(y_i|\mathbf{s}_i, \mathbf{w}) \\ &= \mathcal{N}(\boldsymbol{\Phi}^\top \mathbf{w}, \sigma_n^2 \mathbf{I}) \end{aligned} \tag{5.7}$$

except that $\boldsymbol{\Phi}(\mathbf{S}) = \boldsymbol{\Phi}$, which represents the aggregation of columns $\boldsymbol{\phi}(\mathbf{s})$ for all the training cases, is substituted in place of $\mathbf{S}$.

Basis functions $\boldsymbol{\phi}(\boldsymbol{s})$ can in general accommodate any nonlinear mapping of the inputs. We will focus on a particular form of *radial* basis function (RBF), the Gaussian, such that $\boldsymbol{\phi}(\boldsymbol{s}) = [\phi_1(\boldsymbol{s}), \phi_2(\boldsymbol{s}), \ldots, \phi_J(\boldsymbol{s})]^\top$ where:

$$\phi_j(\boldsymbol{s}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{s} - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{s} - \boldsymbol{\mu}_j)\right) \tag{5.8}$$

and where $\phi_j(\boldsymbol{s})$ is the jth RBF with mean $\boldsymbol{\mu}_j$ and covariance $\boldsymbol{\Sigma}_j$. RBFs are useful as they can represent a wide variety of functions. As an example, Park and Sandberg (1991) show how RBF networks are capable of universal approximation. They have also been used in Reinforcement Learning type problems to good effect, as shown in Lee (2009), where a set of RBFs is used to model the value function in a TD(λ) algorithm. In that case, a linear model using RBFs was shown to be far more effective than *tile coding* (see Sutton and Barto (1998)), for which the state space is discretised into non-overlapping elements.

5.1.2 Bayesian Linear Regression

We already mentioned that a maximum likelihood estimate of the parameters is ultimately inadequate and suffers from overfitting. The first step towards a Bayesian

treatment of the inference process is to specify a prior over the parameters. In the absence of further information, a zero mean Gaussian prior is appropriate:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \Sigma_p) \tag{5.9}$$

where Σ_p is the covariance matrix and a *hyperparameter*. A prior over the weights replaces the need for a regularisation term after the main cost term, which would be necessary with, for example, linear least squares.

The next stage of Bayesian inference is to form the posterior distribution over the weights by combining the likelihood, prior, and marginal likelihood through Bayes' rule:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{\Phi}) = \frac{p(\mathbf{y}|\mathbf{\Phi}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{\Phi})} \tag{5.10}$$

where the marginal likelihood is independent of the parameters, and is given by:

$$p(\mathbf{y}|\mathbf{\Phi}) = \int p(\mathbf{y}|\mathbf{\Phi}, \mathbf{w})p(\mathbf{w})\mathrm{d}\mathbf{w} \tag{5.11}$$

Solving Equation (5.10) gives a posterior distribution which is also Gaussian:

$$p(\mathbf{w}|\mathbf{\Phi}, \mathbf{y}) \sim \mathcal{N}(\frac{1}{\sigma_n^2} A^{-1}\mathbf{\Phi}\mathbf{y},\ A^{-1}) \tag{5.12}$$

where $A = \frac{1}{\sigma_n^2}\mathbf{\Phi}\mathbf{\Phi}^\top + \Sigma_p^{-1}$. The derivation of this result is given in Appendix B.1. The mean of this distribution coincides with the mode and forms the *maximum a posteriori* (MAP) estimate of $\mathbf{w}$ which, although it necessitates the inclusion of a prior over the weights, has no special role in Bayesian inference.

Ultimately we are less interested in the value of $\mathbf{w}$ than our ability to make predictions about the function for new input values. In order to complete the Bayesian inference process, we must marginalise out the parameter values to arrive at the

posterior predictive distribution:

$$p(f_*|\boldsymbol{\phi}_*, \boldsymbol{\Phi}, \mathbf{y}) = \int p(f_*|\boldsymbol{\phi}_*, \mathbf{w}) p(\mathbf{w}|\boldsymbol{\Phi}, \mathbf{y}) \mathrm{d}\mathbf{w}$$
$$= \mathcal{N}(\frac{1}{\sigma_n^2} \boldsymbol{\phi}_*^\top A^{-1} \boldsymbol{\Phi} \mathbf{y},\ \boldsymbol{\phi}_*^\top A^{-1} \boldsymbol{\phi}_*) \tag{5.13}$$

where $\boldsymbol{\phi}_* = \boldsymbol{\phi}(\mathbf{s}_*)$, and $f_* \triangleq f(\mathbf{s}_*)$ is the predicted function value at input $\mathbf{s}_*$. The main bottleneck in making predictions using this equation is the need to invert A, which will be inconvenient if the dimension of the feature space, J, is large.

5.1.3 Limitations of Fixed, Finite Basis Functions

The use of fixed, finite nonlinear basis functions provides us with the ability to find closed-form solutions, and to perform tractable Bayesian inference over the model. Further, with the appropriate choice of basis functions, we can model arbitrary non-linearities between functional inputs and outputs. Although this may seem like it offers an ideal solution to our original problem, unfortunately there are shortcomings which make its applicability as a general purpose framework difficult.

In the first instance, the complexity of our model will always be bounded if the number of basis functions used is finite. This problem could be solved by adding more and more basis functions, and perhaps reasoning about the model complexity, as per Section 2.2.3. However, whilst the state vector in the mountain car problem has only two dimensions, in many real world problems the state vector can grow to be much larger. Unfortunately, as the dimensionality of the input space grows, the number of basis functions needed can grow exponentially fast (Bishop, 2006), making this approach untenable.

Adding to the issue of complexity, in regions not occupied by basis functions the predictive variance given in Equation (5.13) by $\boldsymbol{\phi}_*^\top (\frac{1}{\sigma_n^2} \boldsymbol{\Phi} \boldsymbol{\Phi}^\top + \Sigma_p^{-1})^{-1} \boldsymbol{\phi}_*$ goes to zero as we move away from the basis function centres. This means that the model becomes

very confident in these regions, which is clearly undesirable behaviour. We could try to ensure there is adequate coverage in the part of the function which contains data and is of interest, although this is difficult if we know little about the function or the space it occupies in advance.

5.1.4 Defining a Kernel Function

Ultimately, the success of parametric methods relies heavily on the assumptions made in the model. We must therefore look to an alternative class of *nonparametric* methods to alleviate the problems just outlined. To begin this process, let us first reformulate the Bayesian model up to this point. As shown in Rasmussen and Williams (2006), (5.13) can be rewritten as follows:

$$p(f_*|\boldsymbol{\phi}_*, \boldsymbol{\Phi}, \mathbf{y}) = \mathcal{N}(\boldsymbol{\phi}_*^\top \Sigma_p \boldsymbol{\Phi}(K + \sigma_n^2 \mathbf{I})\mathbf{y}, \\ \boldsymbol{\phi}_*^\top \Sigma_p \boldsymbol{\phi}_* - \boldsymbol{\phi}_*^\top \Sigma_p \boldsymbol{\Phi}(K + \sigma_n^2 \mathbf{I})^{-1} \boldsymbol{\Phi}^\top \Sigma_p \boldsymbol{\phi}_*) \tag{5.14}$$

where $K = \boldsymbol{\Phi}^\top \Sigma_p \boldsymbol{\Phi}$. Importantly, in this rearrangement the feature space enters in such a way that the elements in the matrices are always in the form of $\boldsymbol{\phi}(\boldsymbol{s})^\top \Sigma_p \boldsymbol{\phi}(\boldsymbol{s}')$, where $\boldsymbol{s}$ and $\boldsymbol{s}'$ may appear in either the training or test sets. As this expression appears frequently, let us define a new function $k(\boldsymbol{s}, \boldsymbol{s}') = \boldsymbol{\phi}(\boldsymbol{s})^\top \Sigma_p \boldsymbol{\phi}(\boldsymbol{s}')$, which by examination forms an inner product with respect to Σ_p. To confirm, let us first rewrite $\Sigma_p = (\Sigma_p^{1/2})^2$, which we can do as Σ_p is positive definite. Now defining $\boldsymbol{\Psi}(\boldsymbol{s}) = \Sigma_p^{1/2} \boldsymbol{\phi}(\boldsymbol{s})$, we arrive at the result that $k(\boldsymbol{s}, \boldsymbol{s}') = \boldsymbol{\Psi}(\boldsymbol{s}) \cdot \boldsymbol{\Psi}(\boldsymbol{s}')$, which is a simple dot product.

The benefit of this alternative formulation is that, instead of explicitly introducing a feature vector, we can operate directly on the data through the function $k(\cdot, \cdot)$, which is referred to as a *kernel* or *covariance function*. This process is therefore often referred to as the *kernel trick* or *kernel substitution*. Kernels are by definition a

symmetric function of their arguments, and ultimately evaluate some form of similarity between data points. In order to generate a kernel, we could find the kernel that corresponds to the basis functions we were interested in originally:

$$k(\boldsymbol{s}, \boldsymbol{s}') = \boldsymbol{\phi}(\boldsymbol{s})^\top \Sigma_p \boldsymbol{\phi}(\boldsymbol{s}') = \sum_{j=1}^{J} \Psi_j(\boldsymbol{s}) \Psi_j(\boldsymbol{s}') \tag{5.15}$$

however, as we will see when dealing with Gaussian Processes, it is often desirable to construct kernels directly such that they become the object of primary interest rather than the feature vectors. Bishop (2006) provides a useful introduction into the construction of valid kernels.

It is important to note that, if we are content using a finite basis model, predictions will be more expensive to compute in a kernel space. However, based on our discussion on complexity, it should come as no surprise that the most useful kernels are those that, according to Equation (5.15), emerge from an infinite basis function representation. In this case, operating directly in a kernel space is the only tractable way to make predictions; we find that the parameter space of the model will naturally grow with the amount of data we give it to learn from, averting the need to grow a feature space along with the dimensionality of the problem. Additionally, an infinite set of basis functions means we no longer have to concern ourselves with the locality of our assumptions. For reasons that will become clear, we will from now on mostly refer to kernels using their alternative title of covariance functions.

5.2 Gaussian Processes

In the last section, we noted that the introduction of a prior distribution over the model parameters in a Bayesian linear model resulted in a distribution over functions $f(\mathbf{s})$. Rather than going through the process previously described, we can work directly with distributions over functions in order to generate the predictive distribution.

To see this, let us return to the original basis function model with prior:

$$f(\boldsymbol{s}) = \boldsymbol{\phi}(\boldsymbol{s})^\top \mathbf{w}, \qquad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_p)$$

Given that $f(\boldsymbol{s})$ is a linear combination of Gaussian distributed variables, it is itself Gaussian distributed and we need only find its mean and covariance:

$$\mathbb{E}[f(\boldsymbol{s})] = \boldsymbol{\phi}(\boldsymbol{s})^\top \mathbb{E}[\mathbf{w}] = 0 \tag{5.16}$$

$$\mathbb{E}[f(\boldsymbol{s}), f(\boldsymbol{s}')] = \boldsymbol{\phi}(\boldsymbol{s})^\top \mathbb{E}[\mathbf{w}\mathbf{w}^\top] \boldsymbol{\phi}(\boldsymbol{s}') = \boldsymbol{\phi}(\boldsymbol{s})^\top \Sigma_p \boldsymbol{\phi}(\boldsymbol{s}') \tag{5.17}$$

This means that $f(\boldsymbol{s})$ and $f(\boldsymbol{s}')$ are multivariate Gaussian distributed, with zero mean and covariance $k(\boldsymbol{s}, \boldsymbol{s}') = \boldsymbol{\phi}(\boldsymbol{s})^\top \Sigma_p \boldsymbol{\phi}(\boldsymbol{s}')$, the later being what we previously described as the kernel. In fact, the function $f(\boldsymbol{s})$ evaluated at any set of points $\boldsymbol{s}, \ldots, \boldsymbol{s}_N$ will be jointly Gaussian. This model is therefore a particular example of a *Gaussian Process*.

To understand what we mean by a Gaussian Process (GP), let us start by thinking of a function as a long vector of function outputs associated with an equally long vector of function inputs. For the most part, and certainly within the context of this section, we are interested in functions in which the number of possible inputs (and therefore the length of the function vectors) is infinite. A GP allows us to consider this by forming a probability distribution over a potentially infinite collection of random variables (the function output values), in which any finite subset of them have a joint Gaussian distribution.

In a manner analogous to the familiar Gaussian distribution, a GP is completely defined by its first and second moments: a mean function $\mu : \mathcal{S} \to \mathbb{R}$, which describes the overall trend of the function, and a positive semidefinite covariance function, or kernel, $k : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ which describes how function values are correlated as a function of their locations in the domain. If $\mathbf{f}$ refers to a vector of function outputs at a particular subset of input points $\mathbf{S}$ we are interested in, then we can define a GP

as:

$$\begin{aligned} p(\mathbf{f}|\mathbf{S},\theta) &\triangleq \mathcal{N}(\mathbf{f};\mu_\theta(\mathbf{S}),k_\theta(\mathbf{S},\mathbf{S})) \\ &\triangleq \frac{1}{\sqrt{(2\pi)^D\,|K|}}\exp\left(-\frac{1}{2}(\mathbf{f}-\boldsymbol{\mu})^\top K^{-1}(\mathbf{f}-\boldsymbol{\mu})\right) \end{aligned} \tag{5.18}$$

where θ is a vector of hyperparameters (parameters of the GP prior) which completely specify μ and k. $\boldsymbol{\mu} = \mu_\theta(\mathbf{S})$ and $K = k_\theta(\mathbf{S},\mathbf{S})$ represent a mean vector and covariance matrix defined over $\mathbf{S}$. There exists a wide variety of mean and covariance functions which can be chosen in order to reflect any prior knowledge available about the function of interest. We'll henceforth assume that inputs such as $\mathbf{S}$ are always known, and as such will drop them from explicit representation where appropriate.

The mean function μ_θ expresses our expectation about the function values $\mathbf{f}$ before observations are made. A very common choice is to take $\mu_\theta = 0$, which states that our best initial guess is for the function to be zero at any input value. Other common forms include non-zero constant and polynomial, although in general we can define whatever complexity of function is appropriate to our prior beliefs. The covariance function k_θ indicates the strength of correlations between elements of $\mathbf{f}$. The main requirement in our choice of k_θ is that the resulting covariance matrix K is positive semi-definite. Many choices of covariance function exist (Abrahamsen, 1997; Gibbs, 1997; Gneiting, 2002), and these can express beliefs about the function such as smoothness and periodicity, and even the possibility of changepoints (Garnett et al., 2010a). As we have already said, the hyperparameters θ contain all the parameters required by the mean and covariance functions; we will come to how to manage these shortly.

Although a GP is defined over potentially infinitely many function values, we are only really interested in considering subsets of these. Fortunately, and certainly not without influence on our choice of approach, the Gaussian distribution has some

especially useful properties in this regard. Take, for example, two finite subsets of function values $\mathbf{f}_1$ and $\mathbf{f}_2$. For the time being, let us also take the mean function μ_θ to be zero everywhere. Due to the multivariate Gaussian property, we can write the following:

$$\begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}\right) \tag{5.19}$$

where K_{ij} represents the covariance matrix between subsets i and j, and where $K_{ij} = K_{ji}^\top$. The marginalisation property of the Gaussian distribution means that any subset of variables is also Gaussian distributed, and we have $\mathbf{f}_1 \sim \mathcal{N}(\mathbf{0}, K_{11})$ and $\mathbf{f}_2 \sim \mathcal{N}(\mathbf{0}, K_{22})$. Similarly, the distribution of any subset of variables conditioned on any other subset is also Gaussian. So, imagining that we know $\mathbf{f}_1$, the predictive distribution for $\mathbf{f}_2$ is:

$$\mathbf{f}_2|\mathbf{f}_1 \sim \mathcal{N}(K_{21}K_{11}^{-1}\mathbf{f}_1,\ K_{22} - K_{21}K_{11}^{-1}K_{12}) \tag{5.20}$$

To take the model one step further, and in line with our linear model approach, we should assume a noise model as exact measurements of the function are often not available. To do this we define:

$$p(y|f, \sigma_n^2) := \mathcal{N}(y; f, \sigma_n^2) \tag{5.21}$$

which represents i.i.d. Gaussian observation noise with variance σ_n^2. The noise variance joins the other hyperparameters of the model and as such is included in θ. With this extension to the model, we arrive at an updated prior on the covariance of the noisy observations, such that:

$$\mathrm{cov}(y, y') = k_\theta(\boldsymbol{s}, \boldsymbol{s}') + \sigma_n^2\delta(\boldsymbol{s} - \boldsymbol{s}') \tag{5.22}$$

where $\delta(\cdot)$ is the Kronecker delta function. It should be noted that alternative noise models can be accommodated in the GP framework, although as discussed in Rasmussen and Williams (2006) these come at the cost of losing the convenient analytical form of the Gaussian.

Let us return to the problem of predicting the function $\mathbf{f}_*$ for some input set $\mathbf{S}_*$ after taking a set of noisy observations $\mathcal{D} = (\mathbf{S}, \mathbf{y})$. We can write the joint distribution of $\mathbf{y}$ and $\mathbf{f}_*$ under the prior, now including a mean function, as:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} K + \sigma_n^2 \mathbf{I} & K_*^\top \\ K_* & K_{**} \end{bmatrix} \right) \tag{5.23}$$

where the prior means $\boldsymbol{\mu} = \mu_\theta(\mathbf{S})$ and $\boldsymbol{\mu}_* = \mu_\theta(\mathbf{S}_*)$, and the prior covariances $K = k_\theta(\mathbf{S}, \mathbf{S})$, $K_* = k_\theta(\mathbf{S}_*, \mathbf{S})$, and $K_{**} = k_\theta(\mathbf{S}_*, \mathbf{S}_*)$. Once again, using the properties of the Gaussian we arrive at a predictive distribution over $\mathbf{f}_*$, which is also Gaussian:

$$p(\mathbf{f}_*|\mathbf{y}, \theta) := \mathcal{N}(\mathbf{f}_*; m_\theta(\mathbf{f}_*|\mathbf{y}), C_\theta(\mathbf{f}_*|\mathbf{y})) \tag{5.24}$$

where the posterior mean and covariance are:

$$m_\theta(\mathbf{f}_*|\mathbf{y}) := \boldsymbol{\mu}_* + K_* \left[K + \sigma_n^2 \mathbf{I}\right]^{-1} (\mathbf{y} - \boldsymbol{\mu}_*) \tag{5.25}$$

$$C_\theta(\mathbf{f}_*|\mathbf{y}) := K_{**} - K_* \left[K + \sigma_n^2 \mathbf{I}\right]^{-1} K_*^\top \tag{5.26}$$

Note from comparison with Equation (5.20) that only the predictive mean (not the covariance) is altered by the inclusion of the prior mean. It is also possible, if required, to make predictions for the noisy observations at the test inputs, such that:

$$m_\theta(\mathbf{y}_*|\mathbf{y}) := m_\theta(\mathbf{f}_*|\mathbf{y}) \tag{5.27}$$

$$C_\theta(\mathbf{y}_*|\mathbf{y}) := C_\theta(\mathbf{f}_*|\mathbf{y}) + \sigma_n^2 \mathbf{I}_* \tag{5.28}$$

5.2.1 Marginalising out the Hyperparameters

The previous equations assume that the hyperparameters θ are known, although in fact we can rarely be certain about θ *a priori*. This ignorance can be represented by a suitably uninformative prior distribution $p(\theta)$. Given such a *hyper-prior*, the hyperparameters can be marginalised to calculate the predictive distribution over $\mathbf{f}_*$:

$$p(\mathbf{f}_*|\mathbf{y}) = \frac{\int p(\mathbf{f}_*|\mathbf{y},\theta)p(\mathbf{y}|\theta)p(\theta)\mathrm{d}\theta}{\int p(\mathbf{y}|\theta)p(\theta)\mathrm{d}\theta} \tag{5.29}$$

Unfortunately, such integrals are generally non-analytic, requiring numerical approximation. Randomised Monte Carlo techniques (Chen et al., 2000) form the most popular approaches to numerical integration, although Bayesian alternatives (Rasmussen and Ghahramani, 2003; Osborne et al., 2012) also exist. These techniques estimate the integral given the value of the integrand on a set of sample points, an example of which is given in Figure 5.2, usually via a weighted mixture:

$$p(\mathbf{f}_*|\mathbf{y}) \simeq \sum_i \rho_i p(\mathbf{f}_*|\mathbf{y},\theta_i) \tag{5.30}$$

for some weight vector $\boldsymbol{\rho}$. Unfortunately, the computational expense of evaluating the integrand at sufficient samples to estimate high-dimensional integrals is often prohibitive.

A less computationally demanding alternative is to select only a single sample. Type-II maximum likelihood, or maximum marginal likelihood, approximates

$$p(\mathbf{f}_*|\mathbf{y}) \simeq p(\mathbf{f}_*|\mathbf{y},\theta_{\mathrm{ML}}) \tag{5.31}$$

where θ_{ML} is the hyperparameter vector that maximises the marginal likelihood,

$$\theta_{\mathrm{ML}} = \operatorname*{argmax}_{\theta} \; p(\mathbf{y}|\theta) \tag{5.32}$$

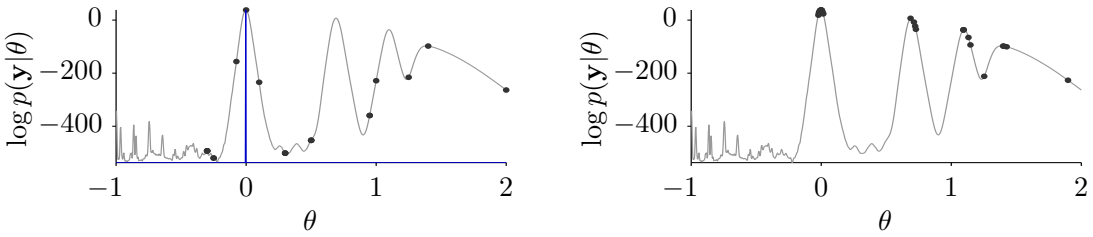


Figure 5.2: (a) Samples (black dots) obtained by optimising the log-likelihood (grey) using a global optimiser, and in blue, the maximum likelihood approximation of the likelihood surface. (b) Samples obtained by taking draws from the posterior using an MCMC method.

As per Figure 5.2(a), (5.31) is equivalent to approximating the likelihood $p(\mathbf{y}|\theta)$ as the Dirac delta function $\delta(\theta - \theta_{\mathrm{ML}})$. (We should state that, as the Figure depicts a log-likelihood function, the likelihood may in fact be well-fitted by a *mixture* of Dirac delta functions.) In general this assumption is a poor representation of our true state of ignorance under low amounts of data, for which the likelihood is typically broad and/or multi-modal. As a consequence, type-II maximum likelihood can often lead to over-fitting. Nevertheless, the less onerous computational requirements of this approach have made it ubiquitous throughout machine learning.

5.3 Approximate Hyperparameter Marginalisation

If we were to take a small number of samples of the value functions in order to generalise the function across the whole state domain, then over-fitting will lead us to pursue a sub-optimal policy, leading to reduced return. As type-II maximum likelihood often suffers from over-fitting, we must find a balance between this tractable yet less robust approach and a Bayesian optimal solution with unmanageable computational requirements. This is not an unusual objective, and we encounter the need to tradeoff these extremes through much of the Reinforcement Learning space (and indeed machine learning in general); in later chapters we will confront a similar situation

when we deal with the apparently competing aims of exploration and exploitation.

A core objective of this thesis is to find a balance between the ideal (inevitably the Bayesian solution) and the tractable, by generating approximate solutions that operate within a reasonable computational budget but are derived from an understanding of optimal methodology. In this section we do this by introducing a novel approximate method for performing inference, using a Gaussian process, which accommodates the inherent uncertainty one should have about the value of certain hyperparameters but does not suffer from the excessive computational expense of full marginalisation or sampling. Our approach, which avoids attempting to approximate the integral in (5.29), uses a modified prior that incorporates and marginalises the uncertainty in the hyperparameters of the covariance *before* it is fused with observations. The result is an adjusted covariance function which can be used interchangeably with a standard covariance function in GP inference. We term the general form of these covariance functions as *Approximately Marginalised Covariance Functions* (AMCFs).

5.3.1 Approximate Hyperparameter Priors

Our derivation of a new method for approximate hyperparameter marginalisation focusses on a particular type of strictly positive *input scale* hyperparameters L, which feature in many common covariance functions. Without further assumptions on their form, we write these covariance functions as $k_L(\boldsymbol{s}, \boldsymbol{s}') = k_L$.

We begin by looking at the prior distribution over $\boldsymbol{f}$, which is generally a non-Gaussian distribution:

$$p(\boldsymbol{f}) = \int p(\boldsymbol{f} \mid L) p(L) \mathrm{d}L \ . \tag{5.33}$$

where $p(\boldsymbol{f} \mid L)$ is a GP and $p(L)$ is a prior distribution over the hyperparameters L. We first aim to approximate $p(\boldsymbol{f})$ by a Gaussian distribution $\tilde{p}(\boldsymbol{f})$, by matching the

first two moments of $p(\boldsymbol{f})$ and $\tilde{p}(\boldsymbol{f})$:

$$\tilde{p}(\boldsymbol{f}) = \mathcal{N}(\tilde{\mu}(\boldsymbol{s}), \tilde{k}(\boldsymbol{s}, \boldsymbol{s}')) \tag{5.34}$$

where $\tilde{\mu}$ is the mean of a GP and $\tilde{k}$ is its covariance function. To clarify the presentation, we assume that the mean of $p(\boldsymbol{f} \mid L)$ is zero for all L, although our approach can be extended to the non-zero mean case. Thus, the mean $\tilde{\mu}$ of $\tilde{p}(\boldsymbol{f})$ is also zero and the covariance of $\tilde{p}(\boldsymbol{f})$ is the second moment of $p(\boldsymbol{f})$:

$$\tilde{k}(\boldsymbol{s}, \boldsymbol{s}') = \int_L \left[\int_{\boldsymbol{f}} \boldsymbol{f}\ \boldsymbol{f}^\top p(\boldsymbol{f} \mid L) p(L) \mathrm{d}\boldsymbol{f} \right] \mathrm{d}L = \int_L k_L(\boldsymbol{s}, \boldsymbol{s}') p(L) \mathrm{d}L \tag{5.35}$$

The problem we now face is how to perform the marginalisation in Equation (5.35). To do this we must first consider the prior over the input scale L, which is the only hyperparameter of concern. As L should be strictly positive, we propose a Gaussian prior for the logarithm of the input scale L, $\beta := \log(L)$, such that

$$p(\beta|\nu, \Lambda) = \mathcal{N}(\beta; \nu, \Lambda) = \frac{1}{\sqrt{2\pi\Lambda}} \exp\left(\frac{-(\beta - \nu)^2}{2\Lambda} \right) \tag{5.36}$$

where ν and Λ are hyperparameters, and where k_L is redefined as k_β. Substituting into (5.35), the marginalisation of the kernel becomes

$$\tilde{k}_{\nu,\Lambda} = \int k_\beta\ p(\beta|\nu, \Lambda) \mathrm{d}\beta \tag{5.37}$$

which unfortunately has a non-analytic solution.

As we do not wish to return to computationally demanding sampling strategies, we instead propose an approximation to the integral in (5.37) by first approximating k_β using a second-order Taylor expansion around the mean of $p(\beta)$ given by ν. Before we do that, however, we note that the approximate kernel must be positive

semi-definite in order to be valid. Whilst not all exponential functions are positive semi-definite, we can enforce positivity by putting k_β in exponential form and then later checking whether the full criteria are met. We therefore note the following identity and substitution:

$$k_\beta = \exp\left(-\ln\frac{1}{k_\beta}\right) = \exp\big(-A(\beta)\big) \tag{5.38}$$

We Taylor expand $A(\beta) = \ln\frac{1}{k_\beta}$ around $\beta = \nu$ as follows:

$$A(\beta) \approx \ln\frac{1}{k_\beta}\bigg|_\nu + (\beta-\nu)\frac{\partial A}{\partial\beta}\bigg|_\nu + \frac{1}{2}(\beta-\nu)^2\frac{\partial^2 A}{\partial^2\beta}\bigg|_\nu \tag{5.39}$$

and note the following result:

$$A'_\nu = \frac{\partial A}{\partial\beta}\bigg|_\nu = -\frac{k'_\nu}{k_\nu} \tag{5.40}$$

$$A''_\nu = \frac{\partial^2 A}{\partial^2\beta}\bigg|_\nu = -\frac{k''_\nu}{k_\nu} + \frac{{k'_\nu}^2}{k_\nu^2} \tag{5.41}$$

where $k_\nu = k_{\beta=\nu}$. Inserting these into (5.38) and (5.39), we arrive at the following expression for k_β:

$$\begin{aligned} k_\beta &\approx \exp\left(-\left(\ln\frac{1}{k_\nu} + (\beta-\nu)A'_\nu + \frac{1}{2}(\beta-\nu)^2A''_\nu\right)\right) \\ &= k_\nu\,\exp\left(-(\beta-\nu)A'_\nu - \frac{1}{2}(\beta-\nu)^2A''_\nu\right) \end{aligned} \tag{5.42}$$

Inserting (5.36) and (5.42) into (5.37), we are left with the following integral to solve:

$$\tilde{k}_{\nu,\Lambda} = k_\nu\int_{-\infty}^{+\infty}\frac{1}{\sqrt{2\pi\Lambda}}\exp\left(\frac{-(\beta-\nu)^2}{2\Lambda}\right)\exp\left(-(\beta-\nu)A'_\nu - \frac{1}{2}(\beta-\nu)^2A''_\nu\right)\,\mathrm{d}\beta \tag{5.43}$$

The solution to this leaves us with the following expression for $\tilde{k}_{\nu,\Lambda}$:

$$\tilde{k}_{\nu,\Lambda} = k_\nu \exp\left(\frac{\Lambda A_\nu'^2}{2\left(1+\Lambda A_\nu''\right)}\right)\frac{1}{\sqrt{1+\Lambda A_\nu''}} \tag{5.44}$$

the derivation of which can be found in Appendix B.2.

5.3.2 Proof of Positive Semi-Definiteness

In order to confirm that our new kernel $k_{\nu,\Lambda}$ is a legitimate covariance function, we must prove that it fulfils the necessary condition of kernels which is positive semi-definiteness. We first note that both a sum of kernels and a product of kernels is also a kernel (Rasmussen and Williams, 2006). In the first instance:

$$\tilde{k}_{\nu,\Lambda} = \int k_\beta \; p(\beta)\mathrm{d}\beta \tag{5.45}$$

should therefore be a legitimate kernel if k_β is also a legitimate kernel as $p(\beta)$ is strictly positive, and simply weights k_β through the integral. In order to confirm whether k_β is indeed legitimate we must use the expression for k_β given in (5.42). We cannot, however, do this until we have provided an expression for k_ν. In this chapter we consider two forms for k_L, the first of which we introduce in the next section along with a proof of how, in that case, k_β represents a legitimate kernel.

5.3.3 Approximately Marginalised Generalised Exponential

The aim of this chapter is to provide a powerful and *versatile* tool for representation of continuous functions in Bellman type equations. As has already been said, prior knowledge about the target function can be encoded in the covariance function. In the case of the mountain car problem we may be able to extract some insight into the value functions by inspection of the original problem, such as an expectation

of smoothness; however, this cannot necessarily be generalised to all problems. It is therefore prudent, when looking at the form of k_L, to consider a general form of covariance that can incorporate a range of prior knowledge.

In order to provide a highly expressive form of covariance we now introduce what we term the *Generalised Exponential* (GE):

$$k_\theta(\boldsymbol{s}, \boldsymbol{s}') = h^2 \exp\left(-\frac{1}{2}\frac{\triangle^2}{L^2}\right), \tag{5.46}$$

where h is a scaling parameter termed the *output scale*, and L is the input scale; the vector θ contains both hyperparameters. This form captures many of the most commonly used covariance functions. If $\triangle^2 = \sum_i (\boldsymbol{s}_i - \boldsymbol{s}'_i)^2$, (5.46) becomes the ubiquitous *squared exponential*. If $\triangle^2 = |\boldsymbol{s} - \boldsymbol{s}'|$, (5.46) is the *exponential covariance*. Many other covariances can be constructed by taking $\triangle^2$ as other metrics, allowing for periodic covariances, warping of the input space (Snelson et al., 2004), and inference in more structured domains (Garnett et al., 2010b). It is important to note, however, that $\triangle^2$ can not be any arbitrary metric; rather it must be an embedding into Euclidean space (see for example Hutter and Osborne (2013)).

The output scale h can be tractably marginalised according to suitable priors (Kennedy, 1998). Even if performing type-II maximum likelihood, the output scale is usually well-specified by even a moderate amount of data. We therefore do not consider this hyperparameter further.

The input scale L affects how closely outputs co-vary as a function of $\triangle^2$. Our predictions will ultimately be very sensitive to the value of the input scale: a short input scale will favour rough functions; and a long input scale will favour smoother functions. Our reasoning for concentrating on L in the first instance is that the likelihood of the input scale is often highly multi-modal: Figure 5.2 illustrated the likelihood surface as a function of $\theta = L$ for a periodic covariance, where the multiple

modes correspond to harmonics of the latent period and sampling frequency. Given limited data, the posterior for the input scale is likely to have large variance. As such, type-II maximum likelihood for this hyperparameter will likely lead to problematic over-fitting.

We proceed with our derivation of the *Approximately Marginalised Generalised Exponential* (AMGE) by rewriting the covariance function in (5.46) in terms of β as follows:

$$k_\beta(\boldsymbol{s}, \boldsymbol{s}') = h^2 \exp\left(-\frac{1}{2} \triangle^2 \exp(-2\beta)\right) = k_\beta \,. \tag{5.47}$$

where $\beta := \log(L)$ as before. From this we can find the derivatives of k_β:

$$k'_\beta = k_\beta \left(\triangle^2 \exp(-2\beta)\right) \tag{5.48}$$

$$k''_\beta = k_\beta \left(\triangle^4 \exp(-4\beta) - 2\,\triangle^2 \exp(-2\beta)\right) \tag{5.49}$$

which, using equations (5.40) and (5.41), are needed to calculate A'_ν and A''_ν, the partial derivatives in (5.39) evaluated at ν:

$$A'_\nu = -\,\triangle^2 \exp(-2\nu) \tag{5.50}$$

$$A''_\nu = 2\,\triangle^2 \exp(-2\nu) \tag{5.51}$$

Inserting these into (5.44), we get the following expression for the AMGE:

$$k_{AMGE} = h^2 \exp\left(-\frac{\triangle^2}{2\exp(2\nu)}\right) \exp\left(\frac{\Lambda\ \triangle^4 \exp(-4\nu)}{2(1 + 2\Lambda\ \triangle^2 \exp(-2\nu))}\right) \frac{1}{\sqrt{1 + 2\Lambda\ \triangle^2 \exp(-2\nu)}} \tag{5.52}$$

Our new kernel is the product of the original kernel k_β evaluated at $\beta = \nu$, and an expression which is a function of the squared distance $\triangle^2$.

As stated in Section 5.3.2, we simply need to confirm that k_β, as defined in

Equation (5.42), is a valid kernel in order to demonstrate the validity of our new kernel. Substituting the results from this section into Equation (5.42) and completing the square, we arrive at the following expression:

$$k_\beta = h^2 \exp\left(-\frac{\triangle^2 \exp(-2\nu)}{4}\right) \exp\left(-\frac{1}{2}\left(\beta - \nu - \frac{1}{2}\right)^2 \triangle^2 \exp(-2\nu)\right) \tag{5.53}$$

Both exponentials in (5.53) form legitimate kernels, which leaves us with the result that their product, k_β, is also a kernel and by definition is positive semi-definite as required.

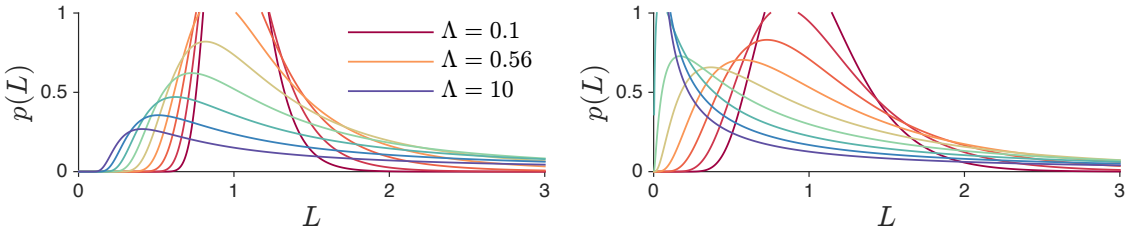
5.3.4 Relationship to the Rational Quadratic

The Rational Quadratic (RQ) covariance function:

$$k_{RQ} = \left(1 + \frac{\Delta^2}{2\alpha L^2}\right)^{-\alpha} = \int k_\beta(\boldsymbol{s}, \boldsymbol{s}) p(\boldsymbol{\beta}) \mathrm{d}\boldsymbol{\beta} \tag{5.54}$$

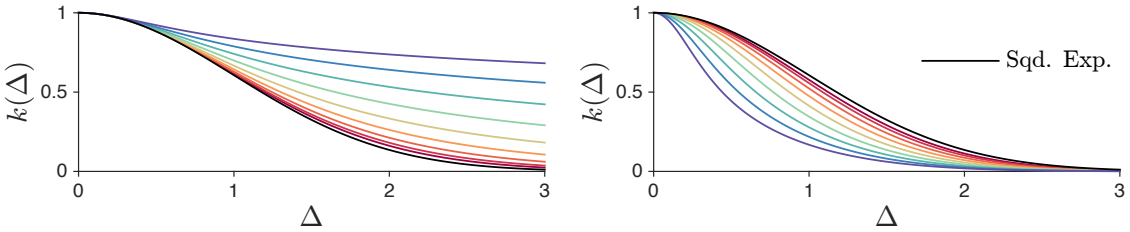
(discussed in Matérn (1960)) represents an alternative means of approximately marginalising the input scale of a squared exponential covariance. We introduce it here to provide a fair comparison with the AMGE when the Euclidean embedding $\triangle^2$ equals the squared difference, meaning the GE becomes the squared exponential. The RQ is only defined for $\triangle^2 = (\boldsymbol{s}_i - \boldsymbol{s}'_i)^2$ and is therefore less flexible than the AMGE, however we feel obliged to compare the two forms given the popularity of the squared exponential.

Where our approach in Section 5.3.1 was to adopt a log-Gaussian prior for the input scale, the RQ assumes a gamma prior on the inverse squared input scales. Problematically, this puts excessive weight on very large input scales (see Figure 5.3), such that the AMGE dominates the RQ for sufficiently small input scales. Figure 5.3(c) shows that as the variance of the gamma prior increases, the RQ covariance gets increasingly broad, favouring an interpretation of the data as being constant.



(a) Gamma prior distributions for $1/L^2$

(b) Gaussian prior distributions for $\log(L)$



(c) Rational quadratic covariances

(d) Approximately Marginalised Generalised Exponential covariances

Figure 5.3: The different priors assumed by RQ and AMGE covariances give rise to different behaviours as the variance in the input scale (Λ) increases. The priors of (a) assign more weight to very large input scales, whereas the priors of (b) assign more weight to small input scales. All figures have $\nu = 1$ and share the legends of (a)) and (d)).

On the other hand, the AMGE covariance becomes increasingly tight as its associated variance increases, favouring an interpretation of the data being i.i.d. We argue that the latter is preferable: if we are truly ignorant about the input scales, assuming that all data is uncorrelated is more reasonable than assuming that it is all perfectly correlated.

To provide an alternative argument, imagine performing type-II maximum likelihood on a small training set to train the input scales of a squared exponential covariance. If the test set is well-modelled by input scales significantly smaller than the trained input scales, the RQ is likely to provide poor predictions relative to the AMGE.

Although the prior distributions for AMGE and RQ are defined over different functions of the input scale, we can ultimately consider them in the same domain by

performing a change of variables such that:

$$p(L|\nu,\Lambda) = \frac{1}{L}\mathcal{N}(\log(L);\nu,\Lambda) \tag{5.55}$$

$$p(L|\alpha,\kappa) = \frac{2}{L^3}\ \text{Gamma}(1/L^2;\alpha,\kappa) \tag{5.56}$$

where α and κ are the shape and mean parameters of the gamma distribution. Whilst these distributions are functionally and parametrically different, in Section 5.4 we show that the performance of the resulting covariance functions becomes closely matched if we are able to match the distributions in some way. For this reason the main objective of our testing, which we discuss next, is to demonstrate the significant advantages an explicit distribution over input scales offers over vanilla methods such as squared exponential and alike. That being said, whilst this will demonstrate how the two methods provide equivalent functionality, we stand by the assertion that the AMGE is better placed for real-world application and practice through its more intuitive interpretation of how uncertainty should be handled.

5.3.5 Approximately Marginalised Matérn

The generalised exponential covariance function provides a highly flexible class of covariance functions in its own right. However, before proceeding to experimental performance analysis, we demonstrate the general efficacy of our approach and introduce the *Matérn class* of covariance functions, named by Stein (1999) after the work of Matérn (1960) and given by:

$$k_{Matern} = h^2\frac{2^{1-\tau}}{\Gamma(\tau)}\left(\frac{\sqrt{2\tau}\Delta}{L}\right)^{\tau}K_{\tau}\left(\frac{\sqrt{2\tau}\Delta}{L}\right) \tag{5.57}$$

where τ and L are positive parameters, K_τ is a modified Bessel function, and $\Delta = |\boldsymbol{s}-\boldsymbol{s}'|$.

A particular feature of the Matérn covariance is that functions drawn from a GP are k-times differentiable if and only if $\tau > k$ (Rasmussen and Williams, 2006). The covariance also becomes especially simple when τ is a half-integer. If $\tau = 1/2$, then the Matérn class gives the exponential covariance function, the corresponding process of which is *mean-square* (MS) continuous but not MS differentiable. This, however, is equivalent to the generalised exponential when $\triangle^2 = |\boldsymbol{s} - \boldsymbol{s}'|$, for which we already have a solution. We therefore concentrate on the next half-integer case, when $\tau = 3/2$:

$$k_{\tau=3/2} = h^2\bigg(1 + \frac{\sqrt{3}\Delta}{L}\bigg)\exp\bigg(-\frac{\sqrt{3}\Delta}{L}\bigg) \tag{5.58}$$

We are interested here in applying our approach to approximately marginalise the L hyperparameter, which is equivalent to the input scale parameter in the GE. Following through with the derivation as we did with the GE, we first rewrite (5.58) in terms of β as follows:

$$k_\beta = h^2\bigg(1 + \triangle\exp(-\beta)\bigg)\exp\bigg(-\triangle\exp(-\beta)\bigg) \tag{5.59}$$

where for simplicity we have redefined $\triangle = \sqrt{3}|\boldsymbol{s} - \boldsymbol{s}'|$. From this we can calculate the first and second derivatives of k_β as follows:

$$k'_\beta = h^2\,\triangle^2\exp\bigg(-2\beta\bigg)\exp\bigg(-\triangle\exp(-\beta)\bigg) \tag{5.60}$$

$$k''_\beta = h^2\,\triangle^2\exp\bigg(-2\beta\bigg)\exp\bigg(-\triangle\exp(-\beta)\bigg)\bigg(\triangle\exp(-\beta) - 2\bigg) \tag{5.61}$$

Once again we can use these to calculate the value of A'_ν and A''_ν:

$$A'_\nu = -\frac{\triangle^2\exp(-2\nu)}{(1 + \triangle\exp(-\nu))} \tag{5.62}$$

$$A''_\nu = \frac{\triangle^3\exp(-3\nu) + 2\,\triangle^2\exp(-2\nu)}{(1 + \triangle\exp(-\nu))^2} \tag{5.63}$$

Inserting these into (5.44) yields the final expression for the Approximately Marginalised Matérn, which for the sake of space we have written as follows:

$$k_{AMM} = h^2\left(1 + \triangle\exp(-\nu)\right)\exp\left(-\triangle\exp(-\nu)\right)\exp\left(\frac{\Lambda A_\nu'^2}{2\left(1+\Lambda A_\nu''\right)}\right)\frac{1}{\sqrt{1+\Lambda A_\nu''}} \tag{5.64}$$

where A_ν' and A_ν'' are defined above.

The expression for the AMMAT is, unfortunately, more complex than that of the AMGE. At this point we admit that an analytical proof of positive semi-definiteness remains elusive. Whilst this means the AMMAT has not been shown definitively to be a legitimate kernel, numerical testing strongly indicates that it does fulfil the necessary requirements: we performed $\mathcal{O}(10^6)$ numerical tests and found all to be PSD. Given this result we proceed with experimental testing on the assumption that a full proof will be forthcoming in future work.

5.4 Performance Analysis

Our objective now is to provide experimental evidence of the advantage brought about by using an AMCF over a standard covariance function, specifically when performing inference without the use of Bayesian or expensive sampling-based techniques. We focus our investigation on the underlying cause of overfitting in maximum likelihood techniques: namely the selection of a single hypothesis, in this case hyperparameter value, where full marginalisation would accommodate a range of hypotheses.

In the next section we introduce an experimental test bed which, in the first instance, generates a range of functions that legitimately describe a single set of observations. We show how our approach improves on the use of standard covariance functions by testing its ability to represent a broader range of these functions. Further to that, we also investigate the relationship between the abundance of observations and performance, and demonstrate that our approach is particularly useful when

observational data is scarce.

5.4.1 Experimental Test Bed

The experimental test bed, an overview of which is given in Figure 5.4, is based around a GP with known covariance and hyperparameters, which we call the 'base-GP'. The base-GP is designed for two purposes: firstly, to generate sets of observations on which the test methods are trained; and secondly, to generate a range of test functions. The choice of base-GP covariance (and hyperparameters) depends on the types of functions we would like to test our methods against; we consider several forms in this section.

Once the base-GP has been defined, the test bed begins by drawing a single function from the GP prior (Step 1). Following on from this, a set of between 3 and 20 observations are generated in the domain $\mathcal{S} \in [-3, 3]$ (Step 2). These observed values are used to train the hyperparameters on the test methods (Step 3). After training we return to the base-GP, and redefine the input scale L within a range defined by $[L_{\min}, L_{\max}]$. Each value of L defines a new GP, from which we draw a test function from the *posterior* (Step 4). Finally, a test set of 100 observations are taken uniformly from the test function, and the log-likelihood (LLH) and root-mean-squared-error (RMSE) is calculated for each test method (Step 5).

The test bed then loops back to Step 4, until 100 posterior test functions with different $L \in [L_{\min}, L_{\max}]$ have been generated. After this the test bed returns to Step 2 and generates a new set of n observations. Steps 2-4 continue cyclically until all $N_{TR} \in [3, 20]$ have been generated. Once this has been completed, the test bed returns to the base-GP (Step 1) and draws a new base function on which to generate observations and so on, a total of 2000 times. In all, the test bed generates 3.6×10^6 different test functions.

One thing to note regarding this set-up is that observations drawn from functions of relatively short input scales may need to be described largely as noise by less flexible

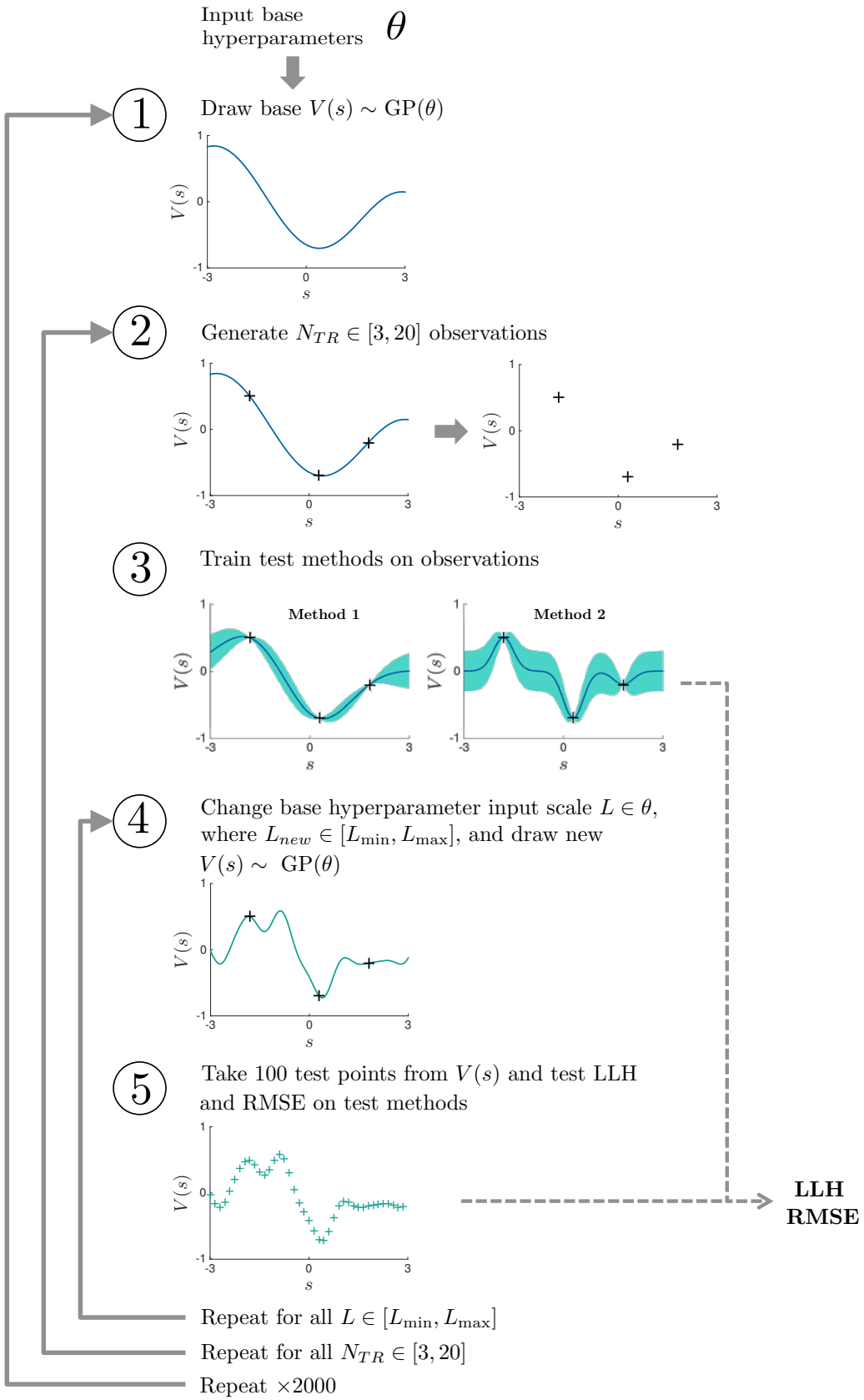


Figure 5.4: Overview of the experimental test bed.

functions, especially if those observations are close together. Rather than emphasise the noise process, we note that this problem does not present itself in reverse; hence in the following tests we set the input scale of the base-GP towards the higher end of the tested range.

5.4.2 Test 1 - Functions with Stationary Input Scales

When performing regression, we usually want to choose a model which is a good representation of the underlying data generating processes. In our first series of tests, we look at how our method performs when the statistical model is perfectly matched to the underlying process, in this case the base-GP. To do this, we simply set the covariance for both the base-GP and the test methods to be of the same functional form.

Squared Exponential

The squared exponential (SE) covariance function, which is equivalent to the generalised exponential with $\triangle^2 = \sum_i (\boldsymbol{s}_i - \boldsymbol{s}'_i)^2$, is where we begin the first set of tests. This allows us to compare the AMGE against the un-modified SE form, as well as to see how it relates to the RQ. The base-GP takes a SE covariance, we set the base input scale to $L = 1$, and the posterior test function input scales are set to range from $L_{\min} = 0.2$ to $L_{\max} = 2$. An example training set, based on the settings just described, can be seen in Figure 5.5, along with two posterior test functions.

In the first part of this test we want to address the similarities between the AMGE and RQ. In both of these cases, the input scale of functions drawn from the prior can take a range of values as a result of the distribution over input scales. When the prior is fused with observations to form a posterior GP, we should find that a much broader range of test functions can be described. In order to demonstrate this, we start by explicitly setting the hyperparameters during the training phase (Step 3 in

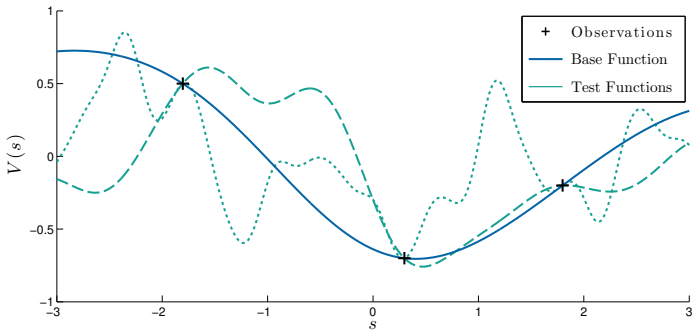


Figure 5.5: Plot of the training observations drawn from the base-GP with squared exponential covariance, along with test functions drawn from posterior GPs with differing input scales.

the test bed), both for the AMGE and RQ as well as an unmodified SE covariance.

In the case of the unmodified SE covariance, we set the input scale equal to that of the base-GP (i.e., $L = 1$). With regards to the AMGE and RQ, Equations (5.55) and (5.56) detail how we can consider the prior distributions over the input scale in the same domain; however it is not possible to set these as equal given they are of different functional forms. In order to match the distributions, we instead specify one of the distributions to be reasonably broad over the possible input scales of the test bed, and then optimise the hyperparameters of the other distribution in order to minimise the KL-divergence between the two. Figure 5.6 shows one such matching of the prior distributions.

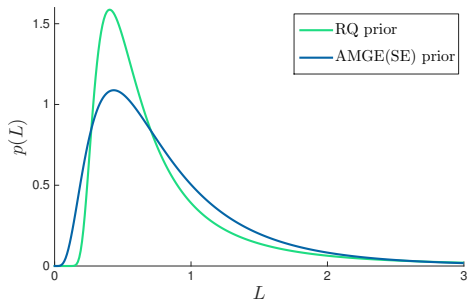


Figure 5.6: Plot of the input scale prior for both RQ and AMGE when matched by minimising their KL-divergence.

The results of the first experiment can be seen in Figure 5.7, which shows the median, upper and lower quartiles of the log-likelihood and RMSE for the test

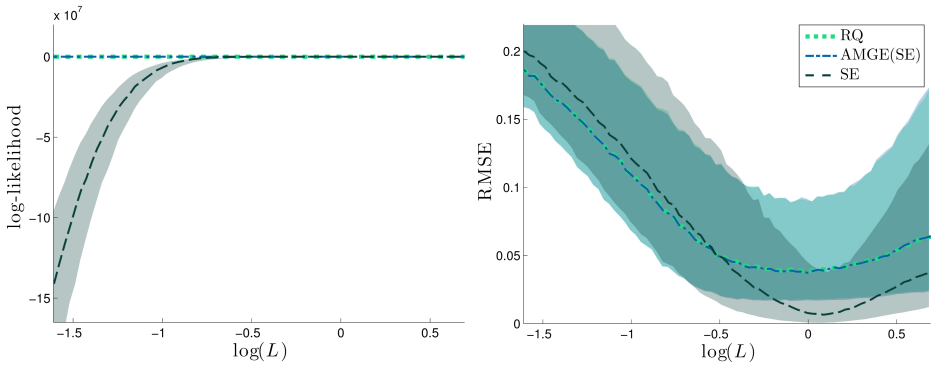


Figure 5.7: Performance of AMGE and RQ with matched hyperparameters, and SE with hyperparameters matched to the base-GP.

bed against the log input scale of the posterior test function. We opt to use the quartiles instead of mean and variance due to the skewed nature of the results. Before continuing with analysis we must first note the difference between log-likelihood and RMSE, and what each tells us. The RMSE helps us understand the predictive power of the GP, by comparing the test function with our best prediction for the true function (the mean of the posterior GP). The RMSE does not, however, assess the uncertainty in this prediction. The log-likelihood, on the other hand, considers how well the test functions can be explained by the trained GP model, and considers the full predictive distribution, rather than just the mean.

Returning to our analysis of Figure 5.7, we note two specific results: firstly, a significant drop off in log-likelihood for the SE covariance as the test function input scale reduces; and secondly, the RMSE for SE is actually lower than the other methods in the region around the true (base) input scale. The first of these results can be well explained by the fact we have explicitly made the AMGE and RQ expressive of a wide variety of input scales. The log-likelihood for these methods reflects this ability to describe the range of resulting functions. The SE is limited to choosing a single input scale and therefore its performance suffers in this regard. The second result is both a product of the SE model being a perfect match to the underlying process (both in covariance and hyperparameters), and also the nature of the AMGE and RQ

priors shown in Figure 5.6, which concentrate their weight away from the true input scale. We also note how the RMSE for all methods reduces as the true input scale is approached, something we will see in all subsequent tests. Ultimately the observations used for training are *always* generated by the base-GP, and we should therefore expect that the mean of the test methods will favour functions with an input scale in the same region as it.

Explicitly setting the hyperparameters of the test methods allows us to show how a distribution over input scales provides representation of a greater number of test functions, although this should not be done in practice. In our next experiment we move towards learning, and train the hyperparameters of both the SE and AMGE covariances on the training data. We drop the RQ from our experiments, as the performance of the AMGE and RQ has been shown to be the same across the entire range of test input scales, and thus these covariances are essentially exchangeable when $\triangle^2 = \sum_i (\boldsymbol{s}_i - \boldsymbol{s}'_i)^2$.

Regarding our inference procedure: given that the test bed uses very low numbers of training points relative to the tested input scales, type-II maximum likelihood will still perform badly. We therefore take a MAP approach by placing a hyper-prior over L for the SE covariance, and ν and Λ for the AMGE. The form of these are broad Gaussian distributions, which allow the training to concentrate on reasonable solutions when the number of training points is particularly low, whilst allowing the data to be expressive as observations increase. It is important to note that, whilst we would like to avoid the setting of any prior parameters, ultimately the test bed investigates an extreme end of the problem domain. Type-II maximum likelihood simply isn't practical here, so we maintain that, provided the prior information is relatively uninformative, this is the right approach to take.

The results of the test bed for both SE and AMGE using MAP estimation are shown in Figure 5.8. We see once again that AMGE outperforms SE across the lower

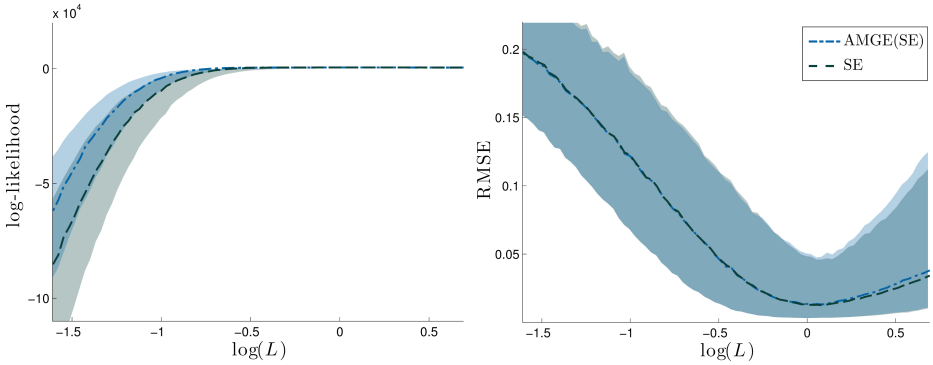


Figure 5.8: Performance of AMGE and SE using MAP estimation.

input scales. We can also see that both methods still perform best in the region of the true input scale, which makes sense given that as data increases, shorter hypotheses are less likely. In Figure 5.9 we zoom into this region and observe that the difference in log-likelihood between the two methods is very small. We might otherwise have expected SE to far outperform AMGE given that the AMGE distributes its weight across a broader range of input scales. We should note, however, that as we move past the base-GP input scale, the SE performance is higher than that of the AMGE. This makes sense if we remember that the AMGE covariance favours the data being less correlated under increased uncertainty.

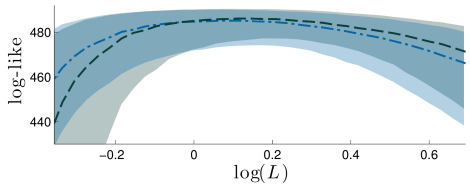
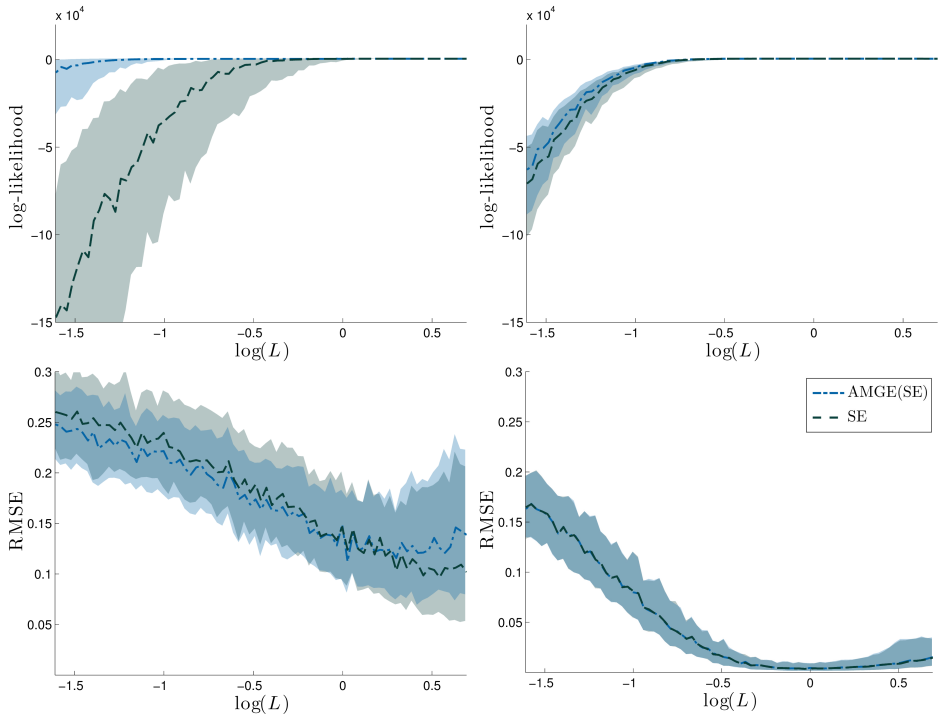


Figure 5.9: Performance of AMGE and SE in the region of the base-GP input scale.

The last two figures demonstrated the performance of both methods across the whole range of observations. In Figure 5.10 we plot the performance at the extreme ends of the tested range, specifically $N_{TR} = 3$ and $N_{TR} = 20$. We must be aware when analysing these figures that, as the number of observations increases, the range of permissible functions decreases. As this happens, the advantage that would have been gained in performing full marginalisation diminishes. Understanding this, we

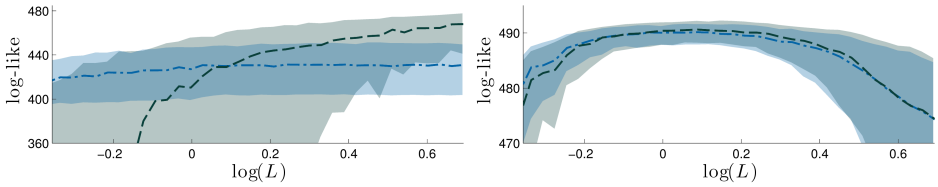


(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.10: Performance of AMGE and SE for different numbers of training observations.

see in the figure that the AMGE approach is particularly useful when the amount of training data is smallest. There is a significant performance difference between AMGE and SE when $N_{TR} = 3$ as AMGE maintains a distribution across the whole range of permissible hypotheses. Further to this, we note that as the data increases, the AMGE collapses to the correct solution (i.e., that of the base-GP) as Λ shrinks towards zero.

As before, we also include a closer inspection of the region around the base-GP input scale, shown in Figure 5.11. Under high uncertainty we see that SE favours longer input scales, although we should note that at the true input scale of $\log(L) = 0$, AMGE actually performs better. Once information increases, there is a negligible difference in performance between the two approaches.



(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.11: Performance of AMGE and SE in the region of the base-GP input scale for different numbers of training observations.

Quasi-Periodic

In our next experiment we move away from the squared exponential and in the first instance redefine $\triangle^2 = \sin(\frac{\pi}{T}(\boldsymbol{s}_i - \boldsymbol{s}'_i))$, which represents a periodic embedding into Euclidean space with time period T. We are interested in comparing the AMGE to the unmodified GE, both with the same periodic embedding. Purely periodic covariances are, however, of limited applicability. We therefore extend their form to be *quasi*-periodic, by multiplying each covariance by a SE covariance with a *secondary* input scale. In the case of the quasi-periodic GE, this means:

$$k_{GE(QP)} = k_{SE}\, k_{GE} \tag{5.65}$$

Figure 5.12 shows an example training set, along with two posterior test functions, for this test setup.

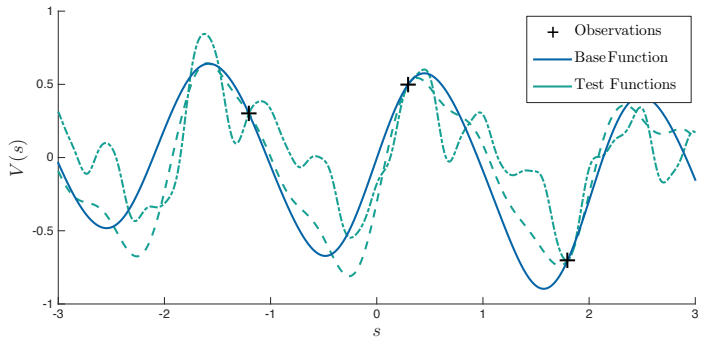


Figure 5.12: Plot of the training observations drawn from the base-GP with quasi-periodic covariance, along with test functions drawn from posterior GPs with differing input scales.

This experiment proceeds in the same way as the previous experiment, although we now use a different hyperparameter set for the base-GP, and investigate a different range of test functions. We set the secondary input scale (of the SE element) to $L_{se} = 4$, and the time period to $T = 2$. Both of these parameters are given to the covariances during the training phase so that we can focus on the primary input scale of interest. The primary input scale (of the GE element) we set to $L = 1$, and we set the posterior test function input scales to range from $L_{\min} = 0.2$ and $L_{\max} = 2$.

Figure 5.13 presents the aggregated results of the test bed. The results are very similar to those of the squared exponential, and we see that AMGE outperforms GE across the lower input scales. Again, both methods perform best in the region of the true input scale. In Figure 5.14 we once again zoom into this region, and observe the difference log-likelihood between the two methods is very small.

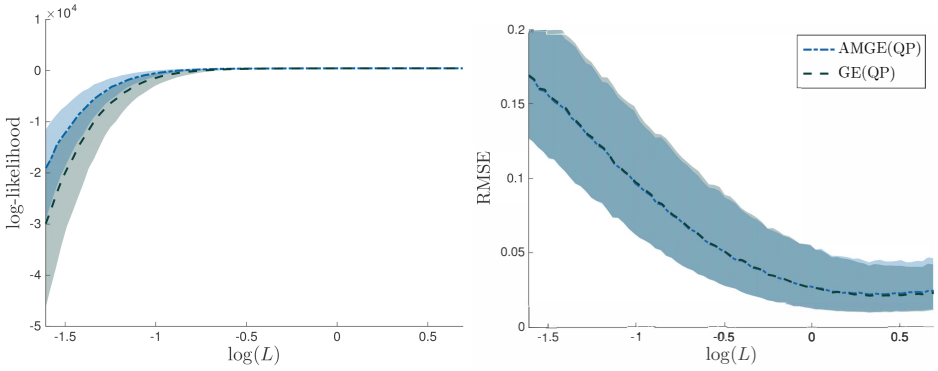


Figure 5.13: Performance of AMGE(QP) and GE(QP) using MAP estimation.

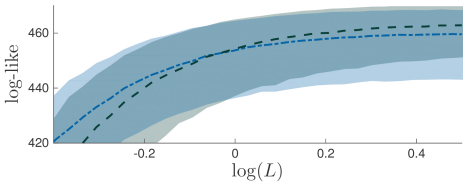
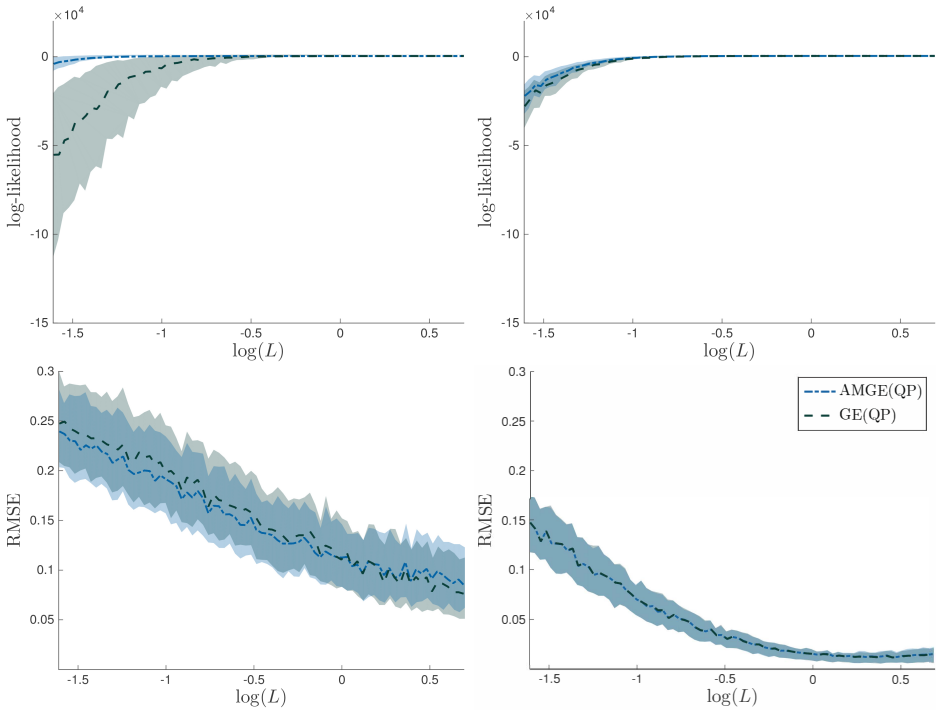


Figure 5.14: Performance of AMGE(QP) and GE(QP) in the region of the base-GP input scale.

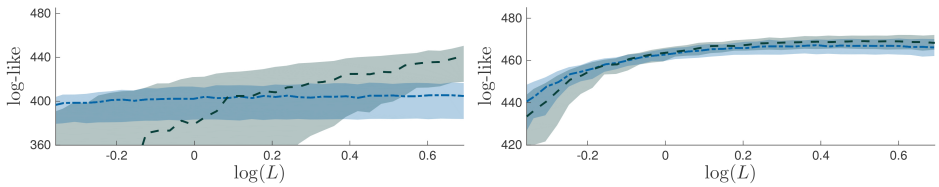
In Figure 5.15 we plot the performance for $N_{TR} = 3$ and $N_{TR} = 20$, and reinforce our statement that the AMGE approach is particularly useful when training data is

scarce. There is a significant performance difference between the AMGE and GE when $N_{TR} = 3$, which substantially reduces towards the other end of the tested range. In Figure 5.16 we zoom in to the region around the true input scale, and note much the same behaviour as we did in the last experiment.



(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.15: Performance of AMGE(QP) and SE(QP) for different numbers of training observations.



(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.16: Performance of AMGE(QP) and SE(QP) in the region of the base-GP input scale for different numbers of training observations.

Matérn

Our final experiment in this test set revolves around the Approximately Marginalised Matérn covariance[1]. We set the input scale of the base-GP to $L = 1$, and those of the test functions to range from $L_{\min} = 0.2$ to $L_{\max} = 2$. Figure 5.17 shows an example training set, along with two posterior test functions, for this test setup.

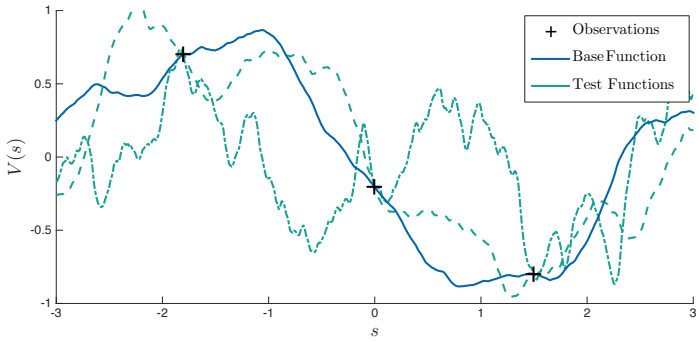


Figure 5.17: Plot of the training observations drawn from the base-GP with Matérn covariance, along with test functions drawn from posterior GPs with differing input scales.

We present the aggregated results from the test bed in Figure 5.18, and the region around the true input scale in Figure 5.19. Figure 5.20 provides the break down by number of observations, and Figure 5.21 the same results around the true input scale. Whilst the results here are analogous to those produced in the last two experiments, we can now say there is a benefit to the general application of the methodology outlined in Section 5.3.1, rather than just for the specific case of the generalised exponential.

[1]We use the term covariance under the assumption that an analytical proof of positive semi-definiteness will be found in future work, as mentioned in Section 5.3.5

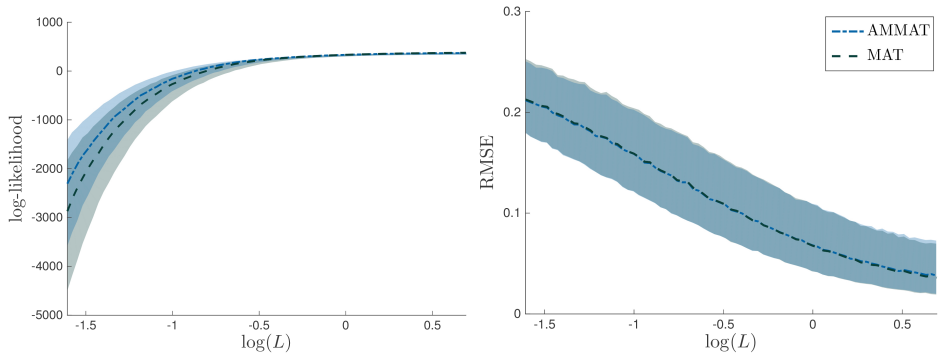


Figure 5.18: Performance of AMMAT and MAT using MAP estimation.

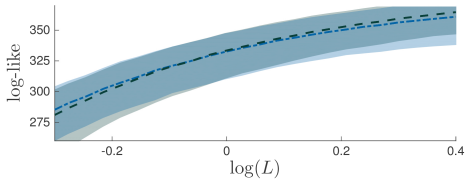
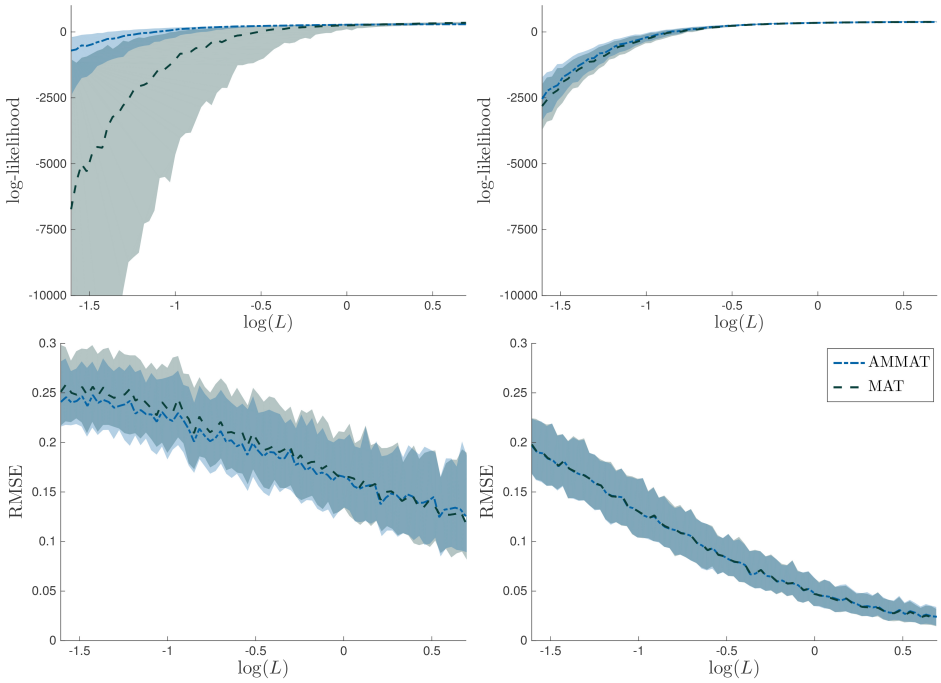
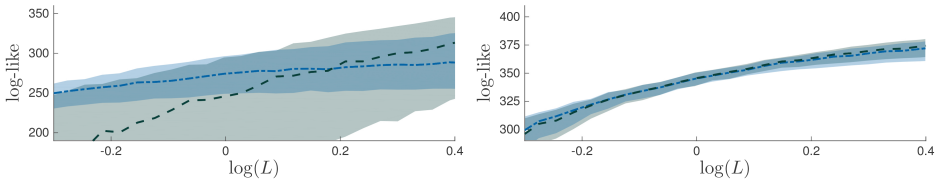


Figure 5.19: Performance of AMMAT and MAT in the region of the base-GP input scale.



(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.20: Performance of AMMAT and MAT for different numbers of training observations.



(a) $N_{TR} = 3$ (b) $N_{TR} = 20$

Figure 5.21: Performance of AMMAT and MAT in the region of the base-GP input scale for different numbers of training observations.

5.4.3 Test 2 - Functions with Non-Stationary Input Scales

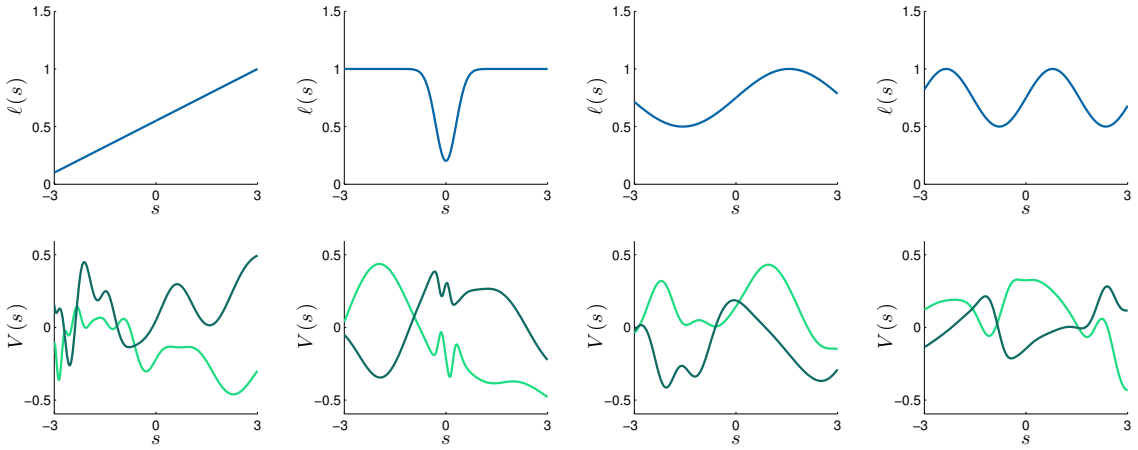
The previous set of experiments concentrated on problems where the model was perfectly matched to the underlying data generating process. This, however, is not necessarily a realistic approximation of the type of value functions we need to model in practice. In this section we look to 'stress-test' our approach against functions generated by a process which fundamentally violates the assumptions of the model.

The covariances considered so far have always been a function of the difference $(\boldsymbol{s}_i - \boldsymbol{s}'_i)$ and are therefore classed as *stationary*, meaning the covariance does not change as a function of the location in the input domain. In order to see how our approach handles functions which violate this property, we look to introduce a *non-stationary* covariance function into the base-GP. As noted in Rasmussen and Williams (2006), we are not free to simply make the input scale of a stationary covariance a function of $\boldsymbol{s}$, as it will not in general yield a valid covariance. Instead, we use the following covariance function derived in Gibbs (1997):

$$k(\boldsymbol{s}, \boldsymbol{s}') = \prod_{d=1}^{D} \left(\frac{2\ell_d(\boldsymbol{s})\ell_d(\boldsymbol{s}')}{\ell_d^2(\boldsymbol{s}) + \ell_d^2(\boldsymbol{s}')} \right)^{1/2} \exp\left(-\sum_{d=1}^{D} \frac{(s_d - s'_d)^2}{\ell_d^2(\boldsymbol{s}) + \ell_d^2(\boldsymbol{s}')} \right) \tag{5.66}$$

in order to generate non-stationary test functions for the test bed. In this case, $\ell(\boldsymbol{s})$ can be any arbitrary positive function of $\boldsymbol{s}$.

To create the experimental test bed, we start by defining a set of $\ell(\boldsymbol{s})$ functions, pictured in Figure 5.22 alongside samples from the resulting GP prior with Gibbs'



(a) linear (b) inverted Gaussian (c) $\sin(s)$ (d) $\sin(2s)$

Figure 5.22: Diagram illustrating the set of $\ell(\boldsymbol{s})$ functions in the test bed alongside samples from the resulting GP prior with Gibbs' covariance function.

covariance function. These ensure the set of test functions is challenging and varied. Given that we are interested in seeing how an AMCF performs when its underlying assumptions are violated, we focus solely on the squared exponential version of the AMGE rather than the whole suite of AMCFs generated thus far. As before, we compare the AMGE against the unmodified SE covariance, but also include a GP using the same Gibbs' covariance as the base-GP, and where $\ell(\boldsymbol{s})$ and all other hyperparameters are known (in essence, the best-case model). The test bed is run four times, once for each of the four $\ell(\boldsymbol{s})$ functions.

In Figure 5.23 we plot the total performance of all three methods on the non-stationary covariance test bed. Concentrating on the log-likelihood, we can see an enormous performance advantage when using AMGE over SE. We also note that the full range of the log-likelihood is far below that pictured for SE, as the whiskers only represent the limit of data within 1.5 IQR (interquartile range) of the quartiles. The Gibbs' covariance, which in theory cannot be improved upon, gives further context to this result. Ultimately the SE simply does not possess the flexibility to acknowledge more than one input scale, making it particularly ill posed for this sort of application. To reinforce this, we note from our testing that under greater amounts of data, the

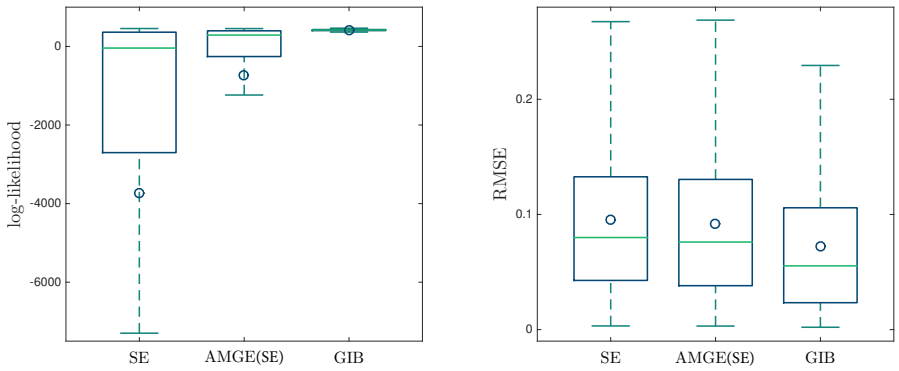
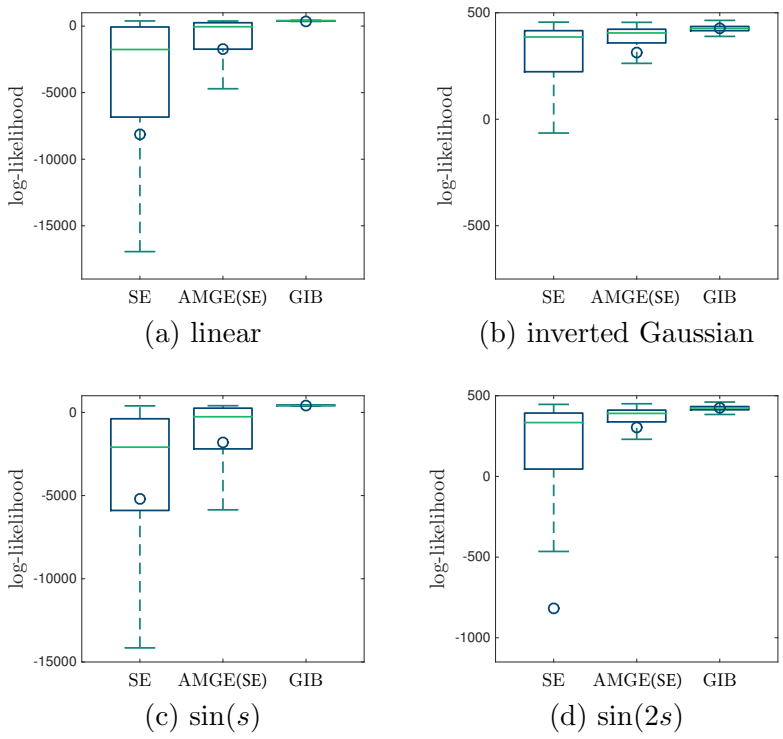


Figure 5.23: Performance of AMGE, SE and Gibbs' covariance with known $\ell(\boldsymbol{s})$ on the non-stationary covariance test bed. The mean is given by a circle, the quartiles by a box, and the median given by a horizontal line within each box. The whiskers represent the limit of data within 1.5 IQR of the quartiles.

MAP surface being optimised was highly concentrated around a Λ value greater than zero, meaning the method favoured maintaining a distribution over input scales *even* when given plenty of data. Looking at the RMSE we see that the performance difference between the two methods is negligible. Taking these two results together, we must conclude that the main problem when using SE is that it exhibits insufficient uncertainty about the underlying function.

For interest purposes we include the individual performances of all three methods on each of the tested $\ell(\boldsymbol{s})$ functions in Figure 5.24. We only present the log-likelihood as once again the difference between the two stationary methods was negligible for RMSE. Each function has its own challenges and we observe substantial performance differences between them, although AMGE always significantly outperforms SE. Referring back to Figure 5.22, certain methods appear as if they may be better modelled by a stationary covariance than others, which would explain the observed differences. Others, such as the inverted Gaussian, have only a small region in which the input scale changes, meaning that the loss encountered by choosing a simpler model would be less than with, for example, the linear function. All together, the results from this test bed should hopefully emphasise the need to test methodology



(a) linear (b) inverted Gaussian

(c) $\sin(s)$ (d) $\sin(2s)$

Figure 5.24: Performance of AMGE, SE and Gibbs' covariance with known $\ell(\boldsymbol{s})$ on the non-stationary covariance test bed, segmented according to the $\ell(\boldsymbol{s})$ function used.

outside of idealised experimental setups, and certainly away from functions which are based on the original assumptions of the model.

Finally, in Figure 5.25 we plot the overall performance of all three methods as a function of the number of training points. In line with our previous experiments, we see that AMGE is much more robust with low numbers of observations than SE, and in fact much closer in performance to the matched Gibbs' covariance. Rather interestingly, we notice an increase in the IQR of the log-likelihood for AMGE as the number of observations increases past around 5. Through investigation we only observed this behaviour on the harder two problems, namely linear $\ell(\boldsymbol{s})$ and $\sin(\boldsymbol{s})$. It seems that under a few observations, a single fixed input scale may begin to look likely, causing performance to drop. As observations increase further, the function looks less stationary, and the method correctly asserts a distribution over the input scale again. We should note that this behaviour is not clearly observed in the median,

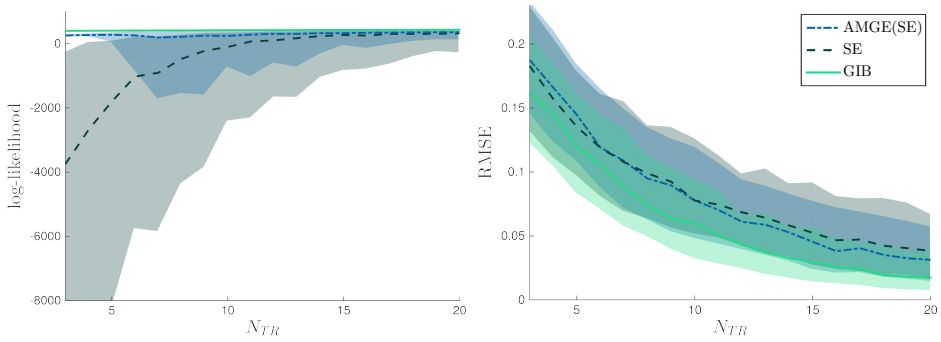


Figure 5.25: Performance of AMGE, SE and Gibbs' covariance on the non-stationary covariance test bed as a function of the number of training points.

suggesting that it only occurs in a subset of problems based on the specific positions of the samples.

5.4.4 Test 3 - Real World Datasets

In our last test, we move away from the experimental test bed and look towards real world problems. We take two examples from the UCI repository (Lichman, 2013): firstly, the *auto-mpg* dataset, which records the fuel consumption in miles per gallon against 3 multivalued discrete and 4 continuous attributes; and secondly, the *Boston housing* dataset, which records the median value of owner-occupied homes against 1 binary and 12 continuous attributes. Given we do not have any specific domain knowledge for either of these, we use the squared exponential flavour of the AMGE and compare this against the unmodified SE covariance.

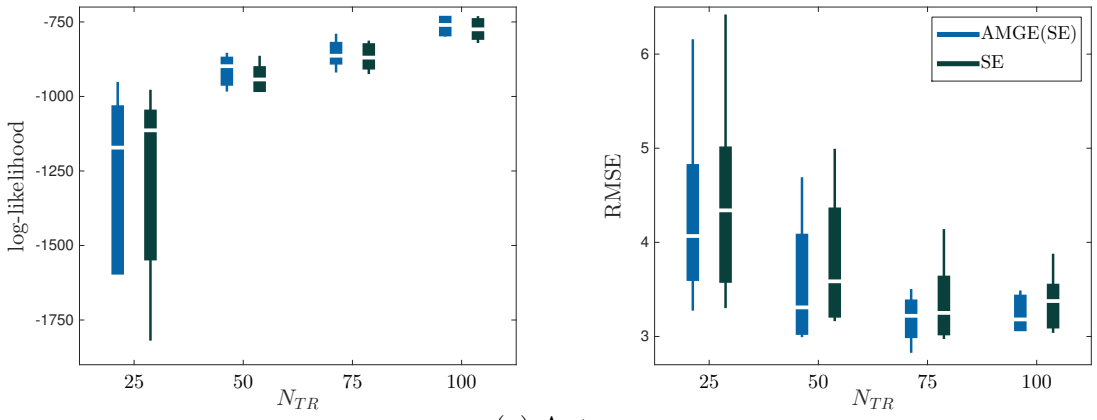
With regards to the auto-mpg dataset, there are 398 data points in the dataset, however 6 of these have missing values which, for simplicity, we exclude from testing. The Boston housing dataset has 506 data points, with no missing values. In order to gain an insight into how the two methods handle different numbers of training points, we test $N_{TR} = 25, 50, 75$ and 100 on each dataset. In each case we select the training points randomly, and repeat the test 10 times for each N_{TR}. The log-likelihood and RMSE are calculated for each method on the remaining data points in the set.

The inference procedure for both methods is substantially the same: we perform type-II maximum likelihood over the input scales of the SE covariance, and the ν hyperparameters of the AMGE(SE) covariance. We introduce broad priors over the Λ hyperparameters of the AMGE(SE) and perform MAP inference for these. Whilst our approach improved on SE without introducing these priors, further performance was gained by doing so. Ultimately we feel that an advantage of using the AMGE(SE) is our ability to inject uncertainty into the inference procedure without making any statement about the true input scales, as would be the case had we placed a prior over ν or L. In any event, the effect of the Λ priors naturally reduces as more training data is available.

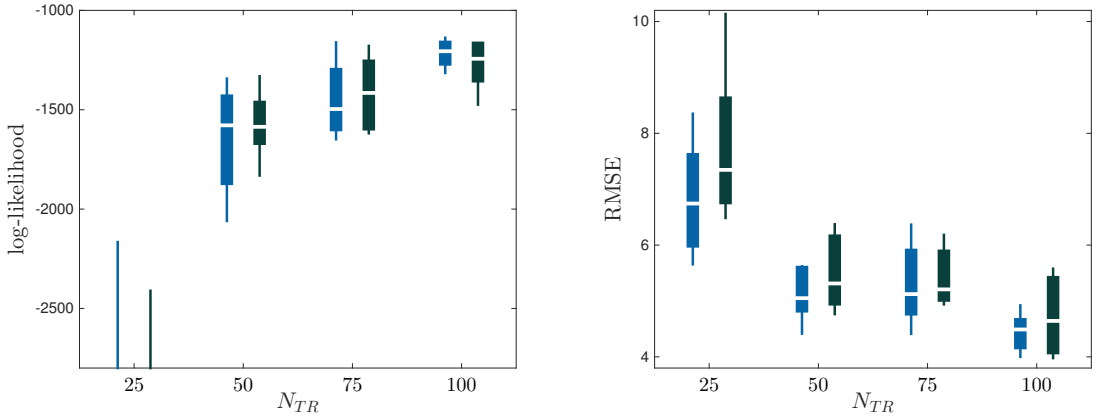
Figure 5.26 presents the results of the testing against the number of training points. The box represents the upper and lower quartiles, the horizontal line gives the median, and the whiskers represent the limit of data within 1.5 IQR of the quartiles. We note that performance generally increases for both methods as the number of training points given increases. With regards to the likelihood, the AMGE(SE) mostly dominates SE, suggesting that uncertainty is better apportioned in the posterior. The predictive performance of the AMGE(SE), given by the RMSE, is also greater than that of the SE. Both of these results suggest that the AMGE(SE) has a lower propensity to overfit than the SE when using sub-optimal inference methods.

5.5 Gaussian Processes for RL

We now return to the original motivation for this chapter, that of Reinforcement Learning under continuous states and actions. Gaussian processes have already been shown by a number of authors to be effective in this domain. As examples, Kuss (2006) uses a GP model to perform both policy iteration and temporal difference learning, whilst Deisenroth (2010) introduces a means of performing dynamic programming,



(a) Auto-mpg



(b) Boston Housing

Figure 5.26: Performance of the AMGE(SE) and SE and covariance functions, plotted against number of training points, on the two real world problems. The log-likelihood quartiles for $N_{TR} = 25$ on the Boston Housing problem are: $[-1.25 \times 10^7, -1.05 \times 10^4, -2.92 \times 10^3]$ for AMGE and $[-2.08 \times 10^8, -1.26 \times 10^4, -3.86 \times 10^3]$ for SE.

as well as a scheme for placing observations in order to reduce uncertainty in the unknown dynamics. These provide convenient frameworks on which to base our algorithms, in particular 'Gaussian Process Dynamic Programming' (GPDP) presented in Deisenroth et al. (2009), which we adapt here.

These methods demonstrate the efficacy of the overall GP approach, although their development is usually independent of the choice of covariance function. Whilst this offers great flexibility and should certainly be considered a benefit, we note that the main motivation in developing the AMGE, specifically the computational bottle-

neck of full Bayesian inference, is of particular concern here. Ultimately we would like to implement decision methods that can provide real-time decision making, and full Bayesian inference simply is not feasible for the time being. As we have discussed already, when using type-II maximum likelihood or other naive inference processes the choice of covariance function is of crucial importance, as uncertainty is not handled appropriately. Given this and the performance advantages demonstrated so far, it is prudent that we test the AMGE alongside other choices of covariance in the context of Reinforcement Learning, as well as focussing on algorithmic development.

5.5.1 Value Iteration

In this section we choose to address the problem of value iteration, in which the objective is to find $V^*(\mathcal{S})$, the optimal value function defined across the state space $\mathcal{S}$. From this it is possible to extract the optimal policy $\pi^*(\mathcal{S})$ that maps states to actions. We will develop two algorithms here, one based on GP representation and a second using a Bayesian linear model as described in Section 5.1. This provides a natural continuation to work in Lee (2009) where, as noted earlier, a linear basis model using RBFs substantially outperformed discretisation techniques. For these algorithms we will also consider the transition function $\mathcal{T}$ to be known; this allows us to focus on representation rather than exploration vs. exploitation problems, which we approach later in the thesis. It is worth noting that, whilst the objective of these algorithms is to find the 'optimal' value function, operating in the continuous domain restricts us to finding approximations, the quality of which determines performance.

Value iteration specifies the overall functionality on which we must build our means of representation. Broadly, the algorithm begins by initialising all states to some value $V_0(\cdot)$, then selects actions across the state space that maximise the expected value under the current estimate. This propagates a new value $V_1(\cdot)$ to the states for the next round of action selection. The algorithm terminates once the

Algorithm 1 Gaussian Process Value Iteration

input $\mathbf{S}, \mathcal{T}, \gamma, \Delta t, \zeta$
$V_0(\mathbf{S}) = r(\mathbf{S}) + \epsilon_n$ ▷ terminal cost
$\mathcal{GP}_v \leftarrow V_0(\mathbf{S})$ ▷ train value GP
$k \leftarrow 0$
repeat
  $k \leftarrow k + 1$
  for all $\mathbf{s}_i \in \mathbf{S}$ **do** ▷ for all support states
    $V_k(\mathbf{s}_i) = r(\mathbf{s}_i) + \epsilon_n + \gamma \max_F \left(m_\theta(V_{k-1}|\mathcal{T}(\mathbf{s}_i, F, \Delta t))\right)$
  end for
  $\mathcal{GP}_v \leftarrow V_k(\mathbf{S})$ ▷ re-train value GP
until $|V_k(\mathbf{S}) - V_{k-1}(\mathbf{S})| < \zeta$ ▷ check for convergence
return $\mathcal{GP}_v$

difference between two value estimations is below a certain threshold, indicating convergence. The question we must solve is how to represent the value estimates, of which we can only have a finite set of observations. As mentioned, here we adapt GPDP for this purpose and represent the value function at a finite set of states $\mathbf{S} \in \mathcal{S}$, which we refer to as *support states*. Observation of the value function at these points allows us to generalise across the continuous state space using either flavour of function approximation.

Gaussian Process Value Iteration (GPVI) is sketched in Algorithm 1. The required inputs are the set of support states $\mathbf{S}$, transition function $\mathcal{T}$, discount factor γ, time step Δt and convergence threshold ζ. Provided there are enough support states we presume to use type-II maximum likelihood, although clearly any priors could also be passed to the GP model. With regards to the support states, it is possible to take these as fixed throughout, although in contrast to GPDP we advocate choosing a new set at each iteration in order to prevent the propagation of systematic errors.

In accordance with standard value iteration, the first step of GPVI is to initialise the value function, which is done by evaluating the reward function at $\mathbf{S}$ and which in general may be corrupted by noise ϵ_n. Using this initial estimate we then perform the first training of the value GP, from which estimation of the value function for any

state $\mathbf{s}_i \in \mathcal{S}$ can be made. As a note, the kth value estimate is given by:

$$V_k(\cdot) \sim \mathcal{GP}_k(m_\theta, C_\theta) \tag{5.67}$$

where m_θ and C_θ are the learnt mean and covariances of the GP. Following the initial value estimate, the algorithm begins alternating between action selection, followed by updating of the value function at the support states and retraining of the GP. Concentrating on the action selection process, although we maintain a distribution over the value at each state, the Bellman equation (and through that value iteration) dictates that we must take an expectation. Fortunately this is given directly by the mean of the distribution, such that $\mathbb{E}[V_k(\mathbf{s}_i)] = m_\theta(V_k|\mathbf{s}_i)$. We should note that for GPDP a separate Q-value GP model is calculated for each support state *before* optimisation takes place. This is required if the immediate reward is a function of both state and action, and thus must also form part of the action optimisation objective. Here we deliberately consider reward to be only a function of state, although acknowledge that this can be extended accordingly. Termination of the algorithm occurs once the difference in value at the support states between two iterations falls below ζ.

Bayesian Linear Model Value Iteration (BLMVI), sketched in Algorithm 2, is functionally very similar to GPVI. We state now that we choose to use full Bayesian inference in the linear model as this process is tractable, and therefore seems a fair comparison to the GP approach. This means the baseline performance to which we compare the GP method is higher than had we chosen to select the parameters through, for example, a maximum likelihood approach. The required input into BLMVI is more elaborate than for GPVI, as we must specify the basis functions $\boldsymbol{\phi}$ through which the support states are mapped. We must also provide the prior covariance over the weight parameters Σ_p, taking the mean of the prior to be zero and the observational noise variance σ_n. As an alternative, Bishop (2006) offers guidance

Algorithm 2 Bayesian Linear Model Value Iteration

input $\mathbf{S}, \mathcal{T}, \boldsymbol{\phi}, \sigma_n, \Sigma_p, \gamma, \Delta t, \zeta$
$\Phi = \boldsymbol{\phi}(\mathbf{S})$ ▷ transform support states
$A = \frac{1}{\sigma_n^2}\Phi\Phi^\top + \Sigma_p^{-1}$ ▷ inverse weight covariance
$V_0(\mathbf{S}) = r(\mathbf{S}) + \epsilon_n$ ▷ terminal cost
$k \leftarrow 0$
repeat
  $k \leftarrow k + 1$
  for all $\mathbf{s}_i \in \mathcal{S}$ **do** ▷ for all support states
    $V_k(\mathbf{s}_i) = r(\mathbf{s}_i) + \epsilon_n + \gamma \max_F \left(\frac{1}{\sigma_n^2} \boldsymbol{\phi}(\mathcal{T}(\mathbf{s}_i, F, \Delta t))^\top A^{-1} \Phi V_{k-1}(\mathbf{S}) \right)$
  end for
until $|V_k(\mathbf{S}) - V_{k-1}(\mathbf{S})| < \zeta$ ▷ check for convergence
return Φ, A, V_k

on how Σ_p and σ_n may be estimated from the data.

The first step in BLMVI is to map the support states through the predefined basis functions in order to generate $\boldsymbol{\Phi} = \boldsymbol{\phi}(\mathbf{S})$. From this the model weight covariance can be calculated, which is independent of observations. The algorithm then proceeds as before and initialises the value function at the support states to the reward function, before entering the iterative action selection procedure. As we take a Bayesian approach with a predefined prior distribution over the linear weights, there is no training of the parameters in the same way as GPVI. For a given set of observations $V_k(\mathbf{S})$, the estimate of the value at state $\mathbf{s}_i$ is given by:

$$p(V_k(\mathbf{s_i})|V_k(\mathbf{S})) = \mathcal{N}(\frac{1}{\sigma_n^2}\boldsymbol{\phi}_i^\top A^{-1}\boldsymbol{\Phi}V_k(\mathbf{S}),\, \boldsymbol{\phi}_i^\top A^{-1}\boldsymbol{\phi}_i) \tag{5.68}$$

where definitions are consistent with Section 5.1. Expected value is also easily found as the mean of this distribution, such that $\mathbb{E}[V_k(\mathbf{s}_i)] = \frac{1}{\sigma_n^2}\boldsymbol{\phi}_i^\top A^{-1}\boldsymbol{\Phi}V_k(\mathbf{S})$.

The last step in the process of both algorithms is to extract the (approximately) optimal policy. In some settings it may be desirable to provide another function approximation that directly maps states to actions. Deisenroth et al. (2009) note that this function can be discontinuous, and develop a model with two GPs along

with a switching element that selects the appropriate output. We observe that any discontinuity is heavily dependent on the problem at hand, and in general may require further model complexity. We also note that actions can be extracted by simply repeating the optimisation process over F already found in both algorithms (lines 8 and 9 respectively), and so for the purpose of testing we choose this approach rather than explicitly calculating a policy model.

5.5.2 The Mountain Car Problem

We test the value iteration algorithms on the mountain car problem, as formulated at the start of this chapter. For the GPVI we test both a standard SE covariance and the squared exponential flavour of the AMGE covariance. For BLMVI we choose to use radial basis functions, as they were shown to be effective in Lee (2009). Using RBFs, one can centre a basis function on each support point as a means of exact interpolation (see for example Bishop (2006)); however we take the approach of Lee (2009) and set the number of RBFs to be much less than the size of $\mathbf{S}$ in order to generate a smooth approximation. Specifically, we choose a grid of 100 bivariate RBFs with $\sigma_x = 0.15$ and $\sigma_{\dot{x}} = 0.3$, as represented in Figure 5.27.

It was mentioned earlier that we advocate providing a new set of support states at each iteration, as this was found to eliminate systematic errors. For the initial set of support states, we take the approach of Kuss (2006) and generate a $[19 \times 19]$ grid on the state space, which is similar to the 400 supports points used to test GPDP. After each iteration we randomly perturb these positions with Gaussian noise, as this means the entire input domain remains well covered whilst assuring that $\mathbf{S}$ changes. In should be noted that not all states in the domain are able to choose an action without violating the restrictions on x and $\dot{x}$. As we automatically set the value surface to be equal to zero outside the permissible range, the value of these areas will also be forced to zero.

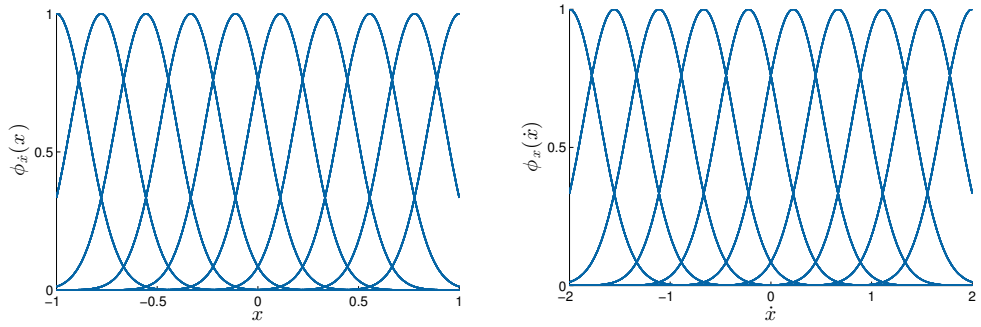


Figure 5.27: Radial basis functions over x and $\dot{x}$, with the opposing variable marginalised. There are 10 basis function centres defined on each dimension, resulting in a grid of 100 bivariate radial basis functions with $\sigma_x = 0.15$ and $\sigma_{\dot{x}} = 0.3$.

The discount and time step tend to vary on each implementation of the mountain car problem; given that the dynamics we use are taken from Kuss (2006) it is appropriate to adopt the setting given there of $\gamma = 0.9$, and $\Delta t = 0.2$. For reference, Rasmussen and Kuss (2004) use $\gamma = 0.8$ and $\Delta t = 0.3$ and achieve similar results. With regards to the convergence factor, the rate of convergence was found to be different for each of the tested methods. We therefore instead opt to run the algorithm for 10 iterations, and find that this gives sufficient convergence in all cases. The final specification we need to provide is the prior weight covariance and observational noise variance for BLMVI. In order to set a reasonably uninformative prior over the linear weights we set $\Sigma_p = 2\mathbf{I}$, and for simplicity we set σ_n equal to the true observational noise variance of 0.01^2.

In Figure 5.28 we plot the value surfaces at various iterations to demonstrate how this evolves. These can be compared with surfaces plotted in Kuss (2006, Figure 7.5) and Sutton and Barto (1998, Figure 8.10). It is quite clear from this figure that the RBFs chosen for the BLMVI provide a much smoother approximation than either of the GPVI methods. This is particularly noticeable after initialisation, when the small reward area is not well represented by BLMVI. In contrast, we notice that the GP methods are able to well represent the initial reward distribution, but the SE covariance results in many more localised peaks as the number of iterations increases.

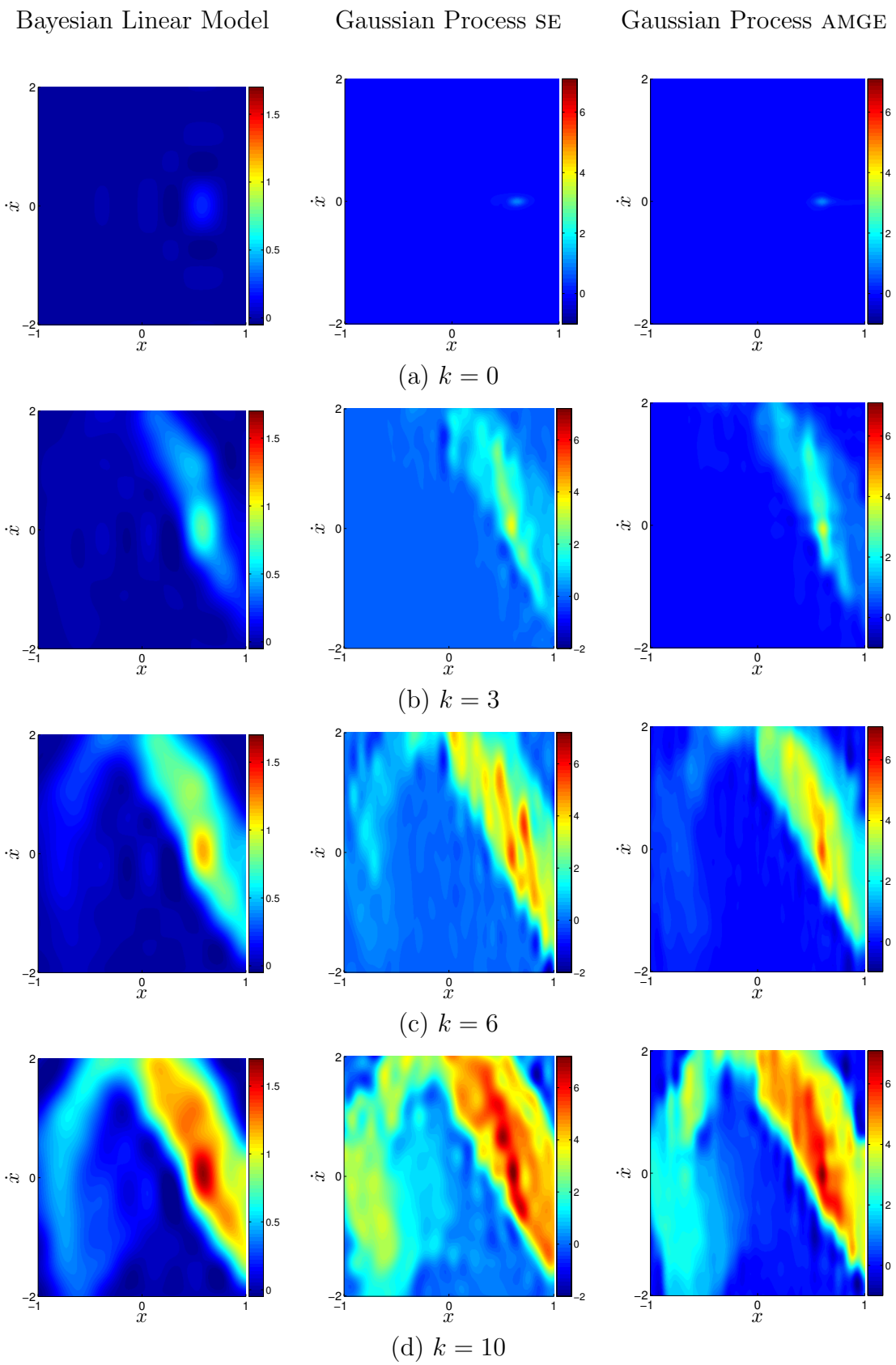


Figure 5.28: Approximate value functions for the mountain car problem during value iteration for the Bayesian Linear model and Gaussian Process model (with SE and AMGE(SE) covariance functions). The mean of each value function is shown after a) initialisation, b) 3 iterations, c) 6 iterations, and d) convergence after 10 iterations.

The AMGE(SE) covariance appears to find a good balance between fine representation whilst retaining the smoothness we should expect from the rest of the surface. We should also note that, due to smoothing, the magnitude of the value surface is less accurate in BLMVI; the true maximum of the surface, found at the centre of the reward distribution, should be $\sum_0^{10} \gamma^i 1 \simeq 6.86$. Lastly, we can see that certain corner areas of the surface remain at zero due to the constraint violation condition already mentioned.

In Figure 5.29 we illustrate the policy extracted from each of the methods after the final iteration. This is shown as state transitions across a grid of starting states, over which a full trajectory is superimposed starting from $\mathbf{s} = [-0.5, 0]$, the base of the hill. We can see that all three methods capture the essence of the solution and direct the car to reverse up one side before accelerating down and then up the other. In Figure 5.30 we plot the x state trajectory for each method against time. The AMGE(SE) method arrives at the reward location first, by a clear margin, followed by SE and then the BLM. Interestingly, the BLM achieves a more accurate representation at the reward location, and thus achieves a better final position than SE, although AMGE(SE) is the closest to the centre.

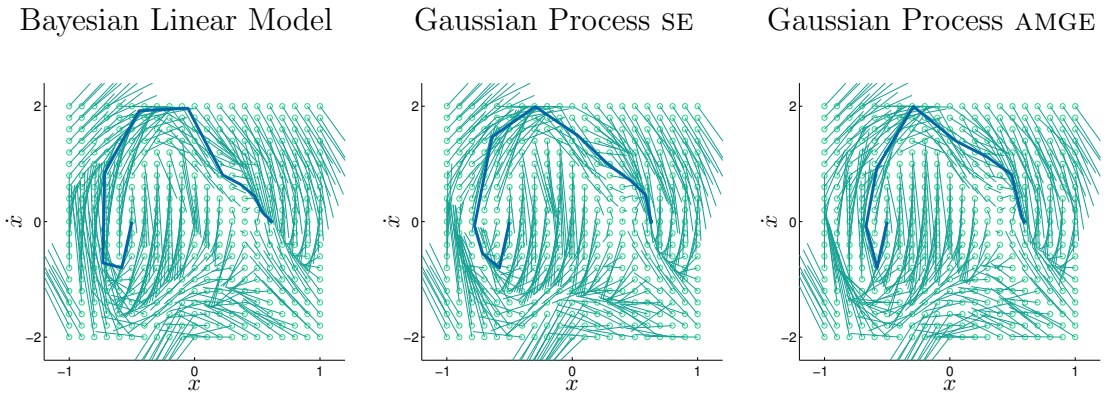


Figure 5.29: State transitions from policy extracted after value iteration using either the Bayesian Linear model, or the Gaussian process model with SE or AMGE covariance. Transitions are shown for a single step when applying the policy across a grid of starting states. A single trajectory of the policy is shown, starting from $\mathbf{s} = [-0.5, 0]$ (the bottom of the valley) and continuing until $t = 5$, where each time step $\Delta t = 0.2$.

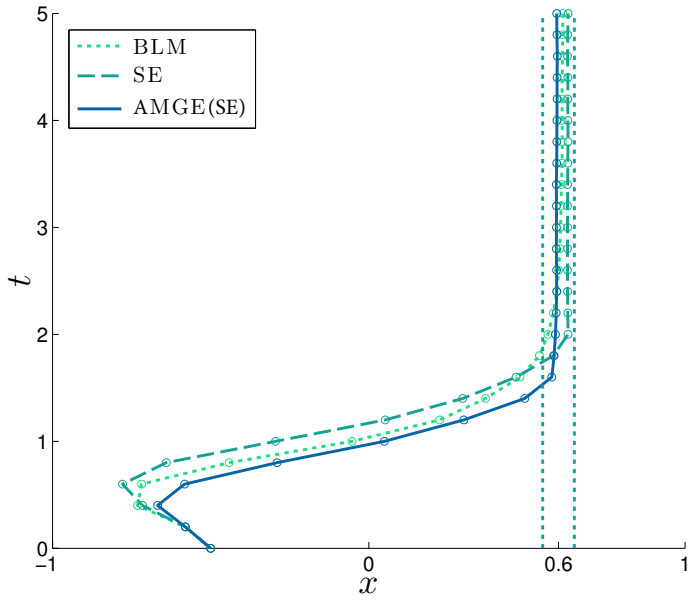


Figure 5.30: Policy trajectories starting from $\mathbf{s} = [-0.5, 0]$ and continuing until $t = 5$, where each time step $\Delta t = 0.2$. The vertical bars represent one standard deviation away from the mean of the reward distribution.

It is clear that the favoured method of all three approaches is that of GPVI using the AMGE(SE) covariance, thus demonstrating the need to encode some form of uncertainty over the input scale. Surprisingly, the BLMVI method also does well on the problem, and ultimately seems to find a more optimal solution than that found using GPVI with a SE covariance. However, let us remember that in Section 5.1 we noted the limitations fixed basis functions can bring, particularly in regard to the fact they must be specified a priori. In order to demonstrate the effect a poor choice can make, we repeat BLMVI using a different set of basis functions, represented in 5.31. Note that the state domain is still well covered, although we choose fewer and broader RBFs than before.

Figure 5.32 illustrates the final value surface and resulting policy trajectory when using the redefined basis function set. Predictably, the surface appears even smoother than before due to the broader basis functions. This results in a poorer representation of the surface around the reward peak, and thus the resulting policy is clearly sub-optimal. Given this result it could be argued that the basis functions

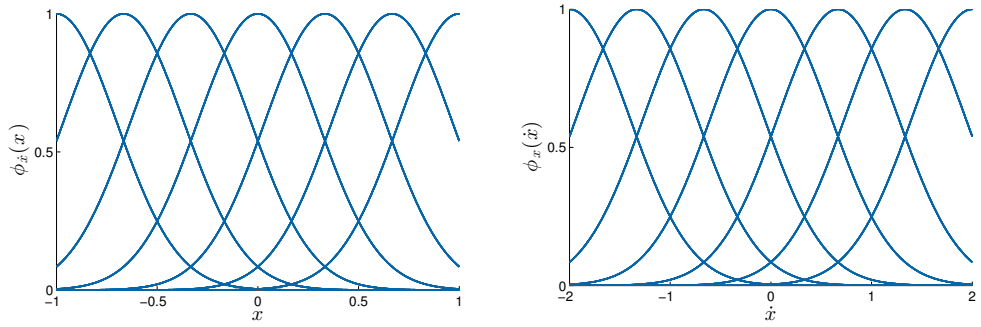


Figure 5.31: A different choice of radial basis functions over x and $\dot{x}$, with the opposing variable marginalised. Here there are 7 basis function centres defined on each dimension, resulting in a grid of 49 bivariate radial basis functions with $\sigma_x = 0.3$ and $\sigma_{\dot{x}} = 0.6$.

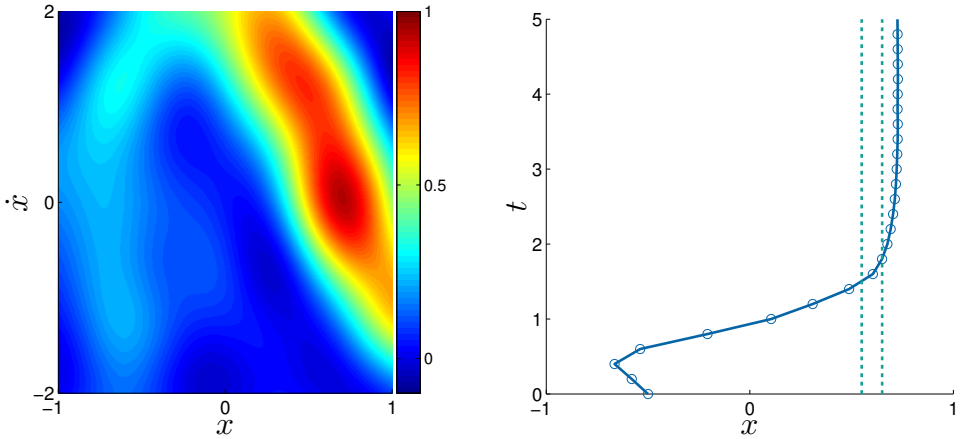


Figure 5.32: The final value surface and policy trajectory from the base of the hill derived from BLMVI with a redefined basis function set. The value surface is smoother, and the resulting policy is clearly sub-optimal.

should be set so that their width is closer to the width of the true reward distribution, although this information may not be known in advance. The benefit in using GPs is that no such specification must be made to ensure good performance.

Whilst we have shown the trajectory prescribed for each of the methods when starting at the base of the hill, an improved test of representation requires us to test from multiple starting states. We therefore select 200 random start locations within the constrained state space and observe the total reward received by following the policy extracted from each value surface. Note that we exclude any start states which do not transition to a permitted state under any action. Figure 5.33 illustrates the

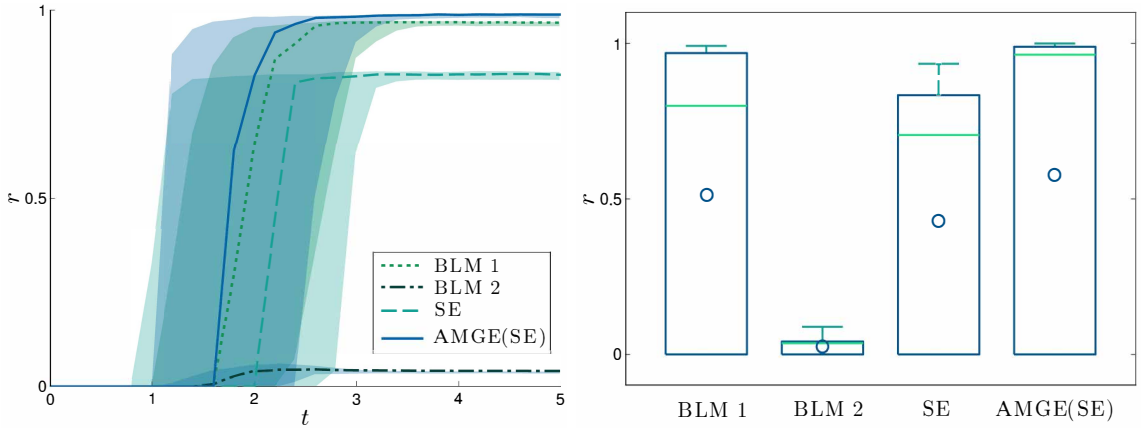


Figure 5.33: Reward performance for each method across 200 randomly selected start states. The left hand plot shows the median, upper and lower quartile reward as a function of time. The right hand plot illustrates overall performance, where the box represents the upper and lower quartiles, the horizontal line gives the median, the whiskers represent the full range and the mean is given by the circle.

distribution of reward both as a function of time as well as total performance across the 200 problems.

Once again, we see that GPVI using the AMGE(SE) exceeds all other methods both in terms of total performance and the time it takes to reach the objective area. We can also see that BLMVI is considerably weaker under a poor choice of basis functions. We therefore believe that GPVI provides a better solution overall. Ultimately, the fewer choices we have to make a priori, the less sensitive the final solution will be to these choices. GPs offer a sophisticated way of representing a wide range of functions, and whilst we need to specify a covariance function, this choice is better grounded in common sense than subjective positioning of basis functions. Clearly there is a requirement that we acknowledge the limitations of less principled inference mechanics, and we did this through the development of AMCFs.

Chapter 6

Solving the Exploration Exploitation Dilemma

> "As soon as we admit the existence of a lack of completely accurate measurement, or, as we shall say, uncertainty, in one part of the system, we must allow for it in every part."
>
> Bellman (1971)

An online search engine, *'Bingle'*, sells sponsored search results, or adverts, to a set of clients. The goal of Bingle is to maximise revenue, which is achieved by maximising the number of adverts that are clicked on. Bingle is entitled to choose which adverts it displays at any given time and is financially rewarded on a per click basis. As clients do not pay for adverts that are displayed but not clicked, Bingle must be careful not to miss out on earning opportunities by choosing to display less popular adverts.

The success of an advert can only be determined by presenting the advert to users and observing the number of click throughs. Unfortunately, this learning process must happen at the same time as the number of click throughs is to be maximised, which generates a difficult decision problem. Bingle must find a balance between *exploring* the profitability of adverts about which little is known whilst *exploiting*

adverts that are known to have high click through rates.

The essence of exploration vs. exploitation problems is the making of decisions and action taking in order to maximise some notion of cumulative reward, whilst simultaneously learning about the system. To reiterate what we said in Chapter 3, for Markov Decision Processes this uncertainty concerns the unknown dynamics of the environment, $\mathcal{P}$. Successfully solving problems of this class relies on the understanding that information about these unknowns has the potential to enable more rewarding action taking in the long run, even if it involves taking actions that appear to be less profitable in the short term. In the case of Bingle, the more information they have about the relative popularities of adverts now, the easier it is for them to set about maximising their profits later on. It makes intuitive sense that information in this context has an intrinsic *value*, just as states and actions that were not in themselves rewarding could still be *valuable*.

Returning to the problem faced by Bingle, in order to see how to encode information into the decision process we must start by defining a model. In general there will be many reasons why a particular user may click on an advert, such as relevance, timing, and personal taste. If there is no information available which is peculiar to individuals, one could assume that each advert has a single 'ground-truth' success probability, or click through rate, which can be inferred from observations. Although the causal structure of an individual's preference is highly complex, this simple model seems tenable in the absence of further information.

Using the model as a basis, imagine Bingle wants to optimise the click throughs for the search term 'chocolate' (Figure 6.1), for which it has three adverts to choose between: advert A selling dark chocolate, advert B selling white chocolate, and advert C selling milk chocolate. Let us imagine that the three adverts have been presented to users to differing extents: A and C to a low number of users, and B to a high number of users (Figure 6.2). Given the observations taken from presenting these

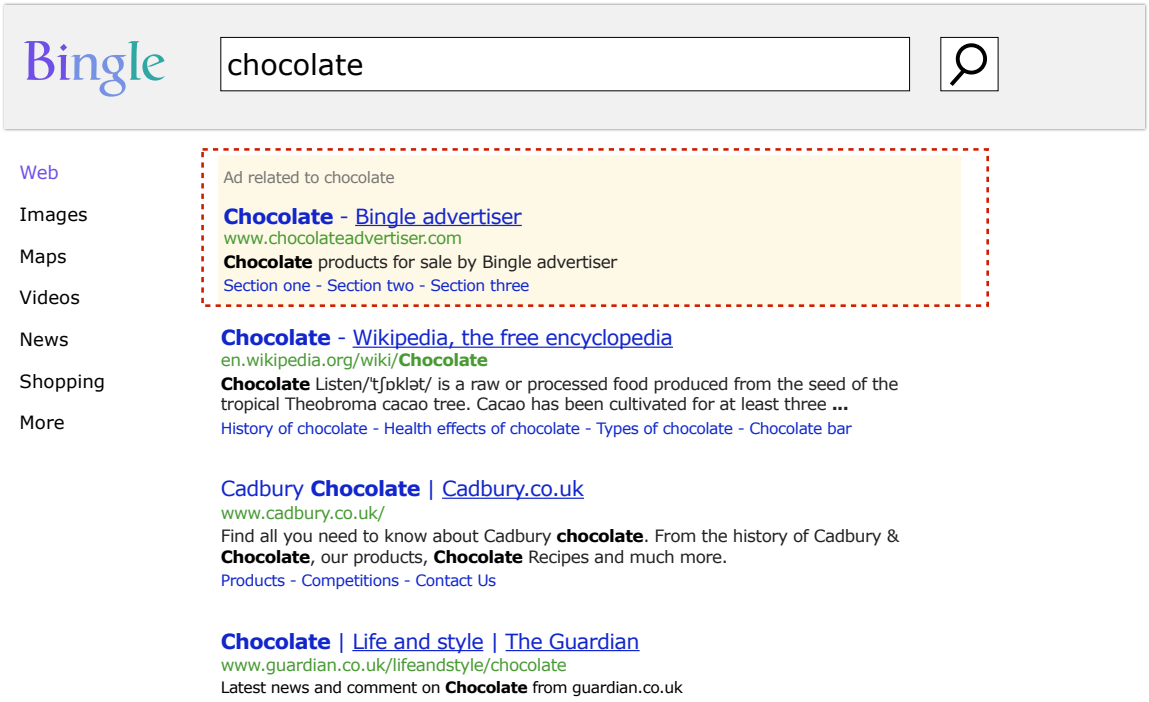


Figure 6.1: Bingle search results for the search term 'chocolate'. A sponsored search result (highlighted in red) appears above organic search results.

adverts to users, we can make inferences as to the true click through rate, illustrated as posterior distributions in the figure. The width of these distributions is determined by the number of times each advert has been displayed. As this number increases the distribution becomes tightly concentrated around, or rather confident in, the 'true' value.

At any point we can evaluate the *expected* click through rates of each advert, marginalising out the uncertainty in each distribution. Intuitively we might think of the expectation as our 'best guess' of which advert will be the most rewarding; hence selecting the advert which maximises this estimate is what we mean by 'exploitation'. Whilst it would be appropriate to do this if the next decision was our last, in general it is sub-optimal as knowledge acquisition is not incorporated into the decision process. In fact, any new information that reduces uncertainty could only be considered a fortunate consequence.

In contrast, our model provides us with some possibilities for evaluating how

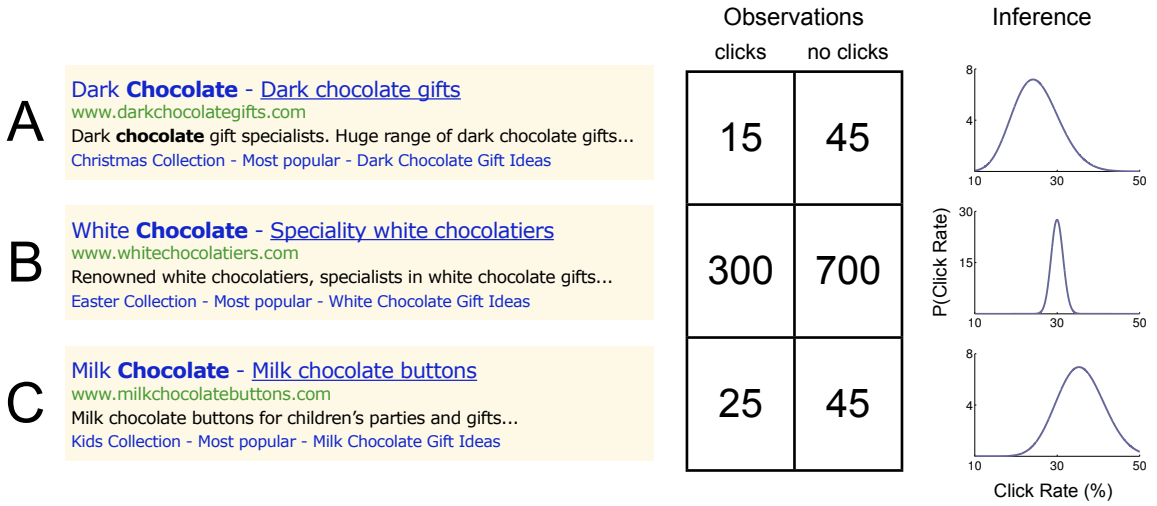


Figure 6.2: Comparison of three adverts associated with the search term 'chocolate'. Observations enable inference about the true click rate.

much information can be gained by displaying each advert. For instance, uncertainty in the posterior distributions would not be an unreasonable place to start defining exploratory actions. However, in doing so we are faced with the problem of how to build *this* into the decision process. Pure exploration, as with pure exploitation, is sub-optimal, and in fact could result in choosing uniformly amongst the possibilities. The nature of the tradeoff is that the best approach seems to lie somewhere between the two extremes. Unfortunately, there is no way to mathematically derive quite where this point is.

Let us take a step back. Whilst we felt obliged to introduce the problem in this way, ultimately dividing the action space into exploitative and exploratory actions only serves to create questions which can only be answered with heuristics. As stated in Chapter 3, by correctly applying Bayesian analysis in an MDP not only do we automatically incorporate knowledge acquisition into the decision process, but ensure no such distinction ever has to be made. In fact, to describe Bayesian decision making in the language of exploration and exploitation can be quite difficult and arguably non-sensical. Having said that, in Chapter 7 we do attempt to decompose Bayesian decision making in terms of these concepts as a means to develop principled

approximations. In terms of this chapter, however, let us just accept that decision making under unknown dynamics is almost universally referred to under the explore exploit moniker, and be aware that it will not feature in the derivation of optimal methodology.

In the next section we introduce the *Multi-Armed Bandit* problem framework, which is a generalisation of the problem faced by Bingle. The MAB is a decision problem that isolates the 'exploration exploitation' process from that of value propagation across physical states, as was discussed in Chapter 4; most real-world problems will typically be a combination of the two. Formulated as an MDP, the MAB environment has only one state, meaning that action taking only affects the decision maker's *understanding* of the system, leaving the environment unaffected. Although it is perhaps the simplest way to examine exploration exploitation problems, the solution was hard earned. According to Peter Whittle, the problem was so demanding of Allied analysts during the war that it was proposed it be 'dropped over Germany as the ultimate instrument of intellectual sabotage' (Gittins, 1979).

6.1 The Multi-Armed Bandit

6.1.1 Problem Formulation

Originally introduced in the statistics literature under the title "the sequential design of experiments" (Thompson, 1933, 1935; Robbins, 1952), the multi-armed bandit is a sequential decision problem analogous to a traditional slot machine, or one-armed bandit, except that there are K levers to choose from instead of one (Figure 6.3). In the slot machine setting, a player inserts a coin, pulls a lever, and receives a payoff determined by the resting position of several spinning drums. In the multi-armed bandit, an agent, or player, is faced repeatedly with a choice between K different actions, $\mathcal{A} = \{a_1, a_2, ..., a_K\}$, analogous to choosing to play one of a set of

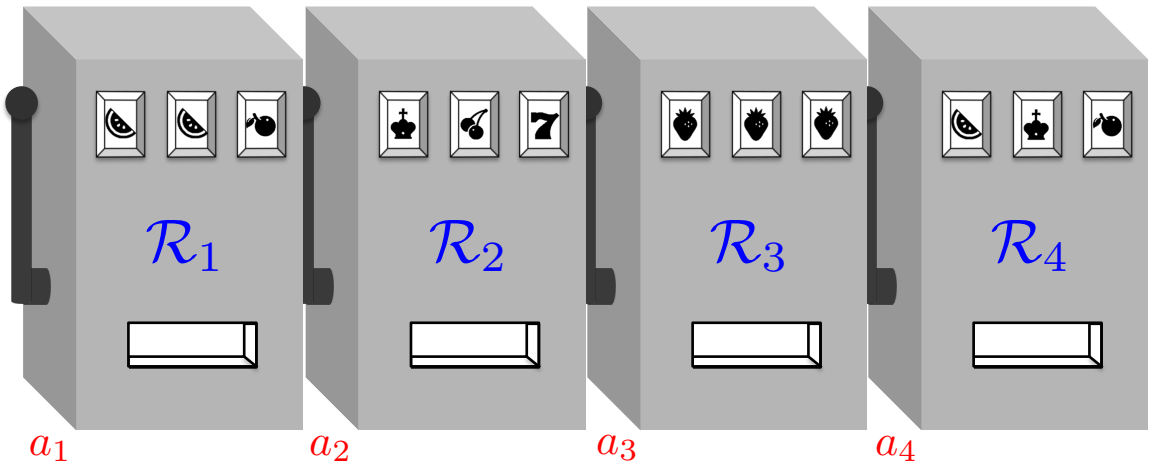


Figure 6.3: The multi-armed bandit: each action is analogous to choosing to play one of a set of traditional slot machines, or one-armed bandits. Each machine has a different probability of payoff, or reward distribution, associated with it, which the decision maker does not necessarily know in advance.

different machines. When an action a_j is taken, a stochastic reward is drawn from a reward distribution $\mathcal{R}_j \in \{\mathcal{R}_1, \mathcal{R}_2, \ldots \mathcal{R}_K\}$ associated with that machine (or action), parameterised by $\theta_j \in \{\theta_1, \theta_2, \ldots, \theta_K\}$. An action in this context does not necessarily incur an immediate cost in the same manner as inserting a coin does in the casino setting. In general the reward distributions will be of the same parametric family across the action space.

The underlying reward distributions are, in the simplest setting, fixed; when an action is taken, a reward is drawn from the corresponding reward distribution and the bandit transitions back to the same state. The problem is formally equivalent to a one-state Markov Decision Process, as shown in Figure 6.4. It is, however, more convenient to imagine the K-armed bandit as being the conjunction of K states, each of which represents a one-armed bandit, as shown in Figure 6.5. When an action is taken, the associated arm is activated, outputting a reward before returning to its resting state. This approach highlights the independence property of arms on a bandit; when an action is taken, all arms except the activated one remain at rest, and as such the decision maker only learns about the underlying reward distribution

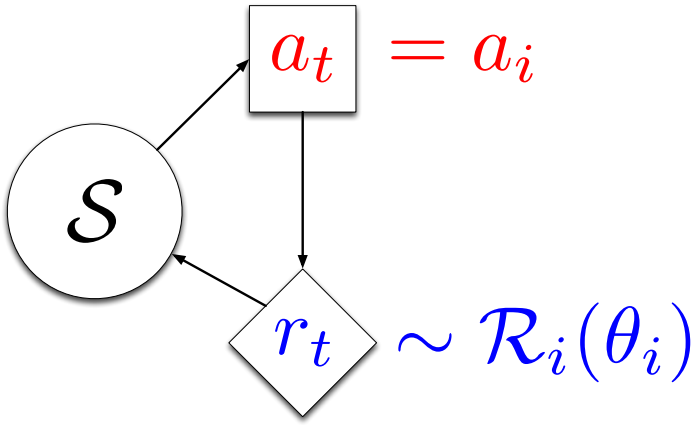


Figure 6.4: The multi-armed bandit, formally equivalent to a one-state Markov Decision Process. An action a_j is chosen at time t, and a reward r is drawn from probability distribution $\mathcal{R}_j$ with associated parameter vector θ_j before transitioning back to the starting state.

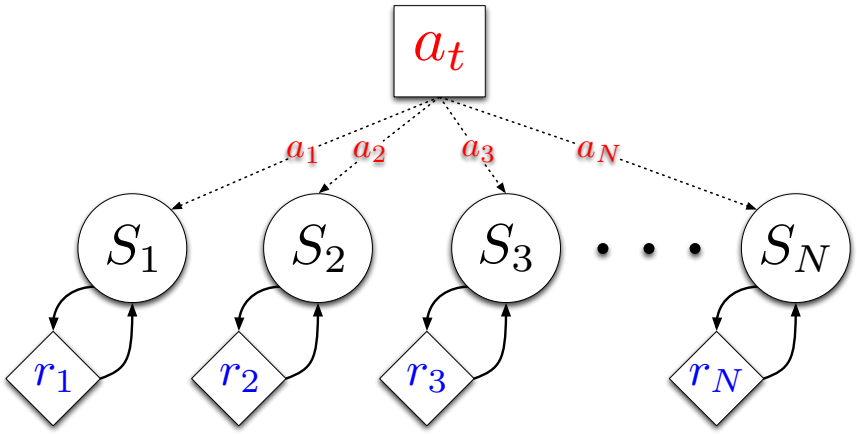


Figure 6.5: The multi-armed bandit, viewed as the conjunction of K one-armed bandits. Bandit j is activated when action a_j is taken, and a single reward $r_j \sim \mathcal{R}_j(\theta_j)$ is generated.

of the activated arm.

In order to perform inference about the underlying reward distributions, and therefore to evaluate the best action to be taken, the decision maker records the history of past rewards, $\mathcal{D}$, drawn from the bandit. As each action is independent, rewards are recorded against the action that was responsible for the received reward. If $\mathbf{d}_{j,t}$ is the current history of rewards received when taking action a_j up to the current time step t, then $\mathcal{D}_t = \{\mathbf{d}_{1,t}, \mathbf{d}_{2,t}, \ldots, \mathbf{d}_{K,t}\}$. The elements of $\mathcal{D}_t$ will inevitably be

of different sizes unless all the actions have been taken to the same extent. As the reward distributions are static, and therefore the rewards drawn from a single action are *i.i.d*, it is not necessary for the rewards to be time indexed. If, however, we want to refer to a specific time indexed reward, we will denote $r_{t+1}^{a_j}$ as the reward received after taking action a_j at time t.

In Section 3.3.1 we described the objective of the agent as that of maximising cumulative reward, or return R, defined by Equation (3.9):

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots + r_{\mathcal{H}}$$

The objective in the MAB is precisely the same, such that the decision maker should still seek to find the policy that maximises expected (discounted) return, as defined in Equations (3.11) and (3.12):

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}\left[R|\pi\right]$$

and

$$\mathbb{E}\left[R_t\right] = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1}\right]$$

Central to the MAB is the fact that the reward function, $\mathcal{R}$, is not only stochastic, but also unknown to the decision maker. If we assume that at least the parametric family $\mathcal{R}$ belongs to is known, then the decision maker will have a belief in what the parameters of the distribution are, represented by $p(\theta|\mathcal{D}_t)$, conditioned on the history of rewards. Using this, the expected immediate reward r_{t+1} for action $a_t = a_j$ can be calculated as follows:

$$\mathbb{E}[r_{t+1}|a_t = a_j, \mathcal{D}_t] = \int_{r_{t+1}} \int_{\theta_j} r_{t+1} \mathcal{R}(r_{t+1}|a_j, \theta_j)\, p(\theta_j|\mathbf{d}_{j,t})\, \mathrm{d}\theta_j\, \mathrm{d}r_{t+1} \tag{6.1}$$

where we have assumed continuous rewards and parameters. We have to integrate over both the uncertain parameters and the reward to calculate the expectation, as the reward function is itself stochastic. By taking actions, observing the rewards received, and updating $\mathcal{D}$, the decision maker can *learn* about the parameters of $\mathcal{R}$, but this must be done at the same time as the return is being maximised.

6.1.2 Common Reward Distributions

The *standard* MAB, as it shall be referred to from now on, is a MAB, as described above, in which the reward distributions remain independent across actions and stationary; the decision maker can only learn about the underlying reward distribution of the action taken, and the rewards drawn when taking a single action repeatedly are *i.i.d.* Although it is not a requirement of either the stationarity or independence properties, it is normal for the reward distributions $\mathcal{R}$ to be of the same parametric family across the entire action space. Different versions of the standard bandit are therefore defined solely by the form of $\mathcal{R}$.

In the literature, the two most common reward distributions used are Bernoulli and Gaussian, outputting discrete and continuous rewards respectively. These two forms offer different challenges, although the same principles should apply to the solution of a standard MAB irrespective of the reward distribution used.

Bernoulli Reward

The Bernoulli reward distribution is a discrete probability distribution which takes the value of 1 with probability $\lambda \in [0, 1]$ and the value of 0 with probability $1 - \lambda$:

$$P(r = 1) = 1 - P(r = 0) = \lambda \tag{6.2}$$

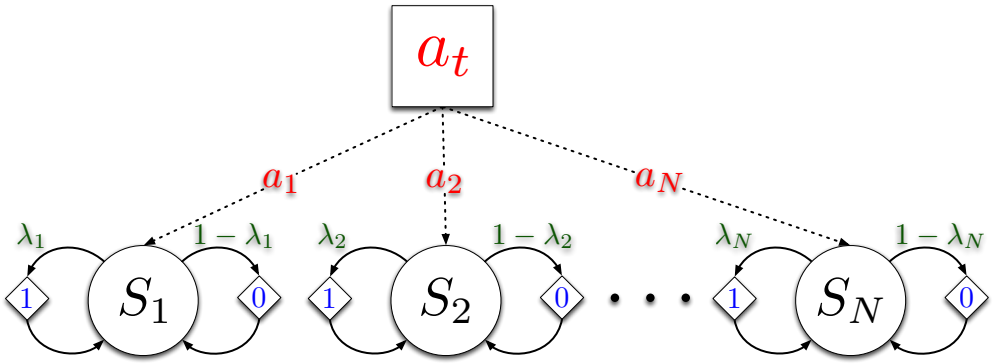


Figure 6.6: A multi-armed bandit with Bernoulli rewards. Each action has an associated success probability $\lambda \in \{\lambda_1, \lambda_2, \ldots, \lambda_K\}$. When an action a_j is taken, the activated arm will either return a unit reward with probability λ_j or a null reward with probability $1 - \lambda_j$ before transitioning back to its resting state.

The probability mass function over the reward is:

$$\begin{aligned} \mathcal{R}(r|\lambda) &= \text{Bernoulli}(r;\lambda) \\ &= \lambda^r(1-\lambda)^{1-r} \qquad \text{for } r \in \{0,1\}. \end{aligned} \tag{6.3}$$

One can think of a single draw from the Bernoulli distribution as being equivalent to the tossing of a weighted coin, in which the probability of the coin landing on heads is equal to λ. By analogy, a Bernoulli reward process would be equivalent to tossing the coin repeatedly, each time receiving a unit reward if the coin landed on heads and nothing if it landed on tails.

In a K-armed Bernoulli bandit, each of the K actions has a different associated $\lambda \in \{\lambda_1, \lambda_2, \ldots, \lambda_K\}$. Figure 6.6 shows the state space of a K-armed Bernoulli bandit, viewed as the conjunction of K one-armed Bernoulli bandits (as in Figure 6.5).

An action that yields a unit reward will often be referred to as a *success*, and consequently λ as the *success probability*. Conversely, an action that yields no reward will be referred to as a *failure*. Once either a success or failure has occurred it must be recorded in the *history* of the bandit, $\mathcal{D}$. The history will contain a record of all the past rewards received during previous action taking: $\mathbf{d}_j = \{w_j, f_j\}$, where

w_j and f_j represent the past successes and failures when taking action a_j, and $\mathcal{D} = \{w_1, f_1, w_2, f_2, \ldots, w_K, f_K\}$. If an action has yet to be taken, then the corresponding w and f will be equal to zero.

Consider a particular action that has been sampled n times with w successes and f failures, where $w + f = n$. The likelihood function over the number of successes w is binomial:

$$\begin{aligned} p(w|\lambda, n) &= \text{Binom}(w; \lambda, n) \\ &= \binom{n}{w} \lambda^w (1-\lambda)^{n-w} \end{aligned} \tag{6.4}$$

for $w \in \{0, 1, \ldots, n\}$, where

$$\binom{n}{w} = \frac{n!}{w!(n-w)!}$$

is the binomial coefficient. The Binomial distribution therefore reduces to the Bernoulli distribution (6.3) when $n = 1$. Given w and n (or w and f), it is possible to infer a distribution over the parameter λ by application of Bayes' theorem:

$$p(\lambda|w, n) = \frac{p(w|\lambda, n)p(\lambda)}{\int_0^1 p(w|\lambda, n)p(\lambda)\mathrm{d}\lambda} \tag{6.5}$$

The conjugate prior to the binomial distribution is the Beta distribution, which is a continuous probability distribution defined on the interval $[0, 1]$ specified by two hyperparameters α_0 and β_0:

$$\begin{aligned} p(\lambda) &= \text{Beta}(\lambda; \alpha_0, \beta_0) \\ &= \frac{1}{B(\alpha_0, \beta_0)} \lambda^{\alpha_0 - 1}(1-\lambda)^{\beta_0 - 1} \end{aligned} \tag{6.6}$$

where B is the beta function:

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha + \beta)} \tag{6.7}$$

and Γ is the gamma function:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} \mathrm{d}t \tag{6.8}$$

In order to find the posterior distribution over λ, a prior distribution must first be defined. A uniform prior over λ is a sensible choice in the absence of further information. This can be achieved by setting the hyperparameters of the prior, α_0 and β_0, to 1, and verified by substituting $\alpha_0 = \beta_0 = 1$ into Equation (6.6).

An expression for the posterior distribution over λ can be generated by inserting Equations (6.4) and (6.6) into (6.5):

$$\begin{aligned} p(\lambda|w, n) &= \frac{\binom{n}{w}\lambda^{w+\alpha_0-1}(1-\lambda)^{n-w+\beta_0-1}/B(\alpha_0, \beta_0)}{\int_0^1 (\binom{n}{w}\lambda^{w+\alpha_0-1}(1-\lambda)^{n-w+\beta_0-1}/B(\alpha_0, \beta_0))\,\mathrm{d}\lambda} \\ &= \frac{\lambda^{w+\alpha_0-1}(1-\lambda)^{n-w+\beta_0-1}}{B(w+\alpha_0, n-w+\beta_0)} \\ &= \mathrm{Beta}(\lambda; w+\alpha_0, n-w+\beta_0) \end{aligned} \tag{6.9}$$

which as expected is equal to another Beta distribution with updated hyperparameters conditioned on the data:

$$\alpha_{\mathcal{D}} = w + \alpha_0 \tag{6.10}$$

$$\beta_{\mathcal{D}} = n - w + \beta_0 \tag{6.11}$$

The expectation and variance of the posterior Beta distribution are given by:

$$\mathbb{E}[\lambda|w,n] = \frac{\alpha_{\mathcal{D}}}{\alpha_{\mathcal{D}} + \beta_{\mathcal{D}}} \tag{6.12}$$

$$\text{var}[\lambda|w,n] = \frac{\alpha_{\mathcal{D}}\beta_{\mathcal{D}}}{(\alpha_{\mathcal{D}} + \beta_{\mathcal{D}})^2(\alpha_{\mathcal{D}} + \beta_{\mathcal{D}} + 1)} \tag{6.13}$$

Using Equations (6.10), (6.11), and (6.13), we can see the posterior hyperparameters $\alpha_{\mathcal{D}}$ and $\beta_{\mathcal{D}}$ will become larger as more samples are taken, which correctly results in less uncertainty in the parameter λ as the variance becomes smaller.

Gaussian Reward

The rewards drawn from a Gaussian MAB are continuous, can take any value on the real line, and are distributed according to a Gaussian distribution with mean μ and variance σ^2:

$$\begin{aligned}\mathcal{R}(r|\mu,\sigma^2) &= \mathcal{N}(r;\mu,\sigma^2) \\ &= \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(r-\mu)^2}{2\sigma^2}\right\} \qquad \text{for } r \in \mathbb{R}.\end{aligned} \tag{6.14}$$

A Gaussian bandit operates largely the same as a Bernoulli bandit, apart from the difference in $\mathcal{R}$. That means that in a K-armed Gaussian bandit, each of the K actions has a pair of parameters $(\mu,\sigma^2) \in \{(\mu_1,\sigma_1^2),(\mu_2,\sigma_2^2),\ldots,(\mu_K,\sigma_K^2)\}$ associated with it. The history of the bandit, $\mathcal{D}$, will once again contain a record of all past rewards. As $\mathcal{R}$ is continuous, rewards must be recorded individually such that $\mathbf{d}_j = \{r_{j1}, r_{j2}, \ldots, r_{jn_j}\}$, where r_{jm} is the mth reward received when taking action a_j, and n_j is the number of times that action has been taken. If an action has yet to be taken, the corresponding $\mathbf{d} \in \mathcal{D}$ will be empty.

Let us again consider a single action that has been sampled n times with resulting set of rewards $\mathbf{d} = \{r_1, r_2, \ldots, r_n\}$, each element of which has been drawn from

$\mathcal{R}(r|\mu, \sigma^2)$. We begin by finding the likelihood function over $\mathbf{d}$, which is a product of Gaussian distributions:

$$\begin{aligned} p(\mathbf{d}|\mu, \sigma^2) &= \prod_{i=1}^{n} \mathcal{R}(r_i|\mu, \sigma^2) \\ &= (2\pi\sigma^2)^{-n/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(r_i - \mu)^2\right\} \end{aligned} \tag{6.15}$$

Let us also define the observation mean and variance:

$$\bar{r} = \frac{1}{n}\sum_{i=1}^{n} r_i \tag{6.16}$$

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(r_i - \bar{r})^2 \tag{6.17}$$

In some forms of the MAB the decision maker will know either μ or σ^2 in advance; however for generality here we will assume that both these parameters are unknown. As noted in Bishop (2006), given both parameters are unknown it is more convenient to work with the precision $\lambda \equiv 1/\sigma^2$ rather than the variance. Incorporating this change of variable, (6.15) becomes:

$$p(\mathbf{d}|\mu, \lambda) = \left(\frac{\lambda}{2\pi}\right)^{n/2} \exp\left\{-\frac{\lambda}{2}\sum_{i=1}^{n}(r_i - \mu)^2\right\} \tag{6.18}$$

which can be written:

$$p(\mathbf{d}|\mu, \lambda) = \left(\frac{\lambda}{2\pi}\right)^{n/2} \exp\left\{-\frac{\lambda}{2}\left[ns^2 + n(\bar{r} - \mu)^2\right]\right\} \tag{6.19}$$

By application of Bayes' theorem we can derive a posterior distribution over both μ and λ:

$$p(\mu, \lambda|\mathbf{d}) = \frac{p(\mathbf{d}|\mu, \lambda)p(\mu, \lambda)}{\iint p(\mathbf{d}|\mu, \lambda)p(\mu, \lambda)\mathrm{d}\mu\,\mathrm{d}\lambda} \tag{6.20}$$

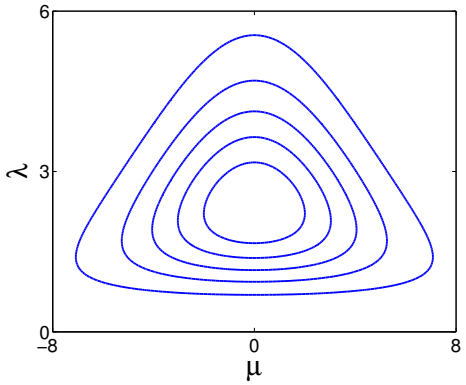


Figure 6.7: Contour plot of the Normal-Gamma distribution (6.21), with hyperparameter values $\mu_0 = 0$, $\kappa_0 = 0.04$, $a_0 = 4$, and $b_0 = 1.5$.

The conjugate prior to the likelihood (6.19) is the *Normal-Gamma* distribution, which is a two dimensional probability distribution specified by four hyperparameters μ_0, κ_0, a_0, and b_0:

$$\begin{aligned} p(\mu, \lambda) &= \mathrm{NG}(\mu, \lambda; \mu_0, \kappa_0, a_0, b_0) \\ &= \mathcal{N}(\mu | \mu_0, (\kappa_0 \lambda)^{-1}) \, \mathrm{Gam}(\lambda | a_0, b_0) \\ &= \frac{b_0^{a_0} \sqrt{\kappa_0}}{\Gamma(a_0)\sqrt{2\pi}} \lambda^{a_0 - \frac{1}{2}} \exp\left(-\frac{\lambda}{2} \left[\kappa_0 (\mu - \mu_0)^2 + 2b_0 \right] \right) \end{aligned} \tag{6.21}$$

Had we elected to continue using the variance instead of the precision, the conjugate prior would have been the *Normal-Inverse-Gamma* distribution.

The choice of hyperparameters for the prior is non-trivial due to the coupling between the precision of μ and the value of λ (Bishop, 2006). In general, and in the absence of further information, the decision maker should aim to set the hyperparameters such that the prior distribution is broad and uninformative over μ and λ. One such setting is shown in Figure 6.7.

Using the result given in Murphy (2007), the solution to (6.20) after inserting (6.21) and (6.19) is a Normal-Gamma distribution representing the posterior over the

parameters μ and λ:

$$p(\mu, \lambda|\mathbf{d}) = \mathrm{NG}(\mu, \lambda; \mu_{\mathcal{D}}, \kappa_{\mathcal{D}}, a_{\mathcal{D}}, b_{\mathcal{D}}) \tag{6.22}$$

with the following posterior hyperparameters:

$$\mu_{\mathcal{D}} = \frac{\kappa_0 \mu_0 + n\bar{r}}{\kappa_0 + n} \tag{6.23}$$

$$\kappa_{\mathcal{D}} = \kappa_0 + n \tag{6.24}$$

$$a_{\mathcal{D}} = a_0 + n/2 \tag{6.25}$$

$$b_{\mathcal{D}} = b_0 + \frac{ns^2}{2} + \frac{\kappa_0 n(\bar{r} - \mu_0)^2}{2(\kappa_0 + n)} \tag{6.26}$$

As the bandit is merely a tool with which to demonstrate how to treat information in an uncertain decision problem, we will not get bogged down in the subtle nuances that each form of bandit provides. Instead, we shall begin by using the Bernoulli bandit as a means of exhibition. As will become obvious, a discrete reward process also allows for simpler graphical demonstration of the principles being used. It should also be obvious that the Bernoulli MAB models precisely the problem faced by Bingle, in which the click through probabilities are equivalent to the success probabilities.

6.1.3 Other Types of Bandit

Whilst we investigate the standard MAB here because it isolates the problem of exploration and exploitation, there a numerous extensions which feature in the literature which the interested reader should consider exploring. For example, Whittle (1988) introduces the concept of *restless bandits* for which states other than the one being sampled change. He motivates this in the context of choosing medical treatments where actions represent alternative treatments for a certain disease. It is quite feasible

that the underlying cause, for example a virus, evolves whilst different treatments are tested. Quite clearly, if this is the case the efficacy (or recovery probability) of each treatment would also change as time goes on.

Another common variation deals with the situation in which actions may share certain similar features. In this case, learning about the value of one action may also provide information about others. This is often utilised as an extension to the advertising problem in which adverts may share similar text, images, themes and so forth. The specific variation is variously known as contextual bandit or bandit with covariates, see for example Yang and Zhu (2002); Pavlidis et al. (2008) and Dudik et al. (2011).

6.2 Introducing Hyperstate

Now that we have defined the MAB and presented a selection of reward functions, we move to the problem of how to make decisions so that we can maximise expected return. In order to make optimal decisions, the decision maker must go through a similar reasoning process regardless of the reward function used; however, as already mentioned, we use the Bernoulli reward bandit to show this process as it is much easier to present how information gain guides decision making in this scenario.

Our discussion starts by understanding what exactly the state variable must represent to enable optimal decision making. If we start with the assumption that state need only include physical state, of which the MAB formally has only one, then for a 2-armed Bernoulli reward bandit the decision tree would look like the section of decision tree presented in Figure 6.8. As there is formally only one physical state in the MAB, the decision tree can be drawn so that each action and reward pair returns to the same state, hence the lack of 'branching' in the diagram. The two labelled nodes, A and B, represent the same physical state; however the number of rewards that have

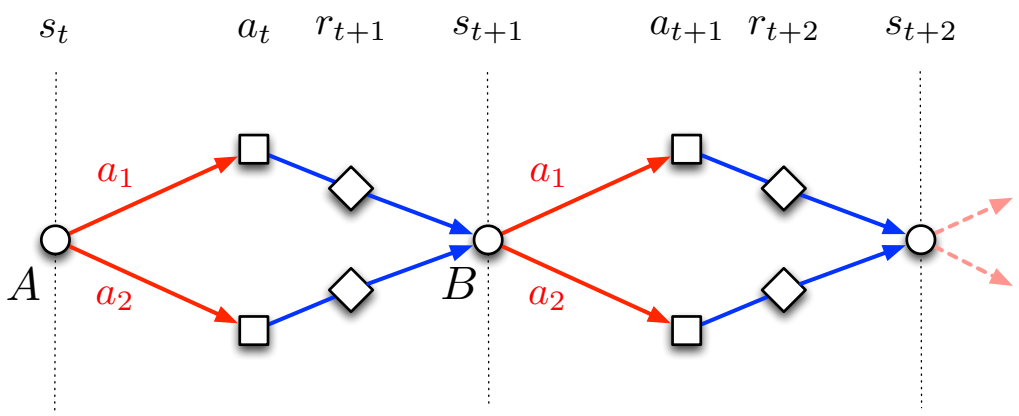


Figure 6.8: A Bernoulli reward bandit decision tree in which only physical state is considered. After each action, a stochastic reward is received and the MDP returns to the original state. The information available at node A will be different from that at the imagined node B, due to the extra reward, r_{t+1}, received; however this is not captured in a state variable containing only physical state.

been observed at each point differs: $\mathcal{D}_{t+1} = \{\mathcal{D}_t, r_{t+1}^{a_t}\}$. Given the extra information available at node B, the expected immediate reward will change for whichever action was taken at time t. This should have the potential to change whatever decision is made; however given that the policy is only a function of the state, $a_t = \pi(s_t)$, a state variable that only captures the physical state component is clearly insufficient in this regard.

We mentioned in Section 3.2.2 that the state variable, alongside representing physical parts of the environment, can also incorporate *states of knowledge*. In order to differentiate between nodes A and B (in a decision sense), it is vital that we do include states of knowledge in the state variable, otherwise in the MAB problem we will always make the same decision, even in the presence of new rewards. To make explicitly obvious what the state variable must incorporate, we take the approach used by Duff (2002) and regard state as an ordered pair or *hyperstate*, $s = \{s_P, s_B\}$, where s_P is the *physical state* encapsulating the properties of the physical world around and including the agent, and s_B is the *information state*, or *belief state*, which represents the agent's epistemic understanding of the world and summarises the past history of

the agent. We prefer the term belief state as this highlights the fact that the variable represents information native to the decision maker. As the MAB formally only has one physical state, which it transitions back to after each action, we can consider the hyperstate variable represented in bandit decision trees to consist entirely of belief state: $s = \{s_B\}$. In discussions on state it is uncommon to explicitly separate physical and information states; however as we have just eluded to, including information into the state variable (and imagining its propagation forward through the decision tree is the vital component needed to optimally solve problems in which the decision maker has uncertainty about elements of an MDP.

Given the inclusion of belief state in the state variable, the decision tree now begins branching as a different belief state arises for each possible reward r_{t+1}. Figure 6.9 gives a section of the decision tree with belief state included. Once again the nodes

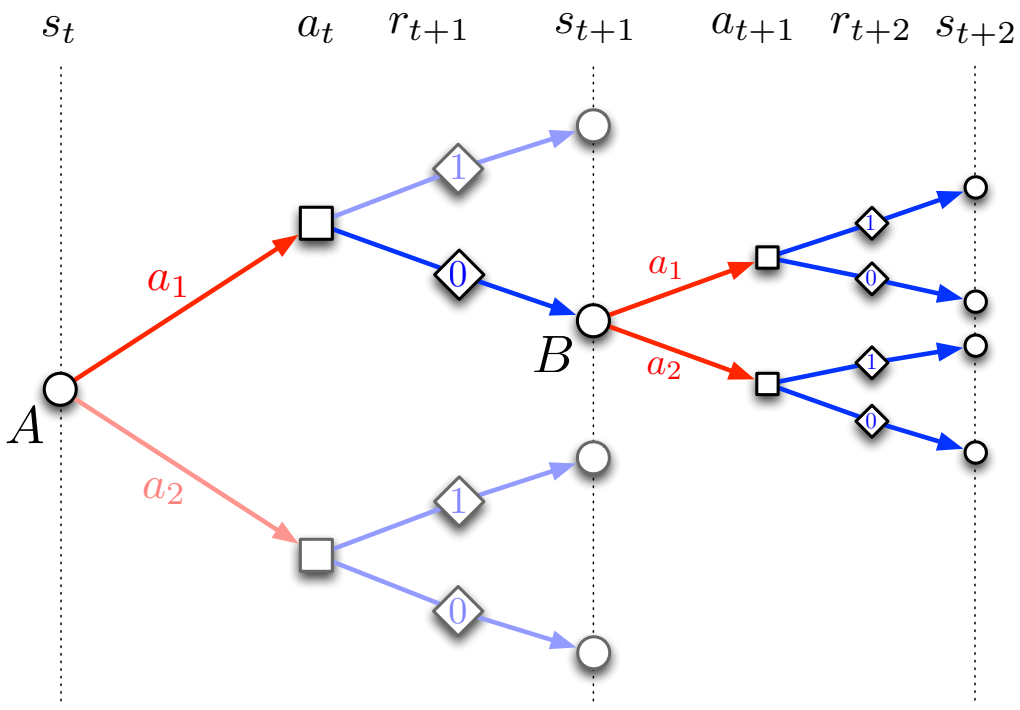


Figure 6.9: A section of a Bernoulli bandit decision tree, in which hyperstate $s = (s_P, s_B)$, where s_P is physical state, and s_B is belief state. The information available at node A will be different from that at node B, due to the 0 reward received. The decision makers belief about λ_1, but not λ_2, will change because of this.

A and B have been labelled, although we see this time that B represents a specific belief state in which an additional reward of 0 has been observed after taking action a_1.

We can look in more detail at how the added observation available at B would affect our *belief* about the environment by evaluating the difference in immediate expected reward at the two points of interest. An expression for immediate expected reward in a Bernoulli reward bandit can be found using Equations (6.1), (6.3) and (6.9), such that the expected reward when taking action a_j is as follows:

$$\begin{aligned}
\mathbb{E}[r|a=a_j,\mathcal{D}_t] &= \int_r\int_{\lambda_j} r\,\mathcal{R}(r|a_j,\lambda_j)\,p(\lambda_j|\mathbf{d}_j)\,\mathrm{d}\lambda_j\,\mathrm{d}r \\
&= \int_0^1\int_0^1 r\ \text{Bernoulli}(r|\lambda_j)\ \text{Beta}(\lambda_j;w_1+\alpha_0,f_1+\beta_0)\,\mathrm{d}\lambda_j\,\mathrm{d}r \\
&= \int_0^1 \lambda_j \frac{\lambda_j^{w_j+\alpha_0-1}(1-\lambda_j)^{f_j+\beta_0-1}}{B(w_j+\alpha_0,f_j+\beta_0)}\,\mathrm{d}\lambda_j \\
&= \frac{w_j+\alpha_0}{w_j+\alpha_0+f_j+\beta_0} \qquad (6.27)
\end{aligned}$$

where $\mathbf{d}_j=\{w_j,f_j\}$ is the history of rewards for action a_j up to that point. In general we will use a flat prior over λ by setting the prior parameters to $\alpha_0=\beta_0=1$, meaning the expected immediate reward becomes:

$$\mathbb{E}[r|a=a_j,\mathcal{D}_t]=\frac{w_j+1}{w_j+f_j+2} \qquad (6.28)$$

If the history of rewards for action a_1 at time t is given by $\mathbf{d}_{1,t}=\{w_1,f_1\}$, the expected immediate rewards at A and B can be calculated directly from Equation

(6.28) as follows:

$$(A) \qquad \mathbb{E}[r_{t+1}|a_t = a_j, D_t] = \frac{w_1 + 1}{w_1 + f_1 + 2} \tag{6.29}$$

$$(B) \qquad \mathbb{E}[r_{t+2}|a_{t+1} = a_j, D^B_{t+1}] = \frac{w_j + 1}{w_j + (f_j + 1) + 2} \tag{6.30}$$

where we have denoted $D^B_{t+1} = \{D_t, r^{a_1}_{t+1} = 0\}$ as the specific information available at B. From this we can see a relative drop in the expected immediate reward at B compared to A, as a result of the 'imagined' change in belief at that point.

It should seem intuitive, at least from the graphic similarity, that the decision tree in Figure 6.9 lends itself to solution via the Dynamic Programming methods we introduced in Chapter 4. However, before jumping in and applying DP, we should be aware exactly what generating a tree in terms of belief state means. As we have just seen when comparing nodes A and B, we are able to explicitly state a different belief about parameters of the environment to the one we currently hold at the start of the decision tree (node A in our example). It may seem a strange notion at first to consider a belief other than that which we hold now; after all that is based on the only 'real' information the decision maker has at his disposal. Instinct makes us think that this and *only this* information should guide decision making, rather than 'imagined' information later in the tree. However, by propagating belief through a decision tree we do not contradict this idea. Instead, we are simply taking the information we have now and putting it in the context of the problem to be solved; a change in belief will *definitely* occur as more information becomes available, so we must acknowledge that as we make a decision now. To do anything other than this would be to say that our evaluation of the parameters of the MDP will never change, which is clearly not the case. Once we are happy with this idea, we can move forward and apply DP to generate a value function across states and actions, just as we did with real-world states.

6.3 Application of Bellman's Equation

In order to apply DP, we once again turn to Bellman's equation to inform us of exactly what the quantities we should be back propagating are. To remind ourselves, Equation (4.15) gave us the Bellman optimality equation:

$$V^*(s_t|\mathcal{D}_t) = \max_{a_t} \mathbb{E}\left[r_{t+1} + \gamma V^*(s_{t+1}) \,|\, s_t, a_t, \mathcal{D}_t\right]$$

This provides us with the recursive relationship needed as the basis for DP. When expanding the expectation operator we now have an additional unknown, defined by our belief distribution over the parameters, which must be integrated out at the same time as the other distributions. For a general distribution over the parameters of $\mathcal{P}$, represented by $p(\theta|D_t)$, the Bellman optimality equation becomes:

$$V^*(s_t|\mathcal{D}_t) = \max_{a_t} \int_\theta p(\theta|D_t) \sum_{s_{t+1}} \sum_{r_{t+1}} \mathcal{P}(s_{t+1}, r_{t+1}|s_t, a_t, \theta) \left[r_{t+1} + \gamma V^*(s_{t+1})\right] \mathrm{d}\theta \tag{6.31}$$

Fortunately, in the MAB setting we do not need to consider an expectation over s_{t+1}, as it is a deterministic function of the previous state and action (s_t, a_t) and the reward received r_{t+1}. Substituting in the parameters of the Bernoulli reward distribution, $\theta = \lambda$, the Bellman optimality equation for the Bernoulli reward bandit becomes:

$$V^*(s_t|\mathcal{D}_t) = \max_{j \in K} \int_{\lambda_j} p(\lambda_j|D_t) \sum_{r_{t+1}} \mathcal{R}(r_{t+1}|a_{j,t}, \lambda_j) \left[r_{t+1} + \gamma V^*(s_{t+1})\right] \mathrm{d}\lambda_j \tag{6.32}$$

where $s_{t+1} = \mathcal{T}(s_t, a_t, r_{t+1})$.

The most important thing to notice from the Bellman equation is that, when calculating the value function for s_t, the belief distribution used in the Bellman equation for that point $p(\lambda|\mathcal{D}_t)$ is conditioned on the data at that point $\mathcal{D}_t$. For some

general state T steps in the future, s_{t+T}, the Bellman equation will be a function of the belief at that future point, given by $p(\lambda|\mathcal{D}_{t+T})$. Conceptually there is no difference in how DP is executed here compared to Chapter 4, other than the inclusion of belief state in the state variable. As before, the action chosen under this future state determines the value to be back propagated to previous states. Figure 6.10 illustrates how the distribution over the parameters evolves into the future in the section of decision tree we presented before.

Now that we have completely defined the Bellman equations for the MAB, we could either directly solve the decision tree if the problem has a horizon, or use an

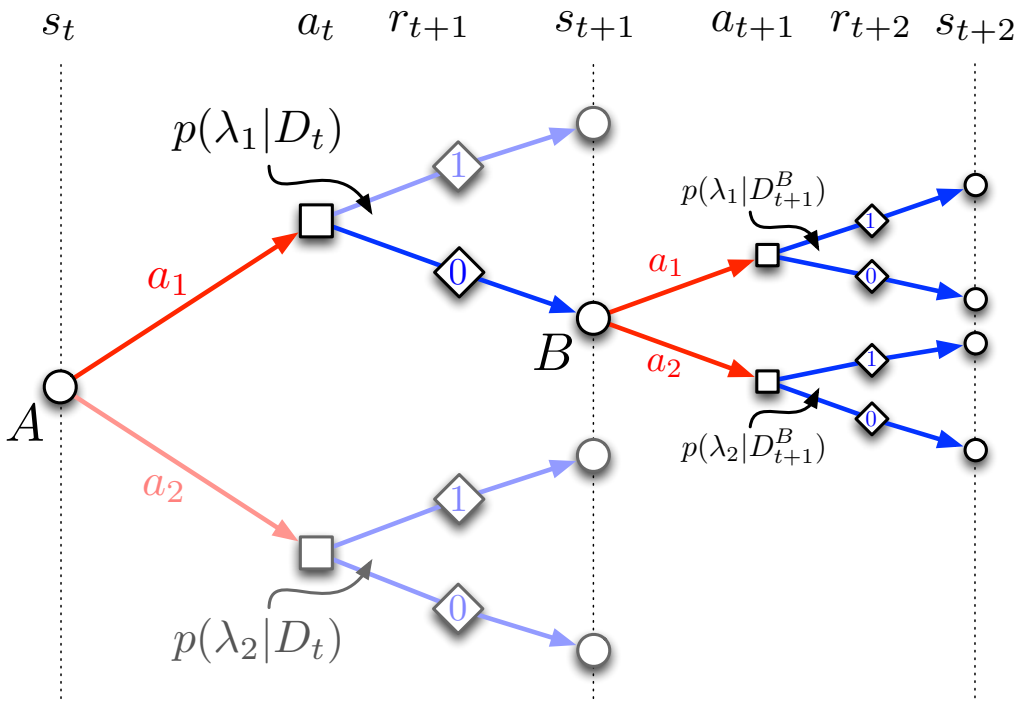


Figure 6.10: A section of a Bernoulli bandit decision tree showing how belief over the parameters of the MDP evolves. The belief at B differs due to the additional reward of 0 observed, which is accounted for in $\mathcal{D}^B_{t+1} = \{\mathcal{D}_t, r^{a_1}_{t+1} = 0\}$. If at A the beliefs over the parameters λ_1 and λ_2 are given by $p(\lambda_1|\mathcal{D}_t) = \text{Beta}(\lambda_1; \alpha_1, \beta_1)$ and $p(\lambda_2|\mathcal{D}_t) = \text{Beta}(\lambda_2; \alpha_2, \beta_2)$, then the updated belief over λ_1 at B is $p(\lambda_1|\mathcal{D}^B_{t+1}) = \text{Beta}(\lambda_1; \alpha_1, \beta_1 + 1)$, whilst the belief over λ_2 remains unchanged as no reward was observed from that reward distribution. The distributions over the parameters further down the tree are vital to solving Bellman's equation and therefore for DP methods to operate.

iterative scheme if it does not or the computational complexity prohibits it, just as we did in Section 4.2.

6.4 Gittins's Indices

The standard MAB, as we have defined it, has the property that taking an action only gives us information about the reward distribution associated with that action; other actions remain unaffected, and we learn nothing about their reward distributions. It was shown by J.C. Gittins and his colleagues in a series of publications, most notably Gittins (1979), that this independence attribute, along with some other minor conditions, allows us to break down the complex intertwined action and state evolution exhibited in the decision tree, into one in which we can deal with each action individually. The paper demonstrates that, as long as actions are independent and do not evolve in any way when not being taken, a particular value can be calculated for each action. Gittins originally termed this the 'Dynamic Allocation Index', but the academic community has since renamed it the 'Gittins Index' (GI). The action that has the highest Gittins Index is the same as that which would be selected using the optimal solution we stated in the last section, and therefore selecting that action maximises the expected discounted return.

The main benefit of the index rule is that it significantly reduces the computational burden on the decision maker, as the K-dimensional bandit problem can be decomposed into K one-dimensional problems (Varaiya et al., 1985) as shown in Figure 6.11. Given the complex relationship between actions and future rewards, captured in decision trees, this ability to decompose the problem is surprising, yet hugely appealing. Proofs of the optimality of the GI have been found using several approaches, and we direct the interested reader to Gittins (1979), Whittle (1980), and Gittins et al. (2011) for further information.

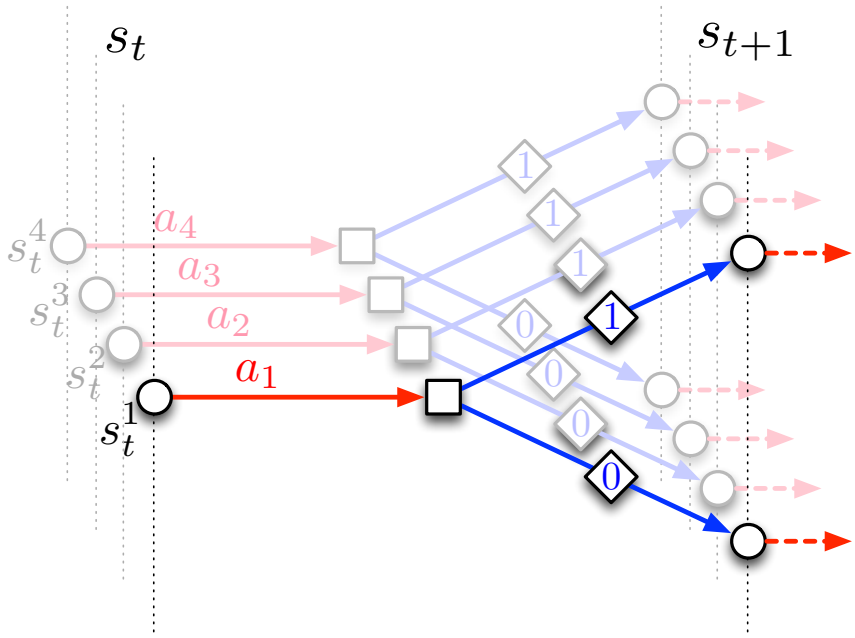


Figure 6.11: The Gittins Index decomposes the K-dimensional bandit problem into K one-dimensional problems. As shown here, a four-armed MAB can be viewed as four one-armed bandits, each of which has its own GI. Here, s_t^j is the state of the subproblem associated with action a_j, which in the case of the Bernoulli reward bandit means $s_t^j = \mathbf{d}_{j,t} = \{\alpha_j, \beta_j\}$.

We emphasise at this point that the independence property exhibited by the MAB is a necessary condition for the Gittins Index (GI) to be used as an *optimal* decision rule. This condition may not be indicative of a lot of real world problems; for example, one can imagine a variety of situations in which learning that a particular action is good implies that similar actions are also good, or the contrary. If actions are not independent the separability result simply does not stand, although it can be used as a heuristic decision method. Other requirements for optimality are that the return is discounted and that the problem has an infinite horizon, although again the GI does have very good heuristic application if these are not the case. Nevertheless there are problems, such as the Bingle example given at the beginning of the chapter, in which we can use this approach to find the optimal solution. Further to that, the GI provides us with an effective way to observe the balance between exploration and

exploitation, which we demonstrate in section 7.1.

6.4.1 Definition of the Gittins Index

The classical definition of the GI for action a_j in state s_t is given by the following equation:

$$G(s_t, a_j) = G(s_t^j) = \sup_{\tau>0} \frac{\mathbb{E}\left[\sum_{m=0}^{\tau-1} \gamma^m r_{t+m+1} \middle| s_t^j\right]}{\mathbb{E}\left[\sum_{m=0}^{\tau-1} \gamma^m \middle| s_t^j\right]} \tag{6.33}$$

where γ is the usual discount parameter. Here, as in Figure 6.11, s_t^j is used to refer to the part of the state variable s_t that is relevant to action a_j; for example, in the case of the Bernoulli reward bandit $s_t^j = \mathbf{d}_{j,t} = \{\alpha_j, \beta_j\}$. We can of course imagine the propagation of this state variable into the future, as shown in Figure 6.12. In fact, we shall see that we must consider the future propagation of s_t^j in order to calculate the GI.

The definition of the GI given in Equation (6.33) bears a strong similarity to the definition of optimal state-value, which can be written as follows:

$$V^*(s_t) = \max_{\pi} \mathbb{E}_\pi \left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \middle| s_t\right] \tag{6.34}$$

The expression on top of both Equation (6.33) and (6.34) is an expected return over either the entire problem in the case of $V^*(s_t)$, or up until a 'stopping time', denoted by τ, in the subproblem defined by $G(s_t, a_j)$. Where in order to calculate $V^*(s_t)$ we maximise the expected return over the policy space, for $G(s_t, a_j)$ we maximise over τ. The GI also has an additional term in the denominator, giving expected discounted time, which means $G(s_t, a_j)$ represents some kind of average reward per trial up to the τth trial (Gittins and Jones, 1979).

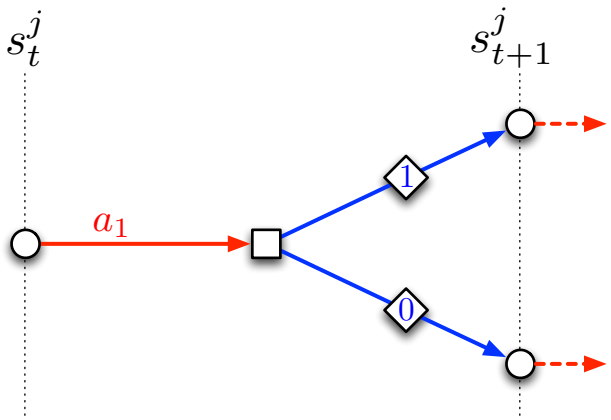


Figure 6.12: Diagram illustrating the propagation of s_t^j into the future.

The classical definition of the GI, given in terms of τ, leads one to describe the index as an *optimal stopping* problem (such as those discussed in Chow et al. (1971)). The general principle behind this begins with the idea that, in calculating the GI for a particular action, we imagine operating the equivalent single armed bandit repeatedly. At some point in the future it would be better to cease operating that single armed bandit than to continue; the further that point is away, the more favourable the arm is compared to others. For example, Weber (1992) presents this optimal stopping point in relation to a 'prevailing charge' which the agent must pay before each action, whilst Whittle (1980) describes it in the context of a retirement reward that the agent receives upon terminating operation. We, however, prefer an interpretation in which each one-armed bandit is compared against another one-armed bandit that outputs a known payoff after each action, underpinning an algorithm used to calculate $G(s_t, a_t)$ given in Gittins (1979), which we discuss in the next section.

As we have said, a GI can be calculated for each action on a MAB solely as a function of the relevant part of the state variable s_t. Figure 6.13 plots the GI for the Bernoulli reward bandit across a range of state values, represented by the variables α and β. Two states, $s_t^j = \mathbf{d}_{j,t}$, are also pictured as an example. Of course, the state variable is purely a function of the reward observations made by the decision maker,

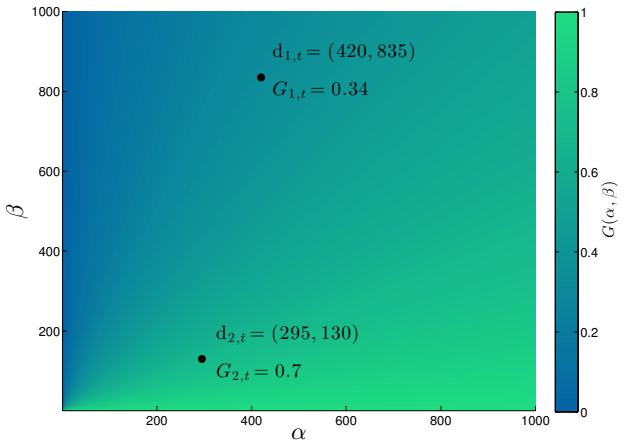


Figure 6.13: Diagram plotting $G(s_t, a_j) = G(\alpha_j, \beta_j)$ for the Bernoulli reward bandit across a range of values of α and β. At any point, a single arm on a MAB sits somewhere on this surface, two examples of which are illustrated in the figure.

so as more observations are made the point on the GI surface related to a particular action will move and the GI may change.

In Figure 6.14 we plot the movement of the GI as state transition histories sampled from three different arms on a Bernoulli reward bandit with different values of λ. As rewards are stochastic, these histories would be different if we were to sample again from each of the arms. On the right hand side of this figure we plot the GI across the histories. We can clearly see from this figure that at points the GI is highest for actions that in reality are not the most rewarding. This reiterates the point that optimality is defined with respect to the decision maker's knowledge, not that of an omniscient observer.

6.4.2 Calculating the Gittins Index

In order to calculate the GI for action a_j on a MAB in state s_t, we can imagine that the decision maker faces a choice at each step between either activating that action's equivalent one-armed bandit and observing the reward, or choosing to play another one-armed bandit which outputs a *known* deterministic payoff of μ. Given that no information is gained by taking the deterministic bandit, if the decision maker decides

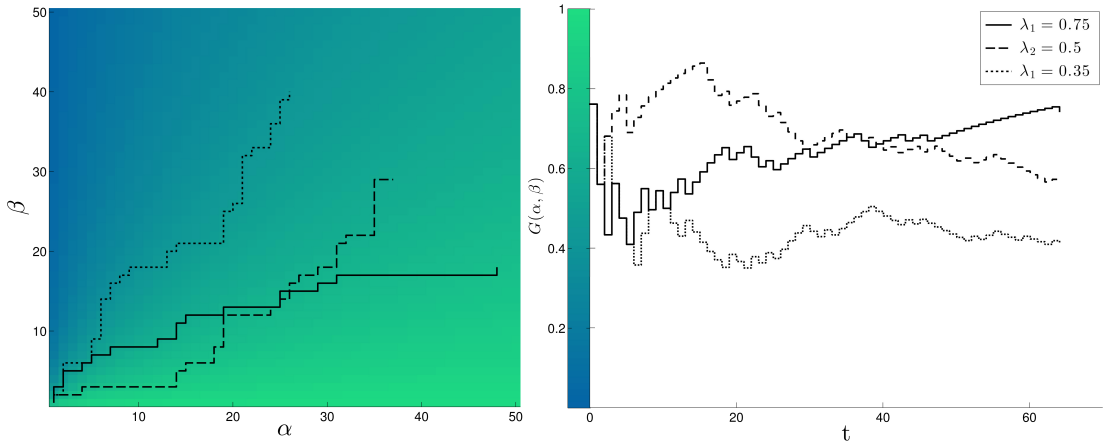


Figure 6.14: Diagram showing sample progressions of state and GI for three arms on a Bernoulli reward bandit. On the left is the state transition history, and on the right is the GI plotted across those histories.

that it will be more rewarding to take the known payoff, he will clearly not return to playing the original bandit, effectively *stopping* its operation. We can therefore see how this interpretation relates to the optimal stopping version of the GI.

The average reward at each time step for the deterministic bandit must equal μ by definition, which given what was said earlier must therefore also equal the GI for that action. The GI for a_j is found by calculating the value of μ for which the decision maker is indifferent between taking either the uncertain or deterministic bandit at the start, which happens when the value of the GI is equal for both. Clearly, the higher μ must be to fulfil that condition, the more favourable a_j will be when compared to other actions on the MAB.

The algorithm introduced in Gittins (1979) approximates the GI by calculating the index for a finite horizon problem. For action a_j with N remaining steps, the approximation is given by the following equation:

$$G^N(\alpha,\beta) = \sup_{\mu} \frac{\mathbb{E}[r|\alpha,\beta] + \sum_{m=1}^{N-1}\gamma^m \sum_{k=0}^{m} Q(k,\alpha,\beta,m,\mu)\mathbb{E}[r|\alpha+k,\beta+m-k]}{1+\sum_{m=1}^{N-1}\gamma^m \sum_{k=0}^{m} Q(k,\alpha,\beta,m,\mu)} \tag{6.35}$$

where

$$Q(k,\alpha,\beta,m,\mu) = P\{(\alpha(m),\beta(m)) = (\alpha+k,\beta+m-k) \cap G^{N-i}(\alpha(i),\beta(i))) \geqslant \mu, \ldots \\ \ldots 1 \leqslant i \leqslant m | (\alpha(0),\beta(0)) = (\alpha,\beta)\}$$

and where the notation has been slightly changed from the original paper. This equation captures the interpretation of a decision maker choosing between a deterministic one-armed bandit with known payoff μ and a Bernoulli reward one-armed bandit starting in state $s_t = (\alpha, \beta)$. The state of the Bernoulli reward bandit m steps into the future is given by $s_{t+m} = (\alpha(m), \beta(m))$, which means that $(\alpha, \beta) = (\alpha(0), \beta(0))$. Although the expression looks rather involved, it actually just expands the classic GI in terms of the parameters of the Bernoulli reward bandit and μ. In fact, the right hand side of Equation (6.35) can be solved with a simple DP scheme, which we now solve, and for which the backup step is shown in Figure 6.15. Notice that the value of each state node is now given by $G^{N-m}(s_{t+m})$ rather than $V^*(s_{t+m})$.

Starting at the last action to be taken, $m = N$, the value of $G^0(s_{t+N})$ is given by the immediate expected reward in state s_{t+N}. Moving backwards, if the value of $G^{N-m}(s_{t+m})$ for a particular node is less that μ, then at that point the decision maker would favour taking the certain payoff indefinitely; as such the value that gets propagated back to the preceding node is 0, as the bandit for action a_j has no value being in that state. Conversely, if $G^{N-m}(s_{t+m})$ is greater than μ, the decision maker will favour continuing to take action a_j and must propagate back $G^{N-m}(s_{t+m})$. It may seem slightly unintuitive to return to a DP process, given that the benefit of the GI is that it decomposes the solution away from the difficult computation of the original problem; however, as pointed out in Whittle (1980), we effectively have a $1\frac{1}{2}$-armed bandit problem rather than two-armed, due to the fact we precisely know the reward we will receive from one of the available actions, vastly simplifying the

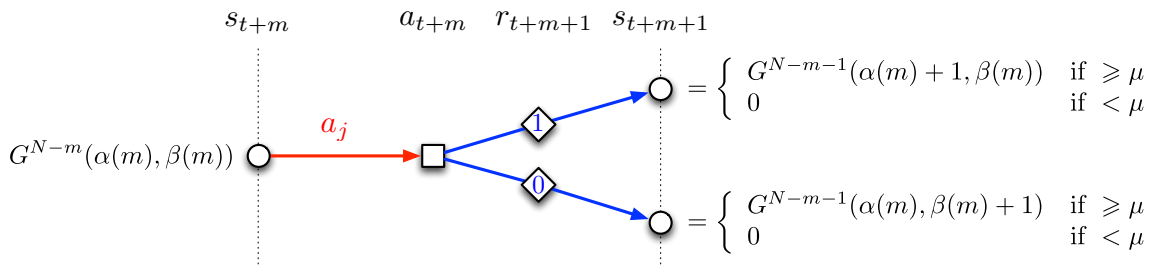


Figure 6.15: A diagram indicating the backup step required to solve Equation (6.35) using DP.

computation.

The DP scheme we just described solves the value function for a particular value of μ, whereas we must find the supremum over μ, as captured in Equation (6.35). This is because the approximate GI for action a_j is found when at the start node, the decision maker is indifferent between taking action a_j or receiving the certain payoff, such that $G^N(\alpha, \beta) = \mu$. To find the supremum, we can use a convergent iterative scheme in which, after each DP sweep, we set μ to the new value of $G^N(\alpha, \beta)$, continuing until the two coincide. Examining Equation (6.35), we can see that the GI must be *at least* equal to the immediate expected reward (when $N = 1$), which can be used as a suitable estimate of μ for the first iteration.

One question remains in this algorithm, and that is what the value of N should be. As we stated earlier, the GI is only defined for infinite horizon problems, which is why our calculation for $N < \infty$ is only an approximation to the true GI, $G(\alpha, \beta)$. As discussed in Gittins (1979), $G^T(\alpha, \beta)$ is increasing in T and tends to $G(\alpha, \beta)$ as T tends to infinity. In practice, we can sequentially increase N and evaluate G^N until it tends to a stable value, the speed of which will depend on the amount of discounting given by γ. Figure 6.16 plots $G^N(\alpha, \beta)$ for several values of γ, α and β. We can see from this that as γ increases the speed of convergence decreases. This makes sense, as later rewards have more of an impact on the value of the GI as discounting increases. Notice also that the magnitude of the GI also increases proportionally with γ. This

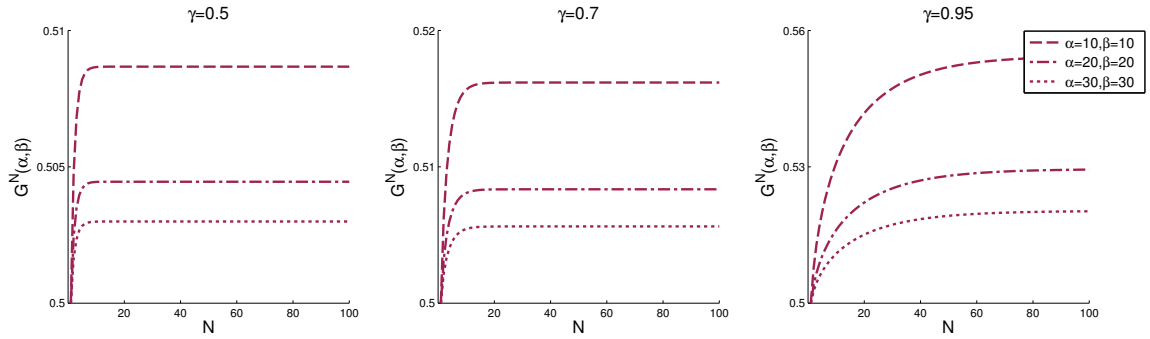


Figure 6.16: Plots of the GI approximation, $G^N(\alpha, \beta)$, for several values of γ, α and β. The speed of convergence is proportional to the value of γ.

behaviour is due to the fact that a greater shift in immediate expected reward requires a greater number of actions to be taken, and a higher value of γ effectively allows us to consider more actions into the future.

6.5 Bayesian Optimisation

This thesis is primarily concerned with the problem of sequential decision making, i.e., how to choose individual actions on the assumption that more can and will be taken in the future. To be more specific, however, we are particularly interested in problems in which our reward function is cumulative, meaning the reward we receive after *each* action contributes to the total reward received for a task. Let us compare this to the problem of *optimisation* (Gill et al., 1981), in which the ultimate task is to return a point in some domain $x_m \in \mathcal{X}$ for which a related function value $y(x_m)$ is a minimum. The reward function in that case is not cumulative; rather a single reward (or loss) is received at the end of the task which is a function of the suggested minimum and the true minimum, for example $\mathcal{L}(x_m) = (y(x_m) - y(x^*))$ or $\mathcal{L}(x_m) = y(x_m)$.

To be clear, the two problems are actually closely related, if not equivalent. When placing a series of observations in order to find the minimum of a function, we note that our actions actually form a sequence of decisions. In order to use information effectively we must consider our ability to place observations in the future,

and that those future decisions can be reevaluated after each observation epoch. The fundamental difference between the two problems is therefore only in the nature of the loss function; in the optimisation case we are only penalised by the final value returned, not the value of each observation.

A great deal of research has been devoted to the subject of optimisation, see for example Nocedal and Wright (2006). The idea of using a *response surface* to model the function being optimised has also been well-investigated (Jones, 2001), which relates to our work in Chapter 5 where we used a Gaussian process as a statistical model of the value function. Further to that, Osborne (2010) uses the GP framework to perform full Bayesian optimisation, and notes that a multi-step lookahead increases performance as well as significantly increasing the computational requirements. He remarks that this cost can be justified so long as the function being minimised is itself even more computationally expensive to evaluate.

Ultimately, it is the relationship between cumulative loss and computational expense that causes the fields of sequential decision making and optimisation to diverge. Our objective function is indeed costly to evaluate (being cumulative in nature), and therefore requires that we *do* consider many actions into the future. For this reason, most sequential decision making literature is interested in finding approximations to the lookahead, rather than effective representation of the objective function. It would, however, be foolish for us not to acknowledge the close relationship these fields share, and note their possible convergence as computational limitations reduce.

Chapter 7

Exploring Approximate Decision Methods

> "Far better an approximate answer to the right question, which is often vague, than the exact answer to the wrong question, which can always be made precise."
>
> *The future of data analysis*, Tukey (1962)

In the last chapter we saw how to optimally make decisions in order to maximise a return when the decision maker has uncertainty about elements of the model. The introduction of an augmented hyperstate which included a belief state effectively solved the exploration vs. exploitation tradeoff, leaving us with an expanded decision tree that can theoretically be solved using standard dynamic programming techniques. Although we discussed in depth the theory behind such optimal decision making, we have yet to investigate or observe in any great detail the effect this has on information gain and how it is balanced against the maximisation of immediate rewards.

Understanding and observing optimal exploration is useful in that it reassures us we are treating information in the correct way; however we look here to accomplish something more practical. As we have already discussed, from a computational perspective these methods can be very difficult to enact. We have already seen that

computationally efficient index methods can be applied in certain problems, and that these offer the same optimal solution as a full dynamic programming method. Unfortunately, such methods are only optimal in a small subset of problems, and it is therefore necessary to develop suboptimal decision methods. In order to develop *principled* and effective methodology, we must understand ideal behaviour first.

We begin our analysis of optimal behaviour by examining the general behaviour of the GI on its own, then quickly follow this by developing our understanding alongside a proposed approximate method using theory that has already been presented. As we have already discussed, index methods which optimally solve problems with a finite horizon do not exist. Nevertheless we utilise the GI approximation $G^N(\alpha, \beta)$, used in the calculation of the true Gittins Index, to show how a suboptimal yet principled decision method can compare favourably to optimal behaviour.

7.1 Features of Optimal Exploration

Whilst we did not decompose actions into exploratory and exploitative subsets in the derivation of optimal decision making, we believe that doing so now may appeal to a human understanding of the solution. We make no judgement on this being the case; rather we carefully define exactly what it is we expect optimal decision making to do in the context of the tradeoff. Simply stating that decision making should combine the two in some way is insufficient, so attempting to be more clear we define the following three features as core *desiderata* of an optimal decision process:

(**i**) Information is valued in the context of the overall goal.

(**ii**) Exploration is balanced by the value of immediate rewards and the horizon of the problem.

(**iii**) Exploitation arises out of reduced uncertainty.

Desideratum (**i**) concerns the fact that information is only useful if it helps in achieving a greater reward; the point being that our objective is *not* to be able to form the best possible predictions about all elements of the model. Elaborating on this, desideratum (**ii**) describes the factors which determine the value of new information in terms of the true objective. There is clearly a relationship between the value of new information and the expectation of immediate rewards; we also note that as the problem draws to a close there is less opportunity to exploit new information, and as such its value should decrease to zero as the horizon (if it exists) is reached. Finally, desideratum (**iii**) states the general 'direction' of behaviour in a decision problem. We should start by gathering information through exploration, which will lead to a reduction in uncertainty. As uncertainty reduces, behaviour should naturally transition to exploitation.

Before going further, let us reiterate that the desiderata should be considered *descriptive* of behaviour in both Bayesian optimal and good approximate methodology. In the first half of this chapter we decompose optimal decision making, specifically in the context of a MAB, to see how the desiderata are naturally encoded. The aim of this is both one of proving consistency with common sense, as well as providing a 'platform of understanding' from which further principled approximations can be made.

In Section 2.4 we noted that one can approach decision problems either from a cogent analysis framework, i.e., Bayesian theory, or by developing heuristic methodology based on an intuitive understanding of desirable behaviour. In the second half of this chapter we compare the performance of principled approximations derived from optimal methodology to that of several heuristic methods, for which the desiderata are *prescriptive*. Through this we will also see that the operations of very few, if any, heuristic methods bear resemblance to optimal methodology.

7.1.1 The Learning Component

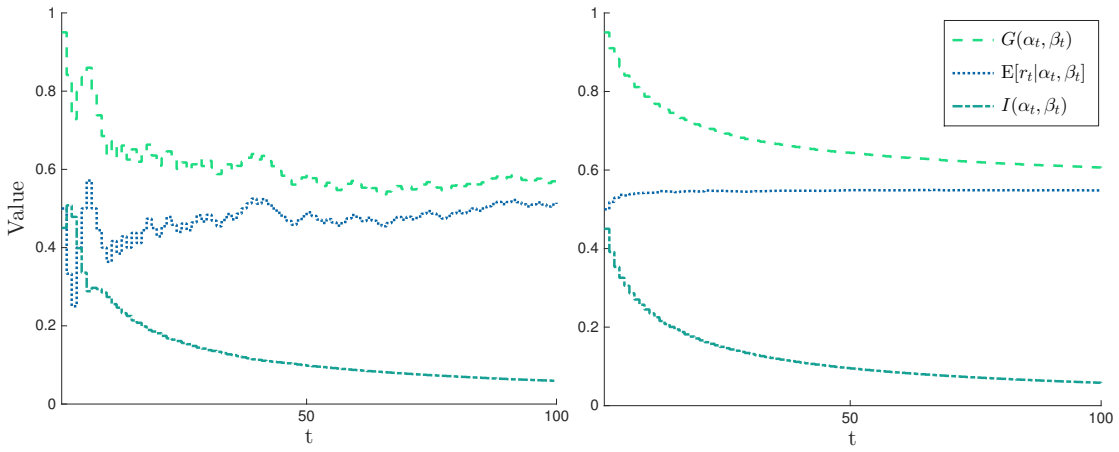
Greater knowledge, in the context of a normal decision problem, can only help a decision maker make more rewarding decisions. Learning about the outcome of an uncertain action should therefore always be associated with a positive value. This positive value should decrease as we gain more information. If we think about the immediate expected reward as the value we attribute to an action without considering future information gain, and the Gittins Index as the value we give an action when we do consider that gain, then the difference between the two is the value we give to information gain. Gittins et al. (1992) termed this value the *learning component* of the indices, and showed that this is a decreasing function of the information currently available about an action[1]. Meuleau and Bourgine (1999) refer to the learning component as an *exploration bonus* and similarly note that it decreases with the number of observations.

In Figure 7.1 we decompose the Gittins Index into immediate expected reward $\mathbb{E}[r_t|\alpha_t, \beta_t]$ and learning component $I(\alpha_t, \beta_t)$ for a single Bernoulli reward process, where we have defined:

$$I(\alpha_t, \beta_t) = G(\alpha_t, \beta_t) - \mathbb{E}[r_t|\alpha_t, \beta_t] \tag{7.1}$$

The figure shows both a single sample trace of the reward process and the mean values taken from 1000 such sample traces, run for 100 time steps. We can clearly see that, as time increases and more rewards are observed, the learning component reduces as uncertainty about the underlying reward distribution shrinks. As this happens, the GI tends towards the immediate expected reward. This should confirm for us that decision making in a problem in which the underlying parameters do not change

[1]The paper proves that the learning component is monotonically decreasing under new information if the expected reward remains unchanged. If the expected reward changes, then the learning component may in fact temporarily increase under this new information.



(a) (b)

Figure 7.1: A diagram plotting the expected reward, Gittins Index and learning component of the Gittins Index as a function of t. (a) is the sample trace from a single Bernoulli reward process with $\lambda = 0.55$, whilst (b) shows the mean values from 1000 runs of the same reward process.

should always move from a period of exploration towards exploitation for it to be optimal.

7.1.2 Long Term Behaviour

In the long term, the learning component of the GI tends to zero as the posterior distribution over λ becomes highly peaked around the true value. As this happens, the value of the GI tends to the immediate expected reward which itself should tend towards the true expected reward. We demonstrate this in Figure 7.2, which shows the GI for three sample Bernoulli reward processes with $\lambda = 0.75$, 0.5 and 0.35. It is an extension of the sample traces shown in Figure 6.14, but with the samples taken further into the future.

The left hand plot of Figure 7.2 shows the sample traces of the three processes plotted on the surface defined by $G(\alpha, \beta)$. A movement away from the origin in this plot indicates a reduction in uncertainty due to an increase in observations. The right hand plot of the figure shows the GI values for each of the three actions as they move across the surface. In both of these plots we see that as the number of observations

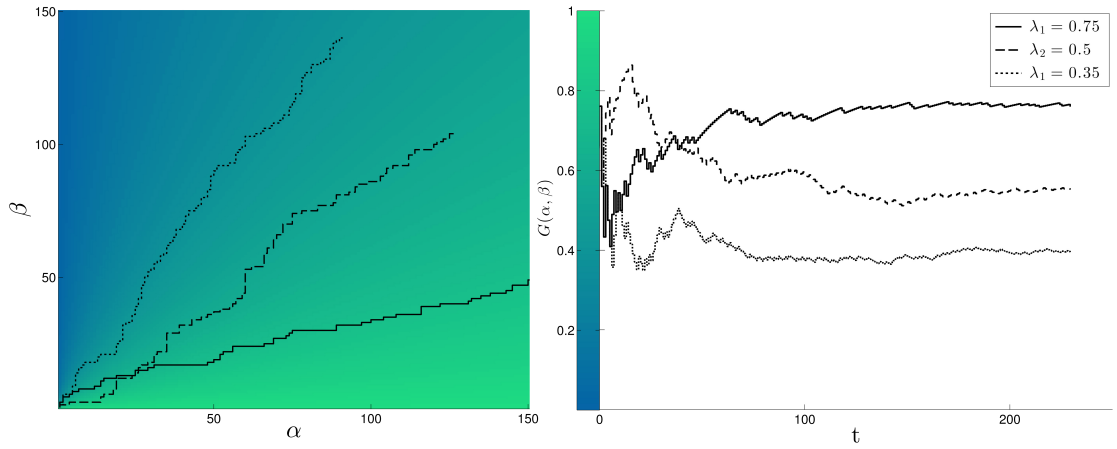


Figure 7.2: Diagram showing the longer term behaviour of the GI for three Bernoulli reward processes. This diagram extends Figure 6.14 by sampling many more times from each process. The GI tends towards the true expected immediate reward in the long run.

increases and uncertainty reduces, the relative change in the GI also reduces.

As we look towards the limit, the GI asymptotes to a steady state value equal to the true expected rewards given by λ. In Figure 7.3 we look even further into the future and plot the traces against contours defined by $G(\alpha, \beta) = \lambda$. We can observe here the sampled traces tending towards these contours which represent the true expected rewards. At this point the reduction in uncertainty has caused the learning component to effectively shrink to zero, which means the reward processes are valued

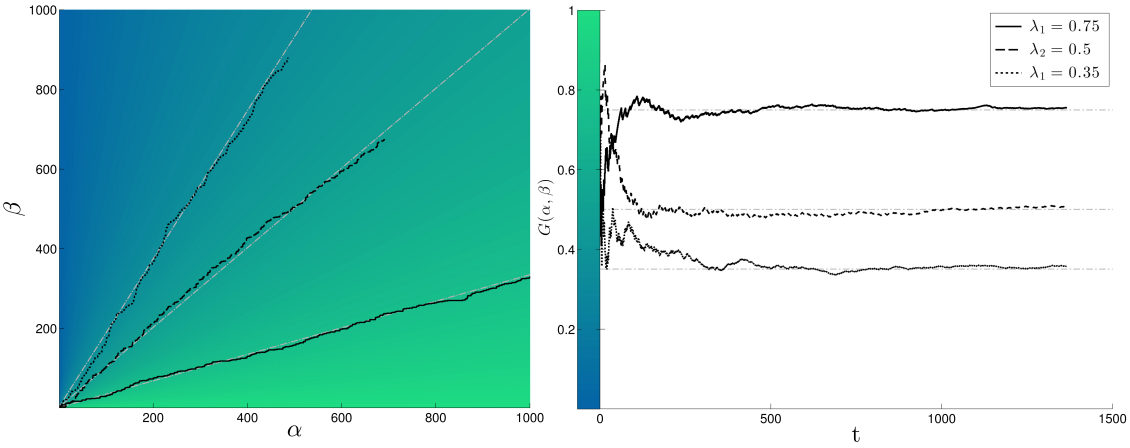


Figure 7.3: Diagram plotting the GI in the long run against contours defined by the true immediate expected reward from three Bernoulli reward processes.

equal to their immediate expected reward. This can be equated to exploitation as a result of low uncertainty, due to the fact that a purely exploitative decision strategy would value actions in the same way.

7.1.3 Observing the Balance Between Exploration and Exploitation

The last two examples dealt with how reward processes are valued independently and as a function of the information available. We now bring these ideas together into a full decision problem, and in Figure 7.4 plot an optimal decision process for a 3-armed Bernoulli reward MAB with infinite horizon. Specifically, we plot the GI for each arm, as well as its decomposition into immediate expected reward and learning component. The last plot in the figure shows the action that is chosen at each time step. Note that this is the first time we have plotted the GI in a decision context, which means that only the value of actions that have been sampled will change.

The optimal policy that generated the actions in the example figure chose the action with the greatest GI, which is plotted in the top plot of Figure 7.4. At various points all actions are sampled, indicated by an update in the value of the index as a new observation is made. Again, the GI for actions that are not sampled remains the same as no further information is gained about that reward process. The immediate expected reward and learning component plots also change as a function of the action taken.

We cannot extract the policy from either the immediate expected reward or the learning component on their own. It is the combination of these two elements, characterised as the GI, which gives us the optimal policy. If we look at the figure, we see that the chosen action does not necessarily have the highest immediate expected reward or the greatest potential information gain at a particular point in time. As time progresses certain actions will be sampled more than others, causing a reduction

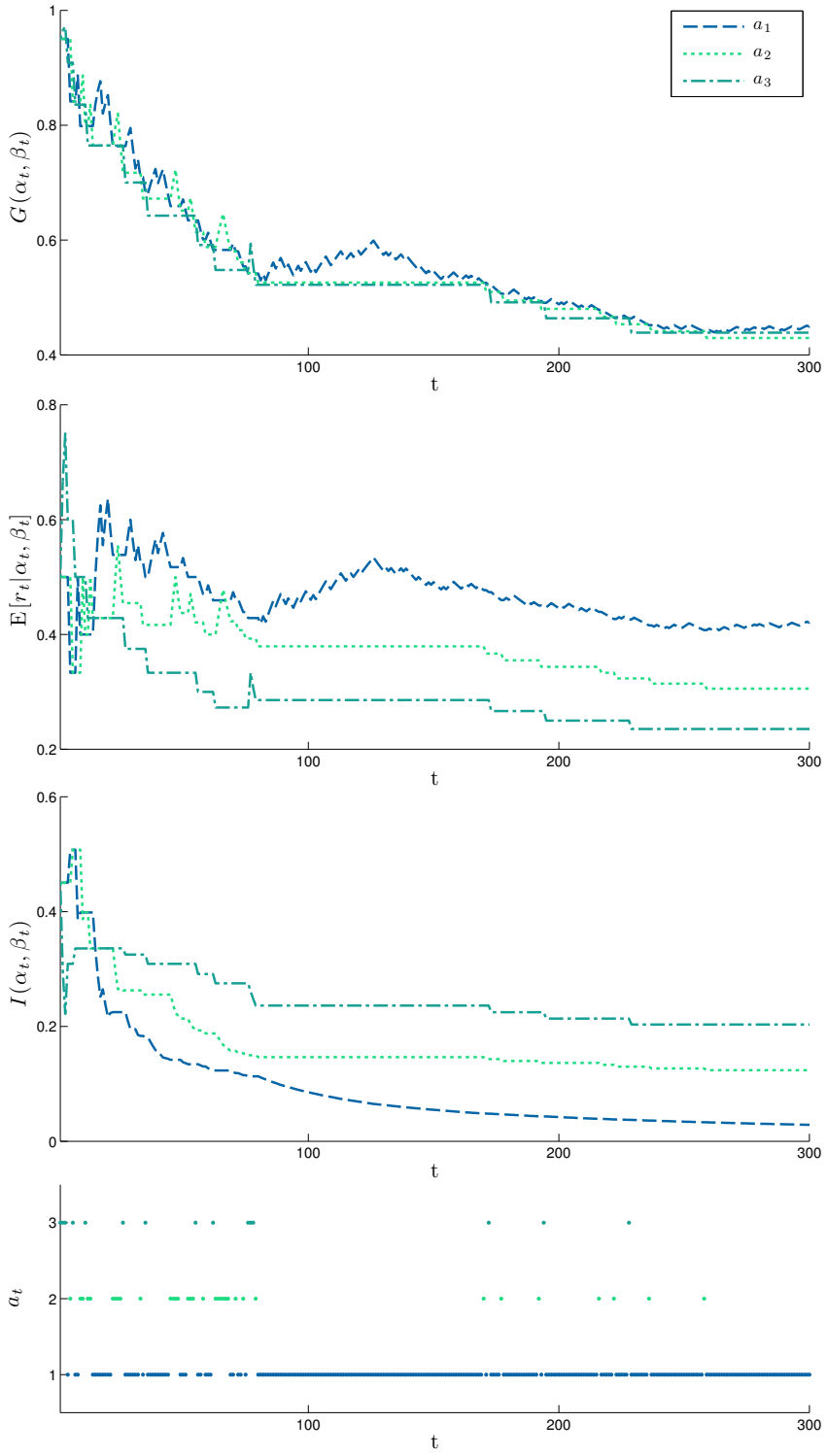


Figure 7.4: Diagram showing a sample optimal decision process for a 3-armed MAB. The plots show the GI, the immediate expected reward, and the learning component for each arm on the MAB as a function of t. The last plot shows the action a_t chosen at time t, which is the action with the highest GI. Actions that are not chosen keep the same GI as no further information is gained about that action.

in the learning component. As this happens, in order to keep choosing those actions their immediate expected reward must be high enough to compensate. The easiest way to see this is to follow a_1 across the plots, and the respective learning component approaching 0. The action continues to be sampled because the immediate expected reward is great enough to outweigh the potential information gain associated with taking other actions.

Whilst we should not get tied down by the nuances of the specific example shown in Figure 7.4, it is interesting to notice what happens when the GI for a_1 drops below that of the other actions. In this case it seems a single observation is enough to reduce the immediate expected reward and through that the GI of those actions, so that a_1 continues to be sampled. In a sense, this shows us the power of optimal decision making, in that it is capable of significantly altering the evaluation of actions based on relatively little information (in this case gathered from just single actions). The optimal use of information allows us to avoid blindly allocating large numbers of actions purely for the sake of information gathering.

Please note that the example presented here, along with several others in this section, is designed to give a flavour of how actions are valued in a real decision problem. It is however only a single example of a stochastic process, and as such caution should be heeded when generalising from this. Later on we present the average performance across a large number of these type of decision problems in order to give a more thorough analysis of decision behaviour.

7.1.4 A Note on Discounting

Discounting is often overlooked in discussions on decisions making, treated as a parameter of necessity only there to provide the means for an optimal solution to be derived. It can, however, make a significant impact on the focus of a decision task as the importance of distant rewards changes. As future rewards become less valuable,

which happens with increased discounting, the value of information decreases and decision making becomes increasingly myopic. We will see later that this can result in higher early rewards by virtue of taking fewer exploratory actions, however this inevitably comes at the price of reduced long term performance.

In Figure 7.5 we plot the GI alongside its decomposition into immediate expected reward and learning component for a Bernoulli reward process, for a range of discount parameters. The plots are the mean values of 1000 runs of the same reward process, equivalent to those shown in Figure 7.1(b). Remember that a lower $\gamma \in [0, 1]$ parameter equates to a higher level of discounting as the reward at time t is scaled by γ^t. We can see from this figure that, as discounting increases, the learning component both decreases in value and converges to 0 faster. Bearing in mind that discounting does not alter the immediate expected reward, an increase in discounting affects decision making in two ways. Firstly, due to the decrease in information value, immediate expected rewards form a greater proportion of the GI. Secondly, as the GI converges to the immediate expected reward faster, decision making transitions to exploitation at an increased rate.

In Figure 7.6 we demonstrate the effect of discounting in a real decision process by returning to a 3-armed MAB example equivalent to the one we used to generate Figure 7.4. In that example we used a very low level of discounting by setting $\gamma = 0.999$. Here we use a value of $\gamma = 0.95$, which is a fairly common setting. The most noticeable comparison we can make between the two examples is the speed at which the policy converges to a single action. In our new example, the higher level of discounting causes the policy to settle much earlier, after taking far fewer exploratory actions. Note once again that this is a single example of a decision process. We will examine average performance on a large test bed later in the chapter.

Whilst on the topic of discounting we feel it is appropriate to address a specific concern that has been levelled at the use of the GI as well as other index policies (for

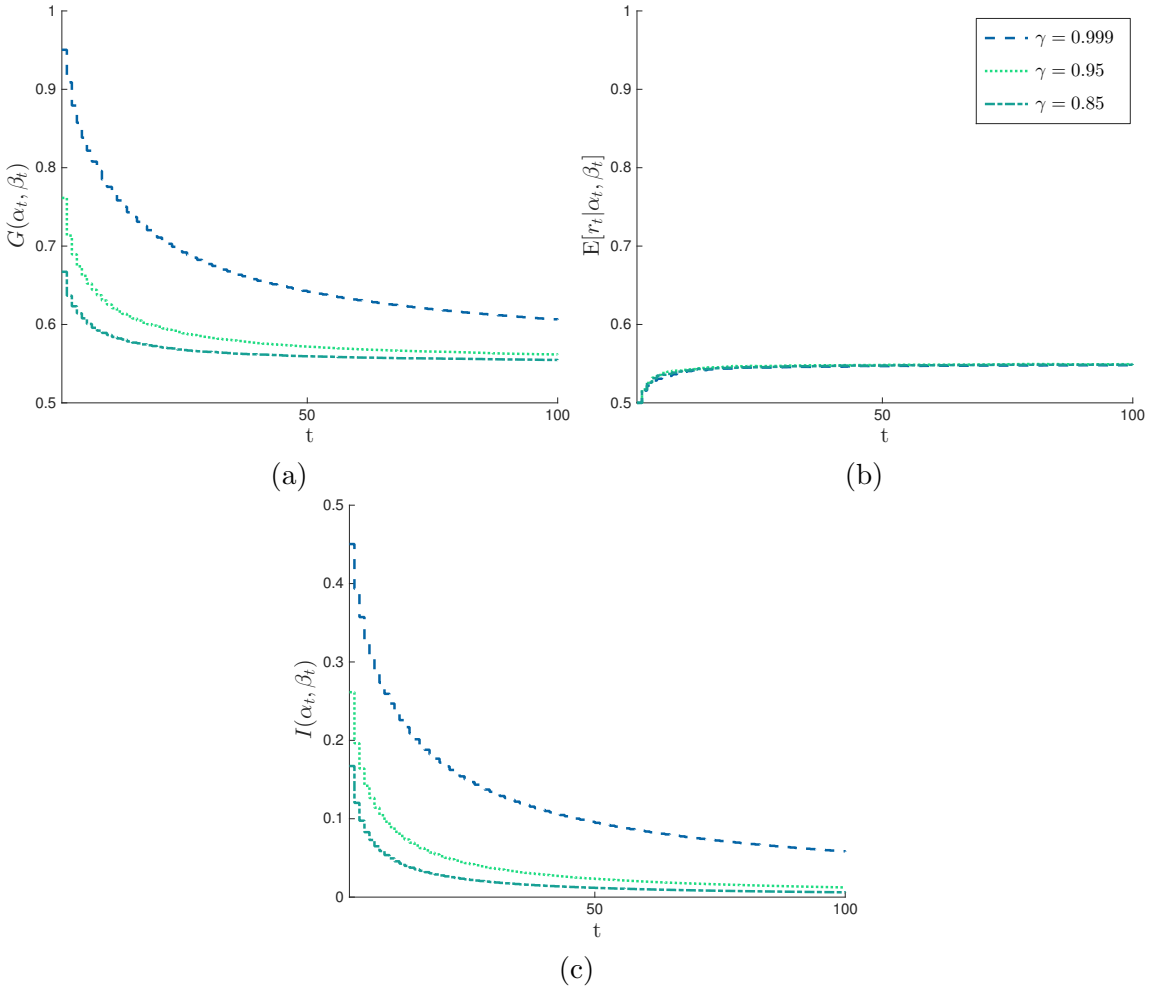


Figure 7.5: A diagram illustrating the effect of discounting on the Gittins Index on a single Bernoulli reward process, equivalent to that given in Figure 7.1. (a) shows the GI, which tends to the immediate expected reward given in (b) at a rate proportional to the level of discounting. (c) shows the learning component which converges to 0, also at a rate proportional to the level of discounting.

example in Scott (2010)). The issue is known as *incomplete learning*, and observes that the GI is an inconsistent estimator. Brezzi and Lai (2000) show that the policy of choosing the action with the highest GI will eventually result in choosing one arm to continue forever when there is a positive probability that the chosen arm does not possess the highest true expected reward. If we consider our inference over the parameters in a MAB, we will see that no matter how many observations we make of an arm, there will always be support for the reward parameter in the range $0 < \lambda < 1$.

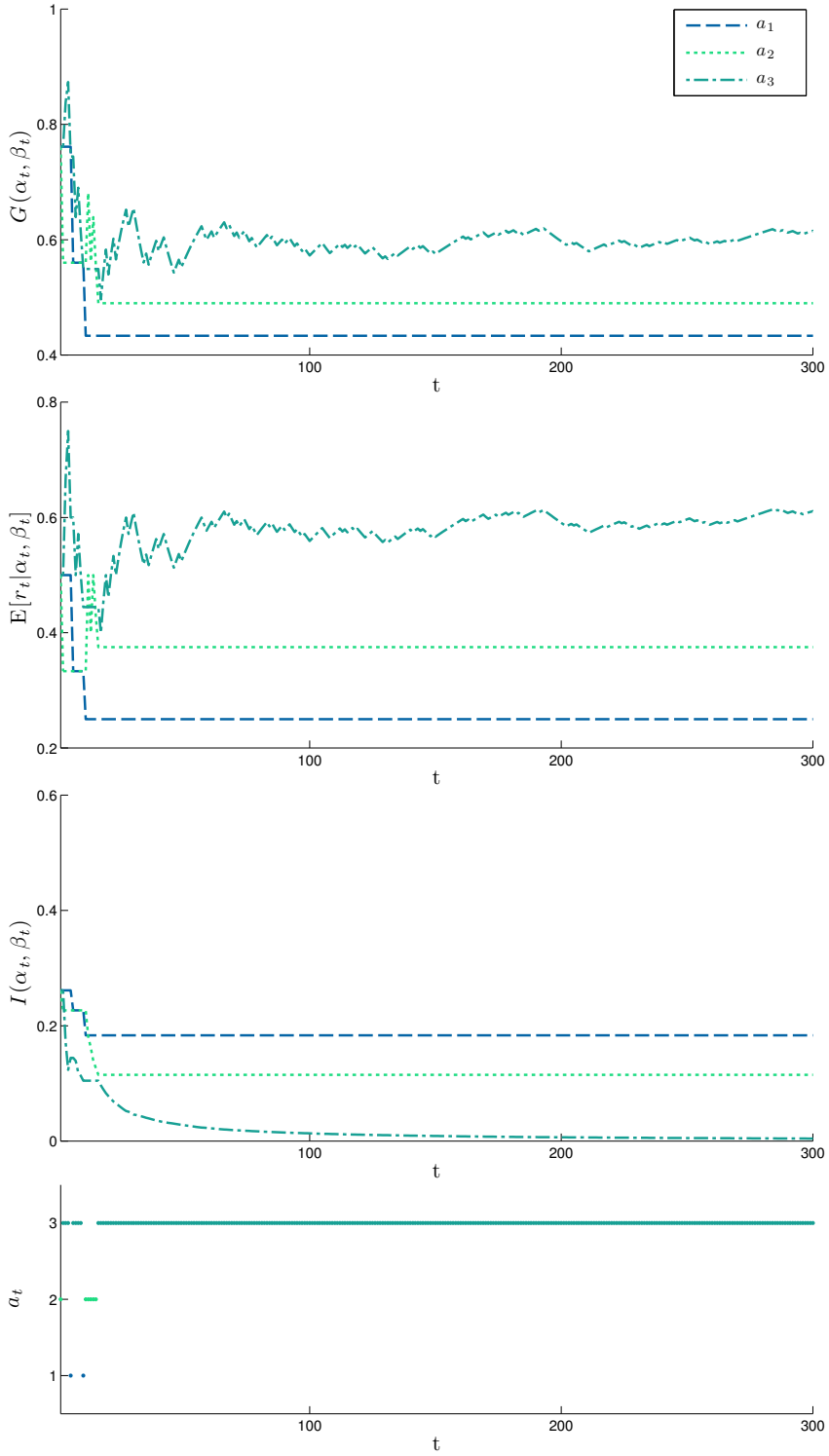


Figure 7.6: Diagram showing a sample optimal decision process for a 3-armed MAB, similar to that given in Figure 7.4 but with a higher level of discounting ($\gamma = 0.95$). The policy converges to a single exploitative action much earlier than the previous example.

Clearly if this is the case, there will always be positive probability that any arm will have the highest true expected reward. As such, any policy that converges to a single action will suffer the same consequence. We should therefore question why this happens in the GI policy, and how this relates to optimality.

As we have already shown, discounting has an effect on the amount of exploration and the speed at which the optimal policy converges to a single action. It turns out that discounting is the reason the GI policy converges at all, even though there is a positive probability that other actions are more rewarding. In effect, discounting causes the infinite horizon problem to appear as a finite horizon problem as it places a limit on the expected return. To remind ourselves, the expected return is given as follows:

$$\mathbb{E}\left[R_t\right] = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1}\right]$$

If the value of r_t can only take the value of 0 or 1, then the expected return is bounded as the sum of a geometric series:

$$\mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i+1}\right] \leqslant \frac{1}{1-\gamma} \qquad \text{if } r_t \in [0,1] \tag{7.2}$$

In a non-discounted finite horizon problem, a similar limit on the expected return would just be the horizon, so in some senses we can think of the above limit as the 'effective horizon' of a discounted infinite horizon problem. Given the *apparent* curtailing of the problem length, it is not surprising that the GI policy converges to a single action as the value of information is proportional to the number of actions available. We should therefore question the validity of discounting in the first place, rather than the GI approach which *is* the optimal solution if discounting defines the problem.

The question of whether discounting is a good approach to take in problems of infinite length should address two points. In the first instance we should ask whether

we only use this approach in order to stem an otherwise infinite expected return equation, which we should follow by asking what we mean by infinite horizon in the first place. With regards to the first point, certainly it seems the case that discounting is principally used to bound the expected return. Let us qualify this by noticing that in these types of problems the reward received after a decision is not itself discounted, as the discount *only* manifests in the expectation. Therefore by using discounting we are explicitly approximating the problem as a shorter horizon problem, as we detailed in the last paragraph. As to the second point, the problem of infinite horizon in many ways should remind us of the St. Petersburg Paradox discussed in Section 2.3.1, in which there is an infinite expected payoff from a potentially infinitely long lottery. A perfectly reasonable resolution to that paradox is to note that the concept of infinities in such a context is patently nonsensical, and any finite length to the problem immediately collapses the expected payoff to a reasonable value. Similarly in the decision problem setting, once we move to a finite horizon problem the original issue of convergence to a single action no longer stands. Perhaps therefore we should not be dealing in the realms of infinite horizons at all.

That said, although an infinitely long decision problem cannot be grounded in reality there are several reasons why this approach might be taken. If the number of actions in a decision problem is unknown and potentially distant into the future, or the horizon has no apparent upper bound, then an infinite horizon model may be a good approximation. In some settings with a very large number of time periods, assuming an infinite number of periods may also simplify the solution. As we will see later, optimal solutions to the infinite horizon model can indeed outperform heuristic methods even over short horizon problems. Given this, it appears that we should remain comfortable with the use of the infinite horizon model as a pragmatic simplification. The method we then choose to bound the return, such as discounting, which without recourse to a better alternative seems entirely reasonable, is simply

part of the approximation in the first instance. Ultimately this means it is imprecise to criticise a method and not the approximation itself. It is equally ill-considered to do this without demonstrating the favourable performance of both the approximation and its solution on the motivating problem.

7.2 Optimal Exploration in Detail

Optimal decision making is in some senses defined by the value placed on information gain over direct rewards. In this section we take an even closer look at how this is accomplished, and attempt to observe the moment a decision maker moderates between actions which are immediately rewarding and those from which there is more to learn. To some extent this can be seen by observing the way different actions are taken throughout a decision problem as we did in Figures 7.4 and 7.6, although this abstracts somewhat from the reasoning process. We therefore take a different approach, based on the assumption that there should be a direct relationship between the number of actions left to take and the optimal action, as indicated in desideratum (**ii**). Specifically, we vary the *horizon* of a problem and observe how the relative value of each action, and therefore the policy, changes.

Our analysis takes place around an MAB in which for each example the states are fixed and only $\mathcal{H}$ varied. For the set of examples we choose a variety of state settings, so as to represent the different levels of uncertainty that can be faced. We start by looking at the one-step decision problem, in which optimal actions should be entirely exploitative being as there is no opportunity to make use of new information. We then look at the multi-step problem, in which we should observe information becoming more important as the number of actions available, determined by $\mathcal{H}$, increases.

Whilst part of our focus is on testing optimal decision making, we also look to demonstrate how a principled approximation can reason in a similar way. The

intention behind this is to affirm our conviction that such methods do more than just fulfil the desiderata. The particular approximation we use, introduced next, makes use of the process used to calculate the GI, which is by definition optimal only for the infinite horizon problem.

7.2.1 A First Principled Approximation

The GI approximation $G^N(\alpha, \beta)$ provides an ever improving estimate of $G(\alpha, \beta)$ as $N \to \infty$. However as we just mentioned, the true GI is only optimal in the infinite horizon problem. A useful property of $G^N(\alpha, \beta)$ is that its value increases away from the immediate expected reward as N increases. Clearly the sort of behaviour that we expect in the setting just described, namely an increase in the value of information, is naturally encoded in $G^N(\alpha, \beta)$. This therefore forms our first principled approximation, in which we set N equal to the horizon $\mathcal{H}$. By doing this we do not approximate the finite horizon problem as an infinite horizon problem, which we would do by making decisions with $G(\alpha, \beta)$. Instead, we approximate the K-action finite horizon decision problem as K independent single action stopping problems. Note that by doing this there are no parameters to tune, which as we shall see is in stark contrast to an overwhelming majority of heuristic decision methods.

7.2.2 One-Step Decision Making

We now look at what decisions are made when only one action remains to be taken. We call this one-step decision making as we only need to look one step forward in our evaluation of future states. We know such a decision must be made when we face the last decision in a finite horizon problem, i.e., where $t = \mathcal{H} - 1$, and as such we covered the correct behaviour when looking at the last decision node in the decision tree as illustrated in Figure 4.8. Of course the same situation could also be faced in one-step problems in which $\mathcal{H} = 1$. The optimal policy in this case is to always take

the action that maximises immediate expected reward:

$$\pi^*(s_{\mathcal{H}-1}) = \underset{a_{\mathcal{H}-1}}{\operatorname{argmax}}\, \mathbb{E}[r_{\mathcal{H}}|a_{\mathcal{H}-1}] \tag{7.3}$$

With regards to the Gittins Index approximation, if in Equation (6.35) we set $N = 1$ then:

$$G^1(\alpha, \beta) = \mathbb{E}[r|\alpha, \beta] \tag{7.4}$$

which is the immediate expected reward in state (α, β); therefore a policy that chooses the action with the largest approximate index is equivalent to that found using Equation (7.3). As such, choosing actions using the Gittins Index approximation is optimal in the one-step case.

To put one-step decision making in the language of common heuristic decision making methods, the action which maximises immediate expected reward (regardless of horizon) is often referred to as the *greedy* action. A *greedy policy* is therefore a policy that naively always chooses the greedy action. This terminology is usually used when referring to decision methods which heuristically use one-step decision making in problems with more than one action left to take, where it is actually optimal to maximise the expected return. Occasionally the greedy action may coincide with the optimal action; however this is certainly not guaranteed.

7.2.3 Multi-Step Decision Making

We now move to much more interesting parts of the problem space and observe how decisions change as we progress from the one-step problem. We hope to see that our intuition about what *should* happen with regards to exploration is automatically encoded in optimal decision making. More specifically, we look to confirm our main assumption that optimal actions become more exploratory as the horizon of the

problem increases.

A good starting point for this analysis is to examine the case when we have very good information about all the parameters of the model. This would occur if we had already observed many rewards from all the available actions. In the case of the Bernoulli reward MAB this would mean that the sum of the parameters α and β for each action would be large, and the posterior Beta distributions over the λ parameters would be highly peaked around the true values. In such a situation new information provides relatively little improvement to our inference about the true λ values, meaning the expected immediate rewards would change very little as we look deeper into the decision tree. Given this, the optimal policy will likely select the same action regardless of how many more actions there are left to take, which in this case would be the one with the highest expected immediate reward.

Figure 7.7 shows our analysis of both optimal decision making and the Gittins Index approximation for each action on a three-armed Bernoulli reward MAB in which the posterior λ distributions are highly peaked around the true values. The top plot shows the optimal action value $Q^*_{\mathcal{H}}(\alpha, \beta)$ as a function of $\mathcal{H}$, where the value of each action is subtracted from the mean of all three, given by $\overline{Q^*_{\mathcal{H}}}$. The reason for subtracting the mean from $Q^*_{\mathcal{H}}(\alpha, \beta)$ is because the values tend to be close to one another, and so removing the mean allows us to better see the relative values which is our primary concern. The middle plot similarly shows the GI approximation for the actions as a function of $\mathcal{H}$, although we do not find the need to subtract the mean from these values. The last plot shows the policy for both optimal and approximately optimal decision making, i.e., the action with the largest $Q^*_{\mathcal{H}}(\alpha, \beta)$ or $G^{\mathcal{H}}(\alpha, \beta)$ respectively. We will see similar plots later in this section as we explore different states of the MAB.

In the case of low uncertainty, as can be seen in the figure, the values of both $Q^*_{\mathcal{H}}(\alpha, \beta)$ and $G^{\mathcal{H}}(\alpha, \beta)$ do not change noticeably as $\mathcal{H}$ increases. Although we make

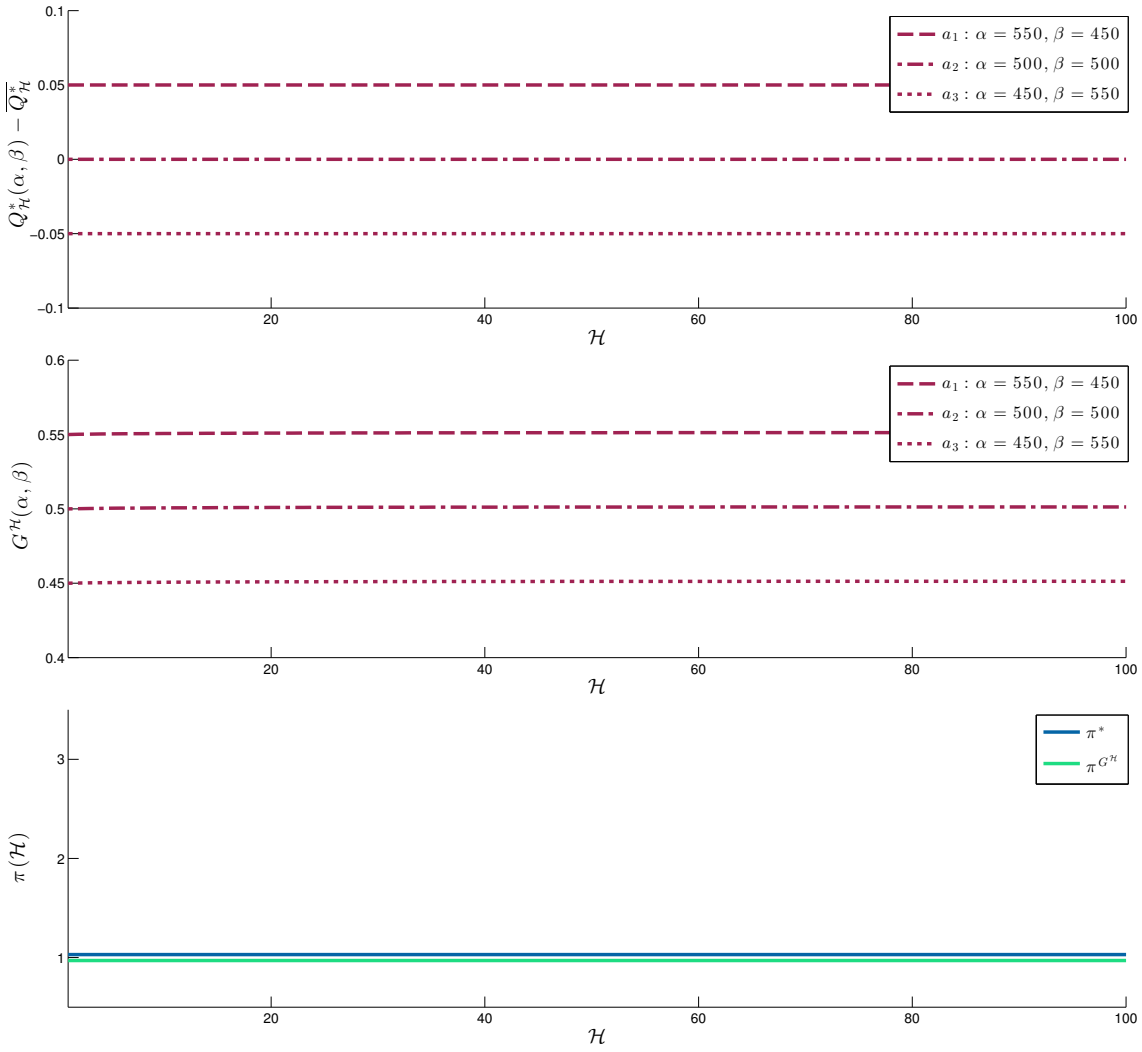


Figure 7.7: A diagram illustrating the optimal action value $Q^*_{\mathcal{H}}(\alpha, \beta)$ and the GI approximation against $\mathcal{H}$ when information about the underlying parameters is very good, along with their respective policies. Both $Q^*_{\mathcal{H}}(\alpha, \beta)$ and $G^{\mathcal{H}}(\alpha, \beta)$ are highly stable as the length of the problem increases, and as such both policies select the action with the highest immediate expected reward (a_1) across all values of $\mathcal{H}$.

no reference to the true λ in this example, we should expect that the value of $G^{\mathcal{H}}(\alpha, \beta)$ will have converged to the immediate expected reward for each action. Given that the value for each action is stable across both these functions, we find that both policies are also stable and favour the action with the highest immediate expected reward, a_1.

Now let us move on to an example in which we have more uncertainty. In Figure

7.8 we present another three-armed Bernoulli MAB, but this time one of the actions, a_3, has been observed a lot less than the others. This action also has the smallest immediate expected reward of all three actions, and so for at least the one-step case will not be selected by either optimal or approximately optimal decision making. Before we analyse the example note that the functions $Q^*_{\mathcal{H}}(\alpha, \beta)$ and $G^{\mathcal{H}}(\alpha, \beta)$ are in reality only defined for integer values of $\mathcal{H}$, as there cannot be fractional time epochs. We interpolate and plot both of these as if they were continuous, however, in order to better demonstrate when a change in policy occurs. Note also that $Q^*_{\mathcal{H}}(\alpha, \beta)$ must be non-decreasing, although this is not apparent due to the subtraction of the mean.

The first thing to notice from the figure is that the policies for both methods change from selecting the action with the highest immediate expected reward to the most uncertain action as the length of the problem increases. Clearly this means that, after a certain point, the potential loss associated with choosing the action with the lower expected immediate reward is outweighed by information gain, as anticipated.

We highlight functional 'crossing points' in the figure to show exactly when there is a change in the ordering of actions according to the two measures. In both the top and middle plot we can see two crossing points as action a_3 initially overtakes a_2, and then later a_1. Let us concern ourselves with the second crossing point for the time being, as this is the point at which the top rated action, and therefore the policy, changes. We will refer to such points as 'upper crossing points', and those which have no effect on the policy as 'lower crossing points'. There is clearly a marginal discrepancy between the position of the optimal upper crossing point and that of the GI approximation. Nevertheless, because both points occur within the same increment in $\mathcal{H}$, the policies agree. We will see an example later when this doesn't quite happen, thus demonstrating why the GI is only 'approximately' optimal.

Now let us draw our attention to the lower crossing point in the example. As previously discussed, the policies are only concerned with which action has the *highest*

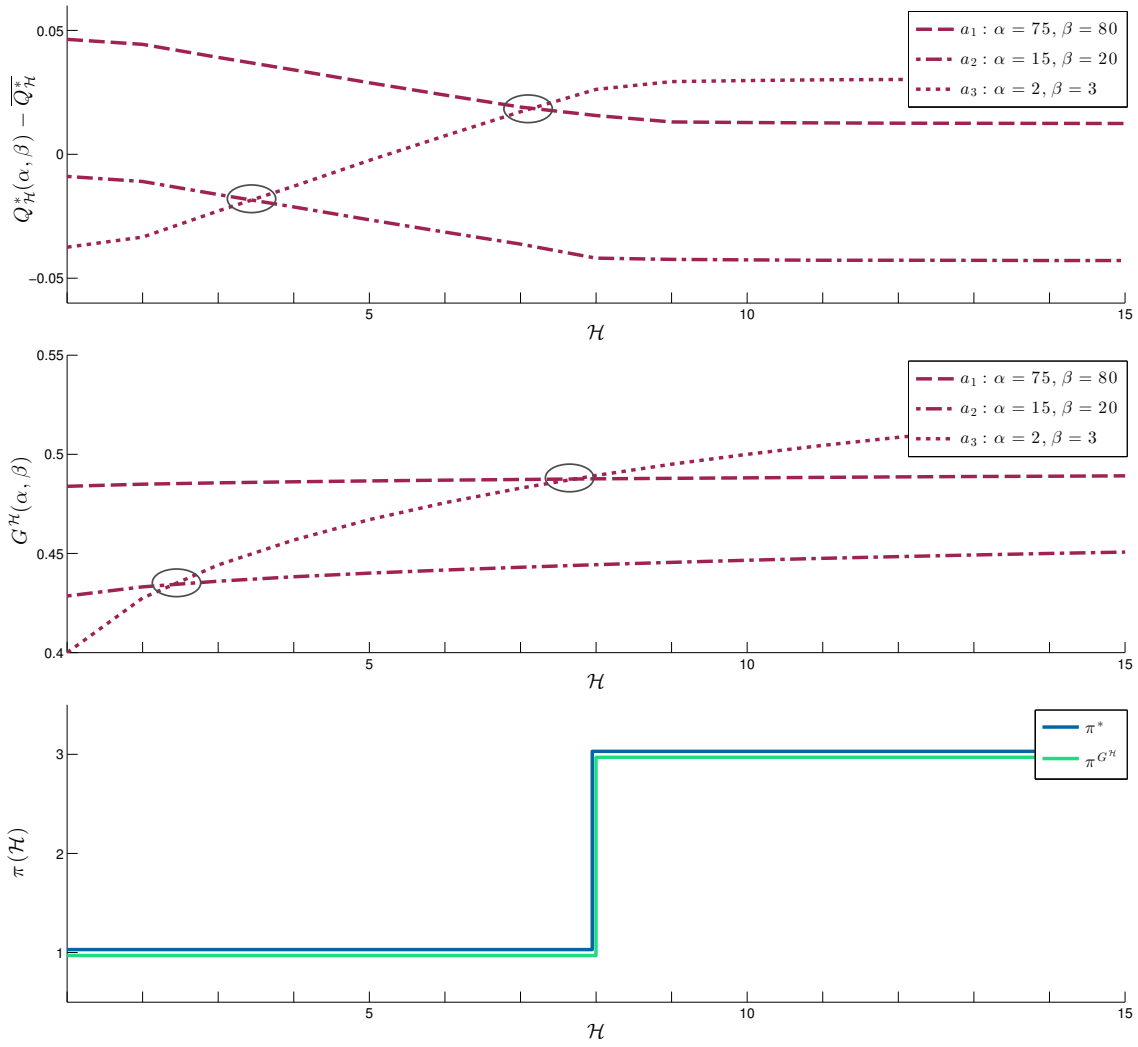


Figure 7.8: A diagram illustrating $Q^*_{\mathcal{H}}(\alpha, \beta)$, $G^{\mathcal{H}}(\alpha, \beta)$, and the resulting policies against N for a three-armed Bernoulli reward MAB. In this example one action, a_3, has been observed a lot less than the other actions. As $\mathcal{H}$ increases away from the one-step case, this action becomes more valuable than both other actions and we see a change in the policy. Functional crossing points are illustrated to demonstrate when a change in the ordering of the actions by value takes place.

value, hence this crossing point has no effect on which action is chosen: action a_1 remains more valuable than either a_2 or a_3. Having said that, we should certainly bear in mind that actions other than the top rated action do still have a value, and this value can be compared against other suboptimal actions. This should be particularly important if we wish to design other approximately optimal decision methods, as we should not consider all suboptimal actions to be associated with the same loss. In

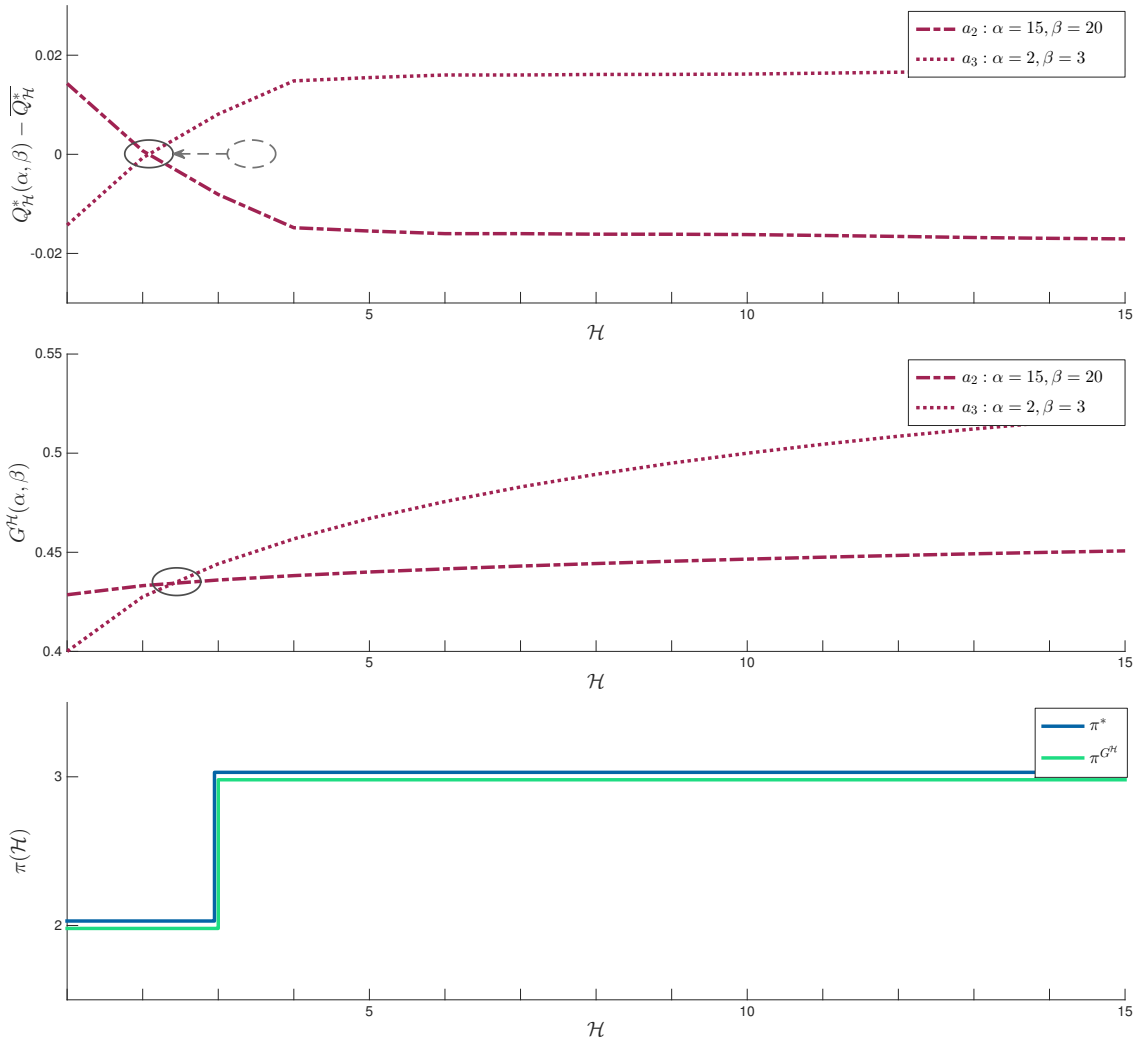


Figure 7.9: A diagram illustrating $Q^*_{\mathcal{H}}(\alpha, \beta)$, $G^{\mathcal{H}}(\alpha, \beta)$, and the resulting policies against N for the three-armed MAB from Figure 7.8, but with action a_1 removed. In this example the GI approximation does not change but the optimal $Q^*_{\mathcal{H}}(\alpha, \beta)$ values do, seen as a lateral shift in the crossing point from the previous figure. The policies of both decision methods coincide in this example.

other words, *not all suboptimal actions are created equal!*

We look in more detail at the a_2/a_3 crossing point in Figure 7.9 by removing action a_1 from the problem altogether. By doing this, we are able to demonstrate a significant difference between optimal decision making and our approximately optimal method. In the approximate method, all actions are considered individually such that the value of an action is only a function of that action's state. If actions are added

to or subtracted from the problem, the value of $G^{\mathcal{H}}(\alpha, \beta)$ remains the same for the remaining actions. However, for optimal decision making the value of each action is inextricably linked to the value of all others in the problem. By removing a_1 from the problem we see a change in $Q^*_{\mathcal{H}}(\alpha, \beta)$ for both a_2 and a_3, which in the figure we can observe as a lateral shift in the position of the crossing point. This is interesting as it means, optimally speaking, that both the relative and absolute value of suboptimal actions is a function of the optimal action. It turns out, however, that in this example the optimal crossing point moves into the same $\mathcal{H}$ increment as the approximately optimal method (it was not previously), and we see the policies agreeing.

Finally, we consider a setting in which the GI approximation does not quite manage to achieve the optimal policy, plotted in Figure 7.10. In this example there are two upper crossing points, and hence two changes in policy, as well as an additional lower crossing point which has no effect on the policy. All three crossing points are fairly localised, which means the policy changes occur in a narrow space of $\mathcal{H}$. As can be seen, both the optimal and approximately optimal functions possess the same number of crossing points in a similar configuration. In this example, however, the second upper crossing point in the GI approximation does not occur in the same $\mathcal{H}$ increment, meaning that the policies differ. This can be seen as a separation between the blue and green lines in the lower plot.

What is most satisfying about this example is that, even though the policies do not completely coincide, the difference between the two methods is very small. We feel that this example in particular reinforces the claim that decision making using $G^{\mathcal{H}}(\alpha, \beta)$ is approximately optimal as opposed to heuristic. As will be discussed in greater detail later, the expected loss attributed to this 'mistake' is relatively small as it doesn't choose an action that was significantly less rewarding. This type of behaviour, we would argue, is where approximately optimal decision making is most important.

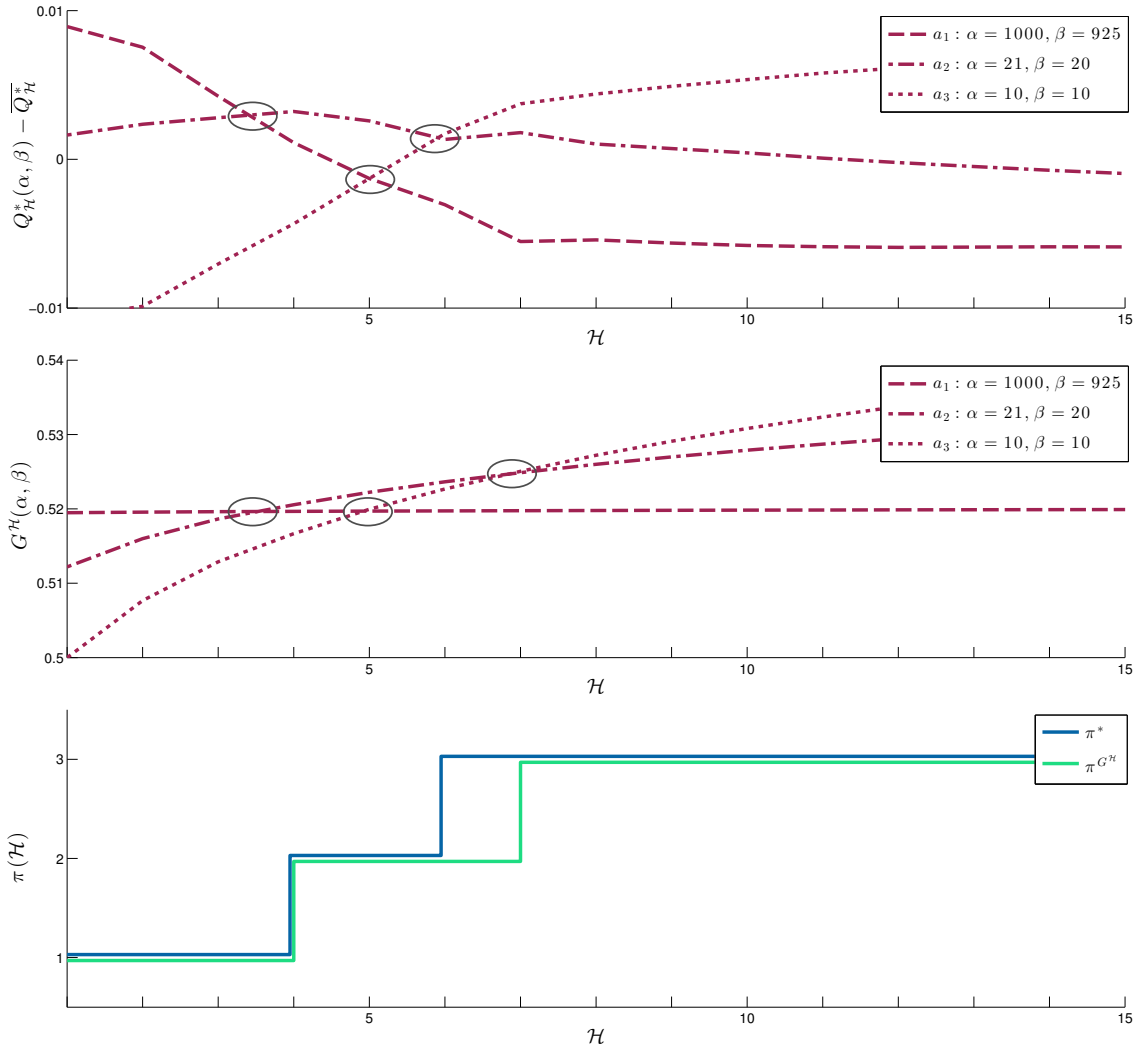


Figure 7.10: A diagram illustrating $Q^*_{\mathcal{H}}(\alpha,\beta)$, $G^{\mathcal{H}}(\alpha,\beta)$, and the resulting policies against N for a three-armed Bernoulli reward MAB, in which the extracted policies do not quite agree. This can be seen as a separation between the blue and green lines in the lower plot.

As a final example for this section, we present an example of the GI approximation being used in a full finite horizon decision problem in order to see how the value of information reduces as we approach the horizon of a problem. Figure 7.11 shows the result of using this policy on a 3-armed MAB problem similar to those we used earlier in this chapter. The value of N used in calculating $G^N(\alpha,\beta)$ is the number of actions available before the horizon is reached, which is given by $\mathcal{H} - t$ where $\mathcal{H} = 300$. Once again we plot the decomposition of the index into the immediate

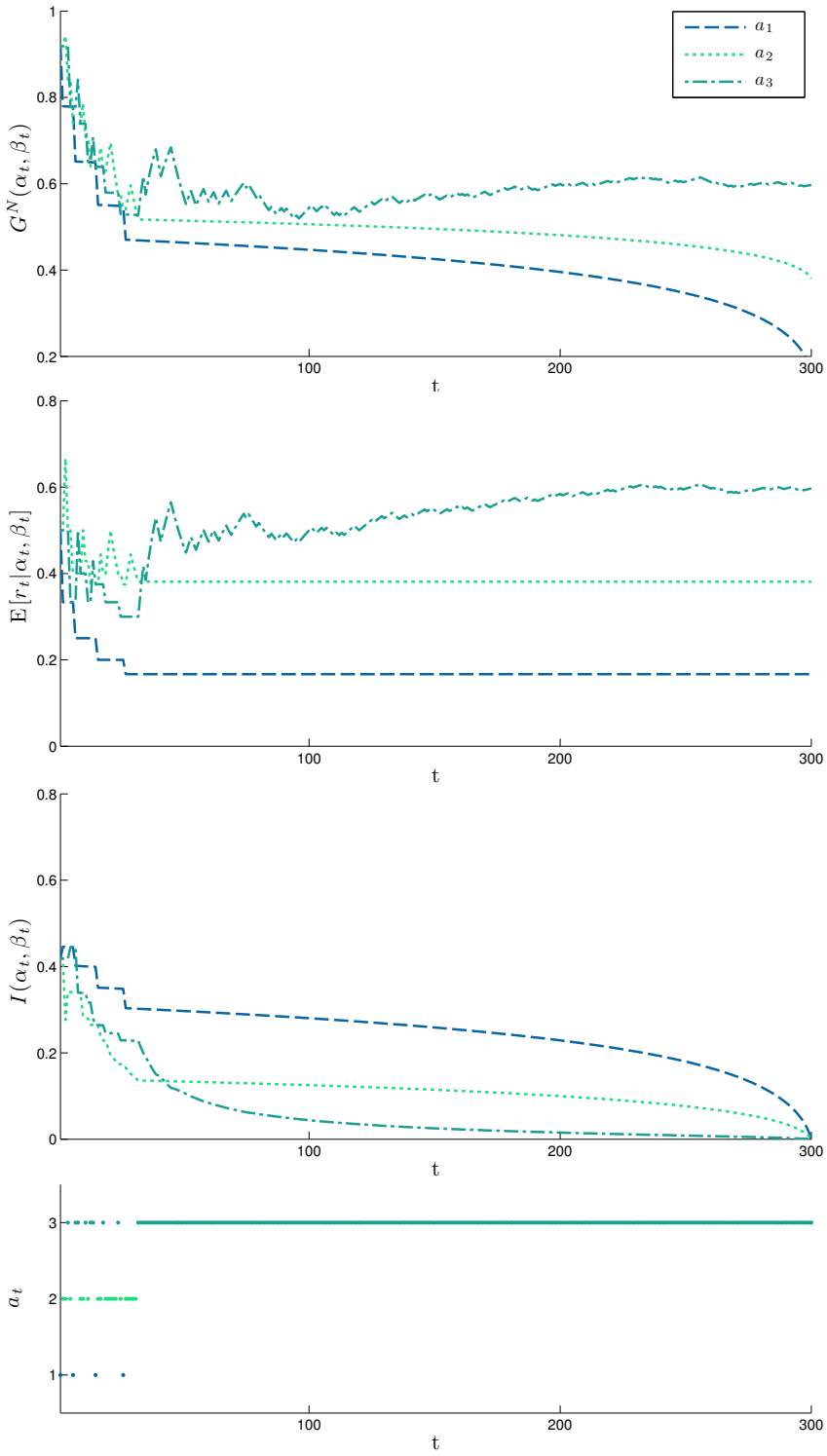


Figure 7.11: Diagram showing a sample decision process for a 3-armed MAB using the GI approximation, $G^N(\alpha, \beta)$, to make decisions. N is given by the number of actions available before the horizon is reached, $\mathcal{H} - t$, where $\mathcal{H} = 300$. All actions demonstrate a reduction in learning component, which using the GI proper would (to its detriment) not occur.

expected reward and the learning component. We can clearly see that as the horizon is approached, the learning components associated with all actions (and not just that associated with the action being sampled) reduce to 0, indicating that further information is not useful. This is precisely the behaviour we would like to see so that we avoid unnecessary exploration and potentially wasted actions.

7.3 Making Suboptimal Decisions

Up until now the discussion has concentrated on how to choose the action which maximises the expected return. Whilst this remains our goal, as we move towards further discussion on approximate decision methods we find the need to answer two questions: what is the loss, or rather loss of reward, associated with taking a suboptimal action; and how can we use this to guide decision making? The first question can in part be answered by a redefinition of our objective into a quantity known as *regret* (Berry and Fristedt, 1985; Cesa-Bianchi and Lugosi, 2006), which provides a means to evaluate actions as they relate to the maximum achievable return, and which we introduce next. The answer to the second question is as open ended as the number of heuristic decision methods we could generate; however, we demonstrate how the relative value of actions can and has been used.

7.3.1 Regret

Despite our focus on one action being better than all the others, it is not so that the same loss can be associated with all suboptimal actions. It is therefore prudent to develop the means to quantify the quality of decisions should we fail in choosing the best action. As just stated, this can be done by redefining the objective in terms of regret. In the first instance regret can be used as an analysis tool, although here we introduce it as a key concept in the development of heuristic methodology, for which

it is used in algorithm design and more commonly to provide long term bounds on performance.

Aside from its use in Reinforcement Learning problems, regret has been investigated in the context of human decision making and economic choice. *Regret Theory* (Loomes and Sugden, 1982; Bell, 1982) refers to models of choice under uncertainty which take into account the negative emotion associated with learning that an alternative decision would have resulted in a better outcome.

Regret, as we use it here, is formally defined as the difference between the maximum available return and the received return, such that after T actions the regret, R_{gt}, is:

$$R_{gt} = \max_{j=1,\ldots,K} \sum_{t=1}^{T} r_t^{a_j} - \sum_{t=1}^{T} r_t^{\pi} \tag{7.5}$$

where r_t^{π} is the reward received at time t after following policy π. In general actions and rewards may be stochastic, leading us to consider two notions of averaged regret - *expected regret*:

$$\mathbb{E}[R_{gt}] = \mathbb{E}\left[\max_{j=1,\ldots,K} \sum_{t=1}^{\mathcal{H}} r_t^{a_j} - \sum_{t=1}^{\mathcal{H}} r_t^{\pi_t}\right] \tag{7.6}$$

and *pseudo-regret*:

$$\overline{R_{gt}} = \max_{j=1,\ldots,K} \mathbb{E}\left[\sum_{t=1}^{\mathcal{H}} r_t^{a_j} - \sum_{t=1}^{\mathcal{H}} r_t^{\pi_t}\right] \tag{7.7}$$

which can be extended to the discounted infinite horizon form as follows:

$$\overline{R_{gt}} = \max_{j=1,\ldots,K} \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t^{a_j} - \sum_{t=1}^{\infty} \gamma^{t-1} r_t^{\pi_t}\right] \tag{7.8}$$

although for simplicity we will tend to use the original form.

In both forms of regret, the expectation is taken with respect to rewards and actions. Pseudo-regret is considered a weaker notion of regret, since the chosen action is judged against the action which is optimal only in expectation rather than on the sequence of reward realisations (Bubeck and Cesa-Bianchi, 2012). MAB liter-

ature, however, focusses on pseudo-regret as the quantity of interest, as it is a more natural characterisation of optimality in stochastic frameworks. We emphasise that maximising expected return also minimises pseudo-regret, and so as already stated the objectives are the same.

For the MAB we can simplify pseudo-regret by making some substitutions. We begin by writing the true expected reward for each action as:

$$\mu_j = \mathbb{E}[r|a_j] \tag{7.9}$$

$$\mu^* = \max_{j=1,\dots,K} \mathbb{E}[r|a_j] \tag{7.10}$$

both of which are calculated with respect to the true reward distribution $\mathcal{R}_j$ and associated parameters θ_j, and are thus unknown to the decision maker. Given these, we can rewrite Equation (7.7) as:

$$\overline{R_{gt}} = \mathcal{H}\mu^* - \sum_{t=1}^{\mathcal{H}} \mathbb{E}\left[\mu_{\pi_t}\right] \tag{7.11}$$

where the expectation here is taken with respect to the belief distribution of the decision maker, as well as the policy if it is stochastic. We also define the following:

$$C_j(\mathcal{H}) = \sum_{t=1}^{\mathcal{H}} \mathbb{1}_{\pi_t = a_j} \tag{7.12}$$

where $C_j(\mathcal{H})$ is the number of times action a_j will be sampled until the horizon under the policy. Finally, using this we can rewrite the pseudo-regret as:

$$\begin{aligned}\overline{R_{gt}} &= \left(\sum_{j=1}^{K} \mathbb{E}\left[C_j(T)\right]\right)\mu^* - \mathbb{E}\left[\sum_{j=1}^{K} C_j(T)\mu_j\right] \\ &= \sum_{t=1}^{K} \Delta_j \mathbb{E}\left[C_j(T)\right]\end{aligned} \tag{7.13}$$

where $\Delta_j \coloneqq \mu^* - \mu_j$.

7.4 Heuristic Decision Making

A variety of methods have been developed to answer the exploration vs. exploitation dilemma from a heuristic point of view. These methods tend to fall into a small set of base methodologies which have been extended over time. In this set we can further categorise heuristic methods into two distinct groups: deterministic methods, and stochastic methods which use randomisation as a means of exploration. In this section we present a number of core methodologies within these categories in order to provide a reasonable overview of heuristic decision making and current avenues of research.

As heuristic methods do not solve the exploration vs. exploitation tradeoff in an optimal sense, a large part of the research in this area involves finding performance bounds on regret which, as noted in Kaufmann et al. (2012b), is a fundamentally frequentist approach. The bounding is typically looked at as a function of the number of actions taken. If suboptimal actions are consistently being taken then the regret will increase as more actions are taken. The extent to which this happens is the main way in which heuristic methods are differentiated.

7.4.1 Upper Confidence Bounds

The first class of heuristic algorithms we discuss were introduced in Lai and Robbins (1985) and are based on upper confidence bounds (UCBs) on the expected reward of each action in a MAB. In that paper the authors found, for particular families of reward distributions, policies satisfying:

$$\mathbb{E}[C_j(t)] \leqslant \left[\frac{1}{D_{KL}(\theta_j||\theta^*)} + o(1)\right] \ln t \qquad \text{for } a_{j\in K} : \mu_j < \mu^* \tag{7.14}$$

where $o(1) \to 0$ as $t \to \infty$, and

$$D_{KL}(\theta_j||\theta^*) = \int \mathcal{R}_j(r|\theta_j) \ln \frac{\mathcal{R}_j(r|\theta_j)}{\mathcal{R}^*(r|\theta^*)} \, \mathrm{d}r \tag{7.15}$$

is the Kullback-Leibler divergence between the reward density $\mathcal{R}^*(r|\theta^*)$ of the action with the highest expected reward and the reward density $\mathcal{R}_j(r|\theta_j)$ of any other action. This means that asymptotically under these policies, the action with the highest expected reward will be taken exponentially more often than all others.

The result of most apparent importance in Lai and Robbins (1985) is not a policy but a proof that asymptotically the regret that arises from Equation (7.14) is in fact the best possible of any policy, provided the reward distributions of the associated problem satisfy some mild assumptions. They showed that for any allocation strategy and any action a_j for which $\mu_j < \mu^*$:

$$\lim_{t\to\infty} \mathbb{E}[C_j(t)] \geqslant \frac{\ln t}{D_{KL}(\mu_j||\mu^*)} \tag{7.16}$$

which yields a lower regret bound of:

$$\lim_{t\to\infty} \overline{R_{gt}(t)} \geqslant \left[\sum_{j:\mu_j<\mu^*} \frac{\Delta_j}{D_{KL}(\mu_j||\mu^*)} \right] \ln t \tag{7.17}$$

This result could be considered as important to heuristic decision making as the Gittins Index is to Bayesian decision making, as it has largely dictated research into theoretical guarantees of such methods. It is proximity to this bound that researchers are referring to when they say a heuristic method is close to 'optimal'. One should therefore be careful not to confuse this definition of optimality with the definition used in the majority of this thesis, specifically actions derived from a solution to the full decision tree. Equally importantly, one should be happy with the fact that this is an asymptotic result, and as such realised regret can in fact be below the bound.

UCB strategies are index methods, in the same vein as the Gittins Index, in that they give to each action a value based on the rewards received whilst taking that action. Once these indices have been calculated, the policy takes the action with the highest value. The UCB policies introduced in Lai and Robbins (1985) are very hard to compute and rely on the entire sequence of rewards generated from each action. More tractable methods have since been developed, starting with work in Agrawal (1995), which generate indices as simple functions of the total reward received from each action. Several UCB strategies appear in Auer et al. (2002) which are frequently referenced in literature. The first strategy, described as UCB1 in the paper, is often credited as simply 'UCB' by many authors (see for example Kaufmann et al. (2012a)) due to it being the first such method to be widely adopted. As such, Auer et al. (2002) is now essentially the seminal piece of research on the subject.

The UCB1 policy is generated by forming an upper confidence $u_{j,t}$ on the true expected reward for each action in relation to the sample mean of received rewards $\bar{r}_{j,t}$, such that with high probability:

$$\mathbb{E}[r_j] \leqslant \bar{r}_{j,t} + u_{j,t} \tag{7.18}$$

where $C_j(t)$ is the number of times action j was taken up until time t, and

$$\bar{r}_{j,t} = \frac{1}{C_j(t)} \sum_{i=1}^{C_j(t)} r_{j,i} \tag{7.19}$$

The policy selects actions using this *optimistic* estimation of the true expected reward, such that:

$$\pi_{UCB1} = \underset{j \in K}{\operatorname{argmax}}(\bar{r}_{j,t} + u_{j,t}) \tag{7.20}$$

Hoeffding's inequality (Hoeffding, 1963) provides an upper bound on the probability that the sum of random variables deviates from its expected value. Applying this we

can find the probability that the true expected reward exceeds the UCB:

$$P\left(\mathbb{E}[r_j] > \bar{r}_{j,t} + u_{j,t}\right) \leqslant e^{-2\,C_j(t)\,u_{j,t}^2} \tag{7.21}$$

where $r \in [0, 1]$. Setting this upper bound on the probability to p we arrive at:

$$u_{j,t} = \sqrt{\frac{-\ln p}{2C_j(t)}} \tag{7.22}$$

Auer et al. (2002) set $p = t^{-4}$ which means $u_{j,t} \to 0$ as $t \to \infty$ and the policy will select the action with the greatest expected reward (assuming the sample means will have converged). The upper bound therefore becomes:

$$u_{j,t} = \sqrt{\frac{2\ln t}{C_j(t)}} \tag{7.23}$$

which inserted into the policy means:

$$\pi_{UCB1} = \underset{j \in K}{\operatorname{argmax}} \left(\bar{r}_{j,t} + \sqrt{\frac{2\ln t}{C_j(t)}} \right) \tag{7.24}$$

This policy results in an upper bound on the regret as follows:

$$\overline{R_{gt}} \leqslant \left[8 \sum_{j:\mu_j < \mu^*} \left(\frac{\ln t}{\Delta_j} \right) \right] + \left(1 + \frac{\pi^2}{3} \right) \left(\sum_{j=1}^{K} \Delta_j \right) \tag{7.25}$$

Importantly, this bound is defined for any value of t, and therefore differs to the asymptotic bound of (7.17).

7.4.2 ϵ-Greedy

A very simple heuristic decision method would be to always take the action which possesses the largest sample mean reward. Ignoring the fact that from a Bayesian

point of view the sample mean is a naive estimator, such *greedy* methods do a poor job of producing large rewards as they fail to adequately explore the full action space (Sutton and Barto, 1998; Scott, 2010). They are also completely unrepresentative of the full decision tree. Stochastic methods offer an improvement to this idea by using randomisation as a means of artificial exploration, forcing samples to be randomly taken from undervalued actions.

The first stochastic decision method we discuss, ϵ-Greedy, is both one of the most widespread and simplest. The method seems to have been first described in Watkins (1989) and is heavily discussed in Sutton and Barto (1998). In its basic form ϵ-Greedy defines a policy which takes the highest rated action a_{tmax} with probability equal to $1 - \epsilon$, otherwise sampling *uniformly* from the remaining, potentially large, action pool:

$$\pi_\epsilon(t): \quad \begin{cases} a_t = a_{tmax} & \text{with Prob} = 1 - \epsilon, \\ a_t \neq a_{tmax} & \text{with Prob} = \epsilon. \end{cases} \tag{7.26}$$

where ϵ must be in the open interval $(0, 1)$. Actions in this method are rated according to the sample mean of received rewards, $\bar{r}_{j,t}$, such that:

$$a_{tmax} = \underset{j \in K}{\operatorname{argmax}} \left(\bar{r}_{j,t}\right) \tag{7.27}$$

The ϵ-Greedy policy, with fixed ϵ, means that as $t \to \infty$ *all* actions will be sampled extensively enough for the mean of their observed rewards to converge to the true expected rewards. Such a method therefore has asymptotic guarantees of finding the optimal action, yet will continue to sample from suboptimal actions with probability defined by ϵ leading to a linear growth in regret. This leads to the notion of ϵ-decreasing strategies in which ϵ decreases as function of t.

Auer et al. (2002) present an ϵ-decreasing strategy called ϵ_t-Greedy. The ϵ_t policy is to select actions according to ϵ-Greedy, but with ϵ a function of the number

of actions taken, t, as follows:

$$\pi_{\epsilon_t}(t): \quad \begin{cases} a_t = a_{tmax} & \text{with Prob} = 1 - \epsilon_t, \\ a_t \neq a_{tmax} & \text{with Prob} = \epsilon_t. \end{cases} \tag{7.28}$$

where

$$\epsilon_t \coloneqq \min\left\{1, \frac{cK}{d^2 t}\right\} \tag{7.29}$$

The value of the parameters c and d is a choice left to the decision maker; however in order for the regret bounds to hold d must be set to less than or equal to the minimum difference between the true expected reward of the optimal and a suboptimal action:

$$0 < d \leqslant \min_{j:\mu_j < \mu^*} \Delta_j \tag{7.30}$$

The ϵ_t policy has better regret bounds than a vanilla ϵ-Greedy strategy. As presented in the original paper, the probability after $t \geqslant cK/d$ plays that the policy chooses a suboptimal action is bounded as follows:

$$P(a_t \neq a^* | t \geqslant cK/d) \leqslant \frac{c}{d^2 t} + 2\left(\frac{c}{d^2}\ln\frac{(t-1)d^2 e^{0.5}}{cK}\right)\left(\frac{cK}{(t-1)d^2 e^{0.5}}\right)^{c/(5d^2)} + \frac{4e}{d^2}\left(\frac{cK}{(t-1)d^2 e^{0.5}}\right)^{c/2} \tag{7.31}$$

For particular values of the user defined parameters this bound is $\mathcal{O}(1/t)$ as $t \to \infty$, which means the regret is near to a $\mathcal{O}(\log t)$ bound (Scott, 2010), and is therefore comparable to the 'optimal' lower bound given in Equation (7.17).

7.4.3 Boltzmann Exploration (Softmax)

The ϵ-Greedy strategy and its variants choose uniformly when allocating exploratory actions. Clearly, some of the actions in the 'exploration pool' will be valued higher

than others, and therefore be less likely suboptimal. A better strategy would choose actions in such a way so as to under sample actions that are more likely to be suboptimal.

A certain class of policies, called probability matching strategies, do just that. Boltzmann exploration (Luce, 1959) otherwise known as Softmax, is one such strategy and randomly assigns actions proportional to their estimated value through a Boltzmann distribution. The Softmax policy selects actions with probability defined as follows:

$$\pi_{BA}(t): \quad p(a_{j,t}) = \frac{e^{\bar{r}_{j,t}/\tau}}{\sum_{i \in K} e^{\bar{r}_{i,t}/\tau}} \tag{7.32}$$

where τ is a temperature parameter that determines the balance between actions. In the limit as $\tau \to 0$ the policy is equivalent to a greedy strategy, whilst as $\tau \to \infty$ the policy selects actions uniformly.

Unsurprisingly, the Softmax strategy has similar asymptotic performance issues to the standard ϵ-Greedy due to the continued sampling of suboptimal actions. Introducing a "cooling schedule", analogous to that used in Simulated Annealing optimisation (Kirkpatrick et al., 1983), can be implemented to decrease $\tau = \tau_t$ over time, and thus achieve a policy similar to an ϵ-decreasing strategy. For example, in Cesa-Bianchi and Fischer (1998) a Softmax strategy with decreasing temperature called SoftMix is presented and analysed. SoftMix uses a temperature decreasing with a factor of $1/\log(t)$ and achieves a $\mathcal{O}(\log^2 t)$ regret bound. Away from the MAB literature, Singh et al. (2000) provide a cooling schedule which satisfies *GLIE* (greedy in the limit with infinite exploration) policy requirements, and prove convergence of the SARSA algorithm under the scheme.

7.4.4 Thompson Sampling

The last stochastic method we discuss, Thompson Sampling (TS), dates back to Thompson (1933), yet has only recently become a popular algorithm for RL problems. Chapelle and Li (2011) suggest this might have been due to a lack of strong theoretical analysis. Like Boltzmann exploration, the algorithm is a probability matching strategy in that it chooses actions proportional to the probability that an action has the highest true expected reward. The method is often classed as Bayesian (see for example Scott (2010) and Agrawal and Goyal (2012)) due to the use of Bayesian inference and prior distributions over the reward parameters; however it certainly does not solve the full decision problem. As a result, articles on the subject often refer to the action with the true highest reward mean as being 'optimal'; once again care should be taken in not confusing this with actions that are derived from a solution to the full decision tree.

TS defines a policy which takes each action with probability equal to the probability that the action will yield the largest expected immediate reward:

$$\pi_{TS}(t): \quad p(a_{j,t}) = w_{j,t} = P\Big(\mu_j(\theta) = \max\{\mu_1(\theta), \ldots, \mu_K(\theta)\}\Big|\mathcal{D}_t\Big) \tag{7.33}$$

where $\mu_j(\theta) = \mathbb{E}[r|\theta, a_j]$ and θ represents the parameters of the reward function as before. As was show in Section 6.1.2, a posterior distribution over θ can be constructed by introducing a conjugate prior with respect to the data likelihood. The posterior distribution $p(\theta|\mathcal{D}_t)$ can then be used to compute Equation (7.33). If $\mathbb{I}_j(\theta)$ is an indicator function which equals 1 if $\mu_j(\theta) = \max\{\mu_1(\theta), \ldots, \mu_K(\theta)\}$ and 0 otherwise, then

$$w_{j,t} = \mathbb{E}[\mathbb{I}_j(\theta)|\mathcal{D}_t] = \int \mathbb{I}_j(\theta)\, p(\theta|\mathcal{D}_t)\, \mathrm{d}\theta \tag{7.34}$$

For some families of reward distribution Equation (7.34) can be calculated analytically or through quadrature, as discussed in Scott (2010). In practice the $w_{j,t}$

probabilities do not need to be computed explicitly and a sampling scheme can be used, hence the naming of the strategy. In this case it is adequate to sample a single draw of the parameters $\theta \sim p(\theta|\mathcal{D}_t)$ and then choose the action with the largest expected reward given the draw. As such the policy becomes:

$$\pi_{TS}(t) = \underset{j \in K}{\operatorname{argmax}} \left(\mu_j(\theta_j) \,\middle|\, \theta_j \sim p(\theta_j|\mathbf{d}_{j,t}) \right) \tag{7.35}$$

In the case of the Bernoulli reward MAB, this is rather trivially:

$$\pi_{TS}(t) = \underset{j \in K}{\operatorname{argmax}} \left(\lambda_{j,t} \sim p(\lambda_j|\mathbf{d}_{j,t}) \right) \tag{7.36}$$

where the samples are taken from a posterior Beta distribution. After each sample and action, the posterior distributions are updated as normal and a new sample is taken. TS can therefore be thought of as maximising the immediate expected reward with respect to a randomly drawn belief.

As already mentioned, theoretical guarantees for TS took a long time coming. Granmo (2010) (under the moniker of a Bayesian Learning Automaton) and May et al. (2012) provided asymptotic (infinite-time) guarantees, whilst Chapelle and Li (2011) carried out extensive experiments with Bernoulli rewards and beyond. Agrawal and Goyal (2012) were, however, the first to provide non-asymptotic regret analysis of the method, achieving a bound of $\mathcal{O}(\left[(\sum_{j:\mu_j<\mu^*} \frac{1}{\Delta_j^2})^2\right] \ln t)$ for the K-armed bandit problem.

TS is certainly easier to enact than optimal decision making, and allows for simple extension to a great deal of reward distributions. The decision maker simply needs to construct a prior over the governing parameters, perform Bayesian inference using observations, and sample from the posterior distribution. As an example, Chapelle and Li (2011) construct a TS strategy using the output of logistic regressor in order to incorporate contextual information into what would otherwise be a

Bernoulli type MAB. The strategy will also continue to sample from actions that have a positive probability of being optimal according to Bayesian inference. Finally, in its basic form there are no parameters for the decision maker to heuristically set, outside from choosing a suitable prior, which means the method is transferable to a range of problems without too much supervision. Nevertheless, the method can not be said to solve the decision tree and so, as with all stochastic methods, is suboptimal.

7.4.5 Hybrid Approaches

The heuristic strategies we have discussed here use a range of methods to try and solve the exploration vs. exploitation dilemma. It is not uncommon for strategies to utilise different elements of these methods to form a hybrid approach. For instance, Audibert et al. (2009) incorporate an *exploration function* in a UCB type strategy in order to moderate the upper bound. This gives further control to the balance between exploration and exploitation, comparable to the way in which the exploration parameter can be varied in an ϵ-Greedy strategy.

One method, Bayes-UCB (Kaufmann et al., 2012a), attempts to bridge the gap between the Bayesian inference on the reward distributions used in TS and the frequentist bounds in UCB. This algorithm appears to do better than a number of the core heuristic methods on a simple test bed, reinforcing the idea that the sample mean is a poor estimator. It seems likely that the incorporation of a prior over the reward distributions used in other stochastic methods would also provide a more effective method, although we do not know of any regret analysis using this approach.

7.5 Critique of Stochastic Decision Making

It should be clear now that heuristics, which in various ways seek to fulfil the desiderata outlined at the beginning of the chapter, do not resemble optimal methodology in

their construction. Where optimal decision making involves the forward propagation of belief through a decision tree, the methods presented in the last section collapse the problem to a myopic one step estimate, requiring some form of ad-hoc procedure to gather information. A large majority of these methods use some form of stochastic reasoning, which by definition means control of part of the decision process is relinquished, generating an additional source of uncertainty. It should be patently obvious that, because of this, such methods require all the more scrutiny.

Let us be very upfront and state that we believe choosing decisions in a stochastic manner is fundamentally unsound. Given our acclamation of Bayesian theory this should come as no surprise, although our acknowledgement of the need for approximation compels us to make it clear. Unfortunately, we face a difficult problem in conveying why stochastic decision making is inadequate. A large part of this is born out of the desiderata being prescriptive for these algorithms. Their very invention relies on fulfilling a set of intuitions that is compatible with some parts of common sense; they seem to do the right thing. We note however that alone, the desiderata of this chapter do not specify a principled treatment of information and uncertainty. At this point we could simply rely on the argument that stochastic decision making is inconsistent with Bayesian theory, however this seems rather escapist. Instead we look to appeal to less obvious violations of common sense as well as experimental results.

One problem with typical long term analyses is that they do very little to develop insight into precisely what is 'going on' in the decision process. Key questions on the fundamental validity of each method can be concealed by favourable average performance. Further to this, asymptotic guarantees of the sort most heuristic methods aspire to, really only serve to create confidence in solutions to problems which are neither the most critical nor the most challenging. For these reasons we begin this section by extending the analyses presented in Section 7.2, where we observed the

manner in which optimal decision making transitions from exploratory to exploitative behaviour on a MAB. By observing specific decisions, we should get a different feel for how the tradeoff is managed under a stochastic policy than that gained from a large test bed of the sort presented later.

7.5.1 Stochastic Decision Making in Detail

The stochastic methods we choose to investigate are ϵ-Greedy and Softmax, for which we test the following parameter settings: $\epsilon = 0.05$, 0.1 and 0.2, and $\tau = 0$, 0.02 and ∞; as well as vanilla Thompson Sampling. We include $\tau = 0$ and $\tau = \infty$ in Softmax for context as they represent greedy and uniform selection respectively. The value of $\tau = 0.02$ was chosen so that operation takes place somewhere between these extremes. Whilst for optimal decision making we could observe exploratory behaviour emerge as a function of the horizon, this is not possible for the core heuristic methods for which exploration is determined either through extemporaneous parameters or as a function of state. For 'anytime' policies like these we have to visualise the decisions they specify in a different manner. The clearest way this can be done is by showing the distribution over actions each method prescribes for the same set of problems investigated in Section 7.2. The first of these is illustrated in Figure 7.12. In order to provide context we include the optimal $Q^*_{\mathcal{H}}(\alpha, \beta)$ values and associated policy as presented before, as well as the posterior distribution from which Thompson Sampling draws samples.

For the first example, we start once again by looking at a problem for which there is low uncertainty across the action space, due to extensive prior sampling. This can be seen as tight, informative distributions in the posterior plot. Quite clearly, in this scenario optimal decision making favours the greedy action, as the policy has already converged. Such behaviour is favourable as no further information is required to improve the policy. To begin the analysis of the stochastic methods, let us draw

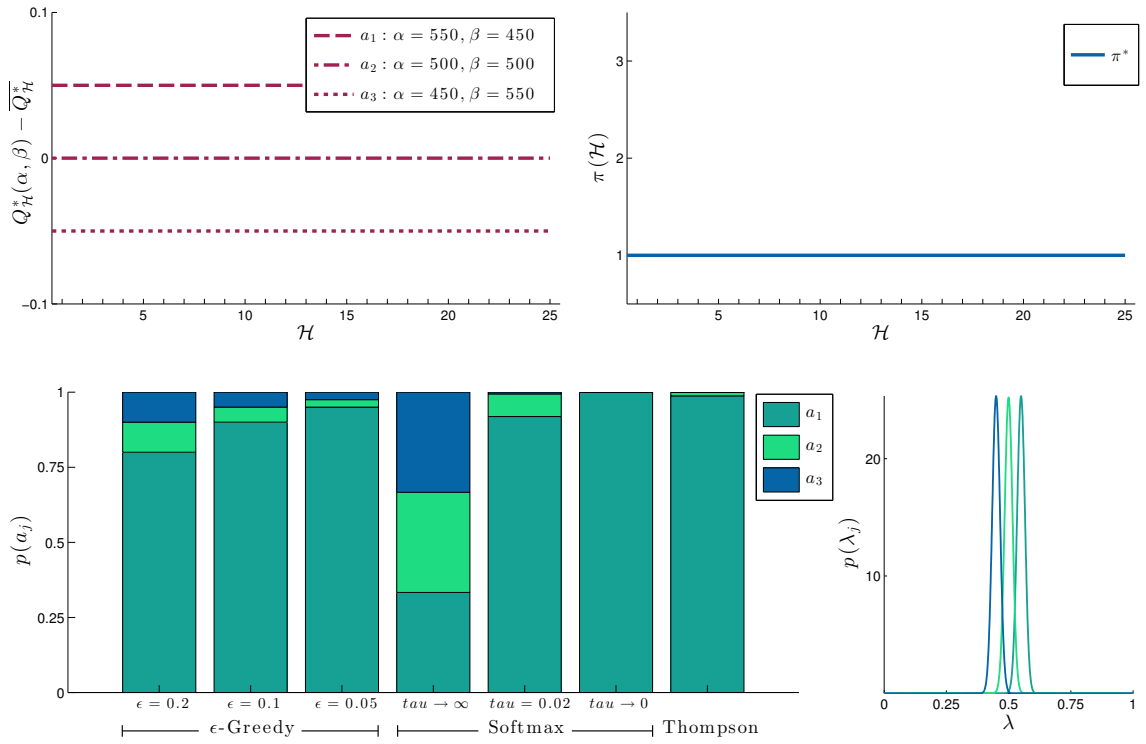


Figure 7.12: A diagram illustrating the action distribution prescribed by a set of stochastic decision methods for a three-armed Bernoulli reward MAB alongside the optimal $Q^*_{\mathcal{H}}(\alpha, \beta)$ values and associated policy. Also pictured is the posterior distribution from which Thompson Sampling draws samples. In this example all three actions have been sampled extensively.

our attention to the three flavours of ϵ-Greedy. Here we can see that action taking is heavily biased towards greedy behaviour whilst allocating equally amongst the alternatives, the amount of which is governed by ϵ. Clearly any exploration is unnecessary at this point, although a fraction of actions are allocated in this way nevertheless. Continuing, in Softmax with $\tau = 0.02$ we see that decisions are also predominantly greedy, with essentially no allocation given to the least rewarding action. Finally, TS appears to have converged to greedy selection of the most rewarding action. It is perhaps not surprising that all three methods perform relatively well on what is approaching the asymptotic end of the problem domain. We now look to understand how this changes as uncertainty becomes more prevalent.

For the second example, illustrated in Figure 7.13, we look at a problem in

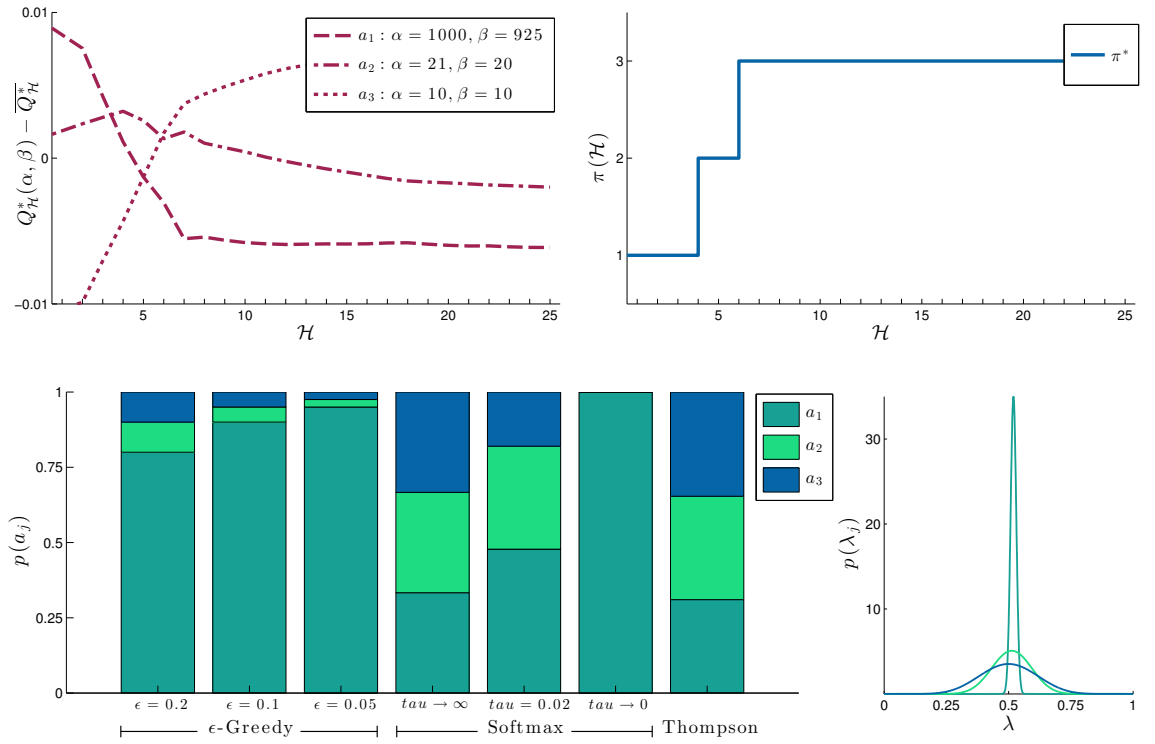


Figure 7.13: A diagram illustrating the action distribution for the same stochastic decision methods as Figure 7.12, on a problem in which only one action has been sampled extensively and the expected immediate rewards from all three are close to one another.

which only one action has been sampled extensively, with that action having the highest expected immediate reward at the outset. Starting again with ϵ-Greedy, we observe the same heavy biasing towards greedy behaviour, with equal allocation amongst alternatives. Comparing this to the optimal policy we see that the majority of problems (in terms of horizon) would in fact favour one of the alternative actions, namely a_3. Clearly in this case ϵ-Greedy substantially down weights the need for exploration. Note also that the policy does not differ at all from the previous example. Moving to Softmax we see more favourable behaviour. The policy has moved away from almost pure exploitation to one in which exploration takes place, with only a slight bias towards greedy behaviour. Finally, TS selects almost uniformly amongst all three actions, with a slight bias towards a_3. This would suggest TS has more of a

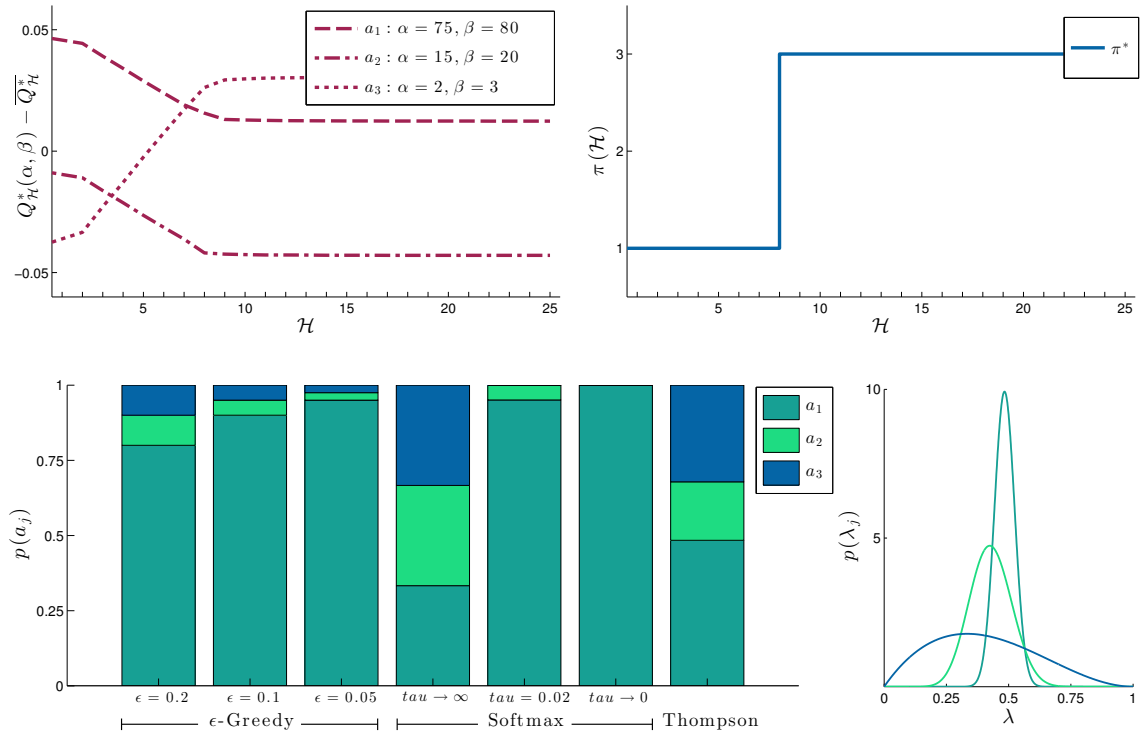


Figure 7.14: A diagram illustrating the action distribution for the same stochastic decision methods as Figures 7.12 and 7.12, on a problem in which there is more uncertainty across the action space than both previous examples.

tendency to explore the action space than the other methods, whilst the results from the last section demonstrate it is capable of converging to the optimal policy under low uncertainty.

Finally, in our last example, illustrated in Figure 7.14, we look at a problem in which there is more uncertainty across the action space than in both of the previous examples. Once again the ϵ-Greedy policies are identical to the previous examples, despite the increase in uncertainty. Softmax now also appears very myopic, and allocates only a tiny fraction to exploration. In observing this we should be aware that Softmax, as with other heuristic methods, can be highly sensitive to the choice of parameter. We will explore this in more detail later in the chapter. Lastly TS, as in the last example, tends towards more exploratory behaviour than the other methods.

One useful insight that can be gained from these examples is related to the

'anytime' nature of the stochastic methods. According to optimal decision making, whether one acts greedily or in an exploratory manner is very much dependent on how many actions are left to take. With the stochastic methods a performance tradeoff has been made to accommodate a whole variety of problem lengths. We mentioned that efforts have been made to vary variables like ϵ and τ over the course of the problem; however those solutions remain based on a one-step approximation.

The three examples are specific decision problems, and therefore one should be careful when generalising. As stated at the beginning of the section, the primary purpose of considering these examples was to give a flavour of how stochastic decision methods operate, as well as an initial comparison between the approaches. In the last part of this chapter we move towards more traditional analysis and observe performance across a large number of decision problems. Before doing this, however, we consider more deeply the uncertainty a stochastic decision process generates, specifically in regards to notions of utility. This will also enable us to better interpret large scale analyses and average performance metrics.

7.5.2 Randomisation and Utility

By enacting a stochastic decision method one must accept that, whilst average performance may be favourable, the results of individual decisions may be significantly against the overall objective. To understand this we appeal to further intuition on how humans make decisions. Admittedly we do this with no reference to cognitive research, although we believe the picture it creates to be powerful nonetheless. Whether or not the human brain can behave in a random way is not assumed; instead we look to a gut instinct on the issues faced.

Let us imagine how one might feel about the notion of stochastic decision making in two real-world problems. For the first problem imagine a person is walking the streets of a new city, trying to find a particular cathedral where a friend of hers is

waiting. Without a map or knowledge of where exactly she is, she would surely engage is some form of exploration whilst maintaining the clear goal of getting to her friend before too long. For the second problem, consider a technology company that is about to invest in a series of products. It is well within the realms of possibility that the company will engage in some early research and panel testing before committing the full extent of their resources.

The two problems are manifestly different, although being as they both define some sequential decision process under uncertainty it seems reasonable that they could be approached in a similar manner. What should be quite clear is that whilst the idea of stochastic decision making in one seems palatable, in the other it is quite the opposite: shareholders would cry villainy if the Board of Directors of their company was choosing product lines at random. Before going further, let us state that by palatability we mean to refer to how much we would contest the use of randomness. We find it hard to believe that a person would truly make a random decision rather than act on some belief in the direction being correct, however vague. We once again defer to research in the cognitive sciences to determine what is actually the case.

The primary distinction we can make as to the difference in judgement between the two examples seems to be related to utility and notions of risk. On one hand we do not stand to lose much by taking a wrong turn, whilst on the other there may be significant financial concerns if the wrong product is launched. It is not difficult to imagine further day-to-day examples where the result of a random choice would be far less detrimental than for problems to which we might dedicate a large amount of forethought. The same reasoning must also be applied to the type of automated decision making we approach in this thesis. If a choice is made to use stochastic methodology, all the more attention should be levelled at utility and the risk this exposes us to.

Part of the reason for this exposition of risk in decision making relates to how

we interpret experimental results. Whilst it is possible to define some alternative utility mapping, most problems considering long term and asymptotic returns indicate that the loss associated with individual actions is very small. Therefore, in these sort of problems average reward across the problem is a reasonable measure, and thus stochastic methods perform comparably well. Further to this, by averaging over many runs of the same problem, we can find similar performance being attributed to very different approaches. We can think of this in the context of high risk versus conservative strategies in which average rewards are similar, but where for the high risk strategies individual returns can be much higher (and lower) than for their conservative counterparts.

At this point the reasons for which we believe stochastic decision making to be problematic should be clear. From a human perspective there seems to be something fundamentally wrong with taking courses of action completely at random. Only when the associated loss is small does it appear that we would even consider such an approach. We equate this to the type of performance guarantees often seen paired with heuristic methods, and observe that good average return from a method which 'on average' performs a certain way is unsurprising. We vehemently believe that this is insufficient and masks the key concerns levelled here. In the next section we move towards more traditional testing, but take special care to record the variability of results when considering large scale test beds and ensure that more challenging shorter term problems are also investigated.

7.6 Performance Comparison

In order to test the effectiveness of different decision strategies, we introduce a suite of test problems similar to that used in Sutton and Barto (1998). The suite consists of a set of $10,000$ randomly generated Bernoulli reward MAB problems, each with $K = 10$

actions. For each action a_j, the associated reward probability is drawn from a Beta distribution $\lambda_j \sim \text{Beta}(4, 4)$, which is resampled for each of the 10,000 problems. The parameters of this distribution are chosen to offer a balance between easy problems in which the reward probabilities are well separated, and harder problems in which they are close together. In keeping with Sutton and Barto (1998), we also refer to this experimental setup as the *10-armed test bed*.

The decision methods we test in this section give a fair representation of the different types of methods available. They are Gittins Index, ϵ-Greedy, Boltzmann exploration or Softmax, Thompson sampling and UCB policies. We do not present the optimal DP strategy, as the calculation for the test bed is not computationally feasible. Several of these methods require parameters to be set which ultimately affect performance, so it is necessary to test a range of values. For the Gittins Index the only parameter that can be changed is the level of discounting; we use $\gamma = 0.85$, 0.95 and 0.999. Within the heuristic set of methods, namely ϵ-Greedy and Softmax policies, we test the following parameter settings: $\epsilon = 0$, 0.05, 0.1 and 0.2; and $\tau = 0$, 0.02 and ∞. For Softmax, we additionally test a cooling schedule where at time t the parameter $\tau = 1/t$. For the remaining methods we use the vanilla TS algorithm and the original UCB1 formulation given in sections 7.4.4 and 7.4.1 respectively.

7.6.1 An Infinite Horizon Problem

Our first test is designed to assess performance on an infinite horizon type task. As we cannot run the test indefinitely, we look at performance over the first 1000 actions. Table 7.1 presents the average return and the average pseudo-regret for each of the methods. The average return is calculated by averaging the total return received after 1000 actions across each of the $10,000$ decision problems. We look at pseudo-regret, which is the form used in the majority of MAB literature (see for example Brezzi and Lai (2002)), as it allows us to remove the stochasticity of the reward distribution itself

Method		Return			Pseudo-Regret		
		Mean	σ	Median	Mean	σ	Median
Gittins	$\gamma = 0.85$	704.00	98.39	706	47.80	62.27	14.82
	$\gamma = 0.95$	715.09	94.27	717	36.63	49.44	12.39
	$\gamma = 0.999$	722.27	89.29	723	29.47	20.71	24.21
ϵ-Greedy	$\epsilon = 0$	693.68	112.86	702	58.06	77.18	14.33
	$\epsilon = 0.05$	700.79	91.86	702	50.92	45.26	29.85
	$\epsilon = 0.1$	694.92	85.21	697	56.71	34.61	44.62
	$\epsilon = 0.2$	674.26	76.82	676	77.28	26.13	72.76
Softmax	$\tau = 0$	692.98	112.19	700	58.77	78.21	14.77
	$\tau = 0.02$	702.08	106.39	709	49.58	66.49	16.86
	$\tau = \infty$	499.06	55.47	500	252.65	68.80	248.12
	$\tau = 1/t$	699.93	107.57	704	51.56	69.54	12.79
Thompson		703.00	89.23	703	48.72	17.62	45.77
UCB1		622.14	71.16	622	129.76	24.58	131.62

Table 7.1: Performance of decision methods on the 10-armed test bed after 1000 actions. The top performing parameter setting in each method class is highlighted.

from our analysis and thus provides a comparison between runs that is complimentary to the return. Finally we have highlighted the top performing parameter settings in each class and plotted box-and-whisker plots for the return and pseudo-regret for each of these in Figure 7.15. These plots show the median, mean, quartiles and range of the return and pseudo-regret across the problem set.

Our original objective was to maximise the return (equivalent to minimising the regret). It is therefore sensible that we look at the average return, given by the mean, as our primary performance measure. Hopefully it should be unsurprising that in this regard, the highest performance of all methods is that of Gittins Index action selection. In fact, all three discount settings perform better than any of the other methods we tested. Of these, the setting with the least amount of discounting, corresponding to $\gamma = 0.999$, performs the best. This is in part because the extent to which the discounting curtails the problem length is reduced, and also because the effective horizon given by Equation (7.2) is equal to 1000 which is the number of actions in each decision problem and is therefore well matched. We will investigate

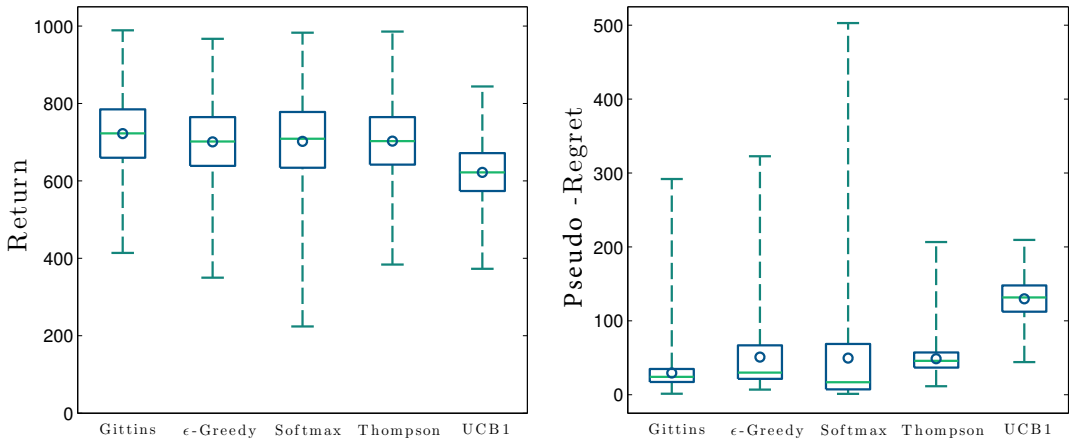


Figure 7.15: Box-and-whisker plots for the return and pseudo-regret of the top performing methods in each class on the 10-armed test bed after 1000 actions. The mean is given by a circle, the quartiles by a box, the median by a horizontal line within each box, and the full range of all the data by the whiskers.

later the relationship between discounting and performance when the problem length is reduced.

The performances of the heuristic methods are fairly close to one another, except for UCB1, which is substantially below that of the others. The performance of UCB1 highlights a potential problem with any bandit algorithm test bed regarding the complexity and horizon of the problem. As we have already discussed, UCB1 has favourable bounds on the regret and will therefore converge to a good solution, yet is not particularly well tuned to our test bed both in regards to the number of actions available and the number of actions we are evaluating over. It would be possible to improve the performance of this algorithm by adjusting the upper bound parameters to better suit this problem, although we do not do this here. This is because our objective is to better understand the nature of different decision methods and indeed to demonstrate the difficulties that can arise when heuristic parameter settings are involved, rather than generate an algorithm that will solve a particular MAB problem.

Whilst the mean is a useful measure of performance, observing the distribution

gives greater insight into the reliability of each decision method. This can be done by either looking at the σ values in table 7.1 or the box-and-whisker plots in Figure 7.15. We can see from these that whilst the distribution of return is relatively similar for all the tested methods, there is greater difference in the regret. Notably, the regret for the heuristic methods, excluding Thompson sampling, is substantially less clustered around the mean than the two other methods. Audibert et al. (2009) note that within a fixed horizon the distribution of the regret is very poorly concentrated around its expectation; however, we should note that this is in part a consequence of using the empirical mean as an estimate of the reward probabilities. Using this approach, each arm must be sampled once at the start of the problem, which occasionally results in the best arm being undervalued and a sub-optimal arm being overvalued. In such a scenario, there is only a small probability that the best arm will be resampled such that its value estimate may improve, leaving it likely that the sub-optimal arm continues to be chosen and the error perpetuated. Even if this happens with low probability, the regret is significant, being of $\mathcal{O}(t)$ rather than $\sim \mathcal{O}(\log(t))$. We do not see such a strong effect when using a Bayesian estimation of the reward probabilities, such as that used by Gittins and Thompson sampling, as the prior distribution serves to prevent low numbers of observations from unduly impacting the value of an action.

At this point it is sensible to establish the confidence of the observed results, and at the same time ensure that the $10,000$ problems in the 10-arm test bed provide enough experimentation to give such confidence. One means of doing this is via hypothesis testing, such as a difference of means or t-test. In this context, a t-test allows us to test whether there is a significant difference between the mean of two sample populations. Confidence in there being a difference is demonstrated by a *p-value* which represents the probability of *at least* the observed difference in the mean, if the means were actually the same (the null hypothesis). Any value of p below 0.05 is generally accepted as statistically significant. If we compare the observed returns of

the top two performing methods given by the top two Gittins Index methods, we find $p = 3.25 \times 10^{-8}$ which represents a highly significant result. Comparing the top Gittins method with the highest performing heuristic method we find $p = 1.29 \times 10^{-47}$, so we should have a high degree of confidence in both there being enough experimentation, and in the top method being Gittins with $\gamma = 0.999$. One potential problem with using a t-test is that it assumes the data is Gaussian distributed. Given what was said in the last paragraph and observation of the long tails on the box-and-whisker plots, this assumption is not appropriate when analysing the regret. An alternative to the t-test is the nonparametric Kolmogorov-Smirnov test (K-S test) which makes no assumptions about the underlying distributions. Using this approach, $p \simeq 0$ (below precision) for the regret comparison between both the top two Gittins methods, and between the top Gittins and heuristic methods. This further confirms confidence in both the results and amount of experimentation.

In Figure 7.16 we can see how the algorithms behave as they gain more experience over time. The figure shows plots of the average (instantaneous) pseudo-regret $(\mu^* - \mu_j(t))$, the average reward, and the percentage of actions that were assigned to the most rewarding (best) action. Averaging is performed across the $10,000$ problems at each time step to provide an indication of how performance improves with learning. Note that the additional noise in the reward curve is a result of averaging over the actual received rewards, rather than an expected value as with the regret curve. A performance increase is associated with a reduction in the average pseudo-regret, and increases in the average reward and percentage allocation to the best action.

All of the methods presented show increased performance with experience, although there are clear differences between the rate this happens and the point at which their performances level off. We can clearly see that Gittins action selection performs better than all other methods for the majority of the problem; however it is interesting to observe that both ϵ-Greedy and Softmax initially do better. Both of

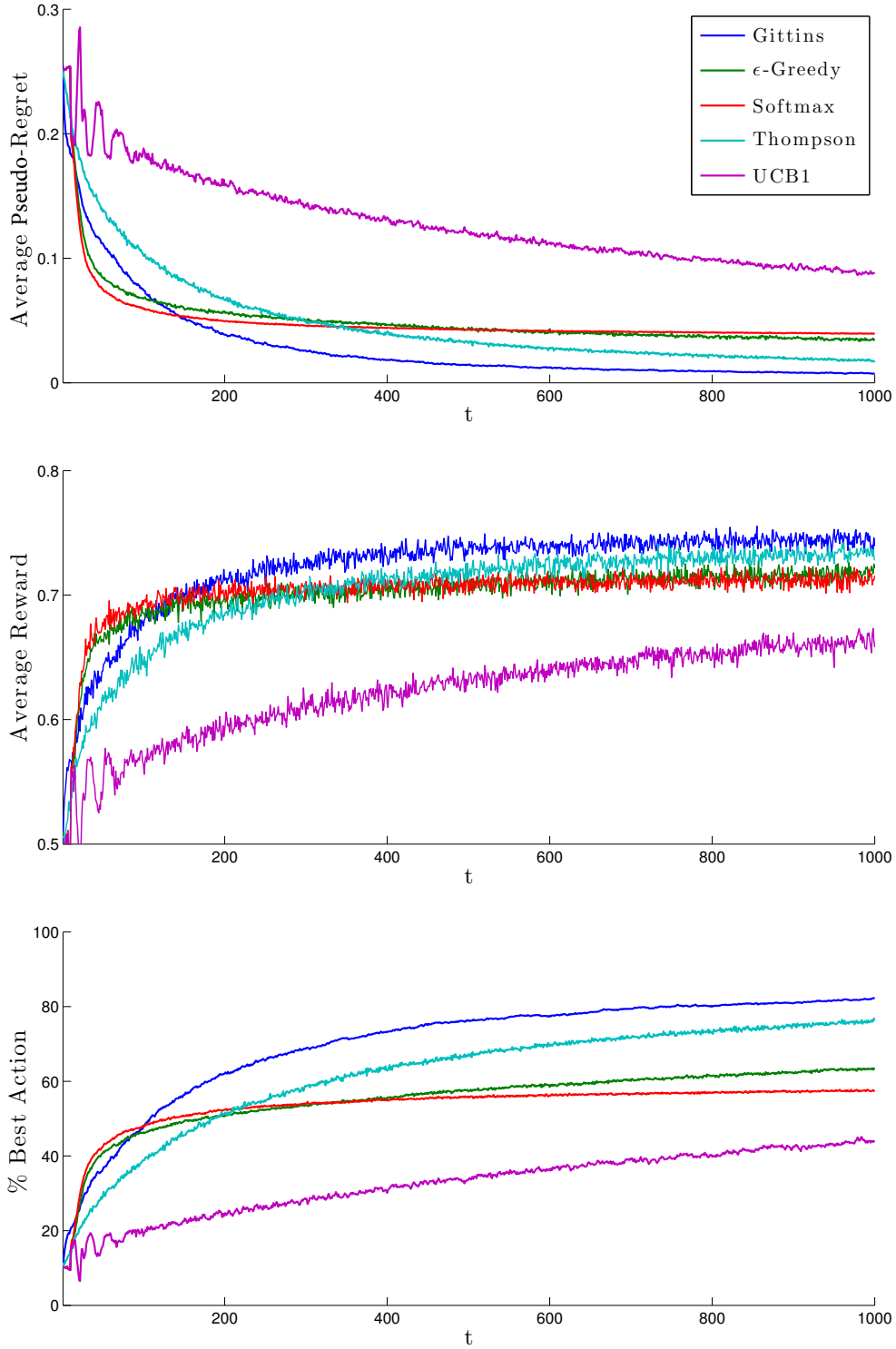


Figure 7.16: Performance over time of the top performing methods in each class on the 10-armed test bed.

these methods are inherently more myopic; as such while other actions are allocating actions to exploration they are to a greater extent maximising immediate rewards based on the information they have available. The penalty for this type of behaviour is an inability to do well in the long run, due to a lack of information about actions that initially appear to be less rewarding. It actually takes relatively little time for the Gittins method and then Thompson sampling to perform better than these strategies, so it is not difficult to see the limitations of this approach. Of course, early exploration does not completely solve the problem as it is important for a decision method to be able to exploit knowledge once it is available, hence the reason for describing a 'tradeoff' in the first place. UCB1, for example, is highly exploratory and therefore likely possesses the highest degree of information about *all* actions; yet it does not move to exploitation fast enough to perform well within the analysed time period.

We feel that the most important part of this analysis to note is that the two methods that rely on a Bayesian estimation of value and that also do not require heuristic parameter setting or optimisation, namely Gittins and Thompson, perform substantially better than all other methods. In fact, at the heart of Thompson sampling, there is very little other than Bayesian estimation, yet its performance eclipses other more complicated-to-explain methods. As Chapelle and Li (2011) note, its simplicity and therefore ease of execution should make TS a standard baseline approach for comparison.

All of the methods shown in the figure continue to show a positive increase in performance at the end of the 1000 steps analysed, and there is an argument for continuing the analysis until all the methods have levelled off. If we look at ϵ-Greedy, for example, eventually we would expect this method to asymptote in performance and select the best action $(1-\epsilon)\times 100\%$ of the time. Also, Thompson sampling should actually converge to selecting the best action almost always, with only a negligible probability of selecting a less rewarding arm. These are predictable outcomes of

continuing the analysis though, so it is sensible to concentrate on the more complex part of the problem, where the interplay between exploration and exploitation is most noticeable and best examined.

7.6.2 Heuristic Parameter Sensitivity

The results presented so far indicate that the performance of certain heuristic methods can be quite sensitive to the value of their parameters. This is problematic, as these parameters can only be set by hand based on experience with the algorithms and the problem domain. In Figure 7.17 we further investigate the challenge this generates and plot the test bed performance of both ϵ-Greedy and Softmax methods using various parameter settings. For comparison, we also plot these results against the best performing decision method given by Gittins action selection.

The chosen parameter in both methods has a clear effect on both the rate of improvement and the asymptotic properties of performance. Focussing in on ϵ-Greedy, we can see that there is a non-trivial relationship between the value of ϵ and the various performance measures. Predictably the most greedy setting results in good early but poor long term performance; yet when ϵ is too large we find that the performance suffers across the whole problem. Additionally, the value of ϵ that gains the highest *return* finishes choosing the best action less often than another more exploratory value. For Softmax the optimum value of τ lies somewhere between 0 and ∞, but there is no way to determine this from theory. As with ϵ-Greedy, a cooling schedule can be used to try and ensure adequate exploration followed by exploitation in the long-term; however this can also increase the complexity of the solution and the number of parameters to optimise. As an example, we can see that the $\tau = 1/t$ cooling schedule performs reasonably well, although the decay rate is ultimately too fierce as $\tau = 0.02$ performs better in the long term. In principal a modified decay parameter could improve performance further.

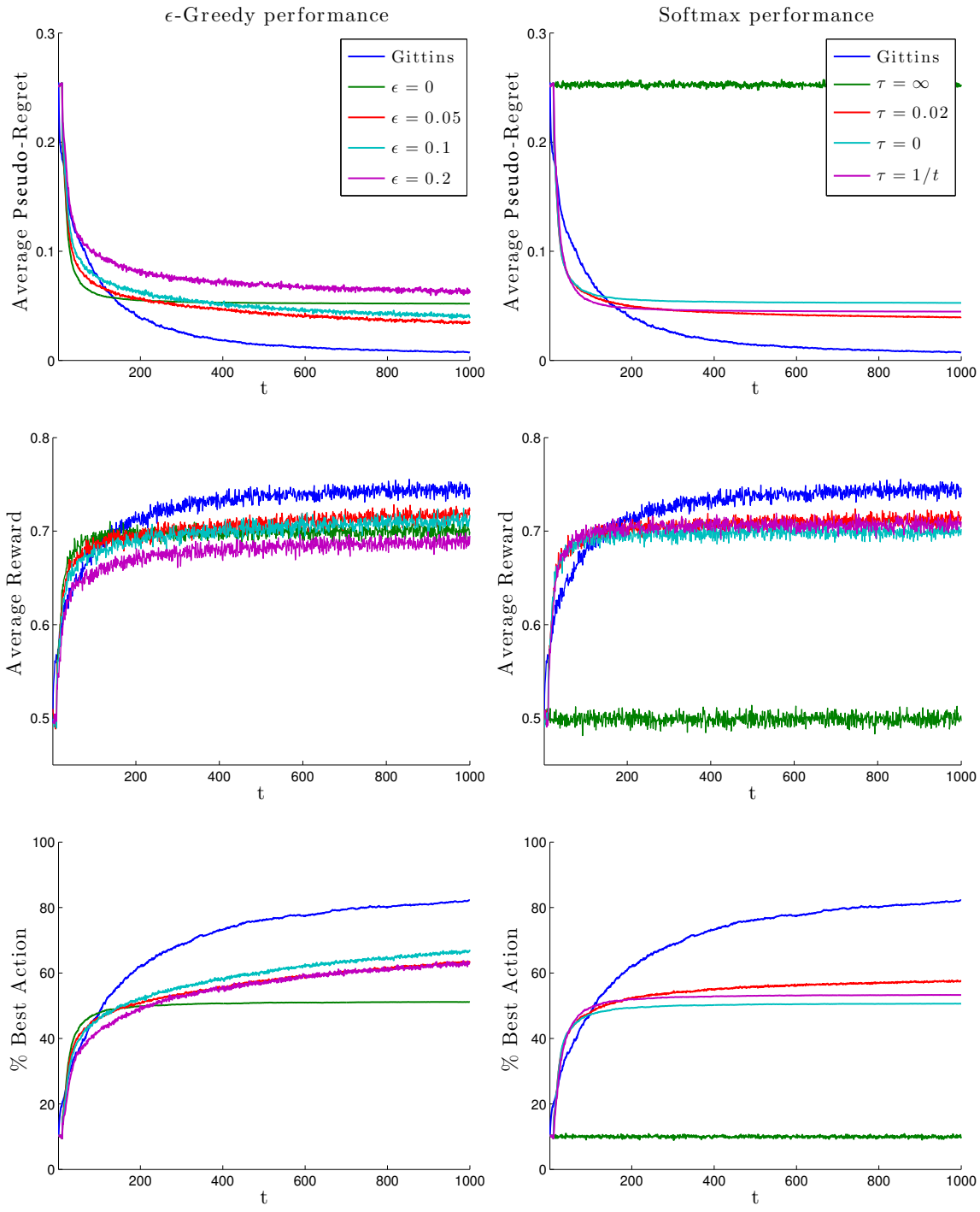


Figure 7.17: A diagram illustrating the difficulties associated with heuristic parameter setting. The performances of both ϵ-Greedy and Softmax are sensitive to the value of heuristic parameters that must be set by hand.

7.6.3 The Effect of Discounting on Performance

Given that we have critiqued the use of heuristic parameters in certain decision methods, in the interest of fairness we must also discuss the effect discounting has on the

Gittins Index solution. Table 7.1 demonstrated that the level of discounting does affect the overall performance of the algorithm, and we extend this analysis in Figure 7.18 as we did with the heuristic methods. Note that from here onwards we no longer include plots of average reward, as they do not add more information to the analysis.

Earlier in the chapter we showed how increased discounting, which corresponds to a lower value of γ, causes the decision algorithm to become increasingly myopic. Reiterating, this is due to a decreased importance in later rewards, which can be characterised as a curtailing of the problem length towards the effective horizon given by Equation (7.2). Looking at Figure 7.18, we can see that as discounting increases the algorithm focusses more on maximising immediate rewards, in much the same way as the previously analysed greedy methods did. Once again, this results in less good long term performance.

As discounting increases, we should also note from the box-and-whisker plots that the regret becomes less concentrated around the mean in the same way as we saw with certain methods in Section 7.6.1. There we diagnosed this behaviour as being in part due to the use of an empirical instead of Bayesian estimator on value. This caused the decision methods to occasionally become stuck taking a suboptimal action, perpetuating an $\mathcal{O}(t)$ error. Here the cause is not poor estimation, but rather the increasingly myopic analysis of the action space. By curtailing the length of the problem, the algorithm does not value exploration in the same way and can end up pursuing over-valued actions in an analogous manner.

Although the spread of regret is proportional to the amount of discounting and the lowest mean regret corresponds to the method with the least amount of discounting, the lowest median appears when discounting is increased. This can be seen by looking at the histograms of observed regret in Figure 7.19, where the distribution of regret becomes increasingly skewed as discounting increases. This result is particularly interesting as it demonstrates that, whilst the lowest level of discounting

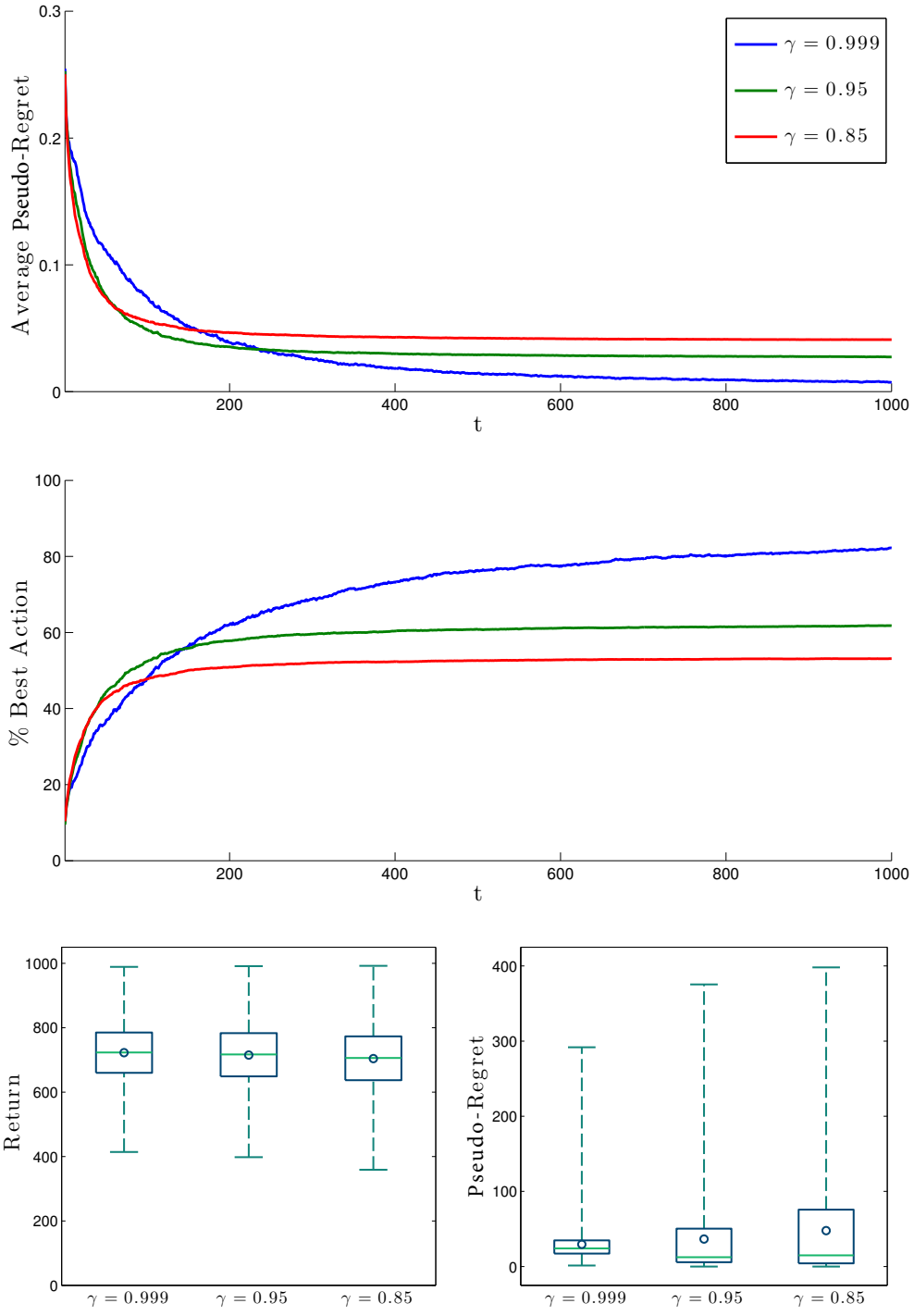


Figure 7.18: A diagram illustrating the effect of discounting on performance of Gittins action selection. As discounting increases through a reduction in γ the algorithm becomes more myopic, resulting in better early performance at the cost of good long term performance.

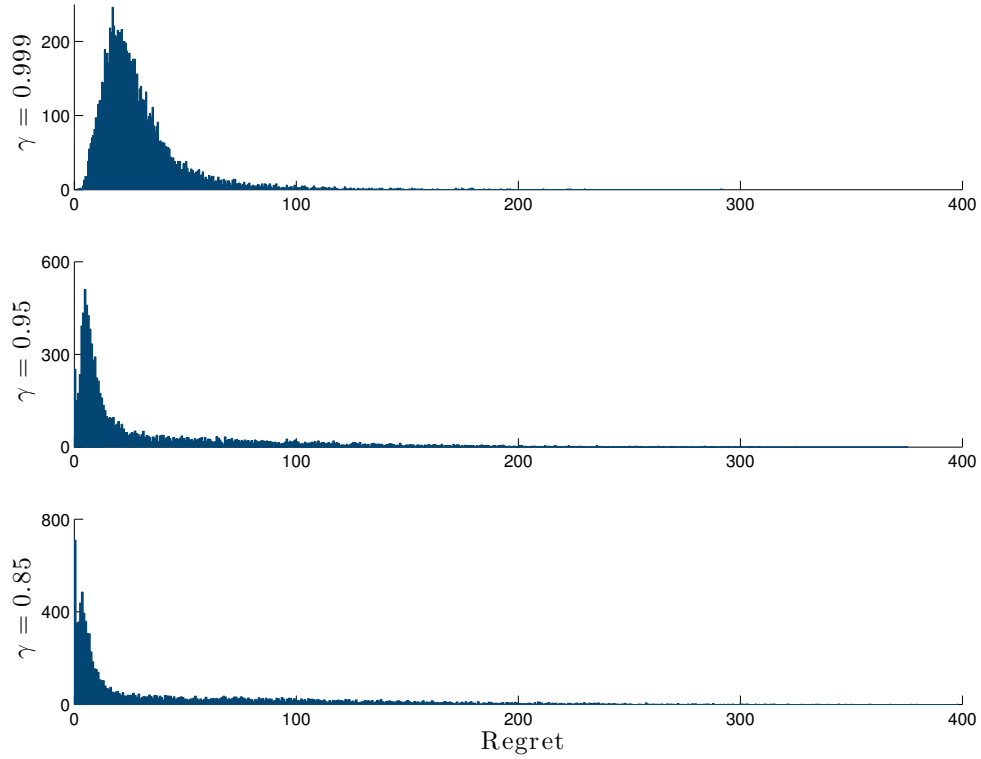


Figure 7.19: Histograms of regret using Gittins action selection on the test bed with different discount values.

corresponds to the least *mean* regret, it is actually possible to achieve better performance (on a case by case basis) with greater discounting. This is easy to understand if we think about completely greedy behaviour. Occasionally the greedy algorithm will end up correctly valuing the best action above all others early in the problem, and as such continue to take that action when other methods would continue to allocate actions to exploration. Hitting upon the best action in this way thus achieves the minimum pseudo-regret. Conversely, if the method chooses the wrong decision, then significant errors can propagate, as already discussed. This principle of 'boom or bust' also explains the bi-modality of the regret distribution seen in the lower histogram plots, and can be linked to our discussion on high risk and conservative strategies in Section 7.5.2. Finally, observation of the regret histogram plots should confirm our suspicion that the regret distribution is not well represented by a Gaussian, and that the K-S test is the most appropriate means of testing statistical significance on this

data.

We have seen the ways in which the value chosen for discounting can affect performance, and we must now determine how it should be set. Thinking about infinite horizon problems, the smaller the amount of discounting we have, the further we look forward in the problem and the better our solution becomes. In theory we would like to set γ as close to 1 as possible; however the closer we get to 1 the more computationally difficult the solution is to find. In the context of the GI, it takes an increasing number of steps for the algorithm described in Section 6.4.2 to converge. If γ is actually set to 1 then the GI would asymptote to 1 for all actions regardless of the information state, which is equally unhelpful. Given this, a reasonable approach is to set γ to as high a value as is computational feasible. We should note that this differentiates discounting from heuristic parameters as we can precisely describe the optimal setting; we are only limited in doing this by the computational resources available.

For finite horizon problems the discount parameter is not required; however we will see in the next section that it can be used to produce novel decision methods with highly favourable performance.

7.6.4 Finite Horizon Problems

Not all interesting problems are those with very long horizons, and in fact the most complex part of the decision problem is often within the early stages. Further, once decision problems become dynamic, the need for good performance over short time scales is even more important. Traditional analyses of heuristic decision methods that look to long term asymptotic performance over infinite horizon problems thus avoid the most intriguing and demanding parts of the decision problem. For these reasons, we view a finite horizon test bed as a critical tool of analysis.

In this section we once again use the 10-armed test bed, but now evaluate

Method		Return			Pseudo-Regret		
		Mean	σ	Median	Mean	σ	Median
Gittins	G^N	136.96	20.85	137	13.43	12.42	9.13
	$\gamma = 0.85$	136.62	21.05	137	13.81	13.29	9.15
	$\gamma = 0.95$	137.63	20.57	138	12.87	10.83	9.30
	$\gamma = 0.999$	133.20	20.18	133	17.25	7.44	16.10
ϵ-*Greedy*	$\epsilon = 0$	134.15	22.77	136	16.05	15.42	9.20
	$\epsilon = 0.05$	133.51	20.36	135	16.85	12.44	12.18
	$\epsilon = 0.1$	131.77	18.97	132	18.73	10.99	15.21
	$\epsilon = 0.2$	128.07	17.14	129	22.28	9.23	20.28
Softmax	$\tau = 0$	134.09	22.71	136	16.19	15.34	9.49
	$\tau = 0.02$	134.71	22.18	136	15.73	14.28	10.23
	$\tau = \infty$	99.92	12.75	100	50.55	14.09	49.53
	$\tau = 1/t$	135.26	21.30	136	15.18	13.21	9.70
Thompson		127.25	18.20	127	23.13	7.39	22.09
UCB1		113.11	13.50	114	37.48	9.55	36.86

Table 7.2: Performance of decision methods on the finite horizon 10-armed test bed with 200 actions. The top performing parameter setting in each method class is highlighted.

performance over specific short term horizon problems. The range of horizons we look at hopefully captures problems bridging the gap between the long 'approximately infinite' horizon analysed in the last section and the very hardest problems in which the number of available time steps approaches the number of available actions. We look at the same set of algorithms as we did in the last section, as well as adding the Gittins approximation G^N discussed earlier in this chapter. When using G^N the value of N is always set to the number of actions left to take; thus it starts equal to the horizon and ends equalling 1. The three lengths of problem we look at here are $\mathcal{H} = 200, 100$, and 40. We begin with $\mathcal{H} = 200$ and present the results in Table 7.2. The box-and-whisker plots for the top performing methods in each class can be seen in Figure 7.20, and the time dependent performance graphs in Figure 7.21.

On the infinite horizon problem greedier methods did better in the short term, and indeed here we can see that those methods do perform better over the 200 available actions than other, more exploratory actions. This reconfirms our belief that

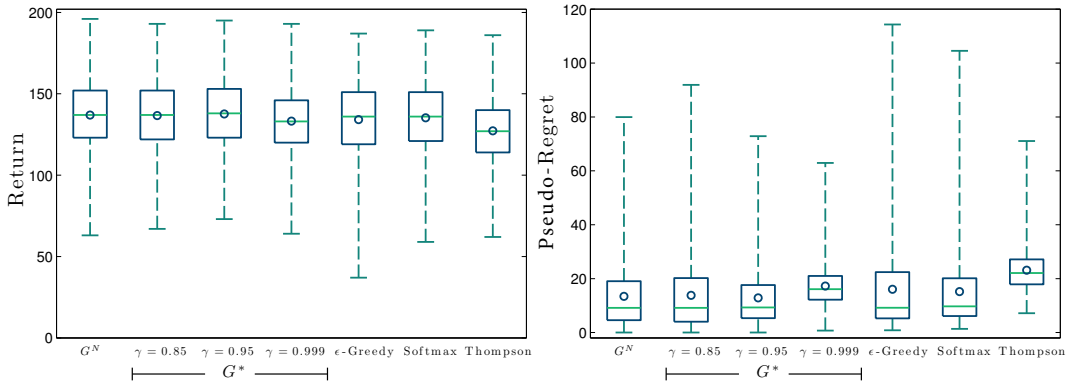


Figure 7.20: Box-and-whisker plots for the return and regret of selected methods from each class on the 10-armed test bed after 200 actions.

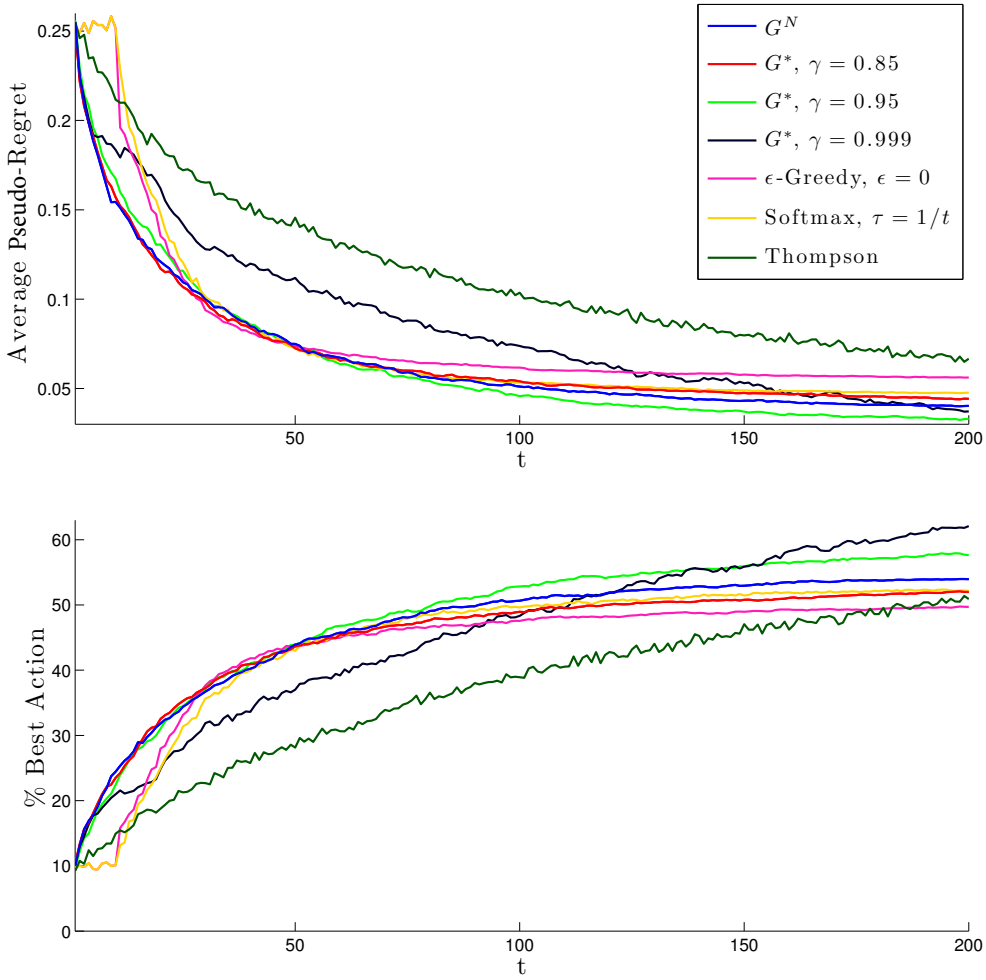


Figure 7.21: Performance over time of selected methods from each class on the 10-armed test bed with $\mathcal{H} = 200$.

exploration should be reduced as the horizon of a problem shortens. With regards to the Gittins Index, the highest performance comes when using a discount setting of $\gamma = 0.95$, which is less than the previous setting. Using Equation (7.2) this represents an effective horizon of only 20 steps which is substantially smaller than the true horizon, although potentially better matched than the effective horizon of 1000 steps found when $\gamma = 0.999$. Do note from Figure 7.21, however, that by the end of the problem $\gamma = 0.999$ takes the best action more often than all other methods due to continued early exploration.

Moving on to the heuristic methods, we see that the best setting for the ϵ-Greedy strategy is $\epsilon = 0$, which corresponds to acting completely greedily. This is surprising due to both the naivety of the method and the fact that the problem is still of modest length. Softmax performs best using the cooling schedule of $\tau = 1/t$, which became exploitative too quickly in the longer problem. This demonstrates that there is still a need to explore to some extent; although we note that the performance difference between this and the purely greedy strategy is relatively small. At this scale we should also note the sudden increase in performance of the ϵ-Greedy and Softmax methods after 10 time steps, which corresponds to the initial sampling of all actions required for empirical estimation of value.

Out of the remaining methods, UCB1 predictably exhibits the worst performance due to its high level of exploration. As it performs considerably below that of the other methods, we have not included it in the figures. Finally, Thompson sampling also performs badly over this shorter time period; Figure 7.21 demonstrates that it is too exploratory early on in the problem. In the vanilla form of the method, the sampling is not modified by the length of the problem, and as such the posterior distributions remain sparse across the action space. This behaviour enables very good asymptotic behaviour, but is clearly not suitable in shorter problems. Should the number of actions increase in the problem, then the uncertainty in the posterior

distributions would only increase further.

Once again we can examine the significance of these results by looking at the t-test when comparing returns and the K-S test for pseudo-regrets. Firstly let us look at the highest performing Gittins (using $\gamma = 0.95$) against a number of other methods. The next best method is given by G^N, shortly followed by Gittins with $\gamma = 0.85$, and these respectively provide values of $p = 0.022$ and $p = 6.15 \times 10^{-4}$ for the return, and $p = 1.52 \times 10^{-15}$ and $p = 1.28 \times 10^{-42}$ for the pseudo-regret. This demonstrates a reasonably close performance with G^N in terms of return, although below the 0.05 significance threshold. Comparing the top Gittins method with the top heuristic method given by Softmax yields a value of $p = 1.39 \times 10^{-15}$ for return and $p = 7.48 \times 10^{-42}$ for the pseudo-regret. Despite the reduced horizon of the problem, the number of experiments provides high confidence in the results. We can also confirm this by comparing the two purely greedy methods given by ϵ-Greedy with $\epsilon = 0$ and Softmax with $\tau = 0$. These methods are theoretically the same, although have slightly different returns and pseudo-regrets due to reward noise; their p-values however are $p = 0.87$ for the return and $p = 0.053$, both above the significance threshold.

In Table 7.3 we present the results for the 10-armed test bed evaluated over 100 actions. The related box-and-whisker and time dependent performance plots are presented in Figures 7.22 and 7.23. We see here that Gittins Index selection performs best when the discount is $\gamma = 0.85$, which equates to an effective horizon of between 6 and 7 steps. Whilst this shows that as the horizon of the problem decreases the level of discounting should increase, the level at which it should do so is greater than one may think. It seems that simply setting the effective horizon to equal the true horizon for the duration of the problem does not provide us with the most effective decision method. To understand this, let us first reaffirm that decision making according to Gittins Index selection is *not* optimal in the finite horizon case.

Method		Return			Pseudo-Regret		
		Mean	σ	Median	Mean	σ	Median
Gittins	G^N	66.20	11.34	66	8.88	6.95	7.06
	$\gamma = 0.85$	66.34	11.22	66	8.99	7.28	7.13
	$\gamma = 0.95$	66.19	11.05	66	9.11	6.27	7.68
	$\gamma = 0.999$	63.10	10.56	63	11.98	4.74	11.31
ϵ-*Greedy*	$\epsilon = 0$	64.76	10.99	66	10.33	7.44	7.74
	$\epsilon = 0.05$	64.16	10.33	65	11.07	6.75	8.99
	$\epsilon = 0.1$	63.49	9.70	64	11.81	6.06	10.20
	$\epsilon = 0.2$	61.60	9.12	62	13.58	5.53	12.55
Softmax	$\tau = 0$	65.01	11.00	66	10.20	7.13	7.77
	$\tau = 0.02$	64.98	10.93	66	10.19	6.91	8.01
	$\tau = \infty$	50.15	7.24	50	25.16	7.11	24.62
	$\tau = 1/t$	64.84	10.57	66	10.37	6.48	8.34
Thompson		60.38	9.23	60	14.87	4.77	14.25
UCB1		54.73	7.36	55	20.43	5.55	20.08

Table 7.3: Performance of decision methods on the finite horizon 10-armed test bed with 100 actions. The top performing parameter setting in each method class is highlighted.

The only known optimal solution is the intractable solution to the full decision tree, which can be found using Dynamic Programming or an alternative iterative method. It is therefore not surprising that Gittins Index selection does not perform predictably in this setting as it is an approximation to the solution. Nevertheless, it seems that, as the problem length reduces, the most effective algorithms are much more myopic than the effective horizon would suggest. In fact, purely greedy behaviour, which as a comparison both ϵ-Greedy and Softmax favour, proves hard to beat.

The reason for myopic behaviour being so favourable is undoubtedly related to the relationship between the number of actions and the length of the problem. This is not represented well by Gittins Index selection, which calculates the value of each action independently. For long problems this is less of a problem; however as the horizon shrinks towards K the coupling between the value of each action becomes much stronger. This must ultimately result in Gittins Index selection becoming less effective as the interaction between actions further violates the independence assump-

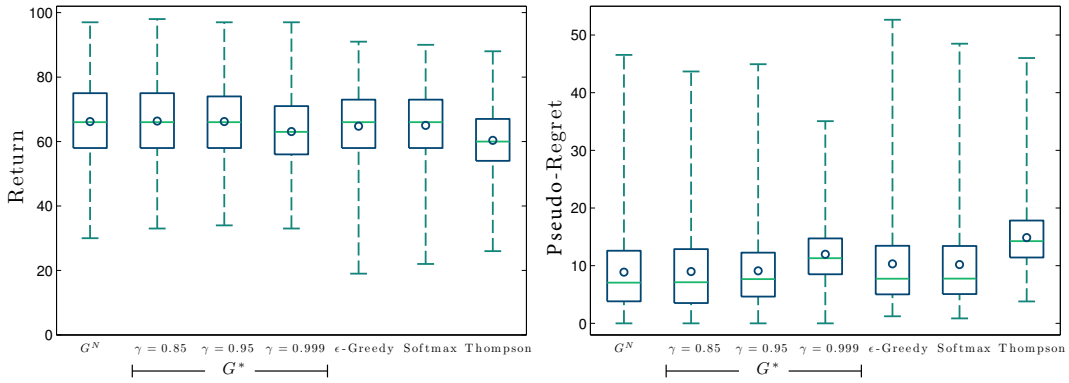


Figure 7.22: Box-and-whisker plots for the return and regret of selected methods from each class on the 10-armed test bed after 100 actions.

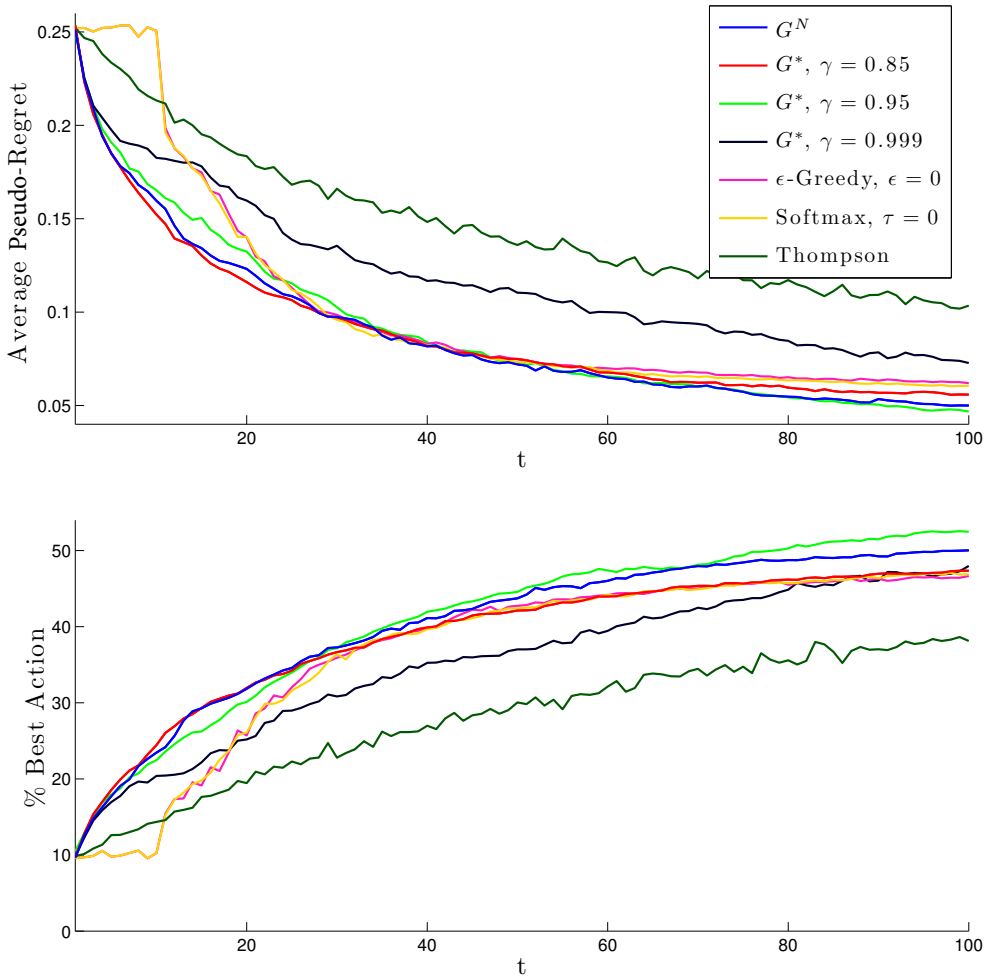


Figure 7.23: Performance over time of selected methods from each class on the 10-armed test bed with $\mathcal{H} = 100$.

tion. Of course, the heuristic methods do not model the interaction between actions and value either; however this is the same for all problem lengths.

There may also be another contributing factor behind the drop in performance of Gittins Index selection as the horizon recedes. Given that the discount is constant throughout the problem, the effective horizon does not reduce along with the number of actions left to take. This means that, when the last action is reached, a decision is made as if there is more than one step left. A potential solution would therefore be to increase the level of discounting along the problem to match the actual number of actions available, not just the overall horizon of the problem. We test this theory in the next section, after examining one more finite horizon problem.

Testing the significance of these results, we find that the top reported method of Gittins with $\gamma = 0.85$ is actually very close to that of G^N with a return p-value of $p = 0.38$. The pseudo-regret does, however, yield a p-value of 1.78×10^{-4}. Comparison with the closest heuristic method yields a p-value of $p = 2.62 \times 10^{-7}$ for return and $p = 1.21 \times 10^{-119}$ for pseudo-regret, which clearly places the Gittins based methods considerably above the best heuristic methods.

As we move to the shortest problem length of $\mathcal{H} = 40$, the number of actions that can be taken is very close to the number of arms available to choose from, $K = 10$. Intuitively, one would think that for the most part only a very small subset of actions should be sampled from, rendering methods like Thompson sampling particularly inefficient. The violation of the Gittins independence property should also become even more significant. Table 7.4 and Figures 7.24 and 7.25 present the results for this problem.

Unsurprisingly the heuristic methods favour completely greedy decision making, as they did for the last problem. Likewise, Thompson and UCB1 are highly ineffective over such a short horizon. The best value of discounting for Gittins Index selection remains at $\gamma = 0.85$, although we do not test a lower parameter value. Pleasingly, the

Method		Return			Pseudo-Regret		
		Mean	σ	Median	Mean	σ	Median
Gittins	G^N	24.88	5.22	25	5.17	3.34	4.82
	$\gamma = 0.85$	24.90	5.11	25	5.22	3.28	4.88
	$\gamma = 0.95$	24.65	5.11	24	5.48	2.98	5.21
	$\gamma = 0.999$	23.62	4.61	24	6.45	2.44	6.16
ϵ-*Greedy*	$\epsilon = 0$	23.86	4.26	24	6.21	2.63	5.75
	$\epsilon = 0.05$	23.61	4.06	24	6.46	2.58	6.04
	$\epsilon = 0.1$	23.31	4.01	24	6.75	2.52	6.38
	$\epsilon = 0.2$	22.72	3.79	23	7.31	2.47	7.02
Softmax	$\tau = 0$	23.95	4.17	24	6.16	2.65	5.67
	$\tau = 0.02$	23.94	4.16	24	6.16	2.60	5.69
	$\tau = \infty$	19.53	3.79	20	10.08	2.90	9.88
	$\tau = 1/t$	23.66	4.11	24	6.43	2.45	6.04
Thompson		22.51	4.23	23	7.56	2.42	7.31
UCB1		21.23	3.77	22	8.83	2.44	8.66

Table 7.4: Performance of decision methods on the finite horizon 10-armed test bed with 40 actions. The top performing parameter setting in each method class is highlighted.

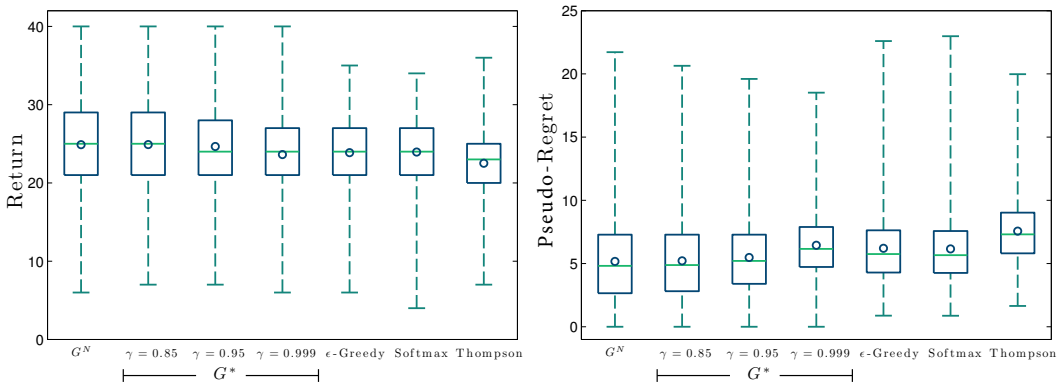


Figure 7.24: Box-and-whisker plots for the return and regret of selected methods from each class on the 10-armed test bed after 40 actions.

Gittins Index approximation performs about as well as the best tested configuration, yielding a comparative p-value of $p = 0.87$ for return and $p = 0.02$ for pseudo-regret. This means G^N has performed consistently well across the whole range of problem lengths, unlike pure Gittins Index selection whose performance can drop off if the level of discounting is not well matched to the horizon.

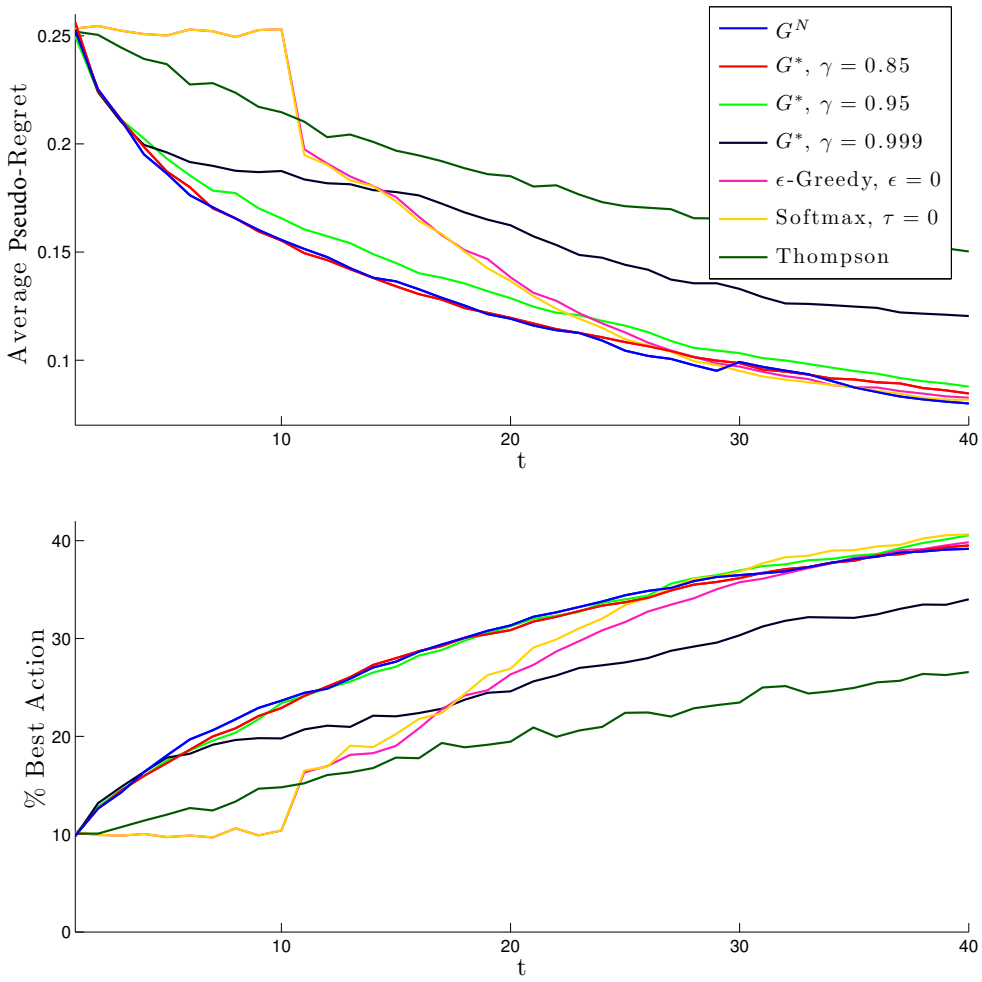


Figure 7.25: Performance over time of selected methods from each class on the 10-armed test bed with $\mathcal{H} = 40$.

7.7 Variable Discounting

Our analysis of finite horizon methods using the Gittins Index led us to a question regarding the use of variable discounting. As was shown, if fixed throughout the problem, the most effective setting for the discount parameter represents an effective horizon that is much shorter than the true horizon of the problem. In effect, a more myopic approach performs better than one would think. An explanation for this could be the fact that fixing the discount throughout the problem results in decisions at the start being discounted in the same way as decisions at the end. As we mentioned in

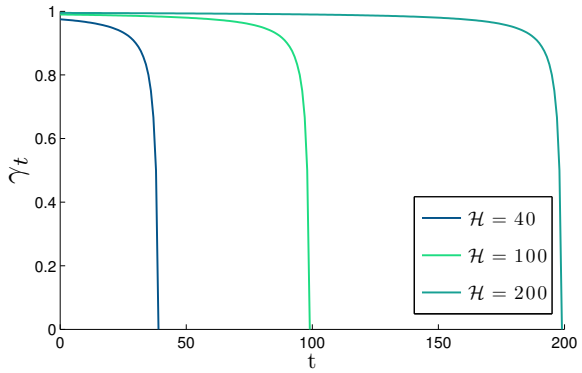


Figure 7.26: Plot of γ_t over time for each of the three finite horizon problems.

the last section, a better approach may therefore be to alter the discount parameter so that the effective horizon is equal to the number of actions left at *each point in the problem*, $\mathcal{H}_t = \mathcal{H} - t$, rather than the total horizon of the problem.

The new policy, which we will call variable discounted Gittins or G^γ, is defined as follows:

$$\pi_{G^\gamma}(t) = \underset{j \in K}{\text{argmax}}\ G^*_j(\gamma_t) \tag{7.37}$$

where the discount at time t, γ_t, is found by rearranging Equation (7.2):

$$\begin{aligned} \mathcal{H}_t &= \frac{1}{1-\gamma_t} \\ \gamma_t &= \frac{\mathcal{H}_t - 1}{\mathcal{H}_t} \end{aligned} \tag{7.38}$$

Figure 7.26 shows how the discount parameter varies over time for each of the three finite horizon problems tested in the last section. We test G^γ on the 10-armed test bed and compare the results against the other tested Gittins based methods. Table 7.5 shows the performance of these methods on the three finite horizon problems. The relevant box-and-whisker plots are presented in Figure 7.27, and the time dependent performance graphs in Figure 7.28.

Overall, the new decision method performs consistently well on the three differ-

Method	Return			Pseudo-Regret		
	Mean	σ	Median	Mean	σ	Median
$\mathcal{H} = 40$						
G^γ	24.97	5.19	25	5.12	3.35	4.72
G^N	24.88	5.22	25	5.17	3.34	4.82
$\gamma = 0.85$	24.90	5.11	25	5.22	3.28	4.88
$\gamma = 0.95$	24.65	5.11	24	5.48	2.98	5.21
$\gamma = 0.999$	23.62	4.61	24	6.45	2.44	6.16
$\mathcal{H} = 100$						
G^γ	66.35	11.14	66	8.91	6.86	7.41
G^N	66.20	11.34	66	8.88	6.95	7.06
$\gamma = 0.85$	66.34	11.22	66	8.99	7.28	7.13
$\gamma = 0.95$	66.19	11.05	66	9.11	6.27	7.68
$\gamma = 0.999$	63.10	10.56	63	11.98	4.74	11.31
$\mathcal{H} = 200$						
G^γ	136.86	21.24	137	13.43	12.61	9.23
G^N	136.96	20.85	137	13.43	12.42	9.13
$\gamma = 0.85$	136.62	21.05	137	13.81	13.29	9.15
$\gamma = 0.95$	137.63	20.57	138	12.87	10.83	9.30
$\gamma = 0.999$	133.20	20.18	133	17.25	7.44	16.10

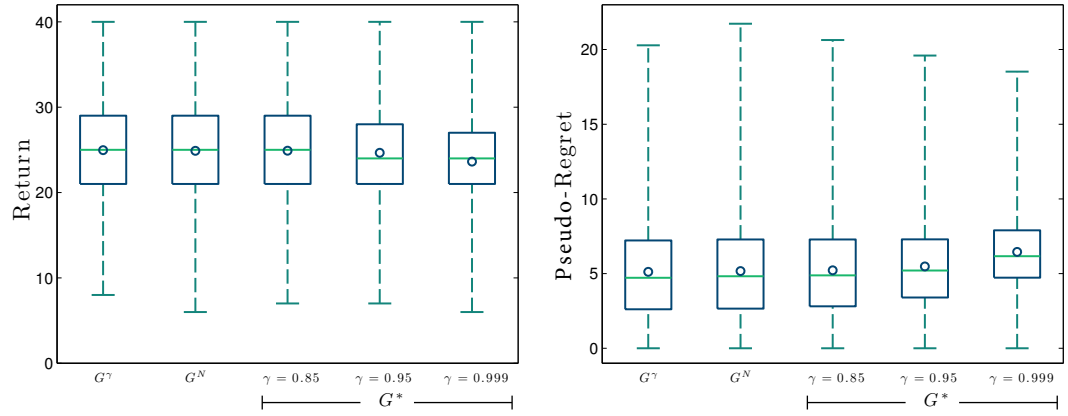
Table 7.5: Performance of Gittins based decision methods including G^γ on the finite horizon 10-armed test bed with various horizons. The top performing method on each problem length is highlighted.

ent horizon problems. In both the $\mathcal{H} = 40$ and $\mathcal{H} = 100$ problems we see G^γ perform the best of all the tested methods. To understand this, let us draw our attention to Figure 7.28, noting that for these two problems the majority of the methods, including G^γ, perform very close to one another. Indeed, comparing the mean return of G^γ and the closest pure Gittins method gives us p-values of $p = 0.33$ and $p = 0.94$ for $\mathcal{H} = 40$ and $\mathcal{H} = 100$ problems respectively. This suggests that it is very hard to improve further on performance in these cases, which seems reasonable given the difficulty of the problem. More interestingly, we see that the performance is not improved for the longest finite horizon problem tested. This is likely related to the way in which the GI converges given enough samples, as was well demonstrated in Figure 7.7. For longer problems, even if γ is set to a higher value, the GI will have converged for well sampled actions. In this case, reducing γ as the horizon approaches will not

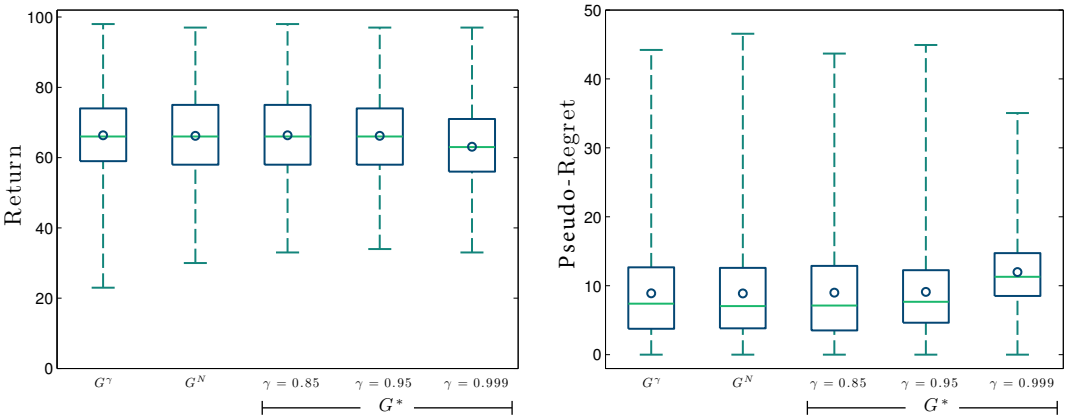
change the ordering of actions.

The results presented in this section may suggest that there is not a lot to be gained by using G^γ or G^N over the original Gittins Index selection strategy. However, we must remember that both of these approaches avert the need to tune the discount parameter to the specific problem, which as we have seen is non-trivial. Indeed, even if we knew how to set the discount parameter as a function of horizon in the 10-armed test bed, as soon as the number of actions or the true distribution of rewards changed then the mapping would likely change. We have already demonstrated that, if the discount parameter is badly matched, the performance of the algorithm can drop off. Having said that, provided the discount parameter is within a reasonable range, then all Gittins based methods perform better than all of the other heuristic based methods tested, each of which (excluding Thompson sampling) is *highly* sensitive to the choice of their parameters.

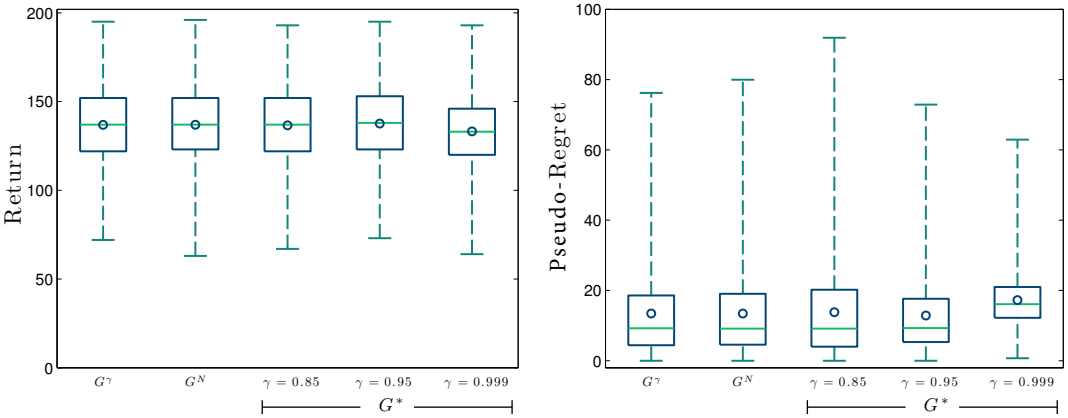
Further to the discussion on performance, we should note that both G^γ and G^N offer an improvement in computational efficiency. In the case of G^N, at each time period the number of calculations required is $\mathcal{O}(KN)$. The computational requirements for G^γ are harder to predict as the algorithm is run until convergence, the point of which is dependent in a non-trivial manner on γ. We can, however, think of the method as providing a sensible scheme for cooling the discount parameter to avoid excessive computation, due to the fact that as γ reduces so too does the number of iterations until convergence is achieved.



(a) $\mathcal{H} = 40$

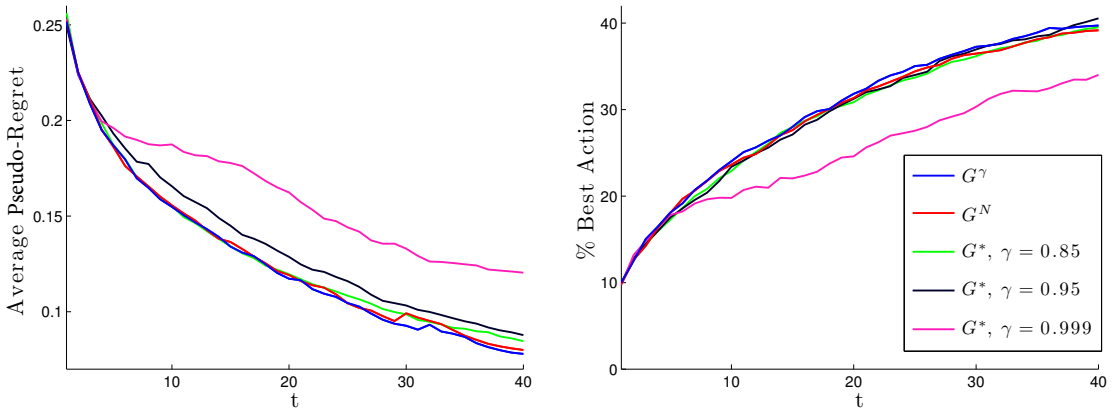


(b) $\mathcal{H} = 100$

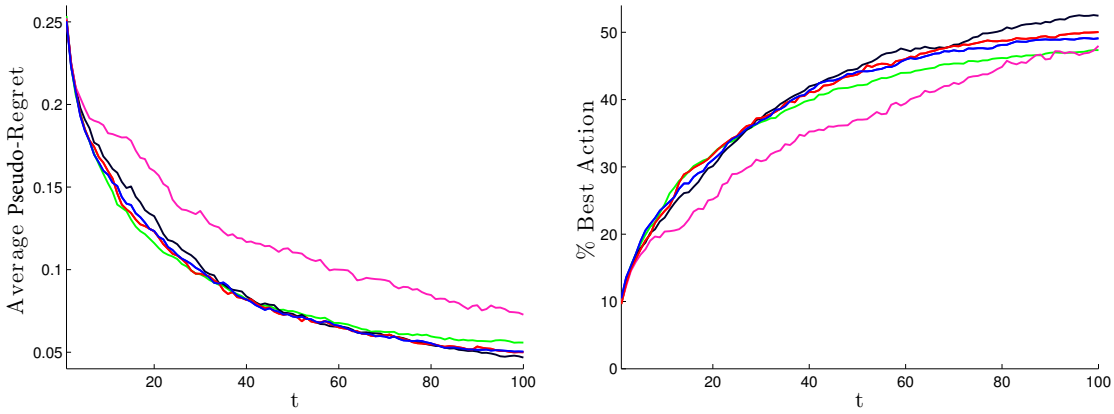


(c) $\mathcal{H} = 200$

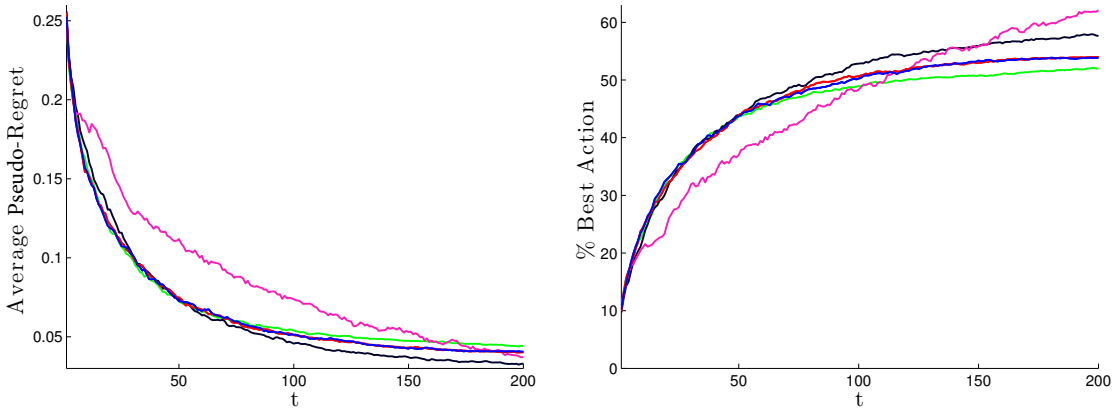
Figure 7.27: Box-and-whisker plots for the return and regret of Gittins based decision methods including G^γ on the finite horizon 10-armed test bed with $\mathcal{H} = 40$, 100 and 200.



(a) $\mathcal{H} = 40$



(b) $\mathcal{H} = 100$



(c) $\mathcal{H} = 200$

Figure 7.28: Performance over time of Gittins based decision methods including G^γ on the finite horizon 10-armed test bed with $\mathcal{H} = 40$, 100 and 200.

Chapter 8

Conclusions

> "The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work - that is, correctly describe phenomena from a reasonably wide area. Furthermore, it must satisfy certain aesthetic criteria - that is, in relation to how much is describes, it must be rather simple."
>
> *Method in the Physical Sciences*, von Neumann (1955)

8.1 Discussion

Intelligent Systems

Biological organisms interact and influence their environments in a variety of ways, although the manner in which this happens is not always underpinned by what we may wish to describe as 'intelligent' processes. Take, for example, a plant that grows in response to a light source through the process of *phototropism*. Without further appreciation of the mechanisms at play, we could happily describe the plant's behaviour using similar terminology to that used throughout this thesis; the plant appears, as a result of external stimuli, to take 'actions' in order to achieve an objective, namely the maximisation of photosynthesis.

At the other end of the biological spectrum we have the human brain which, again without recourse to the underlying mechanisms, also interacts with the environment and which we could equally well describe using the same terminology of inputs, actions, and objectives. The brain in its entirety is, of course, substantially more complex than the plant. More importantly, it is also capable of a far greater array of interactive behaviours, including the ability to 'learn' and 'plan'.

In a manner that parallels the organic world, there is a continuum of choices available in the design of interactive autonomous systems. We could start with explicitly programming our robot brain by defining a series of *if-this-then-that* type statements, or by implementing a basic feedback control system. These are reasonable approaches for simple processes, but quickly become untenable as the range of required behaviours grows, and impossible once we can't adequately describe systems in advance. In contrast, the sorts of systems considered in this thesis generate and reinforce *abstractions* from raw data. In practice, systems such as these are capable of deeper and more complex behaviours, which like the brain include an ability to learn from experience and form hypotheses about the future.

The manner in which we design systems to abstract from raw data, both in terms of model selection and the process of inference, is critical. It is possible to operate high-order models without the use of statistics, although we reiterate our advocacy of Bayesian reasoning. Once we arrive at the desiderata outlined in Chapter 2, which are consistent with common sense, then we know of no other way in which to fulfil them except through the principled application of Bayesian theory. Once we take this approach, desirable behaviour naturally manifests itself; we saw this through our exposition of exploration and exploitation in Chapter 7.

Having said this, in much the same way as our understanding of the physical universe continually evolves through scientific discovery, the ideas developed in this thesis (and indeed those aligned to the broader discipline of artificial intelligence) are

only a step towards a greater understanding of how autonomous systems should be developed and function. The field is young, yet even in its short history our belief in where the solutions lie has changed substantially. It seems prudent to not expect too much and to be receptive to new ways of thinking, especially when our objectives are so complex.

The Necessity Of Approximation

Putting fundamental questions to the side, there is clearly a pressing need to implement tractable yet robust methodology. We are 'practical' Bayesians, and understand that the grand vision to which we aspire is simply not tenable in its entirety. Approximation is a facet of real systems, and must be accommodated if we are to get anywhere. The question is whether we can do this in such a way so as to preserve the essence of Bayesian theory, whilst attaining the scalability of heuristics. We believe that we have managed to do this to some degree, backed up by experimental results. Computationally speaking, the use of the Gittins Index approximation and variable discounting methods are vastly more scalable than a full dynamic programming approach, and yield better performance than the heuristics tested; crucially doing so without the use of ad-hoc parameters. Our development of a covariance function, built on the Gaussian process framework, also shows promise in this regard. There are clear computational benefits to enacting a maximum likelihood or MAP type inference process, yet this disregards in part the uncertainty we wish to accommodate. By understanding what is lost in doing this, we were able to approximate the Bayesian solution in a tractable and principled manner, resulting in greater predictive performance.

Our hope is that, through greater exposition of the ideal, more work will be done to produce further principled approximations in this domain. It is not our opinion that research on the subject conducted outside the realm of Bayesian methodology

views heuristics as anything more than an inevitable consequence of computational limitations, at least for the most part. However, there is often a feeling that the Bayesian approach is intractable on any level, which as we have shown need not be the case. Further to this, we believe that too much effort is focussed on long term bounding of performance and asymptotic performance guarantees. As we made clear in our critique of stochastic decision making, if one looks to average performance in order to judge a method which is designed to perform a certain way *on average*, information and understanding is lost.

8.2 Future Directions

The directions in which work in this field can be taken is unbounded. Technology is moving rapidly away from simple analytics, to the point where intelligent interaction is becoming integral to the operation of many systems. Our 'Bingle' example is very much based on algorithms currently in deployment, and we have no doubt that the principles derived will be relevant in a host of other problem domains. Of course, progress occurs in steps, and so we suggest a few avenues of research which naturally follow on from the work presented here.

Incorporating Computational Cost

In the first instance, we extend the discussion of computational performance to which we referred but did not evaluate to a great extent. Clearly, this is deeply related to how effective a method can be. As new technologies emerge, we hope that this requirement lessens, although it is hard to see how it will not be an issue on some level. As we mentioned when introducing decision making, even the human brain operates under limited resources and seems to allocate effort depending on the importance of the task. We use this idea as inspiration for further work.

Whilst the evaluation of computational guarantees is doubtlessly a worthwhile endeavour, the need to do this can be reformulated in the context of a decision problem. Just as actions themselves are a resource to be allocated, it strikes us that computational complexity can be considered in the same vein. If we look towards the Gittins Index approximation, we see that this is an ever improving estimate of optimal behaviour. However, as we get closer to optimality the computational burden increases. One could think about evaluating the expected improvement of a policy in terms of the depth to which one progresses through a decision tree, at the same time as evaluating the expected computational loss, perhaps combining the two through some utility function. This would certainly be relevant to time critical systems in which the time to action is inversely proportional to the reward received. We can also imagine how function modelling, of the sort dealt with in Chapter 5, could be used as a means of propagating one's uncertainty through the decision tree in order to establish where investigation would be most valuable.

Reasoning About Others

We can extend this idea to concepts involving multiple decision makers. On one level, we can consider external agents as somehow part of the transition dynamics of an environment; however this dilutes the fact they will be reasoning in the same way as our decision maker. Once we start considering external agents as equally reasoned decision makers, models immediately take on a greater complexity. Work into *Game Theory* (von Neumann and Morgenstern, 2007) is interested in these types of problems, and it would be prudent for us to examine them in the context of Bayesian decision making. As stated in the introduction, being able to interact with humans is a clear objective of autonomous systems. Similar approaches will be needed to deal with other autonomous systems operating in the same domain as ours.

In an interesting piece of work lying somewhere in the domain between game

theory and cognitive neuroscience (Yoshida et al., 2008), it is observed that human beings, when faced with a competitive game, try to outwit their opponent by pre-empting their move. This inevitably involves some sort of recursive reasoning: 'I'll think about your move, which thinks about my move thinking about your move', and so on. The optimal strategy is to always out reason the opponent by one step of this process, and it seems from this research that the extent to which someone is willing to go through this recursion is identifiable. This forms an interesting decision problem for us, and we can imagine taking actions akin to exploratory actions on the MAB so as to determine another player's 'type'; this being done at the same time as maximising the game's objective.

Describing Sub-Optimal Methods

Moving back towards the specific types of methodology discussed in the thesis, we note that one of the challenges in analysing heuristic methods alongside Bayesian optimal decision making is that the two approach the problem from manifestly different angles. On the one hand optimal methodology relies on the principles of Bayesian theory to propagate value and uncertainty across a Markov decision process, whilst heuristics tend to ignore the structure of an MDP and collapse the problem to a one-step decision. An interesting idea is to see if we can describe heuristics more competently in the language of Bayesian statistics, either for the purpose of further critiquing the methods or as inspiration for more principled approximations.

One way we can imagine doing this is by placing the decision method in the domain of the utility function, rather than some ad-hoc procedure. By this we mean to say that the decision maker would approach a one step problem; however, where before utility consisted of reward it would now consist of (for example) reward and information gain, these being orthogonal in the explore-exploit domain. In this way, the decision maker would always appear rational and acting in his best interests.

Clearly, in the context of stochastic decision making this would mean the utility function itself would be stochastic. Whilst we don't agree with this as prescriptive to the agent we are designing, it would nevertheless be interesting to see if this type of approach could be used in the context of other agents. With regards to game theory, a central concern is establishing what utility function other agents in the environment are operating under. Of course, in order to determine this, one should understand the decision process by which this objective is maximised. By combining utility and decision making in this way, it may be possible to infer the two together.

Appendices

Appendix A

Experimental Details

A.1 Mountain Car Problem

The mountain car problem is equivalent to the 'puck on the hill' problem as described by Moore and Atkeson (1995). The object of the problem is to drive an underpowered car of mass $M = 1$ out of a frictionless valley, the height of which is given by:

$$H(x) = \begin{cases} x^2 + x & \text{for } x < 0 \\ \frac{x}{\sqrt{1+5x^2}} & \text{for } x \geqslant 0 \end{cases} \tag{A.1}$$

The state of the car is completely characterised by its position and velocity, such that $s_t = \{x_t, \dot{x_t}\}$, which are constrained to $-1 \leqslant x \leqslant 1$ and $-2 \leqslant \dot{x} \leqslant 2$ respectively. Movement of the car can be controlled through a horizontally applied force F in the range $-4 \leqslant F \leqslant 4$.

The dynamics of the system were described in Moore and Atkeson (1995), however we use the corrected version presented in Kuss (2006) which includes a centripetal acceleration term. Writing $H'(x) = \frac{\mathrm{d}}{\mathrm{d}x}H(x)$ and $H''(x) = \frac{\mathrm{d}^2}{\mathrm{d}x^2}H(x)$, the acceleration

of the car is given by:

$$\ddot{x} = \frac{F}{M\sqrt{1 + (H'(x))^2}} - \frac{gH'(x)}{1 + (H'(x))^2} - \frac{\dot{x}^2 H'(x)H''(x)}{(1 + (H'(x))^2)} \tag{A.2}$$

where $g = 9.81$ is the gravitational constant.

In the experimental setup, the force is applied for a fixed time interval Δ_t which, using formulas for constant acceleration, generates the following update equations:

$$\dot{x_{t+1}} = \dot{x}_t + \ddot{x}_t \Delta_t \tag{A.3}$$

$$x_{t+1} = x_t + \dot{x}_t \Delta_t + \frac{1}{2}\ddot{x}_t \Delta_t^2 \tag{A.4}$$

where $\ddot{x}_t$ is the solution to Equation (A.2) with $F = F_t$ and $x = x_t$. We can think of the equations listed in this appendix as defining the transition function $\mathcal{T}$ in the MDP model.

Appendix B

Function Modelling Derivations

B.1 Linear Model

In the standard linear model, observations y are related to inputs s as follows:

$$f(s) = s^\top \mathbf{w}, \qquad y = f(s) + \epsilon \tag{B.1}$$

where $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ is a Gaussian distributed noise variable. We can evaluate the likelihood of a set of observations $\mathbf{y}$ for particular values of the model weights $\mathbf{w}$ as follows:

$$\begin{aligned} p(\mathbf{y}|\mathbf{S}, \mathbf{w}) &= \prod_{i=1}^{n} p(y_i|\mathbf{s}_i, \mathbf{w}) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_n^2}} \exp\left(-\frac{(y_i - \mathbf{s}_i^\top \mathbf{w})^2}{2\sigma_n^2}\right) \\ &= \frac{1}{(2\pi\sigma_n^2)^n/2} \exp\left(-\frac{1}{2\sigma_n^2}|\mathbf{y} - \mathbf{S}^\top \mathbf{w}|^2\right) \\ &= \mathcal{N}(\mathbf{S}^\top \mathbf{w}, \sigma_n^2 I) \end{aligned} \tag{B.2}$$

$\mathbf{S}$ is a $D \times n$ matrix containing n input cases of dimensionality D.

To perform Bayesian inference we must specify a prior distribution over the

model weights, which typically can be taken as a zero mean Guassian:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \Sigma_p) \tag{B.3}$$

where Σ_p is the prior covariance matrix. The posterior distribution over the model weights is calculated through application of Bayes' rule:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{S}) = \frac{p(\mathbf{y}|\mathbf{S}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{S})} \tag{B.4}$$

where the marginal likelihood is:

$$p(\mathbf{y}|\mathbf{S}) = \int p(\mathbf{y}|\mathbf{S}, \mathbf{w})p(\mathbf{w})\mathrm{d}\mathbf{w} \tag{B.5}$$

Given this only serves to normalise the distribution we can exclude it from the following calculation, such that we get:

$$\begin{aligned} p(\mathbf{w}|\mathbf{S}, \mathbf{y}) &\propto \exp\left(-\frac{1}{2\sigma_n^2}(\mathbf{y} - \mathbf{S}^\top\mathbf{w})^\top(\mathbf{y} - \mathbf{S}^\top\mathbf{w})\right)\exp\left(-\frac{1}{2}\mathbf{w}^\top\Sigma_p^{-1}\mathbf{w}\right) \\ &\propto \exp\left(-\frac{1}{2}(\mathbf{w} - \bar{\mathbf{w}})^\top(\frac{1}{\sigma_n^2}\mathbf{S}\mathbf{S}^\top + \Sigma_p^{-1})(\mathbf{w} - \bar{\mathbf{w}})\right) \end{aligned} \tag{B.6}$$

where $\bar{\mathbf{w}} = \sigma_n^{-2}(\sigma_n^{-2}\mathbf{S}\mathbf{S}^\top + \Sigma_p^{-1})^{-1}\mathbf{S}\mathbf{y}$. The posterior distribution is therefore Gaussian:

$$p(\mathbf{w}|\mathbf{S}, \mathbf{y}) \sim \mathcal{N}(\frac{1}{\sigma_n^2}A^{-1}\mathbf{S}\mathbf{y},\, A^{-1}) \tag{B.7}$$

where $A = \frac{1}{\sigma_n^2}\mathbf{S}\mathbf{S}^\top + \Sigma_p^{-1}$. If we are interested in mapping the inputs through a set of basis functions $\boldsymbol{\phi}$ such that $\boldsymbol{\Phi}(\mathbf{S}) = \boldsymbol{\Phi}$, then we can simply substitute $\mathbf{S}$ for $\boldsymbol{\Phi}$ in Equation (B.7) such that:

$$p(\mathbf{w}|\boldsymbol{\Phi}, \mathbf{y}) \sim \mathcal{N}(\frac{1}{\sigma_n^2}A^{-1}\boldsymbol{\Phi}\mathbf{y},\, A^{-1}) \tag{B.8}$$

where $A = \frac{1}{\sigma_n^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \Sigma_p^{-1}$.

B.2 Marginalisation of β for AMCFs

Here we solve the integral:

$$\tilde{k}_{\nu,\Lambda} = \int k_\beta \; p(\beta|\nu,\Lambda)\mathrm{d}\beta \,, \tag{B.9}$$

where the prior on β is given by:

$$p(\beta|\nu,\Lambda) = \frac{1}{\sqrt{2\pi\Lambda}} \exp\left(\frac{-(\beta-\nu)^2}{2\Lambda}\right) \tag{B.10}$$

We use the taylor expansion of k_β originally given in (5.42):

$$k_\beta \approx k_\nu \; \exp\left(-(\beta-\nu)A'_\nu - \frac{1}{2}(\beta-\nu)^2 A''_\nu\right) \tag{B.11}$$

Inserting (B.10) and (B.11) into (B.9), the integral becomes:

$$\begin{aligned}
\tilde{k}_{\nu,\Lambda} &= k_\nu \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\Lambda}} \exp\left(\frac{-(\beta-\nu)^2}{2\Lambda}\right) \exp\left(-(\beta-\nu)A'_\nu - \frac{1}{2}(\beta-\nu)^2 A''_\nu\right) \mathrm{d}\beta \\
&= k_\nu \frac{1}{\sqrt{2\pi\Lambda}} \int_{-\infty}^{+\infty} \exp\left(\frac{-(\beta-\nu)^2}{2\Lambda} - (\beta-\nu)A'_\nu - \frac{1}{2}(\beta-\nu)^2 A''_\nu\right) \mathrm{d}\beta \\
&= k_\nu \frac{1}{\sqrt{2\pi\Lambda}} \int_{-\infty}^{+\infty} \exp\left(B\right) \mathrm{d}\beta
\end{aligned} \tag{B.12}$$

where we have substituted the contents of the bracket for B. In order to get to a convenient form for integration we group together the coefficients in B and complete the square, which gives us:

$$\begin{aligned}
B &= -\left(\frac{A''_\nu}{2} + \frac{1}{2\Lambda}\right)\beta^2 + \left(\frac{\nu}{\Lambda} - A'_\nu + \nu A''_\nu\right)\beta + \left(-\frac{\nu^2}{2\Lambda} + \nu A'_\nu - \frac{\nu^2 A''_\nu}{2}\right) \\
&= b_1\left(\beta - b_2\right)^2 + b_3
\end{aligned} \tag{B.13}$$

where

$$b_1 = -\left(\frac{A''_\nu}{2} + \frac{1}{2\Lambda}\right) \tag{B.14}$$

$$b_2 = \frac{\left(\frac{\nu}{\Lambda} - A'_\nu + \nu A''_\nu\right)}{\left(A''_\nu + \frac{1}{\Lambda}\right)} \tag{B.15}$$

$$b_3 = \frac{\Lambda A'^2_\nu}{2\left(1 + \Lambda A''_\nu\right)} \tag{B.16}$$

Inserting these back into (B.12) yields:

$$\begin{aligned}\tilde{k}_{\nu,\Lambda} &= k_\nu \frac{1}{\sqrt{2\pi\Lambda}} \exp\left(b_3\right) \int_{-\infty}^{+\infty} \exp\left(b_1(\beta - b_2)^2\right) \mathrm{d}\beta \\ &= k_\nu \frac{\sqrt{2\pi\sigma_b^2}}{\sqrt{2\pi\Lambda}} \exp\left(b_3\right) \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(\beta - b_2)^2}{2\sigma_b^2}\right) \mathrm{d}\beta \end{aligned} \tag{B.17}$$

where we have made the substitution $\sigma_b^2 = -\frac{1}{2b_1}$. The right hand side is a Gaussian and integrates to 1, so we are left with:

$$\tilde{k}_{\nu,\Lambda} = k_\nu \frac{\sigma_b}{\sqrt{\Lambda}} \exp\left(b_3\right) \tag{B.18}$$

Using (B.14), (B.16) and the expression for σ_b^2:

$$\tilde{k}_{\nu,\Lambda} = k_\nu \exp\left(\frac{\Lambda A'^2_\nu}{2\left(1 + \Lambda A''_\nu\right)}\right) \frac{1}{\sqrt{1 + \Lambda A''_\nu}} \tag{B.19}$$

which completes the integration.

Bibliography

P. Abrahamsen. *A review of Gaussian random fields and correlation functions.* Norsk Regnesentral/Norwegian Computing Center, 1997.

R. Agrawal. Sample mean based index policies with O(log n) regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27:1054–1078, 1995.

S. Agrawal and N. Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, volume 23, pages 39.1–39.26, 2012.

S. Arnborg and G. Sjodin. On the foundations of bayesianism. In *AIP Conference Proceedings*, pages 61–71. IOP Institute Of Physics Publishing LTD, 2001.

I. Asimov. Runaround. *Astounding Science Fiction*, 29(1):94–103, 1942.

J.-Y. Audibert, R. Munos, and C. Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

A. Barto and T. Dietterich. Reinforcement learning and its relationship to supervised learning. *Handbook of learning and approximate dynamic programming. John Wiley and Sons, Inc*, 2004.

A. Barto, S. Singh, and N. Chentanez. Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the 3rd International Conference on Development and Learning (ICDL 2004)*, pages 112–19, 2004.

D. E. Bell. Regret in decision making under uncertainty. *Operations research*, 30(5):961–981, 1982.

R. Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, 1957a.

R. Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6: 679–684, 1957b.

R. Bellman. *Adaptive control processes: a guided tour.* Princeton University Press, 1961.

R. Bellman. *Introduction to the mathematical theory of control processes*, volume 2. IMA, 1971.

J. M. Bernardo and A. F. Smith. *Bayesian theory*, volume 405. John Wiley & Sons, 2009.

D. Bernoulli. Specimen theoriae novae de mensura sortis [Exposition of a new theory on the measurement of risk]. *Commentarii Academiae Scientiarum Imperialis Petropolitanae [Papers of the Imperial Academy of Sciences in Petersburg]*, 5:175–192, 1738. Translated in Bernoulli (1954).

D. Bernoulli. Exposition of a new theory on the measurement of risk. *Econometrica: Journal of the Econometric Society*, pages 23–36, 1954.

J. Bernoulli. *Ars conjectandi (The Art of Conjecturing)*. Impensis Thurnisiorum, fratrum, 1713a.

N. Bernoulli. Correspondence of Nicolas Bernoulli concerning the St. Petersburg Game. *Letter to Pierre Raymond de Montmort*, 1713b.

D. A. Berry and B. Fristedt. *Bandit problems: sequential allocation of experiments (Monographs on statistics and applied probability)*. Springer, 1985.

D. Bertsekas. *Dynamic programming and optimal control*, volume 2. Athena Scientific Belmont, third edition, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

C. Bishop. *Pattern recognition and machine learning*. Springer New York, 2006.

G. L. Bretthorst. The near-irrelevance of sampling frequency distributions. In *Maximum Entropy and Bayesian Methods*, pages 21–46. Kluwer Academic Publishers, 1999.

M. Brezzi and T. L. Lai. Incomplete learning from endogenous data in dynamic allocation. *Econometrica*, 68(6):1511–1516, 2000.

M. Brezzi and T. L. Lai. Optimal learning and experimentation in bandit problems. *Journal of Economic Dynamics and Control*, 27(1):87–108, 2002.

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

N. Cesa-Bianchi and P. Fischer. Finite-time regret bounds for the multiarmed bandit problem. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pages 100–108. Morgan Kaufmann, San Francisco, CA, 1998.

N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.

M. Chen, Q. Shao, and J. Ibrahim. *Monte Carlo methods in Bayesian computation*. Springer Verlag, 2000.

Y. S. Chow, H. Robbins, and D. Siegmund. *Great expectations: The theory of optimal stopping*. Houghton Mifflin Boston, 1971.

R. T. Cox. Probability, frequency and reasonable expectation. *American journal of physics*, 14(1):1–13, 1946.

N. D. Daw, J. P. O'Doherty, P. Dayan, B. Seymour, and R. J. Dolan. Cortical substrates for exploratory decisions in humans. *Nature*, 441(7095):876–879, 2006.

P. Dayan and Y. Niv. Reinforcement learning: the good, the bad and the ugly. *Current opinion in neurobiology*, 18(2):185–196, 2008.

M. P. Deisenroth. *Efficient reinforcement learning using gaussian processes*, volume 9. KIT Scientific Publishing, 2010.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7):1508–1524, 2009.

M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient optimal learning for contextual bandits. *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011.

M. Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. PhD thesis, University of Massachusetts Amherst, 2002.

A. Einstein. On the method of theoretical physics. *Philosophy of science*, 1(2):163–169, 1934.

Y. Engel, S. Mannor, and R. Meir. Bayes meets bellman: The gaussian process approach to temporal difference learning. In *ICML*, volume 20, No.1, page 154, 2003.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 201–208, 2005.

R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, pages 309–368, 1922.

R. Garnett, M. A. Osborne, S. Reece, A. Rogers, and S. J. Roberts. Sequential bayesian prediction in the presence of changepoints and faults. *The Computer Journal*, 53(9): 1430–1446, 2010a.

R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for sensor set selection. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2010b.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. CRC press, 2013.

M. Ghavamzadeh and Y. Engel. Bayesian actor-critic algorithms. In *Proceedings of the 24th International Conference on Machine Learning*. ACM, 2007a.

M. Ghavamzadeh and Y. Engel. Bayesian policy gradient algorithms. In *Advances in Neural Information Processing Systems 19*. MIT Press, 2007b.

M. N. Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, Department of Physics, University of Cambridge, 1997.

P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization*. Academic press, 1981.

J. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(2):148–177, 1979.

J. Gittins, Y.-G. Wang, et al. The learning component of dynamic allocation indices. *The Annals of Statistics*, 20(3):1625–1636, 1992.

J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices (2nd ed.)*. Wiley, 2011.

J. C. Gittins and D. M. Jones. A dynamic allocation index for the discounted multiarmed bandit problem. *Biometrika*, 66(3):561–565, 1979.

T. Gneiting. Compactly supported correlation functions. *Journal of Multivariate Analysis*, 83(2):493–508, 2002.

J. I. Gold and M. N. Shadlen. The neural basis of decision making. *Annu. Rev. Neurosci.*, 30:535–574, 2007.

L. S. Gottfredson. Mainstream science on intelligence: An editorial with 52 signatories, history, and bibliography. *Intelligence*, 24(1):13–23, 1997.

O.-C. Granmo. Solving two-armed bernoulli bandit problems using a bayesian learning automaton. *International Journal of Intelligent Computing and Cybernetics*, 3(2):207–234, 2010.

I. Hacking. Strange expectations. *Philosophy of Science*, pages 562–567, 1980.

P. Hennig. Optimal reinforcement learning for gaussian systems. *Advances in Neural Information Processing Systems 24*, pages 326–333, 2011.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press and J. Wiley, 1970.

F. Hutter and M. A. Osborne. A kernel for hierarchical parameter spaces. *arXiv preprint arXiv:1310.5738*, 2013.

W. James. The principles of psychology. *Harvard UP, Cambridge, MA*, 1890.

E. T. Jaynes. *Probability theory: the logic of science*. Cambridge university press, 2003.

A. H. Jazwinski. *Stochastic processes and filtering theory*. Courier Dover Publications, 2007.

H. Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007): 453–461, 1946.

D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.

A. G. Journel and C. J. Huijbregts. *Mining Geostatistics*. Academic press, 1978.

E. Kaufmann, O. Cappé, and A. Garivier. On bayesian upper confidence bounds for bandit problems. In *International Conference on Artificial Intelligence and Statistics*, pages 592–600, 2012a.

E. Kaufmann, N. Korda, and R. Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *Algorithmic Learning Theory*, pages 199–213. Springer, 2012b.

M. Kennedy. Bayesian quadrature with non-normal approximating functions. *Statistics and Computing*, 8(4):365–375, 1998.

S. Kirkpatrick, M. Vecchi, et al. Optimization by simmulated annealing. *Science*, 220(4598): 671–680, 1983.

A. Kolmogorov. Interpolation und extrapolation von stationären zufälligen folgen. *Izv. Akad. Nauk SSSR*, 5:3–14, 1941.

M. Kuss. *Gaussian process models for robust regression, classification, and reinforcement learning*. PhD thesis, TU Darmstadt, 2006.

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

P. Laplace. *Essai philosophique sur les probabilités*. Courcier, 1814.

S. Lee. *Sequential Forecasting and Decision Making in Dynamic and Incomplete Environments*. PhD thesis, University of Oxford, 2009.

M. Lewis. Obama's way. *Vanity Fair*, 626:210–217, 2012.

M. Lichman. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2013.

G. Loomes and R. Sugden. Regret theory: An alternative theory of rational choice under uncertainty. *The economic journal*, pages 805–824, 1982.

R. D. Luce. *Individual choice behaviour: a theoretical analysis*. John Wiley & Sons, New York, 1959.

A. Markov. Extension of the law of large numbers to dependent quantities. *Izv. Fiz.-Matem. Obsch. Kazan Univ.(2nd Ser)*, 15:135–156, 1906.

A. Markov. An example of statistical investigation of the text eugene onegin concerning the connection of samples in chains (in russian). *Bulletin of the Imperial Academy of Sciences of St. Petersburg*, 7(3):153–162, 1913.

B. Matérn. *Spatial variation*, volume 49, No.5. Meddelanden från statens Skogsforskningsinstitut, Almänna Förlaget, Stockholm, 1960. Second edition (1986), Springer-Verlag, Berlin.

G. Matheron. The intrinsic random functions and their applications. *Advances in Applied Probability*, pages 439–468, 1973.

B. C. May, N. Korda, A. Lee, and D. S. Leslie. Optimistic bayesian sampling in contextual-bandit problems. *The Journal of Machine Learning Research*, 13(1):2069–2106, 2012.

J. McCarthy, M. Minsky, and N. Rochester. A proposal for the dartmouth summer research project on artificial intelligence. *Technical Report, Dartmouth College*, 1955.

N. Meuleau and P. Bourgine. Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*, 35(2):117–154, 1999.

A. W. Moore. Variable resolution dynamic programming: Efficiently learning action maps in multivariate real-valued state-spaces. In *Proceedings of the Eighth Machine Learning Workshop*, 1991.

A. W. Moore and C. G. Atkeson. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21(3):199–233, 1995.

K. Murphy. Conjugate bayesian analysis of the gaussian distribution. *Technical report*, 2007.

P. Nekrasov. *Filosofiia i Logika Nauki o Massovikh Proiavleniiakh Chelovecheskoi Deiatelnosti (The Philosophy and Logic of the Study of Mass Phenomena of Human Activity)*. Moskva: Universitetskaia tipografiia., 1902.

J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2006.

M. Osborne. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. PhD thesis, Oxford University New College, 2010.

M. Osborne, R. Garnett, S. Roberts, C. Hart, S. Aigrain, N. Gibson, and S. Aigrain. Bayesian quadrature for ratios. *Fifteenth International Conference on Artificial Intelligence and Statistics*, 2012.

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1, 2007.

J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

A. Pascual-Leone, A. Amedi, F. Fregni, and L. B. Merabet. The plastic human brain cortex. *Annu. Rev. Neurosci.*, 28:377–401, 2005.

N. G. Pavlidis, D. K. Tasoulis, and D. J. Hand. Simulation studies of multi-armed bandits with covariates. In *Computer Modeling and Simulation, 2008. UKSIM 2008. Tenth International Conference on*, pages 493–498. IEEE, 2008.

M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

L. Quetelet. *Sur l'homme et le développement de ses facultés ou essai de physique sociale*. Bachelier, 1835.

L. Quetelet. *A treatise on man and the development of his faculties [English translation of Quetelet (1835) with a new preface by Quetelet]*. William and Robert Chambers, 1842.

L. Quetelet. Statistique morale. De l'influence du libre arbitre de l'homme sur les faits sociaux, et particulèrement sur le nombre des mariages. *Royaume de Belgique. Ministère de l'Intérieure. Bulletin de la Commission centrale de statistique*, t. III., 1847.

C. Rasmussen and Z. Ghahramani. Bayesian monte carlo. *Advances in neural information processing systems*, 15:489–496, 2003.

C. E. Rasmussen and M. Kuss. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems 16*, 2004.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

H. Robbins. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc*, 58(5):527–535, 1952.

L. J. Savage. *The Foundations of Statistics*. 1954.

S. L. Scott. A modern bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.

E. Seneta. Markov and the birth of chain dependence theory. *International Statistical Review/Revue Internationale de Statistique*, pages 255–263, 1996.

E. Seneta. Statistical regularity and free will: LAJ Quetelet and PA Nekrasov. *International statistical review*, 71(2):319–334, 2003.

S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

S. Singh, R. Lewis, and A. Barto. Where do rewards come from? In *Proceedings of the annual conference of the cognitive science society*, pages 2601–2606, 2009.

J. Skilling. Fundamentals of maxent in data analysis. *Maximum entropy in action*, pages 19–40, 1991.

E. Snelson, C. E. Rasmussen, and Z. Ghahramani. Warped gaussian processes. *Advances in neural information processing systems*, 16:337–344, 2004.

M. L. Stein. *Interpolation of spatial data*. Springer, 1999.

R. Sutton and A. Barto. *Reinforcement learning: An introduction*. Cambridge Univ Press, 1998.

W. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933.

W. Thompson. On the theory of apportionment. *American Journal of Mathematics*, 57(2): 450–456, 1935.

E. Thorndike. *Animal intelligence*. Hafner, Darien, Conn, 1911.

J. W. Tukey. The future of data analysis. *The Annals of Mathematical Statistics*, pages 1–67, 1962.

P. Varaiya, J. Walrand, and C. Buyukkoc. Extensions of the multiarmed bandit problem: the discounted case. *Automatic Control, IEEE Transactions on*, 30(5):426–439, 1985.

J. von Neumann. Method in the physical sciences. In *The Unity of Knowledge,* edited by L. Leary, pages 157–164. Doubleday, New York, 1955.

J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition)*. Princeton university press, 2007.

C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, 1989.

R. Weber. On the gittins index for multiarmed bandits. *The Annals of Applied Probability*, pages 1024–1033, 1992.

P. Whittle. Multi-armed bandits and the gittins index. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 143–149, 1980.

P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, pages 287–298, 1988.

N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*. MIT Press, 1949.

Y. Yang and D. Zhu. Randomized allocation with nonparametric estimation for a multi-armed bandit problem with covariates. *The Annals of Statistics*, 30(1):100–121, 2002.

W. Yoshida, R. J. Dolan, and K. J. Friston. Game theory of mind. *PLoS computational biology*, 4(12), 2008.