Exploration and Exploitation in Bayes Sequential Decision Problems

James Anthony Edwards

Submitted for the degree of Doctor of Philosophy at Lancaster University.

September 2016

Abstract

Bayes sequential decision problems are an extensive problem class with wide application. They involve taking actions in sequence in a system which has characteristics which are unknown or only partially known. These characteristics can be learnt over time as a result of our actions. Therefore we are faced with a trade-off between choosing actions that give desirable short term outcomes (exploitation) and actions that yield useful information about the system which can be used to improve longer term outcomes (exploration).

Gittins indices provide an optimal method for a small but important subclass of these problems. Unfortunately the optimality of index methods does not hold generally and Gittins indices can be impractical to calculate for many problems. This has motivated the search for easy-to-calculate heuristics with general application. One such non-index method is the knowledge gradient heuristic. A thorough investigation of the method is made which identifies crucial weaknesses. Index and non-index variants are developed which avoid these weaknesses.

The problem of choosing multiple website elements to present to user is an important problem relevant to many major web-based businesses. A Bayesian multi-armed bandit model is developed which captures the interactions between elements and the dual uncertainties of both user preferences and element quality. The problem has many challenging features but solution methods are proposed that are both easy to implement and which can be adapted to particular applications.

Finally, easy-to-use software to calculate Gittins indices for Bernoulli and normal rewards has been developed as part of this thesis and has been made publicly available. The methodology used is presented together with a study of accuracy and speed.

Acknowledgements

I would like to thank my supervisors Professors Kevin Glazebrook, Paul Fearnhead and David Leslie for their guidance and support. Thank you also to my viva examiners Professor Richard Weber and Dr Peter Jacko for their time and helpful comments.

My PhD has been supported by the EPSRC funded Statistics and Operational Research (STOR-i) Centre for Doctoral Training. The work in Chapter 4 was supported by a Google faculty research award. I thank Steve Scott and all others I met with at Google for their time and feedback.

STOR-i has provided an ideal multi-disciplinary environment for both research and training. The frequent opportunities to interact with other students and staff on the programme has helped my research immensely and made studying towards my PhD a much more enjoyable experience. I hope it is adopted widely as a model for doctoral training in the future. I thank all involved.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

James Edwards

Contents

List of Abbreviations

BMAB	Bernoulli multi-armed bandit problem
BSDPs	Bayes sequential decision problems
CTR	Click-through-rate
FHNMAB	Finite horizon normal multi-armed bandit problem
GCM	General click model
GI	Gittins index
GIBL	Gittins index approximation of Brezzi & Lai
GICG	Gittins index approximation of Chick & Gans
GIs	Gittins indices
GPR	Greedy posterior regret
KG	Knowledge gradient
KGI	Knowledge gradient index
MAB	Multi-armed bandit problem
MDP	Markov decision problem
MFUP	Most frequent user preference set choosing method
NAI	Naive set choosing method
NKG	Non-dominated knowledge gradient
NMAB	Normal multi-armed bandit problem
NTS	Thompson sampling with naive set choosing
OUP	Ordered user preference set choosing method

PCM	Probabilistic click model
PKG	Positive knowledge gradient
OPT	Optimal set choosing method
R&S	Ranking-and-selection
RLB	Relative learning bonus
SEQ	Sequential set choosing method
STS	Thompson sampling with sequential set choosing
TCM	Threshold click model
TS	Thompson sampling
UCB	Upper confidence bound policy

Chapter 1

Introduction

What is it that makes a decision difficult? In the past it might have been because the decision was complex with lots of options and factors to consider. However, this challenge can often be overcome by utilising the large computing power that is available today. Instead it is much more often the presence of uncertainty or a lack of information that can make even small problems hard, especially when this uncertainty is about the future.

For example, I have to make a decision of whether to stay in my current job or take a new job in a different city. Although this is an important decision it is not, at first glance, very complex as there are only two alternatives. I just have to make a comparison between my current job and home, which I know a lot about, and my new job and location about which there are uncertainties but which I have researched well. The real difficulty is that to make a good decision I need to think about the future, the different scenarios that may emerge and how I will respond. Will I be promoted if I stay in my current job? If I don't like the new job but do like the location then what other options are there in my new city? What if the city is a problem, should I move again? If so where? What might I learn from the experience that would make moving jobs an easier decision next time? To fully evaluate the question I would need to continue like this considering possible branches into my future where each decision depends on what has gone before and each future depends on what is learnt in the past. Clearly the decision is far more complex than it seemed at first.

This thesis will focus on a crucial part of this problem, how to put a value on the new information that results from a decision. I learn a lot more by moving jobs than by staying where I am. My current job is reliable but limited in what it can become whereas the new job has much greater potential even though it might not work out. This kind of trade-off is known as the *exploration versus exploitation dilemma*. Do I *exploit* my current knowledge (stay with what I know) or *explore* to learn in the hope of finding something better.

This feature is investigated by studying *Bayes sequential decision problems*. These are problems, such as the job example above, which involve taking actions in time sequence in pursuit of some objective. There is information about which we are uncertain which could help our decisions if it were known more accurately *and* we have the opportunity to *learn* this information as a result of taking actions. Therefore our actions affect both the world and our understanding of it which in turn affects our decisions about future actions. This chapter will introduce some of the main ideas of this type of problem and outline the rest of the thesis.

1.1 Markov Decision Processes

A common method of formulating sequential decision problems which will be adopted here is a *Markov decision process* (MDP). These date from at least as far back as Wald (1950) and are an extension of *Markov processes*. Markov processes move through *states* where each move depends on *transition probabilities* associated with the current state. MDPs add a degree of control to the process where a decision-maker can choose from a set of *actions* in each state. The transition probabilities now depend on the action as well as the current state. A *reward* is received which depends on the action and resulting transition. The transition and reward resulting from an action will often be referred to here as the *outcome*. A *policy* is a set of rules that determine which action is taken in any state.

In the basic formulation, which will be assumed throughout this thesis, actions and transitions occur at fixed discrete times $\{0, 1, 2 \ldots\}$ but continuous time versions are frequently used. In a semi-Markov decision process the intervals between each event are random variables which may depend on the start and end states and the most recent action. The time horizon could be of finite or infinite. Where the horizon is infinite the rewards are usually discounted by a fixed factor $\gamma \in (0, 1)$ so that total reward is finite under common simple assumptions. At time t the *return* (the reward received after discounting) is then $\gamma^t y_t$, where y_t is the observed (undiscounted) reward at time t. The standard objective in this setting is to maximise the total discounted expected return over the infinite horizon.

With this objective the quality of any action cannot be judged solely on the immediate expected reward that will be received as it also depends on future rewards. These in turn depend on future states and hence the transition probabilities associated with the action. An action yielding poor reward now may be preferred if it results in a transition to states that yield higher rewards later. Hence to evaluate current actions all future actions and outcomes must be considered. Therefore, as the state and actions spaces increase in size and the transition probability distributions become more complex MDPs can rapidly become very difficult to solve.

Another kind of trade-off between short and long term rewards, as well as another dimension of complexity, comes from adding *learning* to the problem. This arises when there are aspects of the system, such as current state, reward distributions or transition probabilities, which are unknown or partially known but which can be learnt as a result of taking actions and observing outcomes. This leads to a more specialised formulation of the MDP which will be given over the next two sections. The literature for MDPs is extensive and of a more general nature than can be covered here. For a detailed description see, for example, Puterman (2005).

1.2 Learning and Bayesian Inference

We now make two changes to the MDP framework given in the previous section which lead towards the multi-armed bandit problem as will be described in the next section. First, rewards, and therefore decisions, depend on an unknown parameter θ about which we receive information by observing outcomes. Second, the state now incorporates a *belief state* or *informational state* which records our current knowledge of θ. Transitions between states then represent learning and are determined by Bayesian inference as will now be described.

Bayesian statistics provides a framework where our belief about quantities or parameters of interest can be formally represented by probability distributions (see Jaynes and Bretthorst 2003). Let θ be such a parameter which does not change over time and let y be an observation (such as a reward) from the system. The *sampling model* $f(y|\theta)$ gives the probability that the data y would be observed given a known value of θ. The parameter θ is not known exactly but we have knowledge of its distribution given by the *prior* density $g(\theta)$. The *posterior* density $g(\theta|y)$ gives our belief about the value of θ once y has been observed. The transition (or updating) from prior to posterior after an observation is given by *Bayes' rule*:

$$g(\theta|y) = \frac{f(y|\theta)g(\theta)}{\int_\theta f(y|\theta)g(\theta)\mathrm{d}\theta}.$$

Updating easily extends to a sequence of observation steps which makes it very appropriate for sequential decision problems. The posterior of each step becomes the prior for the next step. The belief at time $t = 1, 2, \ldots$ can be found from the belief at time $t-1$ starting with a prior $g(\theta)$ at time 0. Given a sequence of i.i.d. observations, $y_1, \ldots, y_t$ where y_t is the data observed at time t,

$$g(\theta|y_1, \ldots, y_t) = \frac{f(y_t|\theta)g(\theta|y_1, \ldots, y_{t-1})}{\int_\theta f(y_t|\theta)g(\theta|y_1, \ldots, y_{t-1})\mathrm{d}\theta}.$$

The belief at time $t-1$ depends on the data observed up to and including that time. The denominator is called the *marginal likelihood.* This is a normalising constant that ensures that the posterior is a probability density. The posterior distribution can be used to give the *predictive distribution* of the next observation y^*:

$$p(y^*|y_1, \ldots, y_t) = \int_\theta f(y^*|\theta)g_t(\theta|y_1, \ldots, y_t)\mathrm{d}\theta.$$

It is the predictive distribution that models the possible outcomes of actions and which is used in decision making.

Although this framework is very general it is particularly useful for forms of $f(y \mid \theta)$ which enable a choice of prior such that the posterior is of the same family of distributions (the prior and posterior are said to be *conjugate*). For appropriate sampling and belief distributions updating can then be done analytically. For Bernoulli or binomial observations the Beta family provides conjugate beliefs and normal priors provide conjugacy for normal sampling models (self-conjugacy). By using a conjugate prior there is a closed form expression for the posterior which depends upon a small number of defining parameters. The knowledge state can be therefore be expressed and recorded succinctly. Conjugate priors are generally used where possible for these reasons of mathematical tractability.

Bayesian statistics provides a range of methods for the analysis of data that go beyond inference but which link naturally to it within a probabilistic framework. Examples are model choice, design of experiments and decision theory. A good introduction to Bayesian methods is given in Hoff (2009). For a more decision-oriented approach see Robert (2007).

1.3 The Multi-armed Bandit Problem

The multi-armed bandit problem (MAB) is one of the the most well studied of sequential decision problems. It provides the simplest expression of many of the features and challenges of such problems in particular the trade-off of exploration and exploitation. A description of the classical form of the problem will be given here to introduce the main ideas of the thesis but the problem has since spawned into a very large and multi-layered problem class. Chapter 2 will go more deeply into the origins and variations of this wider class and well as the range of solution methods that have been proposed.

The name comes from the analogous problem of a gambler playing a slot machine (a one-armed bandit) in a casino. The gambler can select one machine from those available at each time which pays out a random reward from an unknown distribution which is different for each machine. The distribution for each machine is fixed over time so the gambler can learn about it through playing. How should the gambler play to maximise winnings?

An example of the kind of problem that motivated the study of the MAB was by given by Thompson (1933). Here, the choice is between several drugs. Each drug has a fixed unknown probability of curing a patient of a disease. The reward is binary, either the patient is cured or is not. How should we assign drugs to learn about which drugs are effective and heal the most patients?

This latter problem can be modelled as a MAB with a Bernoulli reward distribution. This is the classical form of the MAB and is as follows. At each decision time $t = 0, 1, 2, \ldots$ an action $a_t \in \{1, \ldots, k\}$ is taken. Action $a_t = a$ corresponds to choosing alternative a (or *pulling arm* a). Associated with each arm a is an unknown parameter θ_a. Action $a_t = a$ results in an outcome or observed reward y_t where $y_t \mid (a_t = a) \sim Bern(\theta_a)$.

Each θ_a has an independent conjugate prior $g_0(\theta_a \mid \alpha_{a,0}, \beta_{a,0}) \sim Beta(\alpha_{a,0}, \beta_{a,0})$. The α and β parameters for all the arms together give the current (informational) state of the system. Following an action $a_t = a$ and outcome y_t the state parameters for arm a update by $(\alpha_{a,t+1}, \beta_{a,t+1}) = (\alpha_{a,t} + y_t, \beta_{a,t} + 1 - y_t)$. For all other arms $b \neq a$ we receive no information so $(\alpha_{b,t+1}, \beta_{b,t+1}) = (\alpha_{b,t}, \beta_{b,t})$.

The total return is given by

$$\sum_{t_0}^{\infty} \gamma^t y_t$$

where $0 < \gamma < 1$ is a discount factor. The objective is to design a policy that takes actions that maximise the *Bayes' return* which is the total return in expectation over both the system realisations and the prior information.

A natural policy to follow is the *greedy policy.* This is greedy with respect to the immediate expected reward of each arm which is given by

$$\mu_{a,t} = \frac{\alpha_{a,t}}{\alpha_{a,t} + \beta_{a,t}}.$$

At any time the arm with the current highest value (the *greedy arm*) is chosen. The greedy policy is not optimal though as it is *myopic* so does not consider the value of information results from actions. An arm with tight posterior (high $\alpha + \beta$) will be less likely to have a high θ than an arm with the same μ about which we are more uncertain (low $\alpha + \beta$). An observation from the latter arm will bring a bigger change in μ than one from the first arm. If the change is negative then there are other arms to fall back on but if it is positive then we will receive good information about a potentially good arm which we can take advantage of in future decisions. This effect is often strong enough that it is worth sacrificing immediate reward by choosing an arm with lower μ if it has greater potential to be the best arm.

The MAB therefore gives a simple representation of the *exploration versus exploitation dilemma.* Do we *exploit* current knowledge to maximise the immediate expected reward or do we *explore* other arms to learn information that will enable us to gain extra reward in the future? The MAB is one of the simplest models for the study of this dilemma and as a result it has been given much attention. Despite this and its apparent simplicity it has been a very difficult problem to solve. Theoretically, techniques such as *stochastic dynamic programming* can be used to solve it but the computation grows rapidly with the number of arms so that this is impractical with all but the smallest problems. A key result, the *Gittins index theorem* (Gittins 1979) enables the problem to be decomposed into smaller subproblems while retaining optimality of the solution. For each arm a *Gittins index* value is calculated and the policy that chooses arms in order of index value is optimal for this problem. Dynamic programming and Gittins indices will be described in more detail in Section 2.3. Simple changes to the MAB mean that policies based on Gittins indices are no longer optimal which provides motivation for the research in this thesis.

1.4 Thesis Outline

Bayes sequential decision problems extend far beyond the classical MAB. Under even simple extensions the optimality guarantee of the Gittins index theorem no longer holds and the calculation of the required indices can become much more difficult. This motivates the search for, and investigation of, other solution methods. Chapter 2 will give an overview of Bayes sequential decision problems and the main solution methods. The focus of the thesis is on problems that exhibit the exploration versus exploitation dilemma, especially as found in extensions to the classical MAB. However awareness of the wider problem area is important in order to understand which solution methods are most appropriate for a given problem.

Chapter 3 investigates *knowledge gradient* (KG) policy. This is a recently developed heuristic solution method which was intended for very general application in MABs. We study its use in a class of MABs which have rewards from the exponential family.

A weakness was found where the policy took actions which were *dominated* in the sense that the action was inferior in both exploration and exploitation. The chapter will give variants which avoid this error. These new policies include an index heuristic which deploys a KG approach to develop an approximation to the Gittins index. This is shown to perform well in a numerical study over a range of MABs including those for which index policies are not optimal. One of the promising features of KG was that it could incorporate correlation between arms into its decision making. However it was found that this did not appear to bring a performance advantage over competitor policies. This chapter has been accepted as a paper (Edwards et al. in press).

Chapter 4 studies the problem of choosing multiple website elements to display to a user of a website. To choose elements that appeal to the user we need to learn the characteristics and general quality of each element while at the same time choosing elements that are appropriate for the user in the short term. Therefore MAB framework is appropriate for this problem. However, elements presented together interact since choosing elements that are very similar will result in redundancy for the user. Therefore rewards need to be considered as function of the set of elements as a whole rather than just some simple combination of the rewards of the individual elements. This brings a number of challenges. Firstly we need to model the interactions in an appropriate way so that rewards realistically reflect user behaviour. The models developed are based on the intuitive idea that we have uncertainty about users' preferences. We investigate the consequences of this model on the *diversity* of the element sets. Secondly we need solution methods which choose and learn about element *sets* while being fast and efficient in an online setting. This is made difficult by the combinatorial explosion in the number of potential element sets. The policy we use combines existing bandit methods with a submodular optimisation algorithm. This requires a method of learning from user feedback that scales with the number of elements rather than the number of element sets. A further complication is uncertainty over the user preferences. A Bayesian scheme is given that deals with these issues.

The Gittins index theorem was a breakthrough result in the study of MABs. It gives a computationally tractable optimal policy for the classical MAB and some variations. Even where it is not optimal, similar index approaches have produced strongly performing policies for a wide range of problems. However, computing the indices required for the policy is not trivial even for common reward distributions. A greater obstacle to their use, though, is the perception that the computation is more difficult than it is. We have produced and made publicly available easy-to-use software to calculate Gittins indices for Bernoulli and normal rewards. Chapter 5 describes the methods used in this software and discusses some of the issues involved in calculating Gittins indices.

Chapter 6 will conclude by giving the main contributions of the thesis and the future research directions emerging from the work involved.

Chapters 3, 4 and 5 can each be read independently and as such some of the basics of the problem area will be repeated several times. There are some differences in notation between chapters but notation is defined within each chapter. The work in Chapter 4 was funded by a Google faculty research award to work on a problem of interest to Google and this motivated the direction of the work.

Chapter 2

Bayes Sequential Decision Problems

Bayes sequential decision problems (BSDPs) have widely varying origins and motivations. The problems and the solution methods developed to solve them have come from a range of disciplines such as statistics, operational research, computing, engineering, machine learning, control theory, artificial intelligence, and economics. Despite large areas of overlap, the different fields naturally had different motivations and expressions of the problems so that solution methods developed in one field may not be appropriate for use with problems for another. Even terms as fundamental as “bandit” or “optimal” can have different interpretations in different areas. This chapter will give an overview of BSDPs, in particular aiming to identify similarities and differences between the different classes of problems so that objectives can be stated more clearly and that solution methods may be better understood and used appropriately.

Section 2.1 give examples of how BSDPs have developed in each of the main disciplines that use them, highlighting the principle features of interest. Section 2.2 will outline the most common objectives in BSDPs and discuss when these different objectives are appropriate. This will lead into Section 2.3 which will give a review of the main solution methods that have been developed for BSDPs. Section 2.4 will give a condensed overview of the main variants of the MAB and their applications.

2.1 Main Features and Origins

BSDPs are optimisation problems where: (i) decisions are made in time sequence, (ii) such decisions involve taking actions that affect a system of interest, (iii) there are features of the system on which decisions may depend which are unknown or partially known, (iv) we learn about the system over time so that later decisions can be better informed than early decisions, and (v) learning is commonly modelled with a Bayesian framework. Many of the models and methods discussed here do not necessarily assume Bayesian learning and can be formulated using other learning and inference models. The interest for this thesis is in problems where *learning is affected by actions* (as distinct from problems where information is accrued over time independently of decisions) but the overview given here will be of a general nature.

The classical MAB, described in Section 1.3, is one of the best known and most studied of BSDPs. One of the original motivations for study of the MAB came from statistics in the *sequential design of experiments* (Robbins 1952). Statistical experimentation is traditionally based on sampling from one or more populations then observing and analysing the results. A modern example is in website optimisation (Scott 2010). We have several possible designs for a commercial website and we want to know which one leads to the best response from users (e.g. the most purchases of a product). A standard method of design of such an experiment (often referred to as *A/B testing* when there are only two alternatives) would be to choose an experiment size (number of users) and, in a suitably controlled manner, present each design to an equally sized subpopulation of the available users. The outcome of the experiment is analysed only when it is over. An alternative approach is to use sequential analysis (e.g. Wald 1973) which instead considers observations one by one, making a decision after each one (i) whether to continue the experiment and if so, (ii) which design should be shown to the next user. This has the advantages that more effort can be devoted to more promising designs and the experiment size does not have to be fixed in advance so can be stopped early if one design is clearly the best. Practical applicability of sequential analysis depends on the ease with which individual observations can be taken and the delay before feedback is outcomes are observed. Web-based applications are ideal since feedback is often instant and the design shown can be changed with minimal effort.

The MAB has since grown into a large problem class of its own, referred to as *multi-armed bandit problems* or just *bandits*. The name can have different interpretations, sometimes referring to the classical MAB and close relatives (as will be the case in this chapter) and other times the wider class or some part of it. The use of the MAB in sequential sampling for inference emphasises the element of learning. However, for bandit problems in general, learning is not necessarily a central element and may not even be a direct consideration in decision making. The bandit framework, particularly in the field of operational research, has become much more general which has allowed theory developed in the study of the MAB to be applied to a much broader class of problems based on sequential resource allocation to competing projects. This is a problem occurring in a wide range of applications such as scheduling, internet-based marketing and commerce, product pricing, industrial research and development, and clinical drug testing. This wider problem class is described in detail in Gittins et al. (2011).

An area in which the study of bandit problems is currently very active is machine learning. This has developed out of computer science but with strong links to statistics. The approach to bandits reflects these origins, combining reinforcement learning ideas (see below) with the sequential learning motivation of Robbins (1952). A fuller review of the numerous variants of bandit problems in machine learning, statistics and operational research will be given in Section 2.4.

Optimal control (Bertsekas 2012) is the study of optimisation over time and has been instrumental in the development of some fundamental solution methods such as *dynamic programming*. Optimal control is connected with *control theory* from the engineering field which is concerned with dynamical systems and has found use in applications such as vehicle control systems. Where these involve feedback in response to an action these problems can share many features with BSDPs. Of note is the idea of *dual control* where the system must simultaneously be controlled and learnt about. This is analogous to the exploration and exploitation trade-off and is referred to as identification and control or, action and investigation.

Reinforcement learning is a very general approach to optimisation in MDPs and related problems which was developed in computer science and artificial intelligence with ideas inspired by behavioural psychology. It treats problem solving as a process of learning about an uncertain environment by interacting with it and observing outcomes. This process could occur in a simulated environment inside a computer rather than the real world which can lead to a different trade-off between exploitation and exploration as will be discussed in Section 2.2. It has had an important role in research into BSDPs and the MAB since the computer science viewpoint has been incorporated into bandit research in machine learning. A full description of reinforcement learning can be found in Sutton and Barto (1998) which also contains further references on the MAB in fields such as computing, engineering and psychology.

A large area of BSDPs which initially seems very similar to the MAB but which has quite different aims is *ranking-and-selection* (R&S). Like the MAB this is concerned with the comparison of a set of alternatives. The comparison can take several forms but most commonly the aim will be to learn which is best (i.e. optimised with respect to some quantity, the value of which is initially unknown). At each each time we choose to measure one alternative and so gradually learn about each over time. This clearly has similarities with the sequential design of experiment view of bandits. The difference, as will be discussed in detail in the next section, is that no reward is received during the process but is only received at the end. A common application of R&S is in *simulation optimisation* (e.g. Fu 2002). For example, we need to choose between competing designs for the layout of a manufacturing plant. Each can be evaluated via computer simulation but the output is stochastic so the simulation will have to be run many times to give a good estimate of each design's true performance. The simulation takes a long time to run due to its complexity and there are a large number of alternatives. The problem is to find a strategy to sequentially divide the available computing budget between the designs. Much analysis of R&S uses frequentist statistical methods (see Bechhofer et al. 1995 for a comprehensive overview) but Bayesian methods are commonly used too (originating with Raiffa and Schlaifer 1968). An introduction to the R&S in general can be found in Kim and Nelson (2006).

2.2 Objectives

The objective for the MAB as given in Section 1.3 is well defined - maximising the Bayes' return over an infinite horizon using a constant discount factor. However, this is not the only objective in BSDPs and this section will present some of the alternatives. Of particular interest is the role learning plays.

2.2.1 Online and Offline Learning Problems

Powell and Ryzhov (2012) makes a useful distinction between *online* and *offline learning problems*. The terms online and offline refer to how and when rewards are collected. A learning problem is said to be *online* if rewards are collected at the same time as information is gathered. The MAB is the classic example of an online learning problem. In *offline* learning problems decisions and feedback still occur in sequence but no reward is collected and the policy is only evaluated according to the state of the system at the end of the process. The focus of this thesis is on online problems so problems and solution methods for purely offline problems will not be discussed in any detail. However, it is still important to be aware of the different objectives since this can inform the use of appropriate models.

An excellent example of an offline learning problem is R&S. For a formal overview of possible objectives in R&S see Kim and Nelson (2006) but, informally, the most common aims are to identify the alternative that is best (or close to the best) (i) with maximum probability by the end of a fixed horizon, or (ii) to a given level of confidence with the minimum number of observations. R&S is appropriate for applications where the actions have no immediate consequences. For example, in simulation optimisation there is no difference between two simulations other than computational resources consumed and the information received. There is no trade-off between exploitation and exploration in offline problems, instead the problem is one of pure exploration.

The related problem of *target processes* was the original motivation for the development of Gittins indices. Chemical compounds are screened sequentially as potential drugs and the problem is to identify, as quickly as possible, one that exceeds some threshold of a given measure. More information and theory on this problem can be found in Gittins et al. (2011) where the target process is one example of bandit processes as a general class of *sampling processes*.

In the sequential design of experiments interpretation of the MAB it is less clear whether the problem is one of online or offline learning. If the experiment in question is a clinical trial then there is a strong argument that immediate outcomes matter as these represents patients' health outcomes and so online is appropriate. If the experiment is one of market research of a range of products before they go on sale then offline would be more appropriate as no products will be sold during the experiment and only the information gained matters. In the first example conducting the experiment sequentially allows short term outcomes to be improved while in the second the motivation is more efficient learning, with the hope that required research budget can be reduced. In the former the MAB is appropriate while in the latter the problem is closer to R&S. Motivated by the former Robbins (1952) introduced a new criteria (based on *regret*) which is often used as an alternative to Bayes' return. This leads to a different balance of exploitation and exploration as will be discussed in the next section.

2.2.2 Regret Criteria

The MAB formulation given by Robbins (1952) used a different objective than given in Section 1.3 (the problem was otherwise the same). This was based on the *regret* $\rho(T)$ at some time T given a history of actions $\{a_1, \ldots, a_T\}$ which was defined to be

$$\rho(T) = T \max\{\theta_1, \ldots, \theta_k\} - \sum_{t=1}^{T} \theta_{a_t}. \tag{2.2.1}$$

This can be defined for general reward distributions by replacing θ with the expected true reward of arms. Note that there is no discounting over time in this setup and that minimising $\rho(T)$ is equivalent to maximising undiscounted reward over the same horizon. The difference between formulations comes from how the regret is used to set objectives.

The most basic objective given by Robbins (1952) is that average regret $\rho(T)/T \to 0$

as $T \to \infty$. This requires infinite exploration to ensure that the best arm is identified and that actions converge to the best arm asymptotically. There are many policies that can achieve this objective, one of which was given by Robbins (1952) for the Bernoulli MAB with $k = 2$ arms. Such policies are called *consistent* by Lai and Robbins (1985) and *uniformly convergent* by Burnetas and Katehakis (1996). No conditions are made on the rate of learning or reward collected over any finite time.

Stronger conditions, based on the worst case growth rate of regret, originate from the result in Lai and Robbins (1985) that the regret must grow at least logarithmically with time. They gave an asymptotic lower bound for regret based on the differences between reward distributions of the arms. That result was for reward distributions with a single unknown parameter but was later extended to multi-parameter unknowns by Burnetas and Katehakis (1996). Lai and Robbins (1985) call any policy that attains this lower bound as $T \to \infty$ *asymptotically efficient* or *optimal.* These terms have since been used more loosely to refer to any policy for which regret grows at some constant multiple of the lower bound, that is $\rho(T) = o(T^b)$ for all $b > 0$ as $T \to \infty$. Burnetas and Katehakis (1996) distinguishes between these two sets of policies with those achieving the lower bound having a *uniformly maximal convergence rate* and those meeting the lesser criterion being *uniformly fast convergent.* Therefore not all policies that are uniformly fast convergent are equivalent and it is unsatisfactory to describe such policies as optimal. For a more formal description of the different regret optimalities with further references see Cowan et al. (2015).

Besides asymptotic results it is desirable to have guarantees on regret after any finite time. Regret criteria are commonly used, especially in machine learning. The current standard of evaluating a policy is to give an upper bound on its regret over time (e.g. Auer et al. 2002). The aim is that this is of $O(\log T)$ but large constant multipliers can mean that these bounds are very loose. As illustrated in Cowan et al. (2015), the difference between a policy's regret bound and its empirical performance may be large, even for long time horizons, and consistent policies can be outperformed by non-consistent policies for a very long time.

2.2.3 Problem Horizon and Discounting

The distinction between maximising Bayes' return and meeting different regret criteria can be seen as a difference of effective time horizon and reward availability. Discounting makes near term rewards and therefore information more important than long term rewards and information. The discount factor γ can be adjusted to control the balance between the importance of short and long term rewards, and therefore the required balance between exploration and exploitation. Which viewpoint is best is a modelling question which should depend on the problem application. Three aspects of this will be discussed briefly here: discounting, asymptotic learning and the cost of information.

The use of discounting has been criticised as artificial especially with a factor that is constant over time (Scott 2010). This can be a reasonable criticism if actions are taken over very irregular time intervals but can also stem from a misunderstanding of the modelling role played by discount factors. The most common view is the economic one that uses discounting to model the reduced value of future money relative to present value. That is, discounting accounts, in a simplified way, for the return that resources could generate if used outside the model. However for many problems this would give a discount factor close to 1 and is not the main reason to use discounting. Instead discounting represents the idea that information becomes less reliable over time. One way this can be seen is to consider what happens if rewards are not guaranteed to always be available for collection - projects can always be cancelled, machines can breakdown or a business might go bankrupt. An undiscounted problem where there is a constant probability that reward process ends prematurely is equivalent to an infinite horizon problem with a constant discount factor (Kelly 1979).

Underlying all regret based optimality criteria given in Section 2.2.2 is the condition that, asymptotically, the process learns which arm has the best true expected reward. This is not true for policies that optimise Bayes' return for the discounted MAB as these can show *incomplete learning* (Brezzi and Lai 2000). This has been given as a failing of such policies but an alternative view is that complete asymptotic learning is incompatible with Bayes' return type optimality and therefore should not always be pursued. It is worth noting that R&S, where the objective is purely about learning, does not seek such guarantees as observations are seen to carry a cost and hence the time horizon is necessarily finite. Glazebrook (1980) did show that a randomised version of the GI could be arbitrarily close to Bayes optimality while retaining asymptotic complete learning but in as much as it is close to Bayes optimal it will be close to the GI-based policy until discounted rewards are small.

Modelling for learning problems cannot be appropriate without considering the cost of information and this goes some way to explaining the different approaches from different fields. Problems in statistics or operational research are more likely to be physical processes where the sampling cost is high and so samples are realistically limited in size. Therefore discounting or a limited horizon length is appropriate. In computing and machine learning sampling is commonly rapid and low cost so effective horizons can be very long. This can be seen especially in reinforcement learning where the aim is to learn a solution to a problem. Typical solution methods (which will be described in Section 2.3.4) balance exploration and exploitation but only as a method of learning. The reward collected is more properly described as a reward *signal* which indicates where learning effort should be focused. Hence the aim is really one of exploration where a long time horizon is available.

2.3 Solution Methods

The main difficulty with sequential decision problems is that decisions cannot be considered in isolation. Any decision must consider all future decisions also. One method to do this is to evaluate all possible *paths* - sequences of actions leading from the starting state to an end state. However, the number of paths quickly becomes large. Even a small deterministic problem with a time horizon of 30 stages and three possible actions at each stage will have $3^{30} \approx 10^{14}$ paths. Time horizons are generally longer and more actions are often available. In addition stochastic problems have multiple outcomes for each action. Therefore other approaches are needed. Such methods, exact and approximate, will be described in this section. All of the methods can be used with BSDPs in general but will be described in terms of the MAB and Bayes' return objectives unless stated otherwise.

2.3.1 Dynamic Programming

Dynamic programming was developed by Bellman (1957) as an optimisation method for sequential decision problems and which has very general use for optimisation in problems beyond BSDPs and bandits. The term "programming" as used here refers to planning rather than computer programming. It has been used extensively for deterministic problems but will be presented here in its stochastic form. It works by breaking the problem down into a series of smaller problems (sub-problems). Solving these gives the solution to the original problem. This approach reduces the required computation in two ways:

1. The solutions to smaller sub-problems can be re-used in solving larger sub-problems which reduces repeated calculation of shared sections of paths.

2. A policy can be found to be suboptimal and so eliminated without evaluating

the whole length of any paths that result from that policy (see the Principle of Optimality below).

Dynamic programming is based on the calculation of the *value* $V(s,t)$ of any state s in the state space $\mathcal{S}$ at time t. Here $V(s,t)$ will be taken to be the maximum total expected reward that can be gained when starting from state s at time t (i.e. when an optimal policy is followed). The value is an important concept because of the Principle of Optimality (which holds for Markov decision problems), as stated in Bellman (1957):

"An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

So to find an optimal policy from the starting state, only policies that are optimal from each subsequent state need be considered. These are exactly the policies that are needed to calculate the state values. Therefore, to find an optimal policy it is sufficient to calculate the values of all states at all times.

Let $r(s,a,t)$ be a reward function and $\pi(s' \mid s,a)$ be the transition probability from state s to any state s' after taking action a, then the value of a state is given by the recursive *value function* (also known as the Bellman equation),

$$V(s,t) = \max_{a \in \mathcal{A}} \left[r(s,a,t) + \gamma \sum_{s' \in S} \pi(s' \mid s,a) V(s', t+1) \right]$$

where γ is a constant discount factor an $\mathcal{A}$ is the available action space. For our purposes here it will be assumed that the process has time T at which the process terminates. The value of states at this time can be found directly,

$$V(s,T) = \max_{a \in \mathcal{A}} [r(s,a,T)].$$

Dynamic programming works backwards sequentially from these end states to give the values of all states. This is called *backward induction.* The value $V(s_0, 0)$ of the initial state s_0 at time 0 is the optimal reward and the recursion sequence that calculates this value gives the optimal policy. A infinite horizon process may not terminate but, for discounted rewards, a finite horizon approximation can be used which tends to the solution as the horizon tends to infinity since received rewards become small as time increases (Blackwell 1962). Regret-based problems use an infinite horizon with undiscounted rewards so dynamic programming is inappropriate in those cases.

Dynamic programming as a solution method for BDSPs has been studied extensively and theoretical results have shown that this method can be used to find the optimal solution (e.g. Rieder 1975). Dynamic programming is a much more efficient method than complete enumeration of paths and has been applied extensively but it still quickly runs into computational limits as the number of calculations required grow exponentially with the number of possible actions and outcomes. A wide range of methods of computation besides backward induction have been developed to calculate the value function of states and the term dynamic programming is often used to refer to this collection of algorithms rather than a single method. For a full overview of DP techniques see Bertsekas (2012).

An even more general family of algorithmic techniques for estimating value functions comes under the term *approximate dynamic programming*. This is itself an example of a BSDP where paths are explored with resources being progressively concentrated on paths more likely to be part of an optimal policy. The problem has been studied in different fields and the main texts on the subject are Powell (2007), Sutton and Barto (1998) and Bertsekas and Tsitsiklis (1996).

Dynamic programming can be applied to the Bernoulli MAB as given in Section 1.3. For each action there are two possible outcomes where $\Pr(y_t = 1 \mid a_t = a, \alpha_{a,t}, \beta_{a,t}) = \mu_{a,t}$. The value function for state $s = (\alpha_{1,t}, \beta_{1,t}, \dots, \alpha_{k,t}, \beta_{k,t})$ at time t is,

$$V(s,t) = \max_{a \in \mathcal{A}} \Big\{ \mu_{a,t} \Big[1 + \gamma V_{t+1}(s^+, t) \Big] + (1 - \mu_{a,t}) \gamma V_{t+1}(s^-, t) \Big\}.$$

where $s^+ = (\alpha_{t,1}, \beta_{t,1}, \ldots, \alpha_{t,a}+1, \beta_{t,a}, \ldots, \alpha_{t,k}, \beta_{t,k})$ and $s^- = (\alpha_{t,1}, \beta_{t,1}, \ldots, \alpha_{t,a}, \beta_{t,a}+1, \ldots, \alpha_{t,k}, \beta_{t,k})$ are the states reached after a success and failure respectively. Using the finite horizon approximation described earlier we terminate the process at some time T where states have value $V(s,T) = \max_{a \in \mathcal{A}} \mu_{a,T}$.

Although there are only two outcomes at each stage the computation required still increases exponentially with the number of available arms and evaluating the value function is impractical for as few as four arms (Gittins et al. 2011). The next section will describe an alternative approach, based on decomposing the dynamic programming problem into smaller subproblems, which gives an optimal policy for the MAB.

2.3.2 The Gittins Index

An *index* is a numerical value that can be calculated for each arm for a given state independently of other arms. An *index policy* always chooses the alternative with the greatest index for its current state. Gittins and Jones (1974) and Gittins (1979) proved that the optimal policy for the MAB *is* an index policy. Such an index was found and termed the *dynamic allocation index* which is now usually referred to as the *Gittins index* (GI). This was a breakthrough since an optimal sequence of actions could now be taken using *forward induction* which is much simpler to perform than the backwards induction of dynamic programming.

The GI values are found by solving a *stopping problem* for each arm. This is itself a MAB involving the arm we wish to find the GI for and a *standard arm* of known constant reward λ. The index value is the value of λ for which we are indifferent between the two arms in the initial action. It is a stopping problem since any time it is optimal to choose the standard arm it will be optimal to continue to choose that arm indefinitely since no new information will be received from that point on. The stopping problem still requires a method such as dynamic programming to solve but this is an independent problem for each arm and so the MAB has been transformed from one k-dimensional problem to k one-dimensional problems. Hence computational cost rises linearly with k rather than exponentially.

A large number of proofs for the index theorem exist in addition to the original publication. For example, Whittle (1980), Tsitsiklis (1994) and Bertsimas and Niño-Mora (1996). An intuitive explanation of why the indexation is optimal comes from the proof given by Weber (1992). In this, each arm has a fixed cost that must be paid each time it is used. It is only profitable for the gambler to play a given arm if the cost is less than the value of playing. There is a cost for any arm, called its fair charge, at which the gambler is indifferent to playing that arm. The fair charge can be compared between different arms and so provides a measure of the relative value of each arm. Calculating the fair charges for each arm will give the optimal arm to play.

Index policies are not optimal for BSDPs in general as changes such as irregular discounting or dependence between the arms can make such a policy suboptimal. Even where they are not optimal index policies can still perform very strongly as will be seen in Chapter 3. More detail of conditions for optimality of index policies as well as their general application can be found in Gittins et al. (2011).

GIs cannot usually be found analytically and can be difficult to calculate. This has led to the development of analytical approximations. Brezzi and Lai (2002) give functional approximations for general bandit problems based on the GI for a Wiener process. These results were refined by Yao (2006) for normal rewards by adding a continuity correction, and further by Chick and Gans (2009). These methods are asymptotically accurate as $\gamma \to 1$. In general, the approximations errors are largest for arms for which we have little information. Thus the decisions affected most are those made at early times when rewards are discounted the least which therefore result in the greatest loss of reward. The performance of policies based on some of these approximations will be assessed empirically in Chapter 3. An detailed discussion of the calculation of the GI will be given in Chapter 5.

2.3.3 Optimistic Index Methods

The GI of any arm a takes the form of $\mu_a + l_a$, $l_a \geq 0$. The quantity l_a has been termed the *learning bonus* or *uncertainty bonus*. It represents, for that arm, the value of exploration, that is, the extra future reward that will result from information gained. A reasonable heuristic approach therefore is to use index policies that approximate this value in some way.

Interval Estimation

Interval estimation was first used by Kaelbling (1993). It uses as an uncertainty bonus the mean reward given $\theta = \theta^*$ where θ^* is some upper quantile of the posterior distribution of θ. For example, if arm rewards had distribution $N(\theta_a, 1)$ the index would be

$$\nu_a = \mu_a + z_\alpha \sigma_a$$

where z_α is the $1 - \alpha$ quantile of the standard normal distribution and σ_a is the (Gaussian) posterior standard deviation for arm a. There is no particular value of α that is correct and so z_α can be regarded as a tunable parameter. Interval estimation was compared to the knowledge gradient policy in Ryzhov et al. (2012) where the performance of the policy was found to be very sensitive to the value of z_α used.

Upper Confidence Bound Methods

Upper confidence bound methods (or *UCB*) are a family of policies designed to meet the regret optimality criteria. The name was originally used for a policy in Auer et al. (2002) but earlier examples exist (Agrawal 1995; Katehakis and Robbins 1995). In order to be asymptotically consistent arms must be sampled infinitely often and the policy does this by making the index a function of the number of times the arm has been pulled up to the current time. There are many upper confidence bound policies, a number of which are designed for a specific problem or using a particular reward distribution. They have been largely studied under regret criteria with many obtaining logarithmic regret growth. Recent methods with general application include Cappé et al. (2013) and Kaufmann et al. (2012).

2.3.4 Stochastic Methods

Semi-uniform Strategies

Semi-uniform strategies are so named as they use a mixed strategy of selecting either (i) the greedy action or, (ii) an action uniformly at random. Therefore there is an explicit mix of pure exploitation with pure exploration.

The simplest and most well known method is *ϵ-greedy*, first described in Watkins (1989). The greedy action is chosen with probability $1 - \epsilon$ where $\epsilon \in [0, 1]$ is a parameter, called the *exploration rate*, to be chosen by the user. With $\epsilon > 0$ the policy is guaranteed to eventually gather all available information. However, it does not use this information well as it explores at a constant rate over time and so does not converge to always selecting the best arm. Ideally there would be more exploration at early times while there is greater uncertainty, becoming more greedy as reward estimates improve.

Two extensions which seek to correct this issue are *ϵ-decreasing* and *ϵ-first*. In *ϵ-decreasing* the value of ϵ decreases over time. In its first presentation in Cesa-Bianchi and Fischer (1998) the value of ϵ used was $log(t)/t$, where t is the current time.

Improved results were found by Auer et al. (2002) with an algorithm that uses the difference between the reward expectations of the two currently best alternatives. The *ϵ-first* approach divides the time into an early pure exploration phase before switching to pure exploitation (see, for example, Mannor and Tsitsiklis 2004).

Semi-uniform strategies have the advantage that they make few statistical assumptions, the only requirement being to identify the greedy arm. For this reason they have been popular in reinforcement learning (see Sutton and Barto 1998). The lack of assumptions can lead to under-performance in problems, such as the MAB, where parametric assumptions are usually made. In addition there are several other weaknesses. Uniform exploration means that the quality of alternatives is not considered when exploring. This leads to a lot of suboptimal choices especially when the number of alternatives is large. A second problem is that all the variants require parameter to be chosen and behaviour will depend on these. Numerical results can exaggerate the quality of such methods by assuming that the best parameters are selected which will rarely be the case in practice.

Softmax

Probability matching strategies are a range of methods which choose actions according to *choice probabilities* assigned to each. These improve on the uniform exploration of semi-uniform strategies by using higher probabilities for better actions (on some measure of quality). The term probability matching should not be confused with Thompson Sampling (discussed below) which, although using a similar idea, has different origins.

One of the most popular of these strategies is *Softmax* or *Boltzmann exploration* (Luce 1959). Here the choice probability for any action a is given by the Boltzmann distribution,

$$\frac{e^{\mu_a/\tau}}{\sum_{j=1}^{k} e^{\mu_j/\tau}}$$

where μ_a is a current estimate of expected reward of action a and τ is a temperature parameter chosen by the user. Low τ tend to pure exploitation while high τ tend to pure exploration. Other estimates besides μ could be used as measures of the quality but μ is most common. Variations exist where the temperature parameter varies with time (Cesa-Bianchi and Fischer 1998) and where the probabilities are weighted by favouring recent high rewards (Auer et al. 2002). As with semi-uniform methods the requirement for a tuning parameter is a downside. In addition, the method ignores any uncertainty in the estimate of expected reward.

Thompson Sampling

Thompson sampling, also known as *probability matching*, was first proposed by Thompson (1933). It selects arms according to the Bayesian probability, taken in expectation over the posterior, that their true expected reward is best. This is an advance over Softmax since it uses the full posterior distribution rather than just the point estimate used by Softmax. Although it is possible to find the choice probabilities for all arms then sample from them, it is simpler and equivalent to sample directly from the arm posteriors. A $\tilde{\theta}_a$ is drawn from the posterior distribution $g(\theta_a \mid \cdot)$ for each arm $a \in \mathcal{A}$ and the arm selected is $\arg\max_a \tilde{\theta}_a$.

Whilst this method has existed for over 80 years, it is only recently that it has become popular (e.g. Scott 2010). Recent work have found it to have good theoretical properties on asymptotic regret-based criteria (e.g. May et al. 2012; Honda and Takemura 2014; Agrawal and Goyal 2013; Korda et al. 2013), including favourable theoretical comparisons with upper confidence bounding methods (Russo and Van Roy 2014; Cowan et al. 2015). These references also give evidence that empirical performance for Thompson sampling is better than for upper confidence bounding methods.

2.4 Multi-arm Bandit Variations

A large number of variations have grown out of the basic bandit framework given in Section 1.3. This section will give a brief overview of some of the more important of these. The variations can loosely be grouped into changes to rewards, available actions or feedback. Chapters 3, 4 and 5 each contain further references relevant to the specific work in each chapter. More extensive references on MABs, the Gittins index and index ideas in general can be found in Gittins et al. (2011) and Mahajan and Teneketzis (2008).

A simple way in which the received rewards change is to alter the discount factor. The most obvious of these is to introduce a finite time after which rewards are no longer received (effectively discounted to zero). Chapter 3 contains work on this variation when rewards up to the horizon end are undiscounted or discounted at a constant rate. However discount sequences can be quite general and a thorough study of these can be found in Berry and Fristedt (1985).

A much more difficult problem is created by allowing the arms themselves to evolve over time. In *restless bandits* (Whittle 1988; Weber and Weiss 1990) such changes can occur at any time whether the arm is selected or not. A special case of this is explored in Mellor (2014) when arm reward distributions reset abruptly with some probability at each time.

The problem where the arms reward distributions stay the same but new arms can arrive over time has been referred to as *arm-acquiring bandits*, *bandit processes with arrivals* or *branching bandits*. This is a very different problem from restlessness and an index policy is optimal (Whittle 1981; Weiss 1988).

In *contextual bandits* (e.g. May et al. 2012; Yang and Zhu 2002) the rewards depend on covariates (called a *context*) which are given at each time before selecting the arm.

The context represents information that is relevant only to the current decision, for example, characteristics (sex, age etc.) of the current patient in a medical trial. It therefore gives a method by which actions can be made more relevant to the current situation. The context is common to all arms and affects only the rewards at the current time.

Another group of extensions restricts the availability of actions in some way. There could be *precedence constraints* which limit the order in which arms can be chosen (Glazebrook 1991) or some arms may not be *available* for selection at all times (Dayanik et al. 2008; Kleinberg et al. 2010). There can also be a penalty (delay or cost) when *switching* from one bandit to another (Van Oyen et al. 1992; Asawa and Teneketzis 1996; Jun 2004). Each of these is a realistic feature of job scheduling problems where jobs have to be completed in order, where machines break down or are unavailable for use, and where changing from one job to another requires a new setup.

Variations can come from extending the available actions. The MAB with multiple plays (Whittle 1988; Pandelis and Teneketzis 1999) allows more than one arm to be selected at a time. Job scheduling is again an example application where multiple machines are available to process a job. Here total rewards are additive over the individual arms rewards but can be a more general function of the arms chosen (Chen et al. 2013). The arms chosen can be thought of as corresponding to a binary vector with a 1 where the arm is selected. This can be generalised so that the action at each time is to choose a vector (possibly chosen from a limited set) with the reward being a function of this vector (Ginebra and Clayton 1995; Rusmevichientong and Tsitsiklis 2010). The exploration part of the problem is then concerned with learning about the reward function rather than arms.

If the arms are *correlated* then this affects both the structure of the underlying arms and the information received from rewards as we now can learn about arms beyond the one selected. This is relevant in applications where we have prior information that some arms are similar. For example if we are testing different dosages of a clinical treatment then we would expect similar dosages to bring a more similar response than that of dosages much larger or smaller. There is no one method of modelling this similarity. In a Bayesian setting we could treat arms as having a joint belief distribution either directly or by using common hyperparameters. An example of the former from Ryzhov et al. (2012) which uses a multivariate normal belief distribution over all arms is investigated in Chapter 3. The latter type is studied in an intelligence setting in Dimitrov et al. (2015) where arms are edges on a graph representing communication links between vertices (representing people). The rewards from the arms/edges depend on adjacent vertices so arms with a vertex in common are correlated. Brown and Smith (2013) used a correlation structure also based on a graph to choose where to explore for oil. Possible sites form clusters which have correlated chances of finding oil. The solution method uses theory from another bandit variant, *bandit superprocesses* (Whittle 1980). Here arms themselves correspond to decision processes - as well as choosing which arm to play there is additional choice about how it is played. In Brown and Smith (2013) the clusters of possible sites form arms and the sites within each cluster the additional option of how to play the arm.

Chapter 3

On the Identification and Mitigation of Weaknesses in the Knowledge Gradient Policy for Multi-Armed Bandits

3.1 Introduction

Bayes sequential decision problems (BSDPs) constitute a large class of optimisation problems in which decisions (i) are made in time sequence and (ii) impact the system of interest in ways which may be not known or only partially known. Moreover, it is possible to learn about unknown system features by taking actions and observing outcomes. This learning is modelled using a Bayesian framework. One important subdivision of BSDPs is between offline and online problems. In offline problems some decision is required at the end of a time horizon and the purpose of actions through the horizon is to accumulate information to support effective decision-making at its end. In online problems each action can bring an immediate payoff in addition to yielding information which may be useful for subsequent decisions. This chapter is concerned with a particular class of online problems although it should be noted that some of the solution methods have their origins in offline contexts.

The sequential nature of the problems coupled with imperfect system knowledge means that decisions cannot be evaluated alone. Effective decision-making needs to account for possible future actions and associated outcomes. While standard solution methods such as stochastic dynamic programming can in principle be used, in practice they are computationally impractical and heuristic approaches are generally required. One such approach is the knowledge gradient (KG) heuristic. Gupta and Miescke (1996) originated KG for application to offline ranking and selection problems. After a period of time in which it appears to have been studied little, Frazier et al. (2008) expanded on KG's theoretical properties. It was adapted for use in online decision-making by Ryzhov et al. (2012) who tested it on multi-armed bandits (MABs) with Gaussian rewards. They found that it performed well against an index policy which utilised an analytical approximation to the Gittins index; see Gittins et al. (2011). Ryzhov et al. (2010) have investigated the use of KG to solve MABs with exponentially distributed rewards while Powell and Ryzhov (2012) give versions for Bernoulli, Poisson and uniform rewards, though without testing performance. They propose the method as an approach to online learning problems quite generally, with particular emphasis on its ability to handle correlated arms. Initial empirical results were promising but only encompassed a limited range of models. This chapter utilises an important sub-class of MABs to explore properties of the KG heuristic for online use. Our investigation reveals weaknesses in the KG approach. We *inter alia* propose modifications to mitigate these weaknesses.

In Section 3.2 we describe a class of exponential family MABs that we will focus on, together with the KG policy for them. Our main analytical results revealing weaknesses in KG are given in Section 3.3. Methods aimed at correcting these KG errors are discussed in Section 3.4 and are evaluated in a computational study which is reported in Section 3.5. In this study a range of proposals are assessed for Bernoulli and exponential versions of our MAB models. Gaussian MABs have characteristics which give the operation of KG distinctive features. The issues for such models are discussed in Section 3.6, together with an associated computational study in Section 3.6.1. Section 3.7 identifies some key conclusions to be drawn.

3.2 A class of exponential family multi-armed bandits

3.2.1 Multi-Armed Bandit Problems for Exponential Families

We consider multi-armed bandits (MABs) with geometric discounting operating over a time horizon $T \in \mathbb{Z}^+ \cup \{\infty\}$ which may be finite or not. Rewards are drawn from exponential families with independent conjugate priors for the unknown parameters. More specifically the set up is as follows:

1. At each decision time $t \in \{0, 1, \ldots, T-1\}$ an action $a \in \{1, \ldots, k\}$ is taken. Associated with each action, a, is an (unknown) parameter, which we denote as θ_a. Action a (*pulling arm* a) yields a reward which is drawn from the density (relative to some σ-finite measure on $\mathbb{R}$)

$$f(y \mid \theta_a) = e^{\theta_a y - \psi(\theta_a)}, y \in \Omega, \theta_a \in \Theta, \qquad (3.2.1)$$

   where $\Omega \subseteq \mathbb{R}$ is the support of f, ψ is a cumulant generating function and parameter space $\Theta \subseteq \mathbb{R}$ is such that $\psi(\theta) < \infty$, $\forall \theta \in \Theta$. Reward distributions are either discrete or absolutely continuous, with Ω a discrete or continuous interval $[\min\Omega, \max\Omega]$ where $-\infty \leq \min\Omega < \max\Omega \leq \infty$. We shall take a Bayesian approach to learning about the parameters θ_a.

2. We assume independent conjugate priors for the unknown θ_a with Lebesgue densities given by

$$g(\theta_a \mid \Sigma_a, n_a) \propto e^{\Sigma_a\theta_a - n_a\psi(\theta_a)}, \theta_a \in \Theta, 1 \leq a \leq k, \tag{3.2.2}$$

where Σ_a and n_a are known hyper-parameters representing, respectively, the Bayesian sum of rewards and the Bayesian number of observations for arm a. This then defines a predictive density

$$p(y \mid \Sigma_a, n_a) = \int_\Theta f(y \mid \theta_a) g(\theta_a \mid \Sigma_a, n_a) d\theta_a \tag{3.2.3}$$

which has mean $\frac{\Sigma_a}{n_a}$. Bayesian updating following an observed reward y on arm a produces a posterior $p(\theta_a|y) = g(\theta_a|\Sigma_a + y, n_a + 1)$. Thus at each time we can define an arm's *informational state* as the current value of hyper-parameters Σ_a, n_a, such that the posterior for θ_a given the observations to date is $g(\theta_a|\Sigma_a, n_a)$. The posterior for each arm is independent so the informational states of arms not pulled at t are unchanged.

3. The total return when reward y_t is received at time t is given by $\sum_{t=0}^{T-1} \gamma^t y_t$, where discount rate γ satisfies either $0 < \gamma \leq 1$ when $T < \infty$ or $0 < \gamma < 1$ when $T = \infty$. The objective is to design a policy, a rule for choosing actions, to maximise the *Bayes' return,* namely the total return averaged over both realisations of the system and prior information.

The current informational state for all arms, denoted $(\mathbf{\Sigma}, \mathbf{n}) = \{(\Sigma_a, n_a), 1 \leq a \leq k\}$ summarises all the information in the observations up to the current time.

When $0 < \gamma < 1, T = \infty$ the Bayes' return is maximised by the *Gittins Index* (GI) policy, see Gittins et al. (2011). This operates by choosing, in the current state, any action a, satisfying

$$\nu^{GI}(\Sigma_a, n_a, \gamma) = \max_{1 \le b \le k} \nu^{GI}(\Sigma_b, n_b, \gamma)\,, \tag{3.2.4}$$

where ν^{GI} is the *Gittins index.* We describe Gittins indices in Section 3.4 along with versions adapted for use in problems with $0 < \gamma \le 1, T < \infty$. Given the challenge of computing Gittins indices and the general intractability of deploying dynamic programming to solve online problems, the prime interest is in the development of heuristic policies which are easy to compute and which come close to being return maximising.

3.2.2 The Knowledge Gradient Heuristic

The *knowledge gradient policy* (KG) is a heuristic which bases action choices both on immediate returns $\frac{\Sigma_a}{n_a}$ and also on the changes in informational state which flow from a single observed reward. It is generally fast to compute. To understand how KG works for MABs suppose that the decision time is t and that the system is in information state $(\mathbf{\Sigma}, \mathbf{n})$ then. The current decision is taken to be the last opportunity to learn and so from time $t+1$ through to the end of the horizon whichever arm has the highest mean reward following the observed reward at t will be pulled at all subsequent times. With this informational constraint, the best arm to pull at t (and the action mandated by KG in state $(\mathbf{\Sigma}, \mathbf{n})$) is given by

$$A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = \arg\max_a \left\{ \frac{\Sigma_a}{n_a} + H(\gamma, s) \max_{1 \le b \le k} E\left(\frac{\Sigma_b + I_a Y}{n_b + I_a} \mid \mathbf{\Sigma}, \mathbf{n}, a \right) \right\}, \tag{3.2.5}$$

where Y is the observed reward at t and I_a is an indicator taking the value 1 if action a is taken at t, 0 otherwise. The conditioning indicates that the reward Y depends upon the current state $(\mathbf{\Sigma}, \mathbf{n})$ and the choice of action a. The constant $H(\gamma, s)$ is a suitable multiplier of the mean return of the best arm at $t + 1$ to achieve an accumulation of rewards for the remainder of the horizon (denoted here by $s = T - t$). It is given by

$$H(\gamma, s) = \begin{cases} \frac{\gamma(1-\gamma^{s-1})}{1-\gamma} & \text{if } 0 < \gamma < 1, T < \infty \\ \frac{\gamma}{1-\gamma} & \text{if } 0 < \gamma < 1, T = \infty \\ s - 1 & \text{if } \gamma = 1, T < \infty. \end{cases} \tag{3.2.6}$$

KG can be characterised as the policy resulting from the application of a single policy improvement step to a policy which always pulls an arm with the highest prior mean return throughout. Note that for $(\gamma, T) \in (0, 1) \times \mathbb{Z}^+$, $H(\gamma, s)$ is increasing in both γ (T fixed) and in T (γ fixed). For any sequence of (γ, T) values approaching the limit $(1, \infty)$ in a manner which is co-ordinatewise increasing, the value of $H(\gamma, s)$ diverges to infinity. This fact is utilised heavily in Section 3.3.

We now develop an equivalent characterisation of KG based on Ryzhov et al. (2012) which will be more convenient for what follows. We firstly develop an expression for the change in the maximal mean reward available from any arm when action a is taken in state $(\mathbf{\Sigma}, \mathbf{n})$. We write

$$\nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n}) = E\left\{ \max_{1\le b\le k} \mu_b^{+1} - \max_{1\le b\le k} \mu_b \mid \mathbf{\Sigma}, \mathbf{n}, a \right\}, \tag{3.2.7}$$

where μ_b is the current arm b mean return $\frac{\Sigma_b}{n_b}$ and μ_b^{+1} is the mean return available from arm b at the next time conditional on the observed reward resulting from action a. Please note that μ_b^{+1} is a random variable. It is straightforward to show that

$$A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = \arg\max_{1\le a\le k} \left\{ \mu_a + H(\gamma, s)\nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n}) \right\}. \tag{3.2.8}$$

Hence KG gives a score to each arm and chooses the arm of highest score. It is not an index policy because the score depends upon the informational state of arms other than the one being scored. That said, there are similarities between KG scores and Gittins indices. The Gittins index $\nu^{GI}(\Sigma_a, n_a)$ exceeds the mean return $\frac{\Sigma_a}{n_a}$ by an amount termed the *uncertainty* or *learning bonus*. This bonus can be seen as a measure of the value of exploration in choosing arm a. The quantity $H(\gamma, s)\nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n})$ in the KG score is an alternative estimate of the learning bonus. Assessing the accuracy of this estimate will give an indication of the strengths and weaknesses of the policy.

3.2.3 Dominated arms

In our discussion of the deficiencies of the KG policy in the next section we shall focus, among other things, on its propensity to pull arms which are suboptimal to another arm with respect to *both* exploitation and exploration. Hence there is an alternative which is better both from an immediate return and from an informational perspective. We shall call such arms *dominated*. We begin our discussion with a result concerning properties of Gittins indices established by Yu (2011).

Theorem 3.2.1. *The Gittins index* $\nu^{GI}\left(c\Sigma, cn, \gamma\right)$ *is decreasing in* $c \in \mathbb{R}^+$ *for any fixed* Σ, n, γ *and is increasing in* Σ *for any fixed* c, n, γ.

We proceed to a simple corollary whose statement requires the following definition.

Definition 3.2.2. *An arm in state* (Σ, n) *dominates one in state* (Σ', n') *if and only if* $\frac{\Sigma}{n} > \frac{\Sigma'}{n'}$ *and* $n < n'$.

Corollary 3.2.3. *The GI policy never chooses dominated arms.*

Proof. We will show that $\nu^{GI}(\Sigma, n, \gamma) > \nu^{GI}(\Sigma', n', \gamma)$ when $\frac{\Sigma}{n} > \frac{\Sigma'}{n'}$ and $n < n'$.

Let $c = \frac{n'}{n}$, then from Theorem 3.2.1 and since $c > 1$ we have $\nu^{GI}(\Sigma, n, \gamma) > \nu^{GI}(c\Sigma, cn, \gamma) = \nu^{GI}(c\Sigma, n', \gamma)$. From the condition $\frac{\Sigma}{n} > \frac{\Sigma'}{n'}$ we have that $c\Sigma > \Sigma'$ and it then follows from the monotonicity of $\nu^{GI}(\Sigma, n, \gamma)$ with Σ given in Theorem 3.2.1 that $\nu^{GI}(c\Sigma, n', \gamma) > \nu^{GI}(\Sigma', n', \gamma)$.

□

Hence pulling dominated arms can never be optimal for infinite horizon MABs. We shall refer to the pulling of a dominated arm as a *dominated action* in what follows. Exploration of the conditions under which KG chooses dominated actions is a route to an understanding of its deficiencies and prepares us to propose modifications to it which achieve improved performance. This is the subject matter of the following two sections.

3.3 The KG policy and dominated actions

3.3.1 Conditions for the choice of dominated actions under KG

This section will elucidate sufficient conditions for the KG policy to choose dominated arms. A key issue here is that the quantity ν_a^{KG} (and hence the KG learning bonus) can equal zero in cases where the *true* learning bonus related to a pull of arm a may be far from zero. Ryzhov et al. (2012) stated that $\nu_a^{KG} > 0$. However, crucially, that paper only considered Gaussian bandits. The next lemma is fundamental to the succeeding arguments. It says that, for sufficiently high γ, the KG policy will choose the arm with the largest ν^{KG}.

Lemma 3.3.1. $\forall\, \mathbf{\Sigma}, \mathbf{n}, t$ *for which* $max_a \nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n}) > 0\ \exists \gamma^*, T^*$ *such that* $\gamma > \gamma^*, T > T^* \Rightarrow A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = \arg\max_a \nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n})$.

Proof. The result is a trivial consequence of the definition of the KG policy in Section 3.2.2 together with the fact that $H(\gamma, s)$ diverges to infinity in the manner described in Section 3.2. □

The next result gives conditions under which $\nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n}) = 0$. Informally, $\nu_a^{KG} = 0$ if no outcome from a pull on arm a will change which arm has maximal mean value.

When $a \in \arg\max_b \mu_b$ this depends on the lower tail of the distribution of Y_a while if $a \notin \arg\max_b \mu_b$ it depends on the upper tail.

Lemma 3.3.2. *Let* $C_a(\boldsymbol{\Sigma}, \mathbf{n})$ *denote* $\max_{b \neq a} \mu_b = \max_{b \neq a} \frac{\Sigma_b}{n_b}$. *If* $a \in \arg\max_b \mu_b$ *and the observation state space,* Ω, *is bounded below with minimum value* $\min \Omega$ *then*

$$\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) = 0 \Leftrightarrow \frac{\Sigma_a + \min \Omega}{n_a + 1} \geq C_a(\boldsymbol{\Sigma}, \mathbf{n}); \tag{3.3.1}$$

while if $a \notin \arg\max_b \mu_b$ *and* Ω *is bounded above with maximum value* $\max \Omega$ *then*

$$\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) = 0 \Leftrightarrow \frac{\Sigma_a + \max \Omega}{n_a + 1} \leq C_a(\boldsymbol{\Sigma}, \mathbf{n}). \tag{3.3.2}$$

In cases where $a \in \arg\max_b \mu_b$ *with* Ω *unbounded below, and where* $a \notin \arg\max_b \mu_b$ *with* Ω *unbounded above, we have* $\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) > 0$.

Proof. Note that

$$\begin{aligned}\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) &= E_{Y_a}\left[\max_b \mu_b^{+1} - \max_b \mu_b \mid \boldsymbol{\Sigma}, \mathbf{n}, a\right] \\ &= E_{Y_a}\left[\max\left(\mu_a^{+1}, C_a(\boldsymbol{\Sigma}, \mathbf{n})\right) \mid \boldsymbol{\Sigma}, \mathbf{n}, a\right] - \max_b \mu_b. \end{aligned} \tag{3.3.3}$$

Hence

$$\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) = 0 \Leftrightarrow E_{Y_a}\left[\max\left(\mu_a^{+1}, C_a(\boldsymbol{\Sigma}, \mathbf{n})\right) \mid \boldsymbol{\Sigma}, \mathbf{n}, a\right] = \max_b \mu_b. \tag{3.3.4}$$

If $a \in \arg\max_b \mu_b$ and so $\max_b \mu_b = \mu_a$ then, observing that

$$E_{Y_a}\left[\max\left(\mu_a^{+1}, C_a(\boldsymbol{\Sigma}, \mathbf{n})\right) \mid \boldsymbol{\Sigma}, \mathbf{n}, a\right] \geq E_{Y_a}\left[\mu_a^{+1} \mid \boldsymbol{\Sigma}, \mathbf{n}, a\right] = \mu_a, \tag{3.3.5}$$

we infer from equation (3.3.4) that $\nu_a^{KG}(\boldsymbol{\Sigma}, \mathbf{n}) = 0$ if and only if

$$\max\left(\mu_a^{+1}, C_a(\boldsymbol{\Sigma}, \mathbf{n})\right) = \mu_a^{+1} \Leftrightarrow \mu_a^{+1} \geq C_a(\boldsymbol{\Sigma}, \mathbf{n}) \tag{3.3.6}$$

with probability 1 under the distribution of Y_a. Under our set up as described in Section 3.2, this condition is equivalent to the right hand side of (3.3.1). If $a \notin \arg\max_b \mu_b$

then $max_b\mu_b = C_a\left(\boldsymbol{\Sigma},\mathbf{n}\right)$ and so, suitably modifying the previous argument, we infer that $\nu_a^{KG}\left(\boldsymbol{\Sigma},\mathbf{n}\right) = 0$ if and only if

$$\max\left(\mu_a^{+1}, C_a\left(\boldsymbol{\Sigma},\mathbf{n}\right)\right) = C_a\left(\boldsymbol{\Sigma},\mathbf{n}\right) \Leftrightarrow \mu_a^{+1} \leq C_a\left(\boldsymbol{\Sigma},\mathbf{n}\right) \tag{3.3.7}$$

with probability 1 under the distribution of Y_a. Under our set up as described in Section 3.2, this condition is equivalent to the right hand side of (3.3.2). The unbounded cases follow directly from the formula for $\nu_a^{KG}(\boldsymbol{\Sigma},\mathbf{n})$ as the change in μ_a due to an observation has no finite limit in the direction(s) of unboundedness. This completes the proof. □

So whether $\nu_a^{KG}(\boldsymbol{\Sigma},\mathbf{n}) = 0$ depends on a different tail of the distribution of Y_a depending on whether $a \in \arg\max_b \mu_b$ or not. This asymmetry is important in what follows.

Theorem 3.3.3. *If Ω is bounded below then there are choices of $\boldsymbol{\Sigma},\mathbf{n}$,$\gamma, T$ for which the KG policy chooses dominated arms.*

Proof. If we consider cases for which

$$\frac{\Sigma_1}{n_1} > \frac{\Sigma_2}{n_2},\ n_1 < n_2;\ \Sigma_b = c\Sigma_2,\ n_b = cn_2,\ 3 \leq b \leq k,\ c \geq 1 \tag{3.3.8}$$

then it follows that $\mu_2 = \mu_b, \nu_2^{KG} \geq \nu_b^{KG}, 3 \leq b \leq k$, and all arms except 1 and 2 can be ignored in the discussion. We first suppose that Ω unbounded above. It follows from Lemma 3.3.2 that $\nu_2^{KG}(\boldsymbol{\Sigma},\mathbf{n}) > 0$. Since $\min\Omega > -\infty$, we can further choose $(\boldsymbol{\Sigma},\mathbf{n})$ such that

$$\frac{\Sigma_1 + \min\Omega}{n_1 + 1} \geq \frac{\Sigma_2}{n_2} = C_1(\boldsymbol{\Sigma},\mathbf{n}). \tag{3.3.9}$$

From the above result we infer that $\nu_1^{KG}(\boldsymbol{\Sigma},\mathbf{n}) = 0$. We now suppose that Ω is bounded above, and hence that $\infty > \max\Omega > \min\Omega > -\infty$. Choose $(\boldsymbol{\Sigma},\mathbf{n})$ as follows: $\Sigma_1 = \max\Omega + 2\min\Omega, n_1 = 3, \Sigma_2 = \max\Omega + 3\min\Omega, n_2 = 4$. It is trivial that these choices mean that arm 1 dominates arm 2. We have that

$$\frac{\Sigma_1 + \min\Omega}{n_1 + 1} = \frac{\max\Omega + 3\min\Omega}{4} = \frac{\Sigma_2}{n_2} = C_1(\boldsymbol{\Sigma},\mathbf{n}) \tag{3.3.10}$$

and hence that $\nu_1^{KG}(\mathbf{\Sigma}, \mathbf{n}) = 0$. Further we have that

$$\frac{\Sigma_2 + \max \Omega}{n_2 + 1} = \frac{2 \max \Omega + 3 \min \Omega}{5} > \frac{\Sigma_1}{n_1} = C_2(\mathbf{\Sigma}, \mathbf{n}) \tag{3.3.11}$$

and hence that $\nu_2^{KG}(\mathbf{\Sigma}, \mathbf{n}) > 0$. In both cases discussed (i.e., Ω bounded and unbounded above) we conclude from Lemma 3.3.1 the existence of t, γ^*, T^* such that $\gamma > \gamma^*, T > T^* \Rightarrow A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = \arg\max_a \nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n}) = 2$, which is a dominated arm, as required. This concludes the proof. □

Although the part of the above proof dealing with the case in which Ω is bounded above identifies a specific state in which KG will choose a dominated arm when $H(\gamma, s)$ is large enough, it indicates how such cases may be identified more generally. These occur when the maximum positive change in the mean of the dominated arm ($\mu_2 \to \mu_2^{+1}$) is larger than the maximum negative change in the mean of the dominating arm ($\mu_1 \to \mu_1^{+1}$). This can occur both when the Y_a have distributions skewed to the right and also where the corresponding means are both small, meaning that a large y can effect a greater positive change in μ_2 than can a small y a negative change in μ_1. A detailed example of this is given for the Bernoulli MAB in the next section. Similar reasoning suggests that the more general sufficient condition for KG to choose dominated arms, namely $\nu_2^{KG}(\mathbf{\Sigma}, \mathbf{n}) > \nu_1^{KG}(\mathbf{\Sigma}, \mathbf{n})$ with arm 2 dominated, will hold in cases with Ω unbounded above if the distribution of Y_a has an upper tail considerably heavier than its lower tail.

3.3.2 Stay-on-the winner rules

Berry and Fristedt (1985) demonstrated that optimal policies for MABs with Bernoulli rewards and general discount sequences (including all cases considered here) have a stay-on-the-winner property. If arm a is optimal at some epoch and a pull of a yields a success ($y_a = 1$) then arm a continues to be optimal at the next epoch. Yu (2011) extends this result to the exponential family considered here in the following way: an optimal arm continues to be optimal following an observed reward which is sufficiently large. The next result is an immediate consequence.

Lemma 3.3.4. *Suppose that Ω is bounded above. If arm a is optimal at some epoch and a pull of a yields a maximal reward ($y_a = \max \Omega$) then arm a is optimal at the next epoch.*

The following result states that the KG policy does not share the stay-on-the-winner character of optimal policies as described in the preceding lemma. In its statement we use $\mathbf{e}_a$ for the $k-$vector whose ath component is 1, with zeros elsewhere.

Proposition 3.3.5. *If Ω is bounded above and below $\exists$ choices of $\mathbf{\Sigma}, \mathbf{n}$,$t, \gamma, T$ and a for which $A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = a$, $A^{KG}(\mathbf{\Sigma} + \max \mathbf{\Omega} \mathbf{e}_a, \mathbf{n} + \mathbf{e}_a, t) \neq a$.*

Proof. For the reasons outlined in the proof of Theorem 3.3.3 we may assume without loss of generality that $k = 2$. As in that proof we consider the state $(\mathbf{\Sigma}, \mathbf{n})$ with $\Sigma_1 = \max \Omega + 2 \min \Omega, n_1 = 3, \Sigma_2 = \max \Omega + 3 \min \Omega, n_2 = 4$. We suppose that a pull of arm 2 yields an observed reward equal to $\max \Omega$. This takes the process state to $(\mathbf{\Sigma} + \max \mathbf{\Omega} \mathbf{e}_2, \mathbf{n} + \mathbf{e}_2, t) = \left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}\right)$, say. We use the dashed notation for quantities associated with this new state. Observe that $\mu_2^{/} > \mu_1^{/}$ and hence that $2 \in \arg\max_b \mu_b^{/}$. We note that

$$\frac{\Sigma_2^{/} + \min \Omega}{n_2^{/} + 1} = \frac{2 \max \Omega + 4 \min \Omega}{6} = \mu_1^{/} = C_2\left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}\right), \tag{3.3.12}$$

which implies via Lemma 3.3.2 that $\nu_2^{KG}\left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}\right) = 0$. We also have that

$$\frac{\Sigma_1^{/} + \max \Omega}{n_1^{/} + 1} = \frac{2 \max \Omega + 2 \min \Omega}{4} > \mu_2^{/} = C_1\left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}\right), \tag{3.3.13}$$

which implies via Lemma 3.3.2 that $\nu_1^{KG}\left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}\right) > 0$. The existence of t, γ, T for which $A^{KG}(\mathbf{\Sigma}, \mathbf{n}, t) = 2$ while $A^{KG}\left(\mathbf{\Sigma}^{/}, \mathbf{n}^{/}, t+1\right) = A^{KG}(\mathbf{\Sigma} + \max \mathbf{\Omega} \mathbf{e}_2, \mathbf{n} + \mathbf{e}_2, t) \neq 2$ now follows from Lemma 3.3.1. □

3.3.3 Examples

We will now give more details of how the KG policy chooses dominated actions in the context of two important members of the exponential family.

Exponential rewards

Suppose that $Y_a \mid \theta_a \backsim Exp(\theta_a)$ and $\theta_a \backsim Gamma(n_a + 1, \Sigma_a)$ which yields the unconditional density for Y_a given by

$$g_a(y) = (n_a + 1)\Sigma_a^{n_a+1}(\Sigma_a + 1)^{-n_a-2}, y \geq 0, \tag{3.3.14}$$

with $E(Y_a) = \frac{\Sigma_a}{n_a}$. Let arm 1 dominate arm 2. For this case $\Omega = [0, \infty)$ and from Lemma 3.3.2, the unboundedness of Ω above means that $\nu_2^{KG}(\mathbf{\Sigma}, \mathbf{n}) > 0$ while $\nu_1^{KG}(\mathbf{\Sigma}, \mathbf{n}) = 0$ if and only if

$$\frac{\Sigma_1}{n_1 + 1} \geq \frac{\Sigma_2}{n_2}. \tag{3.3.15}$$

Hence from Lemma 3.3.1 we can assert the existence of t, γ, T for which KG chooses dominated arm 2 whenever (3.3.15) holds.

Ryzhov and Powell (2011) discuss the online KG policy for exponential rewards in detail. They observe that ν_a^{KG} can be zero but do not appear to recognise that this can yield dominated actions under the policy. Later work, Ding and Ryzhov (2016), showed that this can lead to the offline KG policy never choosing the greedy arm, an extreme case of dominated errors. However, with the online KG policy the greedy arm will eventually be selected as ν_a^{KG} for the other arm tends to zero. These papers note that, in states for which

$$\frac{\Sigma_1}{n_1 + 1} \leq \frac{\Sigma_2}{n_2} \leq \frac{\Sigma_1}{n_1}, \tag{3.3.16}$$

the value of $\nu_1^{KG}(\mathbf{\Sigma}, \mathbf{n})$, while not zero, penalises the choice of the greedy arm relative to other arms in a similar way to the bias which yields dominated actions.

Policies which mitigate such bias are given in the next section and are evaluated in the computational study following.

3.3.4 Bernoulli rewards

Suppose that $Y_a \mid \theta_a \backsim Bern\left(\theta_a\right)$, with $\theta_a \backsim Beta\left(\Sigma_a, n_a - \Sigma_a\right)$ and so $\Omega = \{0,1\}$ and $P\left(Y_a = 1\right) = \frac{\Sigma_a}{n_a} = 1 - P\left(Y_a = 0\right)$. Since Ω is bounded above and below, dominated actions under KG will certainly occur. Demonstrating this in terms of the asymmetric updating of Beta priors can be helpful in understanding the more general case of bounded rewards. Use δ_a^+ and δ_a^- for the magnitudes of the upward and downward change in $E\left(Y_a\right)$ under success and failure respectively. We have

$$\delta_a^+ = \frac{n_a - \Sigma_a}{n_a\left(n_a+1\right)}\ ;\ \delta_a^- = \frac{\Sigma_a}{n_a\left(n_a+1\right)}, \tag{3.3.17}$$

from which we conclude that $\delta_a^+ \geq \delta_a^- \Leftrightarrow \mu_a \leq 0.5$. Prompted by this analysis, consider a case in which $k = 2, \Sigma_1 = \Sigma_2; n_1 + m = n_2$ for some $m \in \mathbb{N}^+$. Arm 1 dominates arm 2. Further, the fact that

$$\frac{\Sigma_1 + \min\Omega}{n_1+1} = \frac{\Sigma_1}{n_1+1} \geq \frac{\Sigma_1}{n_1+m} = \frac{\Sigma_2}{n_2} = C_1\left(\boldsymbol{\Sigma}, \mathbf{n}\right) \tag{3.3.18}$$

implies via Lemma 3.3.2 that $\nu_1^{KG}\left(\boldsymbol{\Sigma}, \mathbf{n}\right) = 0$. From Lemma 3.3.2 we also conclude that

$$\nu_2^{KG}\left(\boldsymbol{\Sigma}, \mathbf{n}\right) > 0 \iff \frac{\Sigma_2 + \max\Omega}{n_2+1} = \frac{\Sigma_1 + 1}{n_1+m+1} > \frac{\Sigma_1}{n_1} = C_2\left(\boldsymbol{\Sigma}, \mathbf{n}\right). \tag{3.3.19}$$

The strict inequality in the right hand side of (3.3.19) will hold whenever $n_1 > (m+1)\,\Sigma_1$. Thus, for suitably chosen t, γ and T, the KG policy will take dominated actions in a wide range of states. Suppose now that $T = \infty$ and hence the immediate claim is that under the condition $n_1 > (m+1)\,\Sigma_1$ the KG policy will take dominated action 2 for γ large enough. We now observe that in practice dominated actions can be taken for quite modest γ. Returning to the characterisation of the KG policy we infer that in the above example, dominated action 2 will be chosen whenever

$$n_1 > (m+1)\,\Sigma_1, \frac{\gamma}{1-\gamma} > \frac{m\,(n_1+m+1)}{\{n_1-(m+1)\,\Sigma_1\}}. \tag{3.3.20}$$

Such errors will often be costly. Note also that the condition $n_1 > (m+1)\,\Sigma_1$ suggests that dominated actions occur more often when arms have small mean rewards. This is investigated further in the computational study following.

Gaussian rewards

Here we have $Y_a \mid \theta_a \backsim N(\theta_a, 1)$ and $\theta_a \backsim N\left(\frac{\Sigma_a}{n_a}, \frac{1}{n_a}\right)$. Hence $\Omega = \mathbb{R}$ is unbounded and if arm a is chosen, the distribution of μ_a^+ is symmetric about μ_a. In this case the KG policy does not choose dominated actions and the value of ν_a^{KG} is always greater for the arm with smaller prior precision n_a. Despite this fact, KG can still take poor decisions by underestimating the learning bonus for the greedy arm. The Gaussian MAB is discussed further in Section 3.6.

3.4 Policies which modify KG to avoid taking dominated actions

In this section we present new policies which are designed to mitigate the defects of the KG approach elucidated in the previous section. The performance of these are assessed along with some earlier proposals, in the numerical study of the next section.

Non-dominated KG (NKG): This proposal modifies standard KG by prohibiting dominated actions. It achieves this by always choosing a non-dominated arm with highest KG score. Any greedy arm is non-dominated and hence one always exists.

Positive KG (PKG): The KG score for a greedy arm reflects a negative change in its posterior mean while that for non-greedy arms reflect positive changes. The PKG policy modifies KG such that for all arms it is positive moves which are registered. It achieves this by modifying the KG scores for each greedy arm a as follows: in the computation of the score replace the quantity $C_a(\mathbf{\Sigma}, \mathbf{n}) = \max_{b \neq a} \mu_b$ by the quantity $C_a^*(\mathbf{\Sigma}, \mathbf{n}) := 2\mu_a - C_a(\mathbf{\Sigma}, \mathbf{n})$. This adjustment transforms the KG scores $\nu_a^{KG}(\mathbf{\Sigma}, \mathbf{n})$ to adjusted values $\nu_a^{PKG}(\mathbf{\Sigma}, \mathbf{n})$. The change maintains the key distance used in the KG calculation as $C_a^* - \mu_a = \mu_a - C_a$ but ensures that it is non-negative. For non-greedy arms b we have $\nu_b^{KG}(\mathbf{\Sigma}, \mathbf{n}) = \nu_b^{PKG}(\mathbf{\Sigma}, \mathbf{n})$.

Theorem 3.4.1. *Policy PKG never chooses a strictly dominated arm.*

Proof. Suppose that arm 2 is strictly dominated by arm 1 such that $\frac{\Sigma_1}{n_1} > \frac{\Sigma_2}{n_2}$ and $n_2 \geq n_1 + 1$. In the argument following we shall suppose that $k = 2$. This is without loss of generality as the addition of any other arm b with $\mu_b \leq \mu_1$ does not effect the PKG score of arm 2 and can only increase the PKG score of the non-dominated arm 1. Given that $\mu_1 > \mu_2$, in order to establish the result, namely that $A^{PKG}(\mathbf{\Sigma}, \mathbf{n}) = 1$ it is enough to establish that $\nu_1^{PKG}(\mathbf{\Sigma}, \mathbf{n}) \geq \nu_2^{PKG}(\mathbf{\Sigma}, \mathbf{n})$. From the definitions of the quantities concerned we have that

$$\begin{aligned}\nu_1^{PKG}(\mathbf{\Sigma}, \mathbf{n}) &= E\left\{\max\left(\mu_1^+ - C_1^*(\mathbf{\Sigma}, \mathbf{n}) \mid \mathbf{\Sigma}, \mathbf{n}{,}1\right), 0\right\} \\ &= E_{Y_1} \max\left\{\left(\frac{\Sigma_1 + Y_1}{n_1 + 1} - \left(\frac{2\Sigma_1}{n_1} - \frac{\Sigma_2}{n_2}\right)\right), 0\right\}, \qquad (3.4.1)\end{aligned}$$

while

$$\nu_2^{PKG}(\mathbf{\Sigma}, \mathbf{n}) = E_{Y_2} \max\left\{\left(\frac{\Sigma_2 + Y_2}{n_2 + 1} - \frac{\Sigma_1}{n_1}\right), 0\right\}. \qquad (3.4.2)$$

However, under the conditions satisfied by $(\mathbf{\Sigma}, \mathbf{n})$ it is easy to show that, $\forall y \in \mathbb{R}$,

$$\max\left\{\left(\frac{\Sigma_1 + y}{n_1 + 1} - \left(\frac{2\Sigma_1}{n_1} - \frac{\Sigma_2}{n_2}\right)\right), 0\right\} \geq \max\left\{\left(\frac{\Sigma_2 + y}{n_2 + 1} - \frac{\Sigma_1}{n_1}\right), 0\right\} \qquad (3.4.3)$$

and hence that

$$\nu_1^{PKG}(\mathbf{\Sigma}, \mathbf{n}) \geq E_{Y_1} \max\left\{\left(\frac{\Sigma_2 + Y_1}{n_2 + 1} - \frac{\Sigma_1}{n_1}\right), 0\right\}. \qquad (3.4.4)$$

But from Shaked and Shanthikumar (2007) we infer that Y_1 exceeds Y_2 in the convex ordering. Since $\max\left\{\left(\frac{\Sigma_2+y}{n_2+1}-\frac{\Sigma_1}{n_1}\right),0\right\}$ is convex in y it follows that

$$\begin{aligned}\nu_1^{PKG}(\mathbf{\Sigma},\mathbf{n}) &\geq E_{Y_1}\max\left\{\left(\frac{\Sigma_2+Y_1}{n_2+1}-\frac{\Sigma_1}{n_1}\right),0\right\}\\ &\geq E_{Y_2}\max\left\{\left(\frac{\Sigma_2+Y_2}{n_2+1}-\frac{\Sigma_1}{n_1}\right),0\right\}\\ &= \nu_2^{PKG}(\mathbf{\Sigma},\mathbf{n}) \qquad (3.4.5)\end{aligned}$$

and the result follows. □

KG-index (KGI): Before we describe this proposal we note that Whittle (1988) produced a proposal for index policies for a class of decision problems called *restless bandits* which generalise MABs by permitting movement in the states of non-active arms. Whittle's indices generalise those of Gittins in that they are equal to the latter for MABs with $0 < \gamma < 1, T = \infty$. Whittle's proposal is relevant for MABs with finite horizon $T < \infty$ since time-to-go now needs to be incorporated into state information which in turn induces a form of restlessness. In what follows we shall refer to Gittins/Whittle indices as those which emerge from this body of work for all versions of the MABs under consideration here.

The KGI policy chooses between arms on the basis of an index which approximates the Gittins/Whittle index appropriate for the problem by using the KG approach. We consider a single arm with (Σ, n) prior, finite horizon t and discount factor $\gamma, 0 \leq \gamma \leq 1$. To develop the Gittins/Whittle index $\nu_t^{GI}(\Sigma, n, \gamma)$ for such a bandit we suppose that a charge λ is levied for bandit activation. We then consider the sequential decision problem which chooses from the actions $\{active, passive\}$ for the bandit at each epoch over horizon t with a view to maximising expected rewards net of charges for bandit activation. The value function $V_t(\Sigma, n, \gamma, \lambda)$ satisfies Bellman's equations as follows:

$$V_t(\Sigma,n,\gamma,\lambda) = \max\left\{\frac{\Sigma}{n} - \lambda + \gamma E_Y[V_{t-1}(\Sigma+Y,n+1,\gamma,\lambda)];\gamma V_{t-1}(\Sigma,n,\gamma,\lambda)\right\}. \qquad (3.4.6)$$

This is a stopping problem in that, once it is optimal to choose the passive action at some epoch then it will be optimal to choose the passive action at all subsequent epochs since the active arm is unchanged by the passive action. Hence, (3.4.6) may be replaced by the following:

$$V_t\left(\Sigma,n,\gamma,\lambda\right)=\max\left\{\frac{\Sigma}{n}-\lambda+\gamma E_Y\left[V_{t-1}\left(\Sigma+Y,n+1,\gamma,\lambda\right)\right];0\right\}. \tag{3.4.7}$$

We further observe that $V_t\left(\Sigma,n,\gamma,\lambda\right)$ decreases as λ increases, while keeping t,Σ,n and γ fixed. This yields the notion of *indexability* in index theory. We now define the Gittins/Whittle index as

$$\nu_t^{GI}(\Sigma,n,\gamma)=\min\{\lambda;V_t(\Sigma,n,\gamma,\lambda)=0\}. \tag{3.4.8}$$

This index is typically challenging to compute.

We obtain an index approximation based on the KG approach as follows: In the stopping problem with value function $V_t(\Sigma,n,\gamma,\lambda)$ above, we impose the constraint that whatever decision is made at the second epoch is final, namely will apply for the remainder of the horizon. This in turn yields an approximating value function $V_t^{KG}(\Sigma,n,\gamma,\lambda)$ which when $0<\gamma<1$ satisfies the equation

$$\begin{aligned}&V_t^{KG}\left(\Sigma,n,\gamma,\lambda\right)\\&=\max\left\{\frac{\Sigma}{n}-\lambda+\frac{\gamma\left(1-\gamma^{t-1}\right)}{\left(1-\gamma\right)}E_Y\left[\max\left(\max\left(\frac{\Sigma+Y}{n+1},\lambda\right)-\lambda;0\right)\mid\Sigma,n\right];0\right\}\end{aligned} \tag{3.4.9}$$

and which is also decreasing in λ for any fixed t,Σ,n and γ. When $\gamma=1$ the constant multiplying the expectation on the r.h.s of (3.4.9) becomes $t-1$. The indices we use for the KGI policy when $T<\infty$ are given by

$$\begin{aligned}\nu_t^{KGI}\left(\Sigma,n,\gamma\right)&=\min\left\{\lambda;V_t^{KG}\left(\Sigma,n,\gamma,\lambda\right)=0\right\}\\&=\min\left\{\lambda;\lambda\geq\frac{\Sigma}{n}\text{ and }V_t^{KG}\left(\Sigma,n,\gamma,\lambda\right)=0\right\},\end{aligned} \tag{3.4.10}$$

where Σ, n, γ are as previously and t is the time to the end of the horizon. Note that the second equation in (3.4.10) follows from the evident fact that the index is guaranteed to be no smaller than the mean $\frac{\Sigma}{n}$.

Trivially $V_t(\Sigma, n, \gamma, \lambda)$ and $V_t^{KG}(\Sigma, n, \gamma, \lambda)$ are both increasing in the horizon t and consequentially so are both $\nu_t^{GI}(\Sigma, n, \gamma)$ and $\nu_t^{KGI}(\Sigma, n, \gamma)$. When $0 < \gamma < 1$ the limits $\lim_{t\to\infty} \nu_t^{GI}(\Sigma, n, \gamma)$ and $\lim_{t\to\infty} \nu_t^{KGI}(\Sigma, n, \gamma)$ are guaranteed to exist and be finite. These limits are denoted $\nu^{GI}(\Sigma, n, \gamma)$ and $\nu^{KGI}(\Sigma, n, \gamma)$ respectively, the former being the Gittins index. We use the indices $\nu^{KGI}(\Sigma, n, \gamma)$ for the KGI policy when $0 < \gamma < 1, T = \infty$.

Theorem 3.4.2. *The KGI policy does not choose dominated arms.*

We establish this result via a series of results.

Lemma 3.4.3. $V_t^{KG}(\Sigma, n, \gamma, \lambda)$ *and* $\nu_t^{KGI}(\Sigma, n, \gamma)$ *are both increasing in* Σ *for any fixed values of* t, n, γ, λ.

Proof. Since the quantity $\left(\max\left(\frac{\Sigma+y}{n+1}, \lambda\right) - \lambda; 0\right)$ is increasing in y and $Y \mid \Sigma, n$ is stochastically increasing in Σ, it follows easily that the expectation on the right hand side of (3.4.9) is increasing in Σ. The result then follows straightforwardly. □

We now proceed to consider the equivalent bandit, but with prior $(c\Sigma, cn)$, where $c > 0$.

Lemma 3.4.4. $V_t^{KG}(c\Sigma, cn, \gamma, \lambda)$ *is decreasing in* c *for any fixed values of* t, Σ, n, γ *and for any* $\lambda \geq \frac{\Sigma}{n}$.

Proof. First note that for $y \geq \frac{\Sigma}{n}$, the quantity $\max\left(\frac{c\Sigma+y}{cn+1}, \lambda\right)$, regarded as a function of c, is decreasing when $\lambda \geq \frac{\Sigma}{n}$. For $y < \frac{\Sigma}{n}, \max\left(\frac{c\Sigma+y}{cn+1}, \lambda\right) = \lambda$ and hence is trivially decreasing in c. Note also that the quantity $\max\left(\frac{c\Sigma+y}{cn+1}, \lambda\right)$, regarded as a function of y, is increasing and convex. We also observe from Yu (2011) that $Y \mid c\Sigma, cn$ is decreasing in the convex order as c increases. It then follows that, for $c_1 > c_2$ and for $\lambda \geq \frac{\Sigma}{n}$,

$$\begin{aligned} E_Y\left(\max\left(\frac{c_1\Sigma + Y}{c_1 n + 1}, \lambda\right) \middle| \, c_1\Sigma, c_1 n\right) &\leq E_Y\left(\max\left(\frac{c_1\Sigma + Y}{c_1 n + 1}, \lambda\right) \middle| \, c_2\Sigma, c_2 n\right) \\ &\leq E_Y\left(\max\left(\frac{c_2\Sigma + Y}{c_2 n + 1}, \lambda\right) \middle| \, c_2\Sigma, c_2 n\right) \end{aligned} \tag{3.4.11}$$

from which the result trivially follows via a suitable form of (3.4.9). □

The following is an immediate consequence of the preceding lemma and (3.4.10).

Corollary 3.4.5. *$\nu_t^{KGI}(c\Sigma, cn, \gamma)$ is decreasing in c for any fixed values of t, Σ, n, γ.*

It now follows trivially from the properties of the index ν_t^{KGI} established above that if (Σ_1, n_1) dominates (Σ_2, n_2) then $\nu_t^{KGI}(\Sigma_1, n_1, \gamma) \geq \nu_t^{KGI}(\Sigma_2, n_2, \gamma)$ for any t, γ. It must also follow that $\nu^{KGI}(\Sigma_1, n_1, \gamma) \geq \nu^{KGI}(\Sigma_2, n_2, \gamma)$ when $0 < \gamma < 1$. This completes the proof of the above theorem.

Closed form expressions for the indices ν_t^{KGI} are not usually available, but are in simple cases. For the Bernoulli rewards case of Subsection 3.3.4 we have that

$$\nu_t^{KGI}(\Sigma, n, \gamma) = \frac{\Sigma}{n + \frac{\gamma(1-\gamma^{t-1})}{(1-\gamma)}\Sigma} + \frac{\gamma\left(1-\gamma^{t-1}\right)}{(1-\gamma)} \frac{\Sigma(\Sigma+1)}{(n+1)\left\{n + \frac{\gamma(1-\gamma^{t-1})}{(1-\gamma)}\Sigma\right\}}. \tag{3.4.12}$$

In general numerical methods such as bisection are required to obtain the indices. If the state space is finite it is recommended that all index values are calculated in advance.

Fast calculation is an essential feature of KG but it should be noted that this is not universal and that index methods are more tractable in general. An example of this is the MAB with multiple plays (Whittle 1980). Here m arms are chosen at each time rather than just one. Rewards are received from each of the arms as normal. For an index policy the computation required is unchanged - the index must be calculated for each arm as normal with arms chosen in order of descending indices. The computation for KG is considerably larger than when $m = 1$. The KG score must be calculated for each possible *combination* of m arms, that is $\binom{n}{m}$ times. For each of these we must find the set of arms with largest expected reward conditional on each possible outcome. Even in the simplest case, with Bernoulli rewards, there are 2^m possible outcomes. For continuous rewards the problem becomes much more difficult even for $m = 2$. It is clear that KG is impractical for this problem.

An existing method with similarities to KG is the Expected Improvement algorithm of Jones et al. (1998). This is an offline method of which KG can be thought of as a more detailed alternative. It was compared with KG in Frazier et al. (2009) in the offline setting. The Expected Improvement algorithm is simpler than KG and always assigns positive value to the greedy arm unless its true value is known exactly. Its arm values are "optimistic" in a manner analogous to the PKG policy described above and it is reasonable to conjecture that it shares that rule's avoidance of dominated actions (see Theorem 3.4.1). As an offline method it is not tested here but it may be possible to develop an online version.

3.5 Computational Study

This section will present the results of experimental studies for the Bernoulli and exponential MAB. A further study will be made for the Gaussian MAB in Section 3.6.1.

3.5.1 Methodology

All experiments use the standard MAB setup as described in Section 3.2.1. For Bernoulli rewards with $k = 2$ policy returns are calculated using value iteration. All other experiments use simulation for this purpose. These are *truth-from-prior* experiments i.e. the priors assigned to each arm are assumed to be accurate.

For each simulation run a θ_a is drawn randomly from the prior for each arm $a \in \{1, 2, \ldots, k\}$. A bandit problem is run for each policy to be tested using the same set of parameter values for each policy. Performance is measured by totalling, for each policy, the discounted true expected reward of the arms chosen. For each problem 160000 simulation runs were made.

In addition to the policies outlined in Section 3.4, also tested are the Greedy policy (described in Section 3.2.1) and a policy based on analytical approximations to the GI (Brezzi and Lai 2002), referred to here as GIBL. These approximations are based on the GI for a Wiener process and therefore assume normally distributed rewards. However, they can be appropriate for other reward distributions by Central Limit Theorem arguments and the authors found that the approximation was reasonable for Bernoulli rewards, at least for n not too small. Other papers have refined these approximations but, although they may be more accurate asymptotically, for the discount rates tested here they showed inferior performance and so only results for GIBL are given.

3.5.2 Bernoulli MAB

The first experiment tests performance over a range of γ for $k \in \{2, 10\}$ arms, each with uniform $Beta(1, 1)$ priors. The mean percentage lost reward for five policies are given in Figure 3.5.1. The results for the greedy policy are not plotted as they

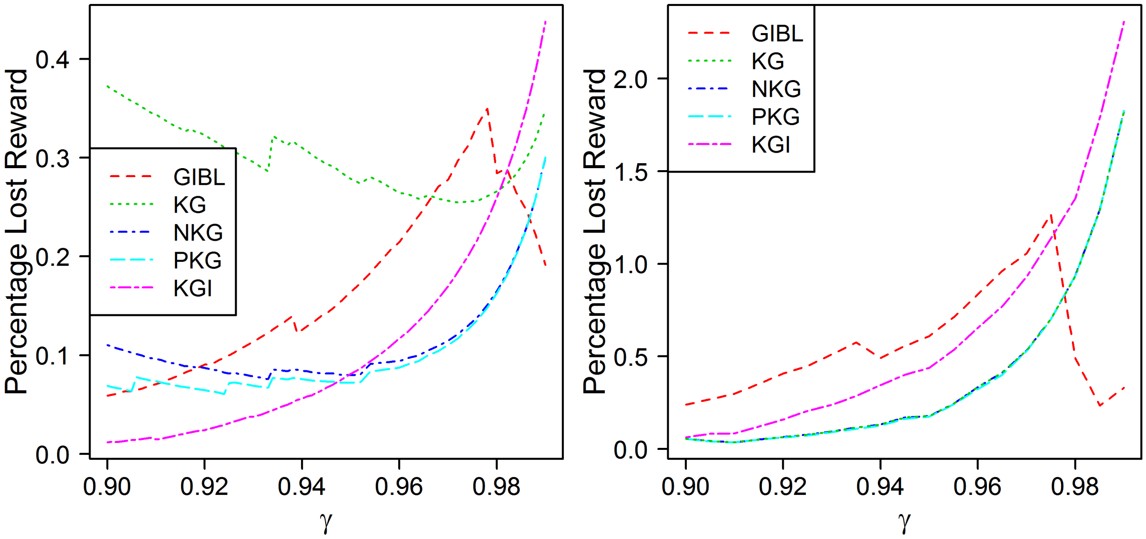

Figure 3.5.1: Mean percentage of lost reward compared to the GI policy for five policies for the Bernoulli MAB with uniform priors and $\gamma \in [0.9, 0.99]$. The left plot shows $k = 2$ while on the right $k = 10$.

are clearly worse than the other policies (percentage loss going from 0.64 to 1.77 for $k = 2$). The overall behaviour of the policies is similar for $k = 2$ and $k = 10$. KGI is strong for lower γ but is weaker for higher γ while GIBL is strongest as γ increases. The sharp change in performance for GIBL at $\gamma \approx 0.975$ occurs because the GIBL index is a piecewise function. Both NKG and PKG improve on KG for $k = 2$ but the three KG variants are almost identical for $k = 10$. The difference between KG and NKG gives the cost for the KG policy of dominated actions. These make up a large proportion of the lost reward for KG for lower γ but, as γ increases, over-greedy errors due to the myopic nature of the KG policy become more significant and these are not corrected by NKG. These errors are also the cause of the deteriorating performance of KGI at higher γ. At $k = 10$ the states given in Section 3.3.3 where KG was shown to take dominated actions occur infrequently. This is because, for larger numbers of arms there will more often be an arm with $\mu \geq 0.5$ and such arms are chosen in preference to dominated arms.

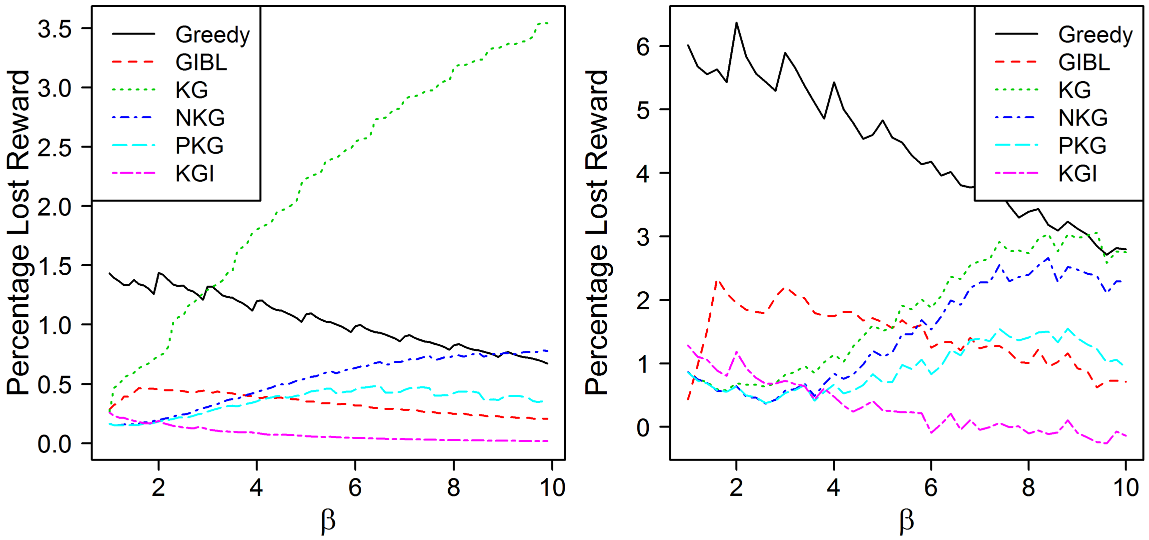

Figure 3.5.2: Percentage lost reward relative to the GI policy for six policies for the Bernoulli MAB with $\alpha = 1, \beta \in [1, 10]$ and $\gamma = 0.98$. The left plot shows $k = 2$ while on the right $k = 10$.

However, states where $\mu < 0.5$ for all arms will occur more frequently when arms have lower θ. Here dominated actions can be expected to be more common. We can test this by using priors where $\beta > \alpha$. Figure 3.5.2 shows the effect of varying the β parameter for all arms. The discount rate $\gamma = 0.98$ is quite a high value where the greedy policy can be expected to perform poorly since exploration will be important. However as β increases the performance of KG deteriorates to the extent that it is outperformed by the greedy policy. This effect is still seen when $k = 10$. The superior performance of NKG shows that much of the loss of KG is due to dominated actions. Policy PKG improves further on NKG suggesting that KG makes further errors due to asymmetric updating even when it does not choose dominated arms. A clearer example of this is given in Section 3.5.3. Both policies based on GI approximations perform well and are robust to changes in β. KGI is the stronger of the two as GIBL is weaker when the rewards are less normally distributed.

The same pattern can also be seen to be present when arms have low success prob-

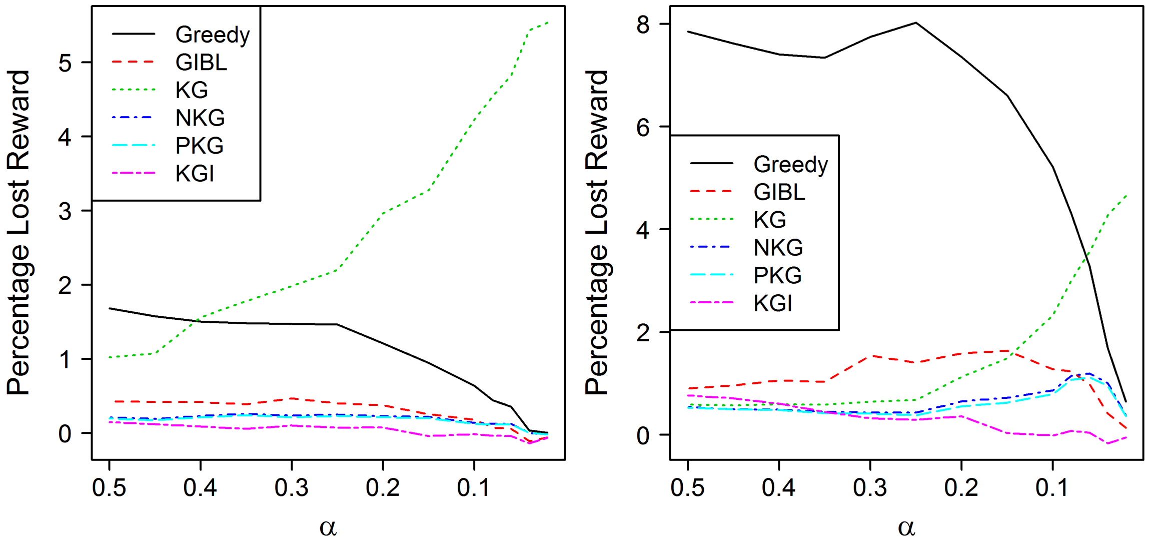

Figure 3.5.3: Percentage lost reward relative to the GI policy for six policies for the Bernoulli MAB with $\beta = 1, \alpha \in [0.02, 0.5]$ and $\gamma = 0.98$. The left plot shows $k = 2$ while on the right $k = 10$.

abilities but prior variance is high. Figure 3.5.3 gives results for $\beta = 1$ with low α. The range shown focuses on lower prior μ which correspond to $\beta \in [2, \ldots 50]$ in the setup of the previous experiment. The higher prior variance makes arms with higher success probabilities more likely than in the previous experiment but as α is reduced the performance of KG can still be seen to deteriorate markedly. The other policies tested do not show this problem.

Arms with low θ are common in many applications. For example, in direct mail marketing or web based advertising where θ is the probability that a user responds to an advert. The unmodified KG is unlikely to be an effective method in such cases.

The equivalent plots with prior $\mu > 1$ do not show any significant changes in behaviour compared to uniform priors.

Another policy that is popular in the bandit literature and which has good theoretical properties is Thompson Sampling (e.g. Russo and Van Roy 2014). Results for this method are not given in detail here as its performance is far inferior on these problems to the other policies tested. For example, on the results displayed in Figure 3.5.1 losses were in the ranges from $1.3-4\%$ and $6-15\%$ for $k=2$ and $k=10$ respectively with the best performance coming for $\gamma=0.99$. It is a stochastic policy and so makes many decisions that are suboptimal (including dominated errors). Its strength is that it explores well in the limit over time, eventually finding the true best arm. However, with discounted rewards or when the horizon is finite it gives up too much short term reward to be competitive unless γ is close to 1 or the finite horizon is long. In addition, note that it will spend longer exploring as k increases as it seeks to explore every alternative. Performance on the other problems in this chapter was similar and so are not given.

3.5.3 Exponential MAB

This section gives the results of simulations for policies run on the MAB with Exponentially distributed rewards as outlined in Section 3.3.3. These are shown in Figure 3.5.4. Here the lost reward is given relative to the KG policy (the negative values indicate that the other policies outperformed KG). Different priors give a similar pattern of results.

The results show a clear improvement over the KG policy by PKG and NKG policies. Notably the PKG earns better reward than the NKG indicating that the bias that causes dominated errors also causes suboptimal choices when arms are not dominated. Policy KGI gives the best performance although similar to PKG.

3.6 The Gaussian MAB

Here we consider the Gaussian case $Y_a \mid \theta_a \backsim N(\theta_a, 1)$ and $\theta_a \backsim N\left(\frac{\Sigma_a}{n_a}, \frac{1}{n_a}\right)$. In the brief discussion in Section 3.3 we noted that KG does not take dominated actions in

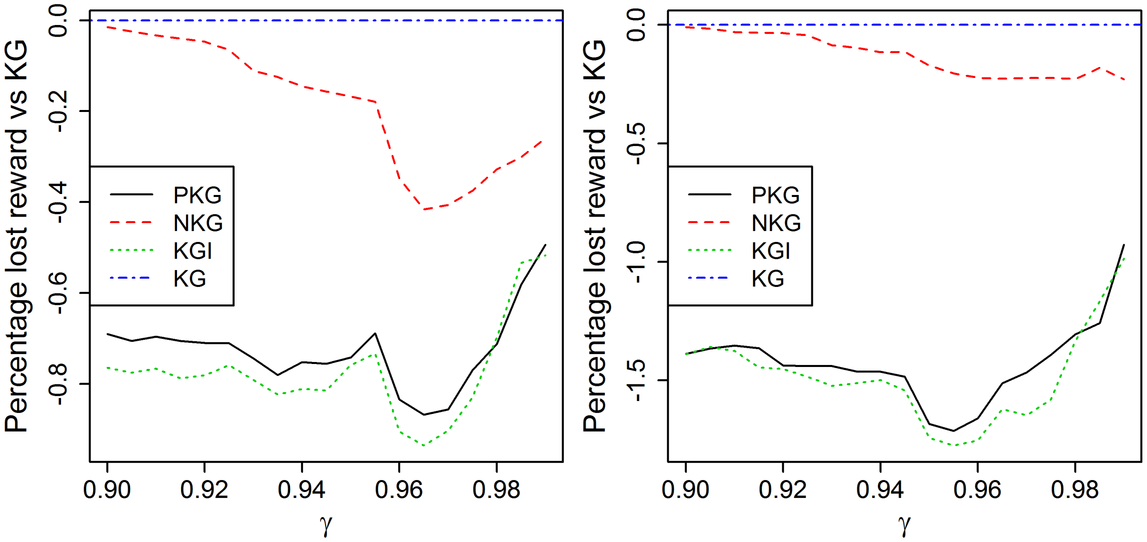

Figure 3.5.4: Mean percentage of lost reward compared to the KG policy for three policies for the exponential MAB with Gamma(2,3) priors and $\gamma \in [0.9, 0.99]$. The left plot shows $k = 2$ while on the right $k = 10$.

this case. While Ryzhov et al. (2012) give computational results which demonstrate that KG outperforms a range of heuristic policies, the policy still makes errors. In this section we describe how errors in the estimation of arms' learning bonuses constitute a new source of suboptimal actions. We also elucidate easily computed heuristics which outperform KG. A major advantage of KG cited by Ryzhov et al. (2012) is its ability to incorporate correlated beliefs between arms. We will later show, in Section 3.6.1, that it is unclear whether KG enjoys a performance advantage in such cases.

We shall restrict the discussion to cases with $k = 2$, $0 < \gamma < 1$ and $T = \infty$ and will develop a notion of *relative learning bonus (RLB)* which will apply across a wide range of policies for such problems. We shall consider *stationary policies* π whose action in state $(\mathbf{\Sigma}, \mathbf{n}) \equiv (\Sigma_1, n_1, \Sigma_2, n_2)$ depends only upon the precisions n_b and the difference in means $\Delta\mu := \frac{\Sigma_2}{n_2} - \frac{\Sigma_1}{n_1}$. We shall write $\pi(\Delta\mu, n_1, n_2)$ in what follows. We further require that policies be *monotone* in the sense of the following definition of the RLB.

Definition 3.6.1 (Relative Learning Bonus). *If π is monotone in $\Delta\mu$ such that $\exists$ function $R^{\pi} : \mathbb{N}^2 \rightarrow \mathbb{R}$ with $\pi\left(\Delta\mu, n_1, n_2\right) = 2 \Leftrightarrow \Delta\mu \geq R^{\pi}\left(n_1, n_2\right)\ \forall\left(n_1, n_2\right)$ then R^{π} is the RLB function.*

This monotonicity is a natural property of deterministic policies and holds for all policies considered in this section since increasing $\Delta\mu$ while holding n_1, n_2 unchanged favours arm 2 in all cases. The RLB gives a method of comparing the actions of index and non-index policies but it is also useful when comparing index policies. A natural method of evaluating an index policy would be to measure the difference in its indices from the GI in the same states. This can be inaccurate. An index formed by adding a constant to the GI will give an optimal policy so it is not the magnitude of the bias that is important but how it varies. The RLB and the idea of index consistency (discussed later) give methods to assess this distinction.

Under the above definition we can set $\Sigma_1 = 0$ without loss of generality. We then have that $\Delta\mu = \mu_2$ and arm 2 is chosen by policy π in state $(\boldsymbol{\Sigma}, \mathbf{n})$ if and only if $\mu_2 \geq R^{\pi}\left(n_1, n_2\right)$. Figure 3.6.1 illustrates this for the GI and KG policies, the former of which determines the optimal *RLB* values. The plots are of slices through the R^{GI} and R^{KG} surfaces with $n_1 = 1$ and with γ set at 0.95. As n_2 increases the GI learning bonus for arm 2 decreases, yielding values of $R^{GI}\left(1, n_2\right)$ which are increasing and concave. Comparison with the $R^{KG}\left(1, n_2\right)$ suggests that the latter is insufficiently sensitive to the value of n_2. This is caused by a KG value close to zero for arm 2 when $n_2 \geq 2$ and results in a mixture of over-exploration and over-exploitation. In practice when the priors of the two arms are close, over-exploration is the main problem. For $n_1 > 1$ the *RLB* curves have a similar shape but with smaller R as the learning bonuses for both policies decrease with increased information over time.

Figure 3.6.2 contains comparative plots of $R^{GI}\left(1, n_2\right)$ and $R^{\pi}\left(1, n_2\right)$ for three other policies π and with γ again set at 0.95. The policies are KGI, described in Section 3.4, and two others which utilise analytical approximations to the Gittins Index, namely

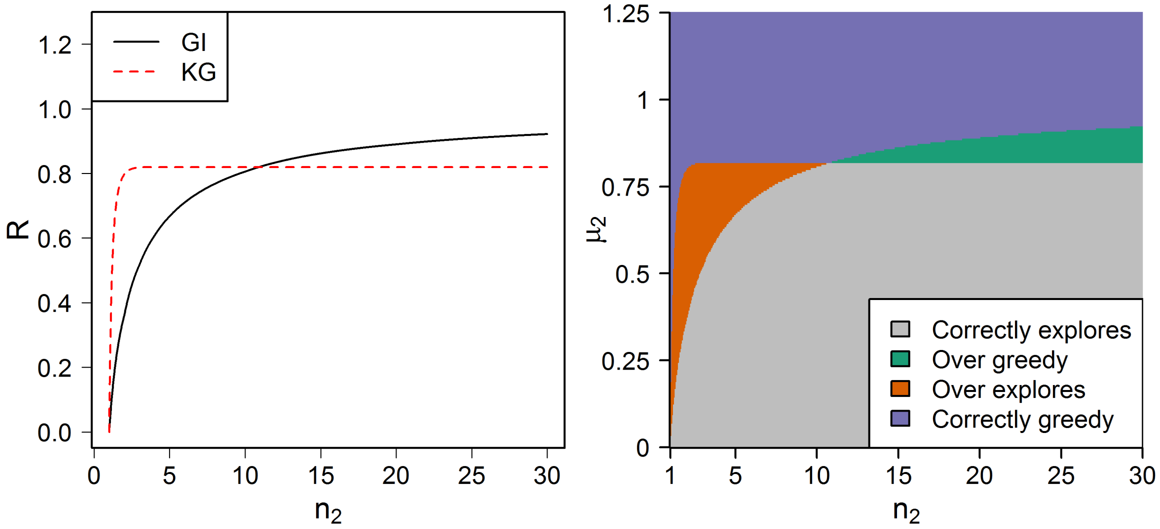

Figure 3.6.1: The left plot shows RLB values for KG and GI policies for $\gamma = 0.95$, $n_1 = 1$, $\mu_1 = 0$. The right plot shows the nature of KG actions over different arm 2 states.

GIBL (Brezzi and Lai 2002) and GICG (Chick and Gans 2009). Although the latter use similar approaches to approximating GI their behaviour appear quite different, with GIBL over-greedy and GICG over-exploring. This changes when γ is increased to 0.99 where both policies over-explore. Although not shown here, the approximations of GI by both GIBL and GICG improve as n_1 increases and the corresponding *RLB* curves are closer. A suboptimal action is often less costly when over-greedy, especially for lower γ since immediate rewards are guaranteed while the extra information from exploration might not yield any reward bonus until discounting has reduced its value. Weber (1992) enunciates a desirable property for policies which is enjoyed by the optimal GI policy. It can be thought of as a generalised stay-on-a-winner property.

Definition 3.6.2. *A policy is index consistent if, once an arm is chosen then it continues to be chosen while its Gittins index remains above its value at the start of the period of continuation.*

The region of over-exploration in the *RLB* plot in Figure 3.6.1 yields states in which

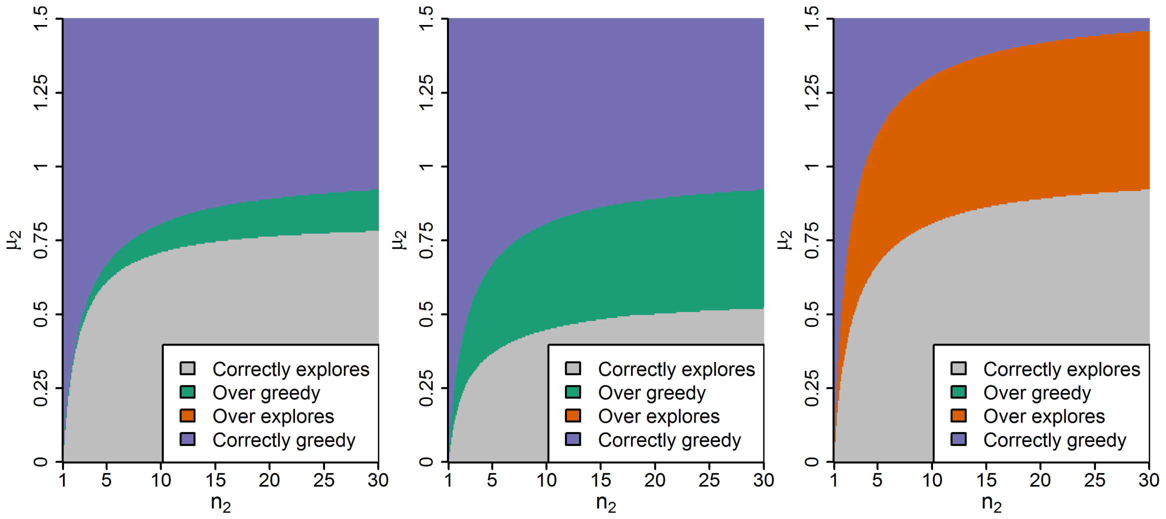

Figure 3.6.2: Plots of actions for KGI, GIBL and GICG (from left to right) for $\gamma = 0.95$, $n_1 = 1$, $\mu_1 = 0$.

KG is not index consistent. It will simplify the proof and discussion of the next result if we write the Gittins index for an arm in state (Σ, n) as $\nu^{GI}(\Sigma, n) = \frac{\Sigma}{n} + l^{GI}(n)$, where l^{GI} is the GI (i.e., optimal) learning bonus for the arm. Note that for notational economy we have dropped the γ-dependence from the index notation. It now follows from the above definition that $R^{GI}(n_1, n_2) = l^{GI}(n_1) - l^{GI}(n_2)$. More generally, if π is an *index policy* we use l^{π} for the learning bonus implied by π, with $R^{\pi}(n_1, n_2) = l^{\pi}(n_1) - l^{\pi}(n_2)$.

To prove Proposition 3.6.3 we use that each policy over-explores, as shown in Figures 3.6.1 and 3.6.2 for KG and GICG and for (e.g.) $\gamma = 0.99$ for GIBL (not shown). The idea of the proof is that a policy that over-explores overestimates the RLB of the arm with lower n. After the arm is pulled n increases and its RLB is reduced. There are values of y such that the arm's GI will increase (as its reduction in RLB is smaller) but its μ will not increase sufficiently to overcome the loss of RLB and so the policy will switch arms.

Proposition 3.6.3. *Policies KG, GIBL and GICG are not index consistent.*

Proof. For definiteness, consider policy KG. From the calculations underlying Figure 3.6.1 we can assert the existence of state $(\mathbf{\Sigma}, \mathbf{n})$ such that $\Sigma_1 = 0, n_1 = 1, n_2 = 2$ and $2R^{KG}(1,2) > \Sigma_2 > 2R^{GI}(1,2)$, equivalently, $R^{KG}(1,2) > \mu_2 > R^{GI}(1,2)$, when $\gamma = 0.95$. It follows that $A^{KG}(\mathbf{\Sigma}, \mathbf{n}) = 1$ and KG over-explores in this state. We suppose that the pull of arm 1 under KG in state $(\mathbf{\Sigma}, \mathbf{n})$ yields a reward y satisfying $\mu_2 > \frac{y}{2} > R^{GI}(1,2) = l^{GI}(1) - l^{GI}(2)$. But $\nu^{GI}(y,2) = \frac{y}{2} + l^{GI}(2) > l^{GI}(1) = \nu^{GI}(0,1)$ and so the Gittins index of arm 1 has increased as a result of the reward y. However, the symmetry of the normal distribution and the fact that $\mu_2 > y$ guarantees that KG will choose arm 2 in the new state. Thus while the Gittins index of arm 1 increases, KG switches to arm 2 and hence is not index consistent. Regions of over-exploration for GIBL and GICG (in the former case when $\gamma = 0.99$) means that a similar argument can be applied to those policies. This concludes the proof. □

An absence of over-exploration does not guarantee index consistency for a policy. However, we now give a sufficient condition for an index policy never to over-explore and to be index consistent.

Proposition 3.6.4. *If index policy π satisfies $0 \leq R^{\pi}(n_1, n_2) \leq R^{GI}(n_1, n_2)$ $\forall n_1 < n_2$ then it never over-explores and is index consistent.*

Proof. Let state $(\mathbf{\Sigma}, \mathbf{n})$ be such that $\Sigma_1 = 0$. This is without loss of generality. For the over-exploration part of the result, we consider two cases. In case 1 we suppose that $\mu_2 > 0$ and the GI policy chooses greedily when it chooses arm 2. This happens when $\mu_2 \geq R^{GI}(n_1, n_2)$. If $n_1 < n_2$ then the condition in the proposition implies that $\mu_2 \geq R^{\pi}(n_1, n_2)$ and policy π must also choose arm 2. If $n_1 \geq n_2$ then the condition in the proposition implies that $R^{\pi}(n_1, n_2) \leq 0$ and hence that $\mu_2 \geq R^{\pi}(n_1, n_2)$ trivially, which implies that policy π continues to choose arm 2. This concludes consideration of case 1. In case 2 we suppose that $\mu_2 \leq 0$ and so the GI policy chooses greedily when it chooses arm 1. If $n_1 < n_2$ then we have $\mu_2 \leq 0 \leq R^{\pi}(n_1, n_2)$ while if $n_1 \geq n_2$

then we must have that $\mu_2 \leq R^{GI}(n_1, n_2) \leq R^{\pi}(n_1, n_2) \leq 0$. Either way, policy π also chooses arm 1 and case 2 is concluded. Hence, under the condition in the proposition, policy π never explores when GI is greedy, and so never over-explores. For the second part of the result suppose that in state $(\mathbf{\Sigma}, \mathbf{n})$, index policy π chooses arm a and that the resulting reward y is such that $\nu^{GI}(\Sigma_a + y, n_a + 1) > \nu^{GI}(\Sigma_a, n_a)$, namely arm a's Gittins index increases. Under the condition in the proposition we then have that

$$\begin{aligned} \frac{\Sigma_a + y}{n_a + 1} + l^{GI}(n_a + 1) &> \frac{\Sigma_a}{n_a} + l^{GI}(n_a) \\ \Leftrightarrow \frac{\Sigma_a + y}{n_a + 1} - \frac{\Sigma_a}{n_a} &> R^{GI}(n_a, n_a + 1) \geq R^{\pi}(n_a, n_a + 1) \\ \Rightarrow \frac{\Sigma_a + y}{n_a + 1} + l^{\pi}(n_a + 1) &> \frac{\Sigma_a}{n_a} + l^{\pi}(n_a) \end{aligned} \tag{3.6.1}$$

and we conclude that policy π will continue to choose arm a. Hence π is index consistent. This concludes the proof. □

Conjecture 3.6.5. *On the basis of extensive computational investigation we conjecture that policy KGI satisfies the sufficient condition of Proposition 3.6.4 and hence never over-explores and is index consistent. We have not yet succeeded in developing a proof.*

3.6.1 Computational Study

This section gives the results of computational experiments on the MAB with normally distributed rewards (NMAB). The same methodology as in Section 3.5 is used. As well as the basic MAB, also considered are the finite horizon NMAB with undiscounted rewards and a problem where arm beliefs are correlated. It extends the experiments of Ryzhov et al. (2012) by testing against more competitive policies (including GI) and by separating the effects of finite horizons and correlated arms. In both of these latter problems the Gittins Index Theorem (Gittins et al. 2011) no longer holds and there is no index policy that is universally optimal. This raises the question of whether index policies suffer on these problems in comparison to non-index policies such as KG.

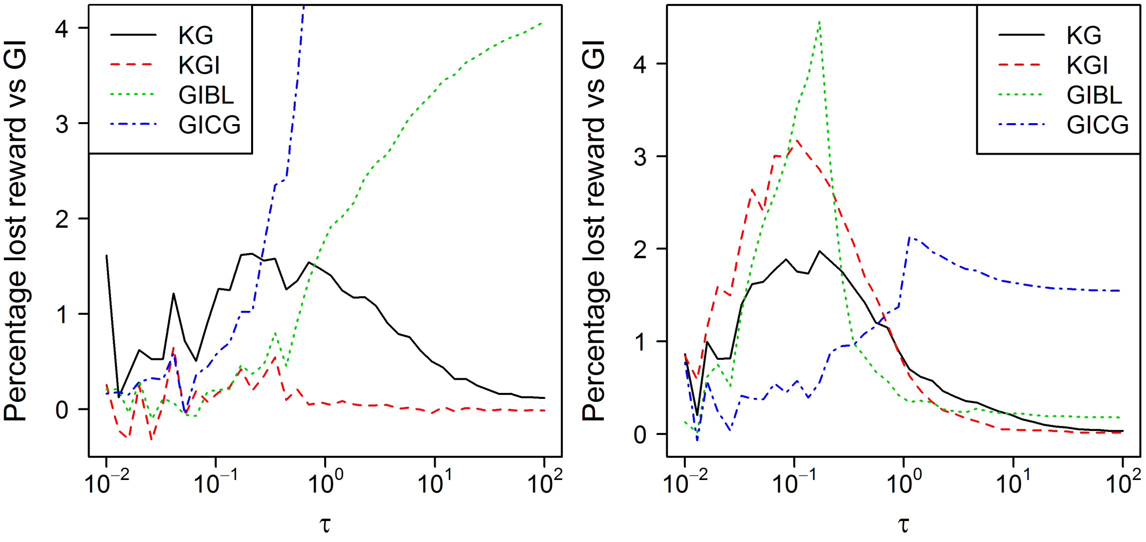

Figure 3.6.3: Lost reward versus optimal for heuristic policies for the NMAB with $\gamma = 0.9$ (left) and $\gamma = 0.99$ (right). There are $k = 10$ arms each with a $N(0, 1)$ prior.

A new parameter, τ for the observation precision is introduced for several of these experiments so that $Y_a \mid \theta_a \backsim N(\theta_a, \tau)$. In Section 3.6 it was assumed $\tau = 1$ but the results given there hold for general τ. The posterior for θ is now updated by $p(\theta_a|y) = g(\theta|\Sigma_a + \tau y, n_a + \tau)$. We take τ to be known and equal for all arms.

Infinite Horizon, Discounted Rewards

The first problem compares KG, KGI, GIBL, GICG against the optimal policy on the standard NMAB over a range of τ. The lost reward as a percentage of the optimal reward is shown in Figure 3.6.3. The plot does not show the loss for GICG for high τ and $\gamma = 0.9$ as it is very high relative to the other policies (rising to $> 38\%$). The Greedy policy has similarly poor performance.

Ryzhov et al. (2012) used the GICG policy as a comparator for the KG policy for the discounted Gaussian MAB. It was described as “the current state of the art in Gittins approximation”. These approximations are supposed to be better than the older GIBL but it appears that the improvements are mainly for large n which may not result in improved performance. The RLB plots earlier in this section suggest that the approximations for low n are not better and it is more important to be accurate in states reached at early times due to discounting. As γ becomes larger these early actions form a smaller portion of total reward and are therefore less significant.

There is no one best policy for all problem settings. Policy KGI is uniformly strong for $\gamma = 0.9$ but is weaker for $\gamma = 0.99$. Both KGI and KG do well for high τ. This is because more information is gained from a single observation and so myopic learning policies become closer to optimal. As τ becomes smaller it becomes important to consider more future steps when evaluating the value of information. However when τ is very low learning takes so long that a simple greedy approach is again effective. Hence KG and KGI are weakest for moderate values of τ between 0.1 and 1, depending on the number of arms.

The Finite Horizon NMAB

This section considers a variant on the NMAB where the horizon is a fixed length and rewards are not discounted (FHNMAB). One strength of KG is that it adapts easily for different horizon lengths and discount rates. GIBL and GICG, however, are designed only for infinite horizons. Ryzhov et al. (2012) got round this problem by treating the discount rate as a tuning parameter. This allowed them to run experiments on a single horizon length ($T = 50$). However, it is not ideal. Firstly, the tuning parameter will need to be different for different horizons and there is no simple way to set this. Secondly, the policy is not appropriate for the problem because a policy for a finite horizon should be *dynamic*, it should change over time by exploring less as the end time approaches, whereas this policy is *static* over time. We give a method here by which any policy designed for an infinite discounted problem can be adapted to a finite horizon one so that it changes dynamically with time. Note that all KG variants given in this chapter (including KGI) are already dynamic when the horizon is finite so do

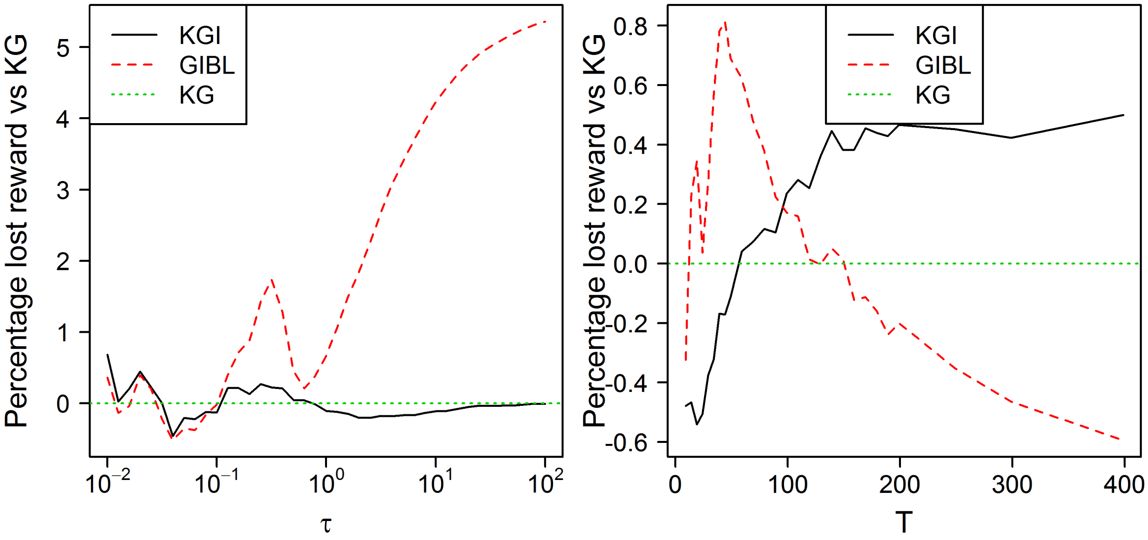

Figure 3.6.4: Lost reward on the FHNMAB. Left shows performance over a $\tau \in [10^{-2}, 10^{2}]$ with $T = 50$, right over $T \leq 400$ with $\tau = 1$. All $k = 10$ arms had the same $N(0, 1)$ prior.

not require any adjustment.

Definition 3.6.6 (Finite Horizon Discount Factor Approximation)**.** *A policy which depends on a discount factor γ can be adapted to an undiscounted problem with a finite horizon T by taking*

$$\gamma(t, T) = \frac{T - t - 1}{T - t}, \quad t = 0, 1, \ldots, T - 1. \tag{3.6.2}$$

This chooses a γ such that $\gamma/(1-\gamma) = T-1-t$ so that the ratio of available immediate reward to remaining reward is the same in the infinite case with γ discounting (LH side) as the undiscounted finite case (RH side).

Figure 3.6.4 shows percentage lost reward versus KG for KGI and the adjusted GIBL (with KG shown as a straight line at zero).

Note that the scale of the vertical axis on the right plot is quite close to zero so that no policies are very distinct from KG here. GIBL shows similar behaviour to that seen in Figure 3.6.3 with infinite horizons, performing similarly to KG at $\tau = 1$ (worse for shorter horizons but better for higher T) but doing very badly as τ increases above 1. KG and KGI show similar results but KG is the preferred policy for $T \geq 60$.

The Correlated NMAB

The NMAB variant where arm beliefs are correlated was studied in Ryzhov et al. (2012) where the ability of KG to handle correlated beliefs was given as a major advantage of the policy. However, being able to incorporate correlation into the KG policy does not mean performance will improve and the experimental results that were given were mixed. A further short experimental study is conducted here for several reasons. Firstly, as shown earlier in this section, the GI approximation used (GICG) performs poorly in many circumstances and the GIBL and KGI policies might offer a stronger comparison. Secondly, Ryzhov et al. (2012) used the finite horizon undiscounted version of the problem. As described earlier the policies based on GI are not designed for this problem so an artificial tuning parameter was introduced. Here we use infinite horizon with discounted rewards as before. This makes it clearer to see the effect of the introduction of correlation without the extra complication of the different horizon.

The problem is the same as the NMAB described earlier in this section except that beliefs are captured in a single multivariate normal distribution for all the arms rather than one univariate normal for each arm. For each simulation run the θ values for all arms are drawn from this true multivariate prior. The belief correlation structure can take many different forms but here we use the same the power-exponential rule used in Ryzhov et al. (2012). Prior covariances of the variance-covariance matrix $\boldsymbol{C}$ are

$$\boldsymbol{C}_{i,j} = e^{-\lambda(i-j)^2}. \tag{3.6.3}$$

where the constant λ determines the level of correlation (decreasing with λ). In the experiments here all prior means are zero.

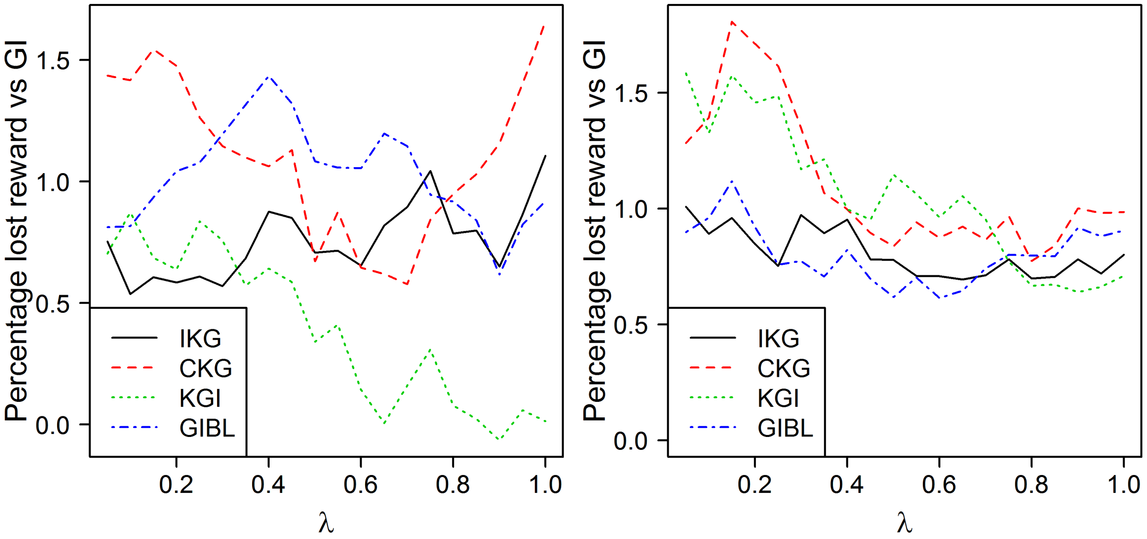

Figure 3.6.5: Lost reward vs GI on the Correlated NMAB for $\lambda \in [0.05, 1]$ with $\gamma = 0.9$ (left) and $\gamma = 0.99$ (right). Both use $k = 10$ with $\tau = 1$.

Two versions of KG are tested, the complete version that incorporates the correlation (CKG) and a version that assumes the arms are independent (IKG). Details of the CKG policy are given in Ryzhov et al. (2012) using an algorithm from Frazier et al. (2009). All policies tested (including IKG) use the true correlated model when updating beliefs but, apart from CKG, choose actions that make the false assumption that arms are independent. Updating beliefs using the independence assumption results in much poorer performance in all cases and therefore these results are not shown. CKG is significantly slower than the other policies, scaling badly with k. This limits the size of the experiment so 40000 runs are used. The results over $\lambda \in [0.05, 1]$ are shown in Figure 3.6.5. The first observation is that, although GI is not optimal for this problem, it still clearly outperforms all the other heuristics indicating that using a index policy is not an obvious handicap. The GI approximation policies' performance follows a similar pattern to the independent NMAB with KGI stronger at $\gamma = 0.9$ and GIBL stronger at $\gamma = 0.99$. IKG compares well to both these policies but again there is no evidence that non-index methods are stronger than index methods. More surprising is that CKG is clearly inferior to IKG. Frazier et al. (2009) found CKG to be stronger in the offline problem but this does not appear to translate to the online problem. Exactly why this is so is not clear as this is a difficult problem to analyse. CKG requires $\mathcal{O}(k^2 \log(k))$ to compute compared to IKG which requires $\mathcal{O}(k)$. CKG's performance would have to be much better to justify its use and in many online problems with larger k it would simply not be practical while IKG and the three simple index policies all scale well.

These experiments only scratch the surface of the correlated MAB problem and there are a number of possible issues. As rewards are observed the correlations in beliefs reduce and the arms tend to independence. Therefore correlations will be most important in problems with short horizons or steep discounting. Secondly, the number of arms used here is quite small. This is partly because CKG becomes very slow as k increases so its use on larger problems would not be practical. A feature of correlated arm beliefs is that we can learn about a large number of arms with a single pull and therefore independence assumptions should be punished with greater numbers of arms. However we still learn about multiple arms as long as belief updating is handled accurately which is easy to do in this Gaussian setting. If this is not done then learning will be much slower and we did find that it is important that belief updating incorporates correlations.

One difficulty with analysing policies on this problem is that it still matters that the policy is effective on the basic MAB problem. Suboptimality in this regard can mask the effect of introducing correlation and changes may improve or worsen performance quite separately from addressing the issue of correlations. For example if a policy normally over-explores then any change that makes it greedier might improve performance. Thompson Sampling is a policy that can easily incorporate correlations (by sampling from the joint posterior) but the high level of exploration that comes from randomised actions does not do well on short horizon problems and any changes due to correlations will be too subtle to change that.

3.7 Conclusion

We identify an important class of errors, *dominated actions* which are made by KG. This involves choosing arms that have *both* inferior exploitative and explorative value. Much of the existing work on KG has focused on Gaussian rewards but these have features (symmetric and unbounded distributions) that avoid the domination problem. For other reward distributions the performance of KG can suffer greatly. Two new variants are given which remove this problem. Of these, NKG is simpler while PKG gives better experimental results by correcting errors besides those arising from dominated actions.

We also introduced an index variant of KG which avoids dominated actions, which we called KGI. For problems where the optimal policy is an index policy, simulation studies indicate that KGI is more robust than other proposed index policies that use approximations to the optimal index. It has computational advantages over KG and performed competitively in empirical studies on all problems tested including those where index methods are known to be suboptimal. One such problem is the MAB with correlated beliefs. Although KG can incorporate these beliefs it was found that any performance gain was, at best, small and did not justify the extra computation involved.

The new variants we introduce give a range of simple heuristic policies, of both index and non-index type. On the problems tested here at least, there did not appear to be any advantage to using non-index methods and, in addition, index methods have computational advantages on some BSDPs. However, this may not always be the case and further research will be needed to be more confident on other variants of this problem.

Chapter 4

Selecting Multiple Website Elements

4.1 Introduction

A common problem faced by websites is to choose which *elements* (e.g. adverts, news stories, or retail items) to display to users. A natural objective is to maximise the number of users that select or *click* an element. To do this we need to present elements that appeal to the users which, for a given user, depends not only on the general attractiveness or quality of the element but also its relevance to that user. For example, a news story about a big football game might attract more interest than a relatively niche story about astronomy but if it is known that the user enjoys science news and has little interest in sport then they may be more likely to click on the latter story.

The challenge of this problem comes principally from two features. Firstly, we do not know exactly either the users' preferences or the characteristics or quality of the available elements. Secondly, elements are not usually presented in isolation but as a part of a list or set of other elements. If they are very similar to one another then there will be *redundancy* in the set. So if our belief about the user in the previous example is wrong then a selection of mainly astronomy stories would be very inappropriate and lead to a dissatisfied customer. This illustrates the intuitive idea that element sets should be *diverse*. Diversity and its role in motivating our choice of model for user click behaviour will be discussed in more detail in Section 4.1.2.

In the version of the multiple elements problem investigated here the users arrive at the website sequentially over time. The user at each time is assumed to be unique with preferences that are independent from those of other users. We have some (noisy) information about each user's preferences which we observe only as they arrive. In response we select a set (of fixed size) of elements to display. The user then either clicks one element or none and this gives us some information about the quality and characteristics of elements which can be used to improve element selections presented to future users. Since users are unique and independent we do not improve our estimates of user preferences over time. The assumption that the user clicks at most one element will not always be realistic but can be seen to be appropriate in the principle motivation for this work which is *search advertising*. Here the user enters a term into a search engine and is presented with adverts alongside the search results. The search term gives us information about the user's current interests and adverts appropriate to this term will be more likely to be clicked. However, since the user is not seeking adverts, clicking more than one will be rare.

4.1.1 Multi-armed Bandit Framework

A natural framework to represent this problem is the *multi-armed bandit problem* (MAB). In the classical form of the MAB we choose one *arm* (equivalent to an element) at each time from those available. We then receive a binary *reward* (equivalent to a click or no click) with a probability associated with the chosen arm but which is unknown. By observing rewards we learn more about the selected arm's reward distribution which can then be used to inform which arms are chosen in future. The difficulty in the problem lies in how best to trade off choosing elements for expected immediate reward given current information (exploitation) against choosing elements to learn about them and so to gain improved reward in the long term (exploration).

The multiple website elements problem differs from the classical MAB since (i) we choose several arms at each time instead of one and, (ii) the reward depends on the user as well as the selected arm set. The dependence of reward on each user is related to the *contextual MAB* where the reward is a function of some *context* which is given at each time prior to choosing an arm. However, in the contextual MAB the context assumed to be known but we treat information about user preferences as uncertain.

Choosing multiple arms changes the problem significantly since, crucially, elements in the set interact so the reward of a set is not a linear combination of rewards of individual arms. Therefore we now need to learn and choose the best *set* for any user rather than just the best individual arms. There is a combinatorial explosion in the number of possible sets as the number of arms grow (which should be expected to be large as these represent, for example, available adverts). Even with a simple model for user click behaviour evaluating expected rewards for all sets will not be tractable in an online setting. In addition we need to define an appropriate model for the reward from a set of arms. We will now discuss one of the main features that such a model should capture.

4.1.2 Diversity

We observed earlier that it is desirable that element sets should be diverse. That is, contain elements that are not too similar to each other. This is done to avoid redundancy in the the arm set since if an element does not appeal to a user then it is unlikely that a similar element will do so either. In discussing diversity we will outline ideas behind the models for user click behaviour that will be used. A formal description of these will be given in Section 4.2.

How to achieve appropriate diversity has been studied in the the field of *recommender systems* which are concerned with recommending items (e.g. documents or other media) to users. A common approach is to maximise a combination of accuracy and diversity (Vargas and Castells 2011). In this chapter we will demonstrate that treating diversity as a separate objective can be unnecessary. Instead diversity arises naturally as a result of maximising a single, realistic objective (click through rate) under uncertainty about user preferences.

These two alternative views of diversity are described by Radlinski et al. (2009) as intrinsic and extrinsic needs for diversity. When the need is intrinsic the diversity is regarded as desirable in itself, for example by providing variety and novelty in a recommender system. The usual approach (e.g. Carbonell and Goldstein 1998; Vargas and Castells 2011; Hurley and Zhang 2011; Zhou et al. 2010) is to trade off accuracy (choosing elements that have similarity to the user in some way) and diversity (choosing elements that are dissimilar to each other). With an extrinsic need, diversity emerges as a characteristic of good solutions even though it is not in itself an objective. The models used here build on the extrinsic viewpoint taken in information retrieval by El-Arini et al. (2009) and Agrawal et al. (2009). We will show that existing models often result in only moderate set diversity, then give an extended family of models in which higher diversity is induced.

Our approach is based on the idea that each user has a latent preference or *state*. This state provides information about what type of element a user is likely to click on. The state could take many forms: it could be the intended meaning of an ambiguous search term, the genre of movie the user wishes to watch, or some subgroup of the user population that the user belongs to. If the state was known then elements could be chosen that are appropriate for that state. However we assume that we are provided only with a probability distribution over the user's states. Therefore it may be desirable to choose elements appropriate for several different states to ensure that there is a good chance of a click whichever of the possible states is true. In this way we choose a diverse set simply by seeking to maximise the probability of a click.

We argue that this approach results in *appropriate* diversity. If we are sure a user wishes to watch an action movie then there is little to be gained from including sport or comedy movies in the recommended set; this would correspond to inappropriate diversity. However when we are uncertain about the user's desires, perhaps because we have little information on the user or because their preferences vary, then we need a variety of genres in the set to cover these different realities. Therefore the diversity of the element set should depend on the reliability of available information about the user's preferences and our models will be seen to reflect this.

4.1.3 Related Work

The *MAB with multiple plays* (e.g. Whittle 1988; Pandelis and Teneketzis 1999) is the MAB where multiple arms are selected at each time. Although this has been well studied it is not appropriate here since rewards are a simple sum of the independent rewards of the individual arms. An area where the MAB with multiple plays is used but where there is also correlation between the arms is in *multi-channel access* in communications (e.g. Ahmad and Liu 2009). The arms are uncontrolled Markov chains and the correlation comes from common transition probabilities which does not fit with our problem.

The *combinatorial MAB* is also concerned with selecting sets of arms at each time. In Chen et al. (2013) rewards can be quite general functions of the arms selected, although it does not include the reward formulation we give in Section 4.2 because of the dependence of rewards on the user state. In addition they assume that rewards from individual arms are observed as opposed to the more difficult problem where only the total reward is observed. This latter more limited feedback occurs in the related *response surface bandit* of Ginebra and Clayton (1995) where we choose at each time the value of a vector (which might be restricted in some way) and observe a reward which is some function of this vector. The actions are used to learn the function. Where the function is linear we have the *linear bandit* problem (e.g. Auer 2002; Dani et al. 2008; Rusmevichientong and Tsitsiklis 2010). Cesa-Bianchi and Lugosi (2012) use the term *combinatorial bandits* to refer to a special case of the linear bandit where the chosen vector is binary.

A very relevant area of research is *information or document retrieval* in particular as applied to search engines. As already mentioned in Section 4.1.2 our models will build on ideas from Agrawal et al. (2009) and El-Arini et al. (2009). However, in both of these, as well as the related work in Yue and Guestrin (2011) and Radlinski et al. (2008), the quality and features of the available documents is assumed to be fixed and known. Where there is learning it is concerned only with the user population. This removes a central feature of the problem which is that we must learn about the available arms. Objectives in information retrieval can extend beyond those already discussed in relation to diversity. For a review of these see Chen and Karger (2006).

The references given above make very different assumptions on how specific or reliable *user feedback* is. These range from non-stochastic rewards, through noisy observations on individual arms to a single noisy reward for the whole set (referred to in Chen et al. (2013) as full information, semi-bandit feedback and bandit feedback respectively). Our problem uses the latter in a Bayesian setting with the added complication that there are two sources of uncertainty, from the users and from the arms. A distinctive alternative type of feedback that makes no assumptions on rewards is that of *duelling bandits* (Yue et al. 2012). This makes pairwise comparisons of arms, receiving only a noisy preference from each pair. This type of feedback can be relevant to information retrieval but we will assume that all feedback comes in the form of clicks. Some issues with interpreting clicks will be discussed in Section 4.2.

4.1.4 Chapter Outline

Section 4.2 will give a formal statement of the problem together with models for user click behaviour that is based on uncertainty about user preferences. Solving this problem is challenging, even when it is assumed that the arm characteristics are known, due to the combinatorial explosion in possible sets. A method for handling this part of the problem is given in Section 4.3. This includes a study of the effect of model choice on set diversity and the implications this has on the validity of the models.

Section 4.4 will give a Bayesian model for learning for when arm characteristics are not known. Dependence on the latent user preferences means that an exact scheme is not practical and we will describe an alternative method. Section 4.5 builds on the work from Sections 4.3 and 4.4 to give a solution method for the full problem which will then be tested in simulations in Section 4.6. The chapter will conclude in Section 4.7 with a summary of contributions and a discussion of issues and future work.

4.2 Problem Formulation

This section will formally state the multiple elements problem. The basic structure of the problem is that of a multi-armed bandit problem as described in Section 4.1 but we need to characterise the information available about each user, define measures of quality for the arms, and combine both of these into a model to determine whether a given user clicks on a chosen set of arms.

4.2.1 Topic Preference Vector

Our model for user click behaviour is based on the idea that each user has a latent preference or *state*. This state provides information about what type of element a user is likely to click on. The state could take many forms. For search advertising in its simplest form it can be thought of as a search objective or the intended meaning of an ambiguous search term but it could also represent more general characteristics of the user such as their interests or some subgroup of the user population that the user belongs to. If the state was known then elements could be chosen that are appropriate for that state. However we will assume throughout that we are provided only with a probability distribution over the user's states which we will refer to as a *topic preference vector*. It represents prior information or beliefs about the user's state. How this information is derived depends on the application and is beyond the scope of this work. However, for search advertising it would reasonably come from the user population search history together with the current search. The available information about searches would be much greater than that available about adverts and for this reason we take the topic preference vectors to be unchanging whilst we do learn about the adverts. The time frame over which an advert is displayed will be small relative to the number of times a search term has been entered. We characterise the quality of arms using weight vectors corresponding to the topic preference vector where a high value in a given entry indicates that the arm has high relevance to a user with that state.

The mathematical formulation of the users and arms is as follows. At each time step $t = 1, 2, \ldots$ a user arrives with a *state* $x_t \in \{1, \ldots, n\}$. This state is hidden but its distribution X_t is observed and is given by a topic preference vector $\mathbf{q}_t$ such that $\Pr(X_t = x) = q_{t,x}$. Each distribution $\mathbf{q}_t$ is random and we will assume throughout that $\Pr(q_{t,x} = 0) < 1$. In response to this we present m arms as an (ordered) set $A_t \subseteq \mathcal{A}$ where $\mathcal{A}$ is the set of k available arms. The user will respond by selecting (or clicking) at most one arm. If any arm is clicked then a reward of one is received, with zero reward otherwise.

Each arm $a \in \mathcal{A}$ is characterised by a weight vector $\mathbf{w}$ of length n with each $w_{a,x} \in (0, 1)$. Let $\mathbf{w}_A$ denote the set of vectors $\{\mathbf{w}_a\}, a \in A$. The weight $w_{a,x}$ represents the probability a user with state x will click element a. Each weight is not known exactly but can be learnt over time as a result of the repeated selection of arms and observation of outcomes. The outcome at each time is given by the reward together with which element (if any) is clicked. Further details on this feedback and learning process and how it can be used to inform future decisions are given in Section 4.4.1.

This framework is complete if $m = 1$ element is to be displayed and x is known: the click probability if a is presented is simply $w_{a,x}$. However if x is latent and $m > 1$ we need to build a model which gives the probability of receiving a click on the set of arms A as well as determining which arm (if any) is clicked. As described earlier we assume that at most one arm is clicked at a given time so we cannot simply sum the rewards from individual arms.

4.2.2 Click Models

We introduce a statistical model of which arm, if any, a user selects at each time. The *click through rate* (CTR) will refer to the expected probability over all relevant unknowns (if any) of a click on some arm in a set of arms. The term *arm CTR* will be used if we are interested in the probability of a click on a specific arm.

A simple and popular model that addresses the question of *which* arm is clicked is the *cascade model* (Craswell et al. 2008). Arms are presented in order and the user considers each one in turn until one is clicked or there are no more left. There are issues with this model as the probability that an arm is clicked is not directly affected by its position while, in reality, it is likely that users would lose interest before looking at arms later in the list. However, it is not the purpose of this work to address these issues and the framework given here can readily be adapted to more complicated models.

Using the cascade model we now give models to determine the CTR of a set of arms. Two intuitive models are presented, which are shown to be extreme examples of a general click model. The consequences of the choice of model will be a major focus of this work. Each model is initially specified for known x_t but easily extends to latent x_t as given at the end of this section.

Definition 4.2.1 (Probabilistic Click Model)**.** *In the Probabilistic Click Model (PCM) the user considers each arm in A_t independently in turn until they click one or run out of arms. This is closely related to the probabilistic coverage model for document retrieval given in El-Arini et al. (2009). At each step, the click probability for arm a is w_{a,x_t} as in the single arm case. Therefore the CTR of the set A_t for known $\mathbf{w}_{\mathcal{A}}$ and x_t is*

$$r_{\text{PCM}}(x_t, A_t, \mathbf{w}_{A_t}) = 1 - \prod_{a \in A_t} (1 - w_{a,x_t}).$$

Although it will be shown later that using this model does encourage diversity in the arm set, an obvious issue is that a set of two identical arms gives a higher CTR than a single such arm, which is inconsistent with our intuition of realistic user behaviour when presented with very similar elements. We present a new model which avoids this problem.

Definition 4.2.2 (Threshold Click Model)**.** *In the Threshold Click Model (TCM) each user has a threshold u_t drawn independently from distribution $U(0,1)$. They consider each arm in turn, clicking the first arm $a \in A_t$ such that $w_{a,x_t} > u_t$. The*

CTR of the set A_t for known $\mathbf{w}_{\mathcal{A}}$ and x_t is then

$$\begin{aligned} r_{\text{TCM}}(x_t, A_t, \mathbf{w}_{A_t}) &= \int_0^1 1 - \prod_{a \in A_t} (1 - \mathbb{1}_{w_{a,x_t} > u_t}) \, \mathrm{d}u_t. \\ &= \int_0^{\max_{a \in A_t} w_{a,x_t}} 1 \, \mathrm{d}u_t + \int_{\max_{a \in A_t} w_{a,x_t}}^1 0 \, \mathrm{d}u_t \\ &= \max_{a \in A_t} w_{a,x_t}. \end{aligned}$$

The TCM thus represents a user who, with preference x_t, will click an element if its relevance w_{a,x_t} exceeds a user-specific threshold u_t. These two models form the extreme ends of the following parameterised continuum of models.

Definition 4.2.3 (General Click Model)**.** *In the General Click Model (GCM) there is a single parameter $d \in [1, \infty)$. Let $a_t^* \in \operatorname{argmax}_{a \in A_t} w_{a,x_t}$ with ties broken arbitrarily. The click probability for arm a_t^* is $w_{a_t^*,x_t}$ and for all other arms $a \in A \backslash a_t^*$ it is $(w_{a,x_t})^d$. Therefore the CTR of the set A_t for known $\mathbf{w}_{\mathcal{A}}$ and x_t is*

$$r^d_{\text{GCM}}(x_t, A_t, \mathbf{w}_{A_t}, d) = 1 - (1 - w_{a_t^*,x_t}) \prod_{a \in A_t \backslash a_t^*} (1 - (w_{a,x_t})^d).$$

Setting $d = 1$ gives PCM and $d \to \infty$ results in TCM. In Section 4.3.3 we will demonstrate how, with latent x_t, the diversity of the arm set with optimal CTR changes with d.

Since x_t is unobserved a more important quantity is the expected reward over $\mathbf{q}$. Since, in any particular instance, $\mathbf{q}_t$ and $\mathbf{w}_A$ are fixed, we write this, the CTR, as

$$\text{CTR}^d(A) = \mathbb{E}_{x_t \sim \mathbf{q}_t}[r^d_{\text{GCM}}(x_t, A, \mathbf{w}_A)]. \tag{4.2.1}$$

The expected reward for PCM is therefore given by $CTR^1(A)$ and for TCM we denote the expected reward by $CTR^\infty(A)$. As $\mathbf{q}$ defines a discrete distribution taking its expectation is easy to do for all click models.

In the full model the arm weights $\mathbf{w}$ are not known exactly but are learnt over time by observing clicks. A model and solution for learning the weights will be given in

Sections 4.4 and 4.5. Where arm weights are known the CTR, as given in Equation 4.2.1, is our objective function. The problem of maximising this is studied in Section 4.3 together with an investigation into set diversity.

4.3 Solution Method and Analysis when Weights are Known

Generally in bandit problems maximising the immediate reward (exploiting) is the simple part of the problem. The difficulty comes from calculating the value of information (exploration). However, in the multiple elements problem, exploiting is not straightforward as there is a combinatorial explosion in the number of possible sets of arms that are available and online evaluation of all of these is computationally impractical. Given the web-based primary applications of this problem it is necessary to use algorithms that are very fast to select arm sets. This section will examine arm set selection in the simplified setting where weight vectors $\mathbf{w}$ are assumed to be known. First a method will be giving for handling the combinatorial aspect of the problem by exploiting a property of the click models then, in Section 4.3.3 the performance of this and other methods will be assessed as well as how exploring how the diversity varies with click model and selection method. Throughout this section the time subscript t will be dropped for simplicity as it is not relevant when weights are known.

4.3.1 Submodularity

The reward functions for our models possess a property, *submodularity*, that has been well studied (e.g. Krause and Golovin 2014) for which there is a simple but effective heuristic algorithm. Submodularity in our context captures the intuitive idea of diminishing returns as the number of elements chosen increases - adding an element to a large set gives a smaller increase in set CTR than adding it to a smaller subset.

Definition 4.3.1. *A set function f is* submodular *if for every $A \subseteq B \subseteq \mathcal{A}$, and $a \in \mathcal{A} \setminus B$ it holds that*

$$f(A \cup \{a\}) - f(A) \geq f(B \cup \{a\}) - f(B).$$

A submodular function for which the $f(A \cup \{a\}) - f(A)$ are all nonnegative is said to be *monotone*.

Proposition 4.3.2. CTR^d *is monotone submodular.*

Proof. $\mathrm{CTR}^d(A \cup \{a\}) - \mathrm{CTR}^d(A)$ is an expectation of

$$\Delta^d_{a,A,x} := r^d_{\mathrm{GCM}}(x, \{a\} \cup A, \mathbf{w}_{\{a\} \cup A}) - r^d_{\mathrm{GCM}}(x, A, \mathbf{w}_A)$$

over $x \sim \mathbf{q}$. It is thus sufficient to show that $\Delta^d_{a,A,x} - \Delta^d_{a,B,x} \geq 0$ for all $A \subseteq B \subseteq \mathcal{A}$, $a \in \mathcal{A} \setminus B$, x and $\mathbf{w}_{\mathcal{A}}$.

We derive formulae for $\Delta^d_{a,A,x}$. Let $a^* = \operatorname{argmax}_{a \in A} w_{a,x}$, resolving ties arbitrarily, and let $C_A = \prod_{b \in A \setminus \{a*\}} (1 - (w_{b,x})^d)$. Directly from Definition 4.2.3, we have $\Delta^d_{a,A,x}$ equal to

$$\begin{array}{ll} (w_{a,x})^d (1 - w_{a^*,x}) C_A, & \text{if } w_{a,x} < w_{a^*,x} \\ \left[w_{a,x} - w_{a^*,x} + (1 - w_{a,x})(w_{a^*,x})^d \right] C_A, & \text{if } w_{a,x} \geq w_{a^*,x}. \end{array}$$

Both are non-negative, so the CTR is monotone.

Now fix arbitrary $A \subseteq B \subseteq \mathcal{A}$, $a \in \mathcal{A} \setminus B$, x and $\mathbf{w}_{\mathcal{A}}$. Also define $b^* = \operatorname{argmax}_{a \in B} w_{a,x}$ and $C_B = \prod_{b \in B \setminus \{b*\}} (1 - {w_{b,x}}^d)$. We wish to show $\Delta^d_{a,A,x} - \Delta^d_{a,B,x} \geq 0$, and are faced with three cases:

1. $\boldsymbol{w_{a,x} \leq w_{a^*,x} \leq w_{b^*,x}}$

$$\begin{aligned} \Delta^d_{a,A,x} - \Delta^d_{a,B,x} &= (w_{a,x})^d \left[(1 - w_{a^*,x}) C_A - (1 - w_{b^*,x}) C_B \right] \\ &\geq (w_{a,x})^d \left[(1 - w_{a^*,x}) - (1 - w_{b^*,x}) \right] C_A \geq 0 \end{aligned}$$

since $C_A \geq C_B$ and $w_{b^*_x} \geq w_{a^*,x}$, as $A \subseteq B$.

2. $\boldsymbol{w_{a^*,x} \leq w_{a,x} \leq w_{b^*,x}}$

$$\begin{aligned}\Delta^d_{a,A,x} - \Delta^d_{a,B,x} &= \left[w_{a,x} - w_{a^*,x} + (1 - w_{a,x})(w_{a^*,x})^d\right] C_A \\ &\quad - (w_{a,x})^d(1 - w_{a^*,x})C_B \\ &\geq \Bigg[\left\{(w_{a,x} - w_{a^*,x}) - \left((w_{a,x})^d - (w_{a^*,x})^d\right)\right\} \\ &\quad + \left\{(w_{a,x})^d w_{b^*,x} - w_{a,x}(w_{a^*,x})^d\right\} \Bigg] C_A.\end{aligned}$$

The first curly braces are non-negative since $x - y \geq x^d - y^d$ for $1 \geq x > y \geq 0$ and $d \geq 1$; the second curly braces are non-negative since $w_{a^*,x} \leq w_{a,x} \leq w_{b^*,x}$.

3. $\boldsymbol{w_{a^*,x} \leq w_{b^*,x} \leq w_{a,x}}$

$$\begin{aligned}&\Delta^d_{a,A,x} - \Delta^d_{a,B,x} \\ &\geq \left[w_{b^*,x} - w_{a^*,x} - (1 - w_{a,x})((w_{b^*,x})^d - (w_{a^*,x})^d)\right] C_A \\ &\geq \left[w_{b^*,x} - w_{a^*,x} - (1 - w_{a,x})(w_{b^*,x} - w_{a^*,x})\right] C_A \geq 0\end{aligned}$$

where again we have used $w_{b^*,x} - w_{a^*,x} \geq (w_{b^*,x})^d - (w_{a^*,x})^d$.

For the TCM, where $d = \infty$, we simply have the limiting case, although a simpler argument can be made based directly on Definition 4.2.2. □

4.3.2 Algorithms for Selecting Element Sets

We now describe a number of heuristic algorithms for choosing element sets which will be evaluated in Section 4.3.3. Maximising a submodular function is NP-hard but for monotone submodular functions a computationally feasible greedy heuristic algorithm is known to have good properties (Nemhauser and Wolsey 1978). This algorithm starts with the empty set then iteratively adds the element that most increases the objective function.

It will be referred to here as the **sequential algorithm (SEQ)**. SEQ is known (Nemhauser and Wolsey 1978) to produce a solution within $1 - 1/e \approx 0.63$ of optimal. In practical terms 0.63 of optimal represents a considerable loss so in Section 4.3.3 we test the method using simulation to better estimate what the true loss is, as well as measuring the robustness of the method to model misspecification and quantifying the diversity of the sets chosen for different click models.

For comparison we present here a number of other algorithms which are natural alternatives.

Optimal (OPT). For small studies, we can optimise CTR^d. Note that we retain the expectation over latent state $x \sim \mathbf{q}$.

Naive (NAI). Elements, a, are ranked in order of independent element CTR given by $\mathbb{E}_{x \sim q} w_{a,x} = \mathbf{q} \cdot \mathbf{w}_a$, and the top m elements are selected in order.

Most Frequent User Preference (MFUP). We fix $\tilde{x} = \text{argmax}_x \, q_x$ and select the m elements with highest $w_{a,\tilde{x}}$.

Ordered User Preference (OUP). For $i \in 1, \ldots, m$, $\tilde{x}_i$ are selected in order of decreasing probability of occurrence q_x.[1] Then select $a_i = \text{argmax}_{a \notin \{a_j : j < i\}} w_{a,\tilde{x}_i}$ for each i.

All of these methods except OPT are computationally fast and scale well. With SEQ, for each of the m elements that it must select it is only necessary to calculate $CTR^d(A \cup \{a\}) - CTR^d(A)$ for the existing set of chosen elements A and each $a \in \mathcal{A} \setminus A$. As the CTR of the existing set is known at each stage from the calculation in the previous stage (note $CTR^d(\emptyset) = 0$) this requires only finding the CTR of each candidate set. Therefore SEQ requires the calculation of CTRs for fewer than mk sets compared to $\binom{k}{m}$ CTR calculations to solve to optimality. NAI requires k such

[1] To ensure that all $q_{\tilde{x}_i} > 0$, if $|\{x : q_x > 0\}| < m$ then fill the remaining slots by repeating the sequence $\{x : q_x > 0\}$ until $|\tilde{\mathbf{x}}| = m$.

calculations. Set CTRs are fast to compute since taking the expectation of $r(\cdot)$ over $\mathbf{q}$ can be done with vector multiplication. The last two methods do not require taking expectations which could be an advantage in more complex models.

4.3.3 Simulation Study

This section will give the results of simulation experiments investigating the sets chosen by each of the algorithms from Section 4.3.2 under the different click models. The arm sets selected by different algorithms will be analysed first for set CTR, then for set diversity.

Results are based on 1000 instances. Each instance represents a unique user. Each weight is drawn for each instance independently from a mixture distribution where each is *relevant* with probability $\xi = 0.5$ and non-relevant otherwise. If relevant the weight is drawn from a $Beta(\alpha, \beta)$ distribution, otherwise the weight is 0.001. The values of the parameters α, β will be given with the results. The low non-relevance weight represents a mismatch between the element and the user's state where it is assumed that the user will click with some low non-zero probability. On each instance, the state distribution $\mathbf{q}$ is sampled from a Dirichlet distribution with all n parameters equal to $1/n$. In response each set choosing algorithm selects a set of m arms from the available k. In all simulations there are $n = 20$ possible states for x with $k = 40$ elements and a set size of $m = 3$. Varying these parameters did not change the overall pattern of results.

Experimental Results: CTR

Table 4.3.1 gives the percentage *lost CTR* which is the the difference between the CTR of the sets chosen by the OPT and those chosen by the heuristic as a percentage of the optimal CTR. This is averaged over all instances. In each instance there is a true click model (either PCM or TCM, as given in the left column) and a set choosing algorithm. The click model assumed by OPT or SEQ is appended to its name (e.g. SEQ-TCM assumes that the click model is TCM - whether that is correct or not). The other methods do not assume any click model. The absolute CTRs of OPT using the correct click model with $\beta = 2$ and $\beta = 9$ respectively are 0.877 and 0.451 for PCM, and 0.773 and 0.316 for TCM.

Click Model	β	Set Choosing Method						
		OPT-PCM	OPT-TCM	SEQ-PCM	SEQ-TCM	NAI	MFUP	OUP
PCM	2	0%	9.2%	0.0%	8.9%	4.2%	17.0%	9.8%
PCM	9	0%	19.9%	0.0%	19.8%	0.6%	13.6%	20.7%
TCM	2	3.4%	0%	3.4%	0.1%	10.6%	25.6%	1.1%
TCM	9	6.6%	0%	6.6%	0.1%	9.6%	27.2%	1.4%

Table 4.3.1: Lost reward as a percentage of optimal reward over 1000 instances with $n = 20$, $k = 40$, $m = 3$, $\xi = 0.5$, $\alpha = 1$ and β as shown.

It can be seen that SEQ performs similarly to OPT with performance being much better than the theoretical guarantees when the assumed click model is correct. However, both methods do badly when the click model is incorrectly specified. The other methods perform poorly on at least one of the click models with OUP better on TCM and NAI better on PCM. These preferences can be explained by the set diversity for each as will be given in the next section.

The performance of NAI illustrates an issue with PCM as a choice for click model. Despite NAI ignoring interaction effects, for PCM with $\beta = 9$ it performs well. When $\mathbf{w}_a$ is small, $r_{\text{PCM}}(x, A, \mathbf{w}_A) = 1 - \prod_{a \in A}(1 - w_{a,x}) \approx \sum_{a \in A} w_{a,x}$, and NAI is optimal for the problem of maximising the expected value of this last quantity. So if weights are small (as would be common in problems such as web advertising) then using PCM does not result in sets where interactions between elements are important. This goes against the intuition that interactions are important.

Experimental Results: Diversity

To compare set diversity of different methods and click models we first need a measure of diversity appropriate to our problem. From Vargas and Castells (2011), the "pairwise dissimilarity between recommended items" is commonly used. Based on this we define the following simple measure of redundancy or overlap:

Definition 4.3.3 (Overlap and Diversity)**.** *The* overlap *of a set of two arms* $A = a_1, a_2$ *is*

$$\mathit{overlap}(A) = \frac{\sum_{x=1}^{n} \min(w_{1,x}, w_{2,x})}{\min[\sum_{x=1}^{n}(w_{1,x}), \sum_{x=1}^{n}(w_{2,x})]} .$$

For sets of arms larger than 2 the overlap is given by

$$\frac{2}{|A|(|A|-1)} \sum_{a_i, a_j \in A, i<j} \mathit{overlap}(\{a_i, a_j\}).$$

The diversity *of the set* A *is given by* $1 - \mathit{overlap}(A)$*.*

Note that these measures are independent of the click model used. For two arms, if $\mathit{overlap} = 1$ then the weights for one arm are all larger than the corresponding weights of the other arm. In that case the lesser of the two arms contributes nothing under TCM. However, it does contribute under PCM. If $\mathit{overlap} = 0$ then the weight vectors of all arms are pairwise orthogonal and $\mathrm{CTR}^d(A) = \sum_{a \in A} \mathrm{CTR}^d(a)$ for all $d \in [1, \infty)$ as well as for TCM.

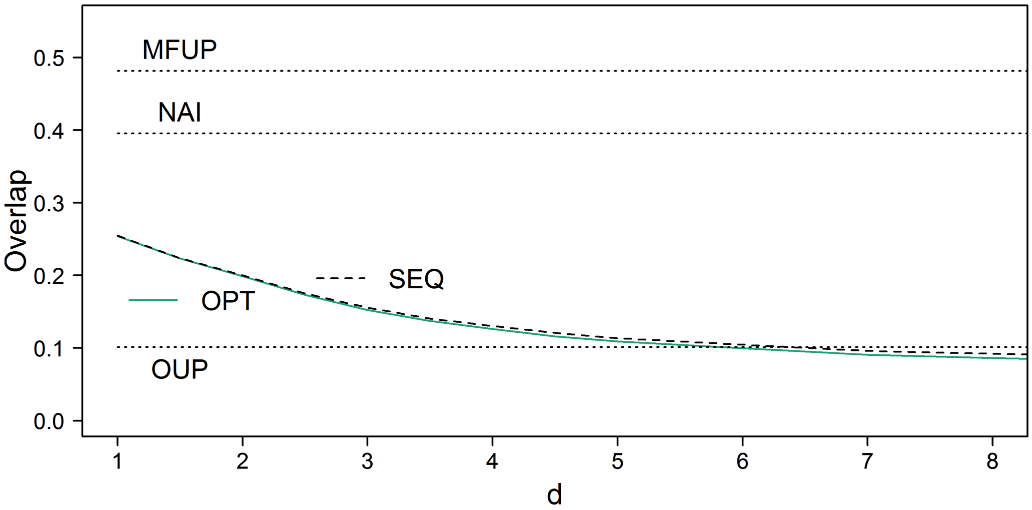

Figure 4.3.1: Overlap for set choosing algorithms over $d \in [1, 8]$ averaged over 1000 instances with $m = 3$, $n = 20$, $\xi = 0.5$ and $k = 40$ arms, each with $\alpha = 1$ and $\beta = 2$.

To investigate diversity we measure the mean overlap values of element sets chosen by algorithms. These are given in Figure 4.3.1 over a range of GCM parameter values d used by the OPT and SEQ algorithms. The other methods ignore the click model and so have unchanging overlap over d.

The overlap of the optimal sets decreases with d which shows how the diversity requirements change with the click model. SEQ shows a very similar pattern and chooses similar sets to OPT but with generally slightly greater overlap.

NAI and MFUP choose sets that are insufficiently diverse for any model which explains their poor performance in the CTR simulation results particularly on TCM. OUP has low overlap which is at an appropriate level for TCM and higher d but which is too low for PCM which fits with its poor CTR on PCM. No method is as diverse as OPT and SEQ for high d indicating that maximising rewards under the TCM model is an effective method to produce diverse sets.

4.4 Unknown Weights - Inference

In the last section arm weights were assumed to be known in order to explore set selection in a simplified setting. From now on we will assume that weights are initially unknown but can be learnt over time. This requires (i) a model for learning about the weights from user actions and, (ii) a method of selecting sets so that weights are learnt effectively while still choosing sets with good CTR. The second of these challenges will be addressed in Section 4.5 where we will give a simple method by which existing bandit algorithms can be adapted to learn the true weights while still retaining the CTR performance of the SEQ algorithm as weights are learnt.

This section will consider the first question, that of creating a model of learning for weights. Such a model requires an estimate of our current knowledge of each weight (a point estimate and an estimate of uncertainty) together with a method to record how that knowledge changes as user actions are observed. A Bayesian model will be used to do this which is presented in Section 4.4.1. There are practical computational problems with implementing this exactly so an approximate version is detailed in Section 4.4.2. An analysis of this method is given in Section 4.4.3 together with a discussion of conditions on $\mathbf{q}$ required for learning to be reliable.

This section and the next will concentrate on PCM and TCM but the methods given apply to all GCM and can easily be adapted.

4.4.1 Feedback and Learning

A Bayesian framework will be used to quantify our knowledge of weights in the form of a joint probability distribution over all weights with density given by $p(\mathbf{w}_{\mathcal{A}})$. Given this density a single step of learning proceeds as follows. We are presented with a user with topic preference $\mathbf{q}$. In response we choose a set of arms A, to which the user gives feedback in the form of either a click (together with which arm was clicked) or no click. Based on this feedback, the belief density $p(\mathbf{w}_{\mathcal{A}})$ is updated which is then used in the next step. Since only a single updating step is described in this section no time subscripts will be used in the notation for simplicity. Updating through time is simply a series of single updating steps. Formally $p(\mathbf{w}_{\mathcal{A}})$ would then need to be conditioned on the history of all relevant information up to the current time but, again, for notational simplicity this is omitted here.

To summarise user feedback two new variables will be introduced: a binary variable y where $y = 1$ if the user clicked some arm and $y = 0$ otherwise and m^*, which is the number of arms considered by the user. Under the cascade model $m^* = i$ if arm a_i is clicked or $m^* = m$ if no arm is clicked. This allows us to distinguish between two possible interpretations for arms that are not clicked. For arms $a_i, i \leq m^*$ that are considered by the user we receive information which affects the updating but for arms $a_i, i > m^*$ not considered no information is received. To simplify notation in the following it is useful to define the set of arms considered by the user but not clicked as

$$A' = \begin{cases} A & \text{if } y = 0 \\ \\ \{a_1, \ldots, a_{m^*-1}\} & \text{if } y = 1. \end{cases}$$

PCM. The joint distribution for all weights given a feedback step is updated as given below. In the following note that $p(\cdot|\mathbf{q}, x)$ simplifies to $p(\cdot|x)$ and that $p(\mathbf{w}_{\mathcal{A}}, x|\mathbf{q}, A) = p(\mathbf{w}_{\mathcal{A}})q_x$ since $p(x \mid \mathbf{q}) = q_x$ which is independent from $p(\mathbf{w}_{\mathcal{A}})$. The weight belief posterior after a user action is,

$$\begin{aligned}
p(\mathbf{w}_{\mathcal{A}}|y, m^*, \mathbf{q}, A) &= \sum_{x=1}^{n} p(\mathbf{w}_{\mathcal{A}}, x|y, m^*, \mathbf{q}, A) \\
&= \sum_{x=1}^{n} \frac{p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A)p(\mathbf{w}_{\mathcal{A}}, x|\mathbf{q}, A)}{p(y, m^*|\mathbf{q}, A)} \\
&= \frac{1}{p(y, m^*|\mathbf{q}, A)} \sum_{x=1}^{n} \left\{ \left[(w_{a_{m^*},x})^y \prod_{a \in A'} (1 - w_{a,x}) \right] p(\mathbf{w}_{\mathcal{A}}) q_x \right\} \\
&= \frac{p(\mathbf{w}_{\mathcal{A}})}{p(y, m^*|\mathbf{q}, A)} \sum_{x=1}^{n} \left[q_x (w_{a_{m^*},x})^y \prod_{a \in A'} (1 - w_{a,x}) \right].
\end{aligned}$$

TCM. The updating equation for $\mathbf{w}_{\mathcal{A}}$ for TCM is similar to that for PCM except that $p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q})$ is different due to the presence of the user threshold u:

$$\begin{aligned}
p(\mathbf{w}_{\mathcal{A}}|y, m^*, \mathbf{q}, A) &= \sum_{x=1}^{n} p(\mathbf{w}_{\mathcal{A}}, x|y, m^*, \mathbf{q}, A) \\
&= \sum_{x=1}^{n} \frac{p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A)p(\mathbf{w}_{\mathcal{A}}, x|\mathbf{q}, A)}{p(y, m^*|\mathbf{q}, A)} \\
&= \frac{p(\mathbf{w}_{\mathcal{A}})}{p(y, m^*|\mathbf{q}, A)} \int_{u=0}^{1} \sum_{x=1}^{n} \left[q_x (\mathbb{1}_{\{w_{a_{m^*},x} > u\}})^y \prod_{a \in A'} \mathbb{1}_{\{w_{a,x} \leq u\}} \right] du.
\end{aligned} \tag{4.4.1}$$

The updating equations for both PCM and TCM therefore involve finding the joint distribution over a large number of variables which cannot be decomposed due to dependency on $\mathbf{q}$. Not knowing the state x means we do not know which $w_{a,x}$ to attribute any click or refusal to click. TCM has the added complication of dependency on the latent user threshold u which is common to all arms. This means conjugate updates are not possible and exact updating is not practical. The next section will describe an approximate updating method that can be used instead.

4.4.2 Updating Weight Beliefs

The standard way to resolve the issues in updating caused by dependency on latent variables, such as found in the previous section, is to use an *expectation maximisation* algorithm for which online versions exist (e.g. Cappé and Moulines 2009; Larsen et al. 2010). The approach used is to sample an $\tilde{x}$ from the belief distribution for x and then update the weights using $\tilde{x}$ as the state. By conditioning on $\tilde{x}$ instead of $\mathbf{q}$ we can treat user actions for any arm a as a Bernoulli trial and attribute successes or failures to $w_{a,\tilde{x}}$. For PCM this allows beliefs for all points to be independent Beta distributions. Each weight $w_{a,x}$ has a belief distribution $W_{a,x} \sim Beta(\alpha_{a,x}, \beta_{a,x})$ and the joint distribution of the weight beliefs for all arms in $\mathcal{A}$ is $\mathbf{W}_{\mathcal{A}}$. The belief state is given by the α and β values so $2kn$ values are required to store the belief state. The model is now conjugate and the update conditional on $\tilde{x}$ is given by $\alpha_{a_{m^*},\tilde{x}} \leftarrow \alpha_{a_{m^*},\tilde{x}} + y$ and $\beta_{a,\tilde{x}} \leftarrow \beta_{a,\tilde{x}} + (1-y)$ for all $a \in A'$ with all other α, β values unchanged.

A key observation in Larsen et al. (2010) is that the belief distribution from which $\tilde{x}$ is drawn should be dependent on the user feedback just observed (rather than just $\mathbf{q}$). This posterior $\tilde{\mathbf{q}} = (\tilde{q}_1, \ldots, \tilde{q}_n)$, which depends on $\mathbf{W}_{\mathcal{A}}$, y, m^*, $\mathbf{q}$ and A is found by,

$$\begin{aligned}
\tilde{q}_x &= \int_{\mathbf{w}_{\mathcal{A}} \sim \mathbf{W}_{\mathcal{A}}} p(x|\mathbf{w}_{\mathcal{A}}, y, m^*, \mathbf{q}, A)\, \mathrm{d}\mathbf{w}_{\mathcal{A}} \\
&= \int_{\mathbf{w}_{\mathcal{A}} \sim \mathbf{W}_{\mathcal{A}}} \frac{p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A) p(x|\mathbf{w}_{\mathcal{A}}, \mathbf{q}, A) p(\mathbf{w}_{\mathcal{A}})}{p(\mathbf{w}_{\mathcal{A}}, y, m^*|\mathbf{q}, A)}\, \mathrm{d}\mathbf{w}_{\mathcal{A}} \\
&= \int_{\mathbf{w}_{\mathcal{A}} \sim \mathbf{W}_{\mathcal{A}}} \frac{p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A) p(x|\mathbf{w}_{\mathcal{A}}, \mathbf{q}, A)}{p(y, m^*|\mathbf{w}_{\mathcal{A}}, \mathbf{q}, A)}\, \mathrm{d}\mathbf{w}_{\mathcal{A}} \\
&= q_x \int_{\mathbf{w}_{\mathcal{A}} \sim \mathbf{W}_{\mathcal{A}}} \frac{p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A)}{\sum_{x=1}^{n} p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A)}\, \mathrm{d}\mathbf{w}_{\mathcal{A}},
\end{aligned}$$

where the last step is because the prior for the state $p(x|\mathbf{w}_{\mathcal{A}}, \mathbf{q}, A) = p(x|\mathbf{q}) = q_x$ depends only on $\mathbf{q}$. It remains to find $p(y, m^*|\mathbf{w}_{\mathcal{A}}, x, \mathbf{q}, A)$. Under PCM this is easily found since, given x, the probability of clicking any arm a considered by the user is the same as its independent click probability (as though it were the only arm in the set) and is independent from all weights except $w_{a,x}$. That is,

$$\begin{aligned}\int_{\mathbf{w}_\mathcal{A}\sim\mathbf{W}_\mathcal{A}} & p(y, m^* = 1|\mathbf{w}_\mathcal{A}, x, \mathbf{q}, A = \{a\})\,\mathrm{d}\mathbf{w}_\mathcal{A} \\ &= \int_{w_a\sim W_{a,x}} p(y, m^* = 1|w_a, x, A = \{a\})\,\mathrm{d}w_a \\ &= (\mu_{a,x})^y(1-\mu_{a,x})^{(1-y)},\end{aligned}$$

where $\mu_{a,x} = \frac{\alpha_{a,x}}{\alpha_{a,x}+\beta_{a,x}}$ is the expectation of $W_{a,x}$. Therefore,

$$\tilde{q}_x = q_x \frac{(\mu_{a_{m^*},x})^y \prod_{a\in A'}(1-\mu_{a,x})}{\sum_{j=1}^n [q_j(\mu_{a_{m^*},x})^y \prod_{a\in A'}(1-\mu_{a,x})]} \,. \tag{4.4.2}$$

The complete method is shown in Algorithm 1.

Algorithm 1 Posterior Sampled Bayesian Updating

```
Input: Weight belief distributions $\mathbf{W}_A$ for the set of arms $A$ presented to the user;
    a user response given by $y$ and $m^*$; the state probability vector $\mathbf{q}$ for the $n$ states.
  Calculate the posterior state probabilities $\tilde{\mathbf{q}} = (\tilde{q}_1, \ldots, \tilde{q}_n)$, given in (4.4.2).
  for all arms $a_i : i = 1, \ldots, m^*$ do
    Draw $\tilde{x}_i$ from $\tilde{\mathbf{q}}$.
    Update $w_{a_i,\tilde{x}_i} \leftarrow w_{a_i,\tilde{x}_i}|y, m^*$.
  end for
Output: A set of updated belief distributions $\mathbf{W}_A$.
```

By studying the stochastic approximation methods (e.g. Larsen et al. 2010) it becomes clear that deterministic averaging over x is equally valid. This is done by updating elements of $\mathbf{W}_\mathcal{A}$ in proportion to $\tilde{\mathbf{q}}$, that is, by setting $\alpha_{a_{m^*},x} \leftarrow \alpha_{a_{m^*},x} + y\tilde{q}_x$ and $\beta_{a,x} \leftarrow \beta_{a,x} + (1-y)\tilde{q}_x$ for all $a \in A'$. All other α, β values are unchanged as before. These two methods will be compared in Section 4.4.3.

TCM does not have the advantage that the click probabilities are independent if x is known and therefore it does not reduce to a simple updating model even given a sampled x. Adapting (4.4.1) in Section 4.4.1, the updating for known x is

$$\begin{aligned}
p(\mathbf{w}_{\mathcal{A}}|y, m^*, x) &= \frac{p(\mathbf{w}_{\mathcal{A}})}{p(y, m^*|\mathbf{q})} \int_{u=0}^{1} q_x (\mathbb{1}_{\{w_{a_{m^*},x} > u\}})^y \prod_{a \in A'} \mathbb{1}_{\{w_{a,x} \leq u\}} du \\
&= \frac{q_x p(\mathbf{w}_{\mathcal{A}})}{p(y, m^*|\mathbf{q})} \int_{u=0}^{1} (\mathbb{1}_{\{w_{a_{m^*},x} > u\}})^y \mathbb{1}_{\{u > \max_{a \in A'}(w_{a,x})\}} du \\
&= \frac{q_x p(\mathbf{w}_{\mathcal{A}})}{p(y, m^*|\mathbf{q})} \left[(w_{a_{m^*},x})^y - \max_{a \in A'}(w_{a,x}) \right].
\end{aligned}$$

This would not fit into a simple updating scheme. In addition, to use this in the posterior sampled Bayesian updating in Algorithm 1 it would be necessary to obtain the posterior for $\mathbf{q}$ which would require taking integrals over the multiple belief distributions. Therefore the heuristic updating method used for PCM will not work for TCM. The approach we use to handle this difficulty is to record and update beliefs as though the click model was PCM. For the arm a_1 in the first slot the models are the same but for subsequent arms we are making an independence assumption. This loses information but this approach will be shown to work well in simulations in Section 4.6. The same method can also be used for any GCM.

4.4.3 Approximate Updating Analysis

In Section 4.4.2 two approximate updating methods were given, each using $\tilde{\mathbf{q}}$ the posterior of $\mathbf{q}$ once the user actions have been observed. The first updates using a sampled value from $\tilde{\mathbf{q}}$ while the second updates deterministically in proportion to $\tilde{\mathbf{q}}$. A simple comparison of these can be made by taking the single arm case where $k = m = 1$ so that the results are unaffected by click model or set choosing algorithms.

The simulation was run 500 times over $N = 1000$ time steps. Each $\mathbf{q}$ is an i.i.d. Dirichlet distribution with all n parameters equal to $1/n$. At each time, the absolute error $|\mu_{1,x} - w_{1,x}|$ for each state x was recorded. This was averaged over all the runs and is shown on the left in Figure 4.4.1. In addition, the state thought to have the highest weight $\text{argmax}_x(\mu_{1,x})$ was compared to the truth $\text{argmax}_x(w_{1,x})$ and the proportion that this was incorrect was recorded. The purpose of this was to assess how often the updating had got the state weights mixed up, a problem found to happen occasionally in a similar problem in Larsen et al. (2010). This is shown on the right in Figure 4.4.1. On both measures the deterministic version performed better. Similar patterns are found with other values of n and β.

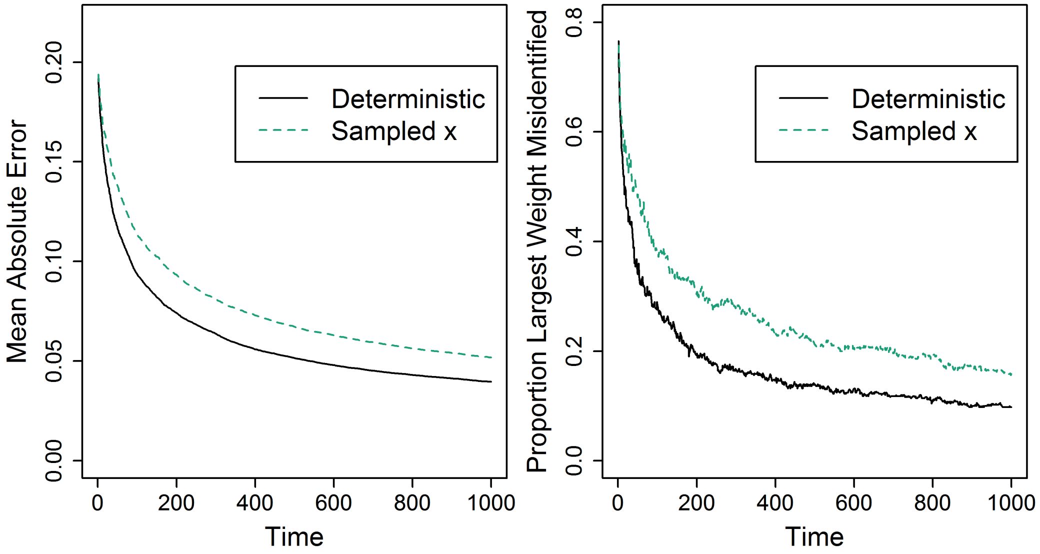

Figure 4.4.1: The mean absolute error of $\hat{w}$ (left) and the proportion of times that the highest weight is misidentified (right) over 500 runs for a single arm with $n = 5$, $\alpha = 1$ and $\beta = 2$.

It is important to note that this relies on $\mathbf{q}$ being varied over time since if $\mathbf{q}$ is fixed then inference is unreliable as illustrated in Figure 4.4.2. This is an identifiability issue due to there being insufficient information to solve the problem, rather than an issue with the updating method. This can be seen by considering the offline version of the problem for the simplest case where $n = 2$ given T observations. We then have a system of equations $\mathbf{y} = Q\mathbf{w} + \boldsymbol{\epsilon}$ where $\mathbf{y}$ is a vector of T observed rewards, Q is a $T \times 2$ matrix where each row t is $\mathbf{q}_t$, and $\boldsymbol{\epsilon}$ is a noise term. The least squares solution is given by $(Q^\top Q)^{-1} Q^\top \mathbf{y}$ which has a unique solution if and only if Q is of rank 2. Therefore using a constant $\mathbf{q}_t$ will not give a single solution.

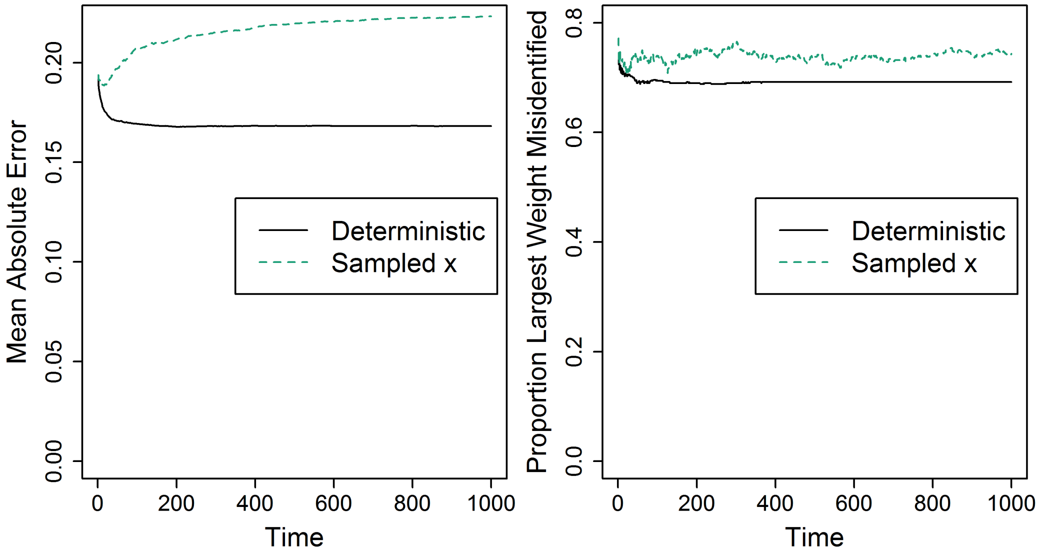

Figure 4.4.2: The mean absolute error of $\hat{w}$ (left) and the proportion of times that the highest weight is misidentified (right) over 500 runs for a single arm with $n = 5$, $\alpha = 1$ and $\beta = 2$ and $q = (0.8, 0.05, 0.05, 0.05, 0.05)$ fixed.

This may be important in practice. Say a given $\mathbf{q}$ comes from a particular search term for which we must choose a set of adverts then the problems of learning with a fixed $\mathbf{q}$ indicate that we cannot reliably learn the qualities of the adverts by observing clicks with only a single search term or very similar search terms. Simulations suggest that the algorithm attributes rewards most accurately and reliably when each q_x is sometimes large. It may be that in practice some states are always more likely and in these situations the estimates of weights for states occurring less often may not be accurate. So feedback for a variety of search terms on a range of preferences gives better information even if we are only interested in learning about responses to a particular search or related group of searches.

4.5 Unknown Weights - Exploration

Section 4.3 gave a method of selecting a set of arms when weights are known. This corresponds to the exploitation part of a bandit problem. With weights only known up to distribution it is necessary to choose arms to learn the true weights effectively while also favouring actions that are likely to give the best immediate reward. This is the exploration versus exploitation problem. The objective is still to achieve high CTR but not for a single time or even a particular time horizon but up to any given time. In this section a method is given whereby a range of existing bandit algorithms (or *policies*) can be easily adapted to work with this problem.

Index policies are a family of bandit policies which assign to each arm a real valued index and then chooses the arm with the highest index. Examples include: the Greedy policy for which the index is the expected immediate reward; the Gittins index (Gittins et al. 2011) where the index is found by solving a stochastic optimisation problem on each arm; Thompson sampling (e.g. May et al. 2012) which uses a stochastic index by sampling from the posterior for the arm; and the upper-confidence-bound policy family (e.g. Auer et al. 2002) where the index is based on some optimistic estimate of the arm's value.

To use any of these methods in the multiple arm problem given here the index is substituted for the true weight in a set selection algorithm as given in Section 4.3. Let I_t be the history up to time t which consists of all available information that is relevant to the current decision, namely $\mathbf{q}_1, \ldots, \mathbf{q}_{t-1}$; the weight priors, past actions $A_1, \ldots, A_{t-1}$; and user responses given by $m_1^*, \ldots, m_{t-1}^*$ and $y_1, \ldots, y_{t-1}$. For the purposes of choosing arms this information is used only via the current posterior weight belief distributions $\mathbf{W}_{\mathcal{A},t}$. Formally, a policy for our problem consists of two parts: an exploration algorithm $\nu(\mathbf{W}_{\mathcal{A},t}|I_t)$ which maps posterior distributions $\mathbf{W}_{\mathcal{A},t}$ to real valued $\tilde{\mathbf{w}}_{\mathcal{A},t}$ and a set selection algorithm $S(\mathcal{A}, \tilde{\mathbf{w}}_{\mathcal{A},t}, m)$ which takes the output of the exploration algorithm and outputs a set A of m chosen arms. Each $\tilde{w}_{a,t,i}$ in $\tilde{\mathbf{w}}_{\mathcal{A},t}$ can be thought of some proxy for the unknown weight $w_{a,t,i}$. As such it is natural for it to be in $(0, 1)$ but can take a more general form as long as it is an appropriate input to the set selection algorithm used. For the algorithms given in Section 4.3, where a CTR calculation is required, inputs would need to be in $[0, 1]$ although a monotone mapping from real valued $\tilde{\mathbf{w}}_{\mathcal{A},t}$ could be made if needed.

A desirable property of any bandit algorithm is that, as weight estimates become increasingly good, the arms chosen will converge to the arms that would be chosen if the true weights were known. It is for this reason that the analysis given in Section 4.3.3 is important - the expected reward rate as arm weights are learnt is limited by the expected reward of the set with known weights.

We will not attempt to compare the general performance of the many possible index policies that are available as there are plenty of examples of this elsewhere but instead we will concentrate on the performance of one, Thompson Sampling (TS), and any issues that arise as it is adapted to this problem. TS is chosen because it has properties that fit the requirements of this problem, namely that it is quick to run and in similar long horizon bandit problems it has been shown to work well and explore effectively (see e.g. Russo and Van Roy 2014, May et al. 2012). The algorithm used is given in Algorithm 2. Note that this makes a single draw from each weight posterior which is then used for each of the $1, \ldots, m$ arms selected. An alternative is to make separate draws for each slot which would avoid extra correlation between the selected arms but in practice the difference is small and it does not affect the results given here.

Algorithm 2 Multiple Action Thompson Sampling

```
Input: The available arms $\mathcal{A}$ with posterior weight beliefs $\mathbf{W}_{\mathcal{A},t}$; the number of arms
    to be selected $m$; a set choosing algorithm $S(\mathcal{A}, \tilde{\mathbf{w}}_{\mathcal{A},t}, m)$.
    for all $W_{a,i}, a \in \mathcal{A}, i = 1, \ldots, n$ do
        Draw $\tilde{w}_{a,t,i} \sim W_{a,t,i}$
    end for
    Select an arm set using set choosing algorithm $S(\mathcal{A}, \tilde{\mathbf{w}}_{\mathcal{A},t}, m)$
Output: A set of chosen arms $A$ of size $m$.
```

The next lemma works towards Theorem 4.5.2 that will give conditions under which Algorithm 2 using the deterministic updating scheme from Section 4.4.2 will select each arm infinitely often. The strongest condition is on the distribution of each $\mathbf{q}_t$, and will be discussed after the proof of the theorem.

Let $R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t \mid I_t) = \sum_{x=1}^{n}(q_{t,x}\tilde{w}_{a,t,x})$ denote the stochastic index for the multiple action TS policy for a single arm a where each $\tilde{w}_{a,t,x} \sim W_{a,t,x}$. Then under SEQ or NAI the arm chosen in slot one is the one with the highest index: $a_{t,1} = \operatorname{argmax}_{a\in\mathcal{A}} R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t | I_t)$.

Lemma 4.5.1. *Let $\tau_{a,T}$ be the set of times $t = 1, \ldots, T$ at which $a \in A_t$. Let $q^* = \min_{t,x} \Pr(q_{t,x} = 1)$ and $w^* = \max_{a\in\mathcal{A},x} w_{a,x}$ and, from these, set $\eta = q^*(1 - w^*)^m$. If $q^* > 0$ then under the deterministic updating scheme given in Section 4.4.2 using any click model from Section 4.2.2,*

$$\Pr\left(R^{TS}(a, \mathbf{W}_{a,T}, \mathbf{q}_T \mid I_t) \leq \frac{1}{1+\eta-\delta_1} + \delta_2\right) \to 1 \qquad \textit{as } |\tau_{a,T}| \to \infty$$

for any $a \in \mathcal{A}$ and any δ_1, δ_2 such that $\eta > \delta_1 > 0$ and $\delta_2 > 0$.

Proof. For any $a \in \mathcal{A}$, $x = 1, \ldots, n$ we will give bounds for expected rate at which $\alpha_{a,t,x}$ and $\beta_{a,t,x}$ increase as the arm a is selected over time (an upper bound for $\alpha_{a,t,x}$ and a lower bound for $\beta_{a,t,x}$). This will give an asymptotic upper bound less than 1 on each posterior mean $\mu_{a,t,x} = \mathbb{E}\left[W_{a,t,x}\right]$ as $|\tau_{a,t}| \to \infty$. Showing that $Var(W_{a,t,x}) \to 0$ as $|\tau_{a,t}| \to \infty$ then gives the required result. Throughout we will consider a to be an arbitrary arm in $\mathcal{A}$ and x to be an arbitrary state in $\{1, \ldots, n\}$.

Let $\alpha_{a,0,x}$ and $\beta_{a,0,x}$ be values of the parameters of the Beta prior placed on $w_{a,x}$, then an upper bound for $\alpha_{a,T,x}$, $T \geq 1$ is simply

$$\alpha_{a,T,x} \leq \alpha_{a,0,x} + |\tau_{a,T}| \tag{4.5.1}$$

since $\alpha_{a,T,x}$ can only increase by at most one at times when when $a \in A_t$ and is unchanging at other times.

For a lower bound on $\mathbb{E}\left[\beta_{a,T,x}\right]$ we consider only times when $a \in A_t$, $q_{t,x} = 1$ and $y_t = 0$. Note that $y_t = 0$ guarantees that arm a is considered by the user and $q_{t,x} = 1$ means that the failure to click can be attributed to $w_{a,x}$. Therefore for $t \geq 1$ we have

$$\beta_{a,t+1,x} \mid (q_{t,x} = 1, y_t = 0, a \in A_t, \beta_{a,t,x}) = \beta_{a,t,x} + 1. \tag{4.5.2}$$

At all other times $\beta_{a,t+1,x} \geq \beta_{a,t,x}$ since the β parameters cannot decrease. For PCM,

$$\Pr(y_t = 0 \mid q_{t,x} = 1, A_t, \mathbf{w}_{A_t}) = \prod_{b \in A_t} (1 - w_{b,x})$$

which is no larger than the corresponding probability for all other GCM click models and TCM. The probability that $y_t = 0$ can therefore be bounded below. Let $w^* = \max_{b \in \mathcal{A}, x} w_{b,x}$ and $q^* = \min_{t,x} \Pr(q_{t,x} = 1)$ then for any $A_t \subset \mathcal{A}$,

$$\Pr(y_t = 0 \mid A_t, \mathbf{w}_{\mathcal{A}}) \geq q^*(1 - w^*)^m. \tag{4.5.3}$$

We can now give a lower bound on $\mathbb{E}\left[\beta_{a,T,x} \mid I_1\right]$ where the expectation is joint over all $\mathbf{q}_t$, y_t, m_t^* for $t = 1, \ldots, T$, and I_1 is just the priors for $\mathbf{W}$. Using (4.5.2) and (4.5.3), we have at any time T,

$$\begin{aligned} \mathbb{E}[\beta_{a,T,x} \mid I_1] \geq \beta_{a,0,x} + & \\ & \sum_{t \in \tau_{a,T}} \Big[\Pr(q_{t,x} = 1) \Pr(y_t = 0 \mid q_{t,x} = 1, a \in A_t, \mathbf{w}_{A_t}) \Big] \\ & \geq |\tau_{a,T}| q^*(1 - w^*)^m. \end{aligned} \tag{4.5.4}$$

Let $\eta = q^*(1-w^*)^m$ and note that $\eta > 0$ since $w^* < 1$ by the problem definition and $q^* > 0$ by the assumption given in the statement of the Lemma. Combining (4.5.1) and (4.5.4) gives, for any $\tau_{a,T}$,

$$\begin{aligned}\mathbb{E}\left[\frac{\beta_{a,T,x}}{\alpha_{a,T,x}} \mid I_1\right] &\geq \frac{1}{\alpha_{a,0,x} + |\tau_{a,T}|}\mathbb{E}\left[\beta_{a,T,x} \mid I_1\right] \\ &\geq \frac{|\tau_{a,T}|\eta}{\alpha_{a,0,x} + |\tau_{a,T}|}\end{aligned}$$

and so by the strong law of large numbers,

$$\begin{aligned}\lim_{|\tau_{a,T}|\to\infty} \frac{\beta_{a,T,x}}{\alpha_{a,T,x}} \mid I_1 &\geq \frac{|\tau_{a,T}|\eta}{\alpha_{a,0,x} + |\tau_{a,T}|} \\ &= \eta\ . \end{aligned} \tag{4.5.5}$$

Note that

$$\mu_{a,T,x} = \frac{\alpha_{a,T,x}}{\alpha_{a,T,x} + \beta_{a,T,x}} = \frac{1}{1 + \frac{\beta_{a,T,x}}{\alpha_{a,T,x}}};,$$

and so from (4.5.5),

$$\Pr\left(\mu_{a,T,x} \leq \frac{1}{1+\eta-\delta_1}\right) \to 1 \quad \text{as} \quad |\tau_{a,T}| \to \infty \tag{4.5.6}$$

for any δ_1 such that $\eta > \delta_1 > 0$.

Then, using the variance of a Beta distribution and (4.5.4) we have

$$\begin{aligned}Var(W_{a,T,x}) &= \frac{\alpha_{a,T,x}\beta_{a,T,x}}{(\alpha_{a,T,x} + \beta_{a,T,x})^2(\alpha_{a,T,x} + \beta_{a,T,x} + 1)} \\ &< \frac{(\alpha_{a,T,x} + \beta_{a,T,x})^2}{(\alpha_{a,T,x} + \beta_{a,T,x})^2(\alpha_{a,T,x} + \beta_{a,T,x} + 1)} \\ &= \frac{1}{(\alpha_{a,T,x} + \beta_{a,T,x} + 1)} \to 0 \quad \text{as} \quad |\tau_{a,T}| \to \infty,\end{aligned}$$

and so for any $\delta_2 > 0$ the sampled $\tilde{w}_{a,T,x} \sim W_{a,T,x}$ satisfy

$$\Pr\left(\tilde{w}_{a,T,x} \mid \mu_{a,T,x} \leq \mu_{a,T,x} + \delta_2\right) \to 1 \quad \text{as} \quad |\tau_{a,T}| \to \infty. \tag{4.5.7}$$

By definition $R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t \mid I_t) = \sum_{x=1}^n (q_{t,x}\tilde{w}_{a,t,x}) \leq \max_x \tilde{w}_{a,t,x}$ where $\tilde{w}_{a,t,x} \sim W_{a,t,x}$. Therefore, to complete the proof it is sufficient that $\Pr(\tilde{w}_{a,T,x} < 1/(1+\eta-\delta_1)+\delta_2) \to 1$ as $|\tau_{a,T}| \to \infty$ for all $a \in \mathcal{A}$, $x = 1, \ldots n$ and any δ_1, δ_2 such that $\eta > \delta_1 > 0$ and $\delta_2 > 0$, which follows from (4.5.6) and (4.5.7). □

Theorem 4.5.2. *Let* $\Pr(q_{t,x} = 1) > 0$ *for all* $x = 1, \ldots, n$ *and* $t \geq 1$ *then the multiple action TS algorithm given in Algorithm 2 with SEQ or NAI as set choosing method samples infinitely often from each arm for any click model from Section 4.2.2. That is,* $\Pr(|\tau_{a,T}| \to \infty$ *as* $T \to \infty) = 1$ *for any arm* $a \in \mathcal{A}$*, where* $|\tau_{a,T}|$ *is set of times* $t = 1, .., T$ *that* $a \in A_t$*.*

Proof. We will assume that there is a non-empty set of arms $\mathcal{A}_F \subset \mathcal{A}$ whose members are sampled finitely often as $t \to \infty$ and show that this leads to a contradiction. By this assumption $\sum_{b \in \mathcal{A}_F} |\tau_{b,\infty}| < \infty$ and so there is a finite time $M = \max_{b \in \mathcal{A}_F} \tau_{b,t}$ even as $t \to \infty$.

Let $\mathcal{A}_I = \mathcal{A} \setminus \mathcal{A}_F$ be the set of arms sampled infinitely often (which must be non-empty). Let $q^* = \min_{t \geq 1, x} \Pr(q_{t,x} = 1)$, $w^* = \max_{a \in \mathcal{A}, x} w_{a,x}$ and $\eta = q^*(1 - w^*)^m$ as in the proof of Lemma 4.5.1 and Note that $\eta > 0$ since $w^* < 1$ by the problem definition and $q^* > 0$ by the given condition. Then fix some $0 < \delta_1 < \eta$ and $0 < \delta_2 < 1 - 1/(1 + \eta - \delta_1)$. Then by Lemma 4.5.1 for all $a \in \mathcal{A}_I$,

$$\Pr\left(R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t) \leq \frac{1}{1 + \eta - \delta_1} + \delta_2\right) \to 1 \text{ as } t \to \infty.$$

So there exists a finite random time $T > M$ such that

$$\Pr\left(R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t) \leq \frac{1}{1 + \eta - \delta_1} + \delta_2\right) > 1 - \delta_2 \text{ for } t > T \text{ and all } a \in \mathcal{A}_I. \quad (4.5.8)$$

Let $\epsilon = \min_{b \in \mathcal{A}_F} [\Pr(R^{TS}(b, \mathbf{W}_{b,T}, \mathbf{q}_T \mid I_T) > 1/(1 + \eta - \delta_1) + \delta_2)]$. Then for all $t > T$, $b \in \mathcal{A}_F$ we have

$$\Pr\left(R^{TS}(b, \mathbf{W}_{b,t}, \mathbf{q}_t \mid I_t) > \frac{1}{1 + \eta - \delta_1} + \delta_2\right) \geq \epsilon, \quad (4.5.9)$$

since no arm in $\mathcal{A}_F$ is selected at times $t > T > M$ and so $\mathbf{W}_{b,t}$ is unchanged over these times. We know that $\epsilon > 0$ since $\Pr(\tilde{w}_{b,T,x} > 1/(1 + \eta + \delta_1) + \delta_2) > 0$ for all b, x because $1/(1 + \eta - \delta_1) + \delta_2 < 1$ and $W_{b,T,x}$ is a Beta distribution with support $(0, 1)$.

Combining (4.5.8) and (4.5.9),

$$\Pr\left[R^{TS}(b, \mathbf{W}_{b,t}, \mathbf{q}_t \mid I_t) > R^{TS}(a, \mathbf{W}_{a,t}, \mathbf{q}_t \mid I_t), \forall a \in A\right] > \epsilon(1 - \delta_2) \quad (4.5.10)$$

for all $t > T$. Therefore

$$\sum_{t=T}^{\infty} \Pr(b \in A_t \text{ for some } b \in \mathcal{A}_F) > \sum_{t=T}^{\infty} \epsilon(1-\delta_2)^{|\mathcal{A}_I|} = \infty.$$

Using the Extended Borel-Cantelli Lemma (Corollary 5.29 of Breiman 1992) it follows that $\sum_{b \in \mathcal{A}_F} |\tau_{b,\infty}| = \infty$ which contradicts the assumption that $|\tau_{b,\infty}|$ is finite for all $b \in \mathcal{A}_F$. Therefore some arm in $\mathcal{A}_F$ is selected infinitely often and since $\mathcal{A}_F$ was of arbitrary size it follows that $\mathcal{A}_F = \emptyset$. □

The condition that all $\Pr(q_{t,x} = 1) > 0$ in Theorem 4.5.2 will not always hold in practice. The reason that this is needed is that it is possible, due to approximate updating, that $\mu_{a,t,x} \to 1$ as $|\tau_{a,t}| \to \infty$ even though $w_{a,x} < 1$. In practice this would be improbable unless the prior $W_{a,0,x}$ was unrealistically concentrated close to 1. An alternative method to avoid this problem would be to put an upper bound close to 1 on the value of $\mu_{a,t,x}$ used to calculate $\tilde{q}_{t,x}$. This would give a lower bound for the amount by which $\beta_{a,t,x}$ increases whenever $a \in A_t$, $y = 0$ and $q_{t,x} > 0$. Using this, Theorem 4.5.2 would hold without the $\Pr(q_{t,x} = 1) > 0$ condition.

4.6 Policy Performance for Unknown Weights

4.6.1 Regret Simulations

The proposed solution outlined in this section brings together a number of different elements. Previous sections have analysed these elements separately: the sequential set choosing algorithm when weights are known (Section 4.3.3) and the ability of the approximate updating scheme (Section 4.4.3) to learn the true weights given sufficient observations. Using TS ensures that there will be enough observations of the arms and that, if weight estimates converge to the true weights, the best arm sets will be chosen with increasing probability. This section uses simulations to test all these elements together using the objective of the full problem - maximising the total reward earned over time or, equivalently, minimising the cumulative *regret*. At any time T this is given by

$$\frac{1}{T}\sum_{t=1}^{T}\left[CTR^d(A_t^{SORACLE},\mathbf{w}_{\mathcal{A}},\mathbf{q}_t)-CTR^d(A_t^{\pi},\mathbf{w}_{\mathcal{A}},\mathbf{q}_t)\right]$$

where A_t^{π} and $A_t^{SORACLE}$ are the arm sets chosen at time t by, respectively, the policy being tested and the SORACLE policy. The SORACLE policy uses SEQ assuming the true weights are known.

For each simulation run a policy is selected. This consists of a set choosing method (either SEQ or NAI) and a bandit exploration algorithm (either the Greedy policy, which uses the posterior mean as an estimate of the true weights, or TS). Each policy is denoted by a two-part name where the first part gives the set choosing algorithm, either N for NAI or S for SEQ, and the second part gives the bandit algorithm, either TS Thompson Sampling or G for Greedy. For example, NTS indicates NAI paired with TS.

The true weights for the run are drawn from a mixture distribution where each is relevant with probability $\xi = 0.5$ and non-relevant otherwise. If relevant the weight is drawn from a $Beta(\alpha = 1, \beta = 2)$ distribution, otherwise the weight is 0.001. Each weight is given an independent prior belief which matches the distribution from which relevant weights are drawn, here $Beta(1, 2)$. At each time $t = 1, 2, \dots, T$ a state distribution $\mathbf{q}_t$ is sampled from a Dirichlet distribution with all n parameters equal to $1/n$. Note that this does not satisfy the assumption on $\mathbf{q}$ given in Therorem 4.5.2. In response the policy chooses a set of m arms from the available k. A user action is then simulated and weight beliefs updated. The values of m, T, n and k used are given with the results. This process is repeated with the other policies to be tested using the same weights, $\mathbf{q}_t$ and common random numbers. Both PCM and TCM are used as click models and the policies use the correct click model where relevant. Section 4.6.2 will consider learning when the click model is misspecified. Where the click model is used by a policy it will be appended to its name e.g. STS-PCM.

At each time, for each of the 500 simulation runs and each policy the cumulative regret is calculated using the chosen arm set and the true weights.

There are clearly many possible settings for these simulations and it is not practical to attempt to present the results of an exhaustive study of all possible combinations. Instead the simulations given here are intended as a complement to the earlier theory and to illustrate behaviour of various policies on the full problem.

We use two different sizes of problem to give an idea of how the learning rates scale. For these the cumulative sequential regret averaged over all runs is shown in Figure 4.6.1 for the smaller problem and in Figure 4.6.2 for the larger one. It can be seen that the overall pattern is the same for each but on different timescales.

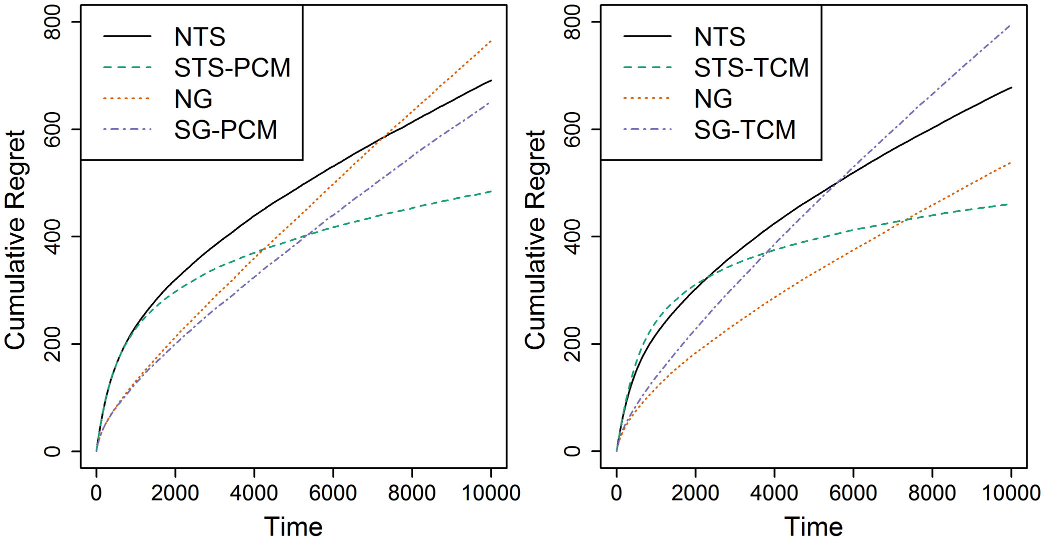

Figure 4.6.1: Mean cumulative sequential regret. Simulation setting are $n = 5$, $k = 20$, $m = 2$, $\xi = 0.5$, $\alpha = 1$ and $\beta = 2$. The true click model is PCM on the left and TCM on the right.

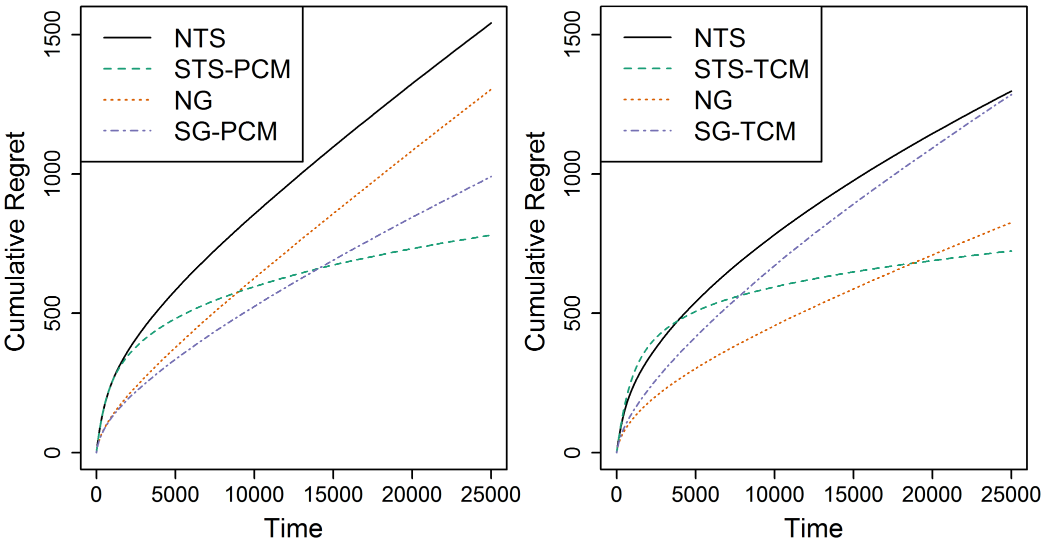

Figure 4.6.2: Mean cumulative sequential regret. Simulation setting are $n = 10$, $k = 40$, $m = 3$, $\xi = 0.5$, $\alpha = 1$ and $\beta = 2$. The true click model is PCM on the left and TCM on the right.

Some aspects of the results are as would be expected given previous analyses. For policies using TS, as the weights are learnt, SEQ chooses higher reward sets than NAI and so STS outperforms NTS. Greedy is more effective than TS at earlier times but does not learn well and so falls behind later. It takes longer for the superior learning of TS to pay off for TCM than PCM.

There are some surprises though. The two Greedy policies perform very differently on PCM and TCM. SG does better than NG on PCM as would be expected but on TCM the order is reversed indicating that the SG-TCM does not learn well. It appears that NTS does not learn well as it is slow to catch up with the Greedy policies and it is not clear whether it is catching the best of the Greedy policies at all on the larger problem. This supports the use of combination of SEQ and TS on this problem. Further simulations in Section 4.6.2 will look at learning rates and these issues in more detail.

It is worth noting that learning rates are generally slow in this problem. There are kn parameters to learn compared to just k for the standard MAB. In addition there is the problem of unknown x which makes feedback more noisy than normal. Offsetting this we get to choose multiple arms at each time but due to the cascade model this multiplies the learning rate by less than m. The contextual nature of the problem means that there is no single best arm and this decreases the policy's ability to focus learning on a small subset of arms over time. Learning would be faster with more accurate priors as the ones used ignore the possibility of non-relevance of the arms. However, our purpose here is to produce a general working algorithm rather than one that is optimised to particular problem settings.

4.6.2 State of Knowledge Simulations

The simulations in the previous section compared policies based on SEQ and NAI using cumulative regret over time. An issue with this form of simulation is that it does not answer the question of how well the methods explore since the inferior set choosing ability of NAI once weights are known (see Section 4.3.3) limits the reward gained even if learning is superior. This section tests how NAI or SEQ affects the learning capabilities of policies. In addition, the previous section's experiments assumed the correct click model was known but misspecified models will also be tested here.

To measure how effectively a policy has learnt we run the same simulations as in Section 4.6.1 but rather than record the cumulative regret, we instead record the *greedy posterior regret* (GPR). To define this we first define the *greedy posterior reward* of a policy. At any time this is the expected reward using the SG policy with the true click model if no further learning occurs beyond that time. The GPR is then the difference between this value and the expected reward of the SORACLE policy which knows the true weights. This is a more useful measure of learning than more general measures such as the Kullback Leibler difference because, rather than give a general measure of learning or convergence, GPR gives a measure of *effective* learning. It estimates how well the policy has focused on the most important arms which are those likely to have larger weights and which are therefore more likely to be part of optimal sets.

The simulation estimates the GPR for a policy as follows. A $\mathbf{q}_t$ is generated as usual and a single action taken using the SG policy using the posterior mean of all weights at time t. This is done for a 100 different simulated values of $\mathbf{q}_t$ for each run of the original simulation and the regret averaged. The GPR over time for TS-based policies with PCM and TCM as the true click model are shown in Figure 4.6.3. As well as comparing SEQ and NAI it also tests for the effects of click model misspecification. So STS-PCM assumes PCM while STS-TCM assumes TCM. NTS acts the same for either click model. The time axis starts at $t = 1000$ because GPR values are high at early times before the policies have had much time to learn. GPR values for the Greedy-based policies are much higher and so these are shown on a separate plot in Figure 4.6.4 with just NTS for comparison.

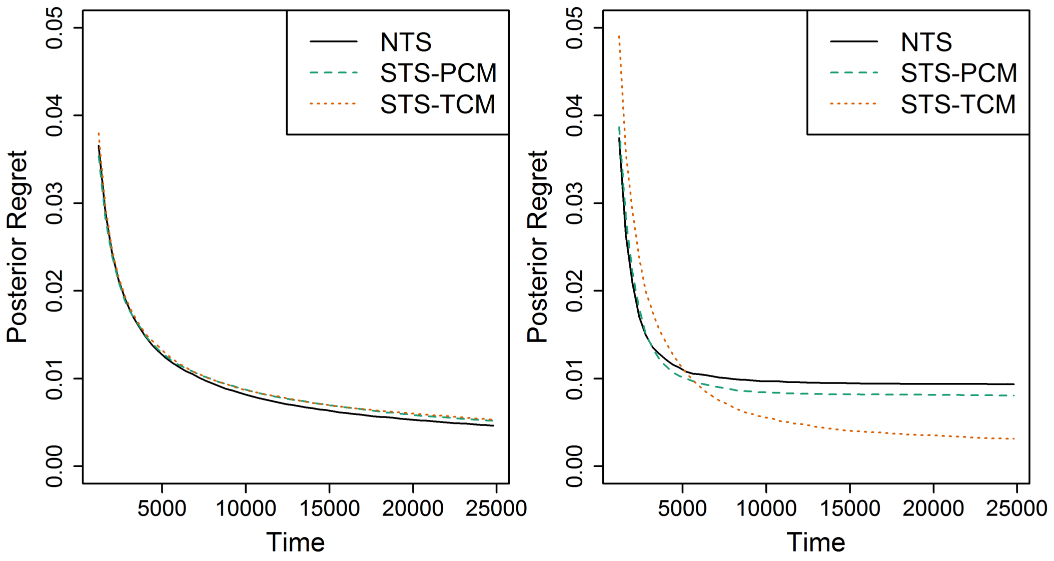

Figure 4.6.3: Mean greedy posterior regret for TS-based policies with PCM (left) and TCM (right). Simulation setting are $n = 10$, $k = 40$, $m = 3$, $\xi = 0.5$, $\alpha = 1$ and $\beta = 2$.

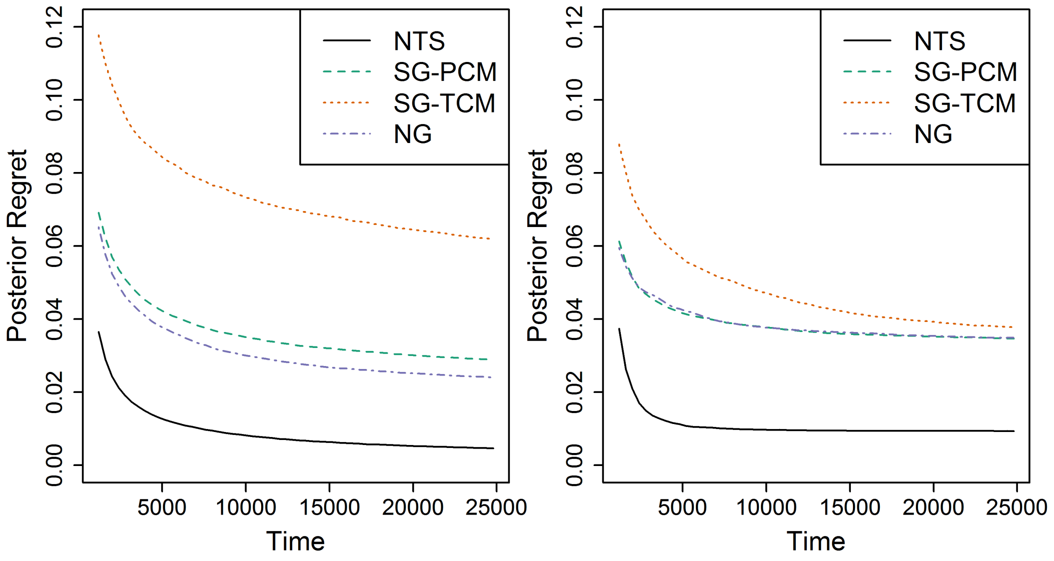

Figure 4.6.4: Mean greedy posterior regret for Greedy-based policies with PCM (left) and TCM (right). Simulation setting are $n = 10$, $k = 40$, $m = 3$, $\xi = 0.5$, $\alpha = 1$ and $\beta = 2$. The NTS policy is shown for comparison.

For PCM all of the TS policies are very similar. On TCM both STS policies do better than NTS with STS-TCM best of all. The learning for the Greedy policies is, as expected, clearly worse than for TS but in addition there are bigger differences between the different Greedy variants. In particular SG-TCM learns poorly on both PCM and TCM. This explains the regret results from Section 4.6.1 showing that the problem is with the click model assumed by the policy rather than the true click model. Therefore, if using a Greedy-based policy, it is more robust for learning to assume PCM. However this is not the case with TS policies. The results here and the regret simulations in Section 4.6.1 suggest that, in addition to superior exploitation of knowledge, STS also learns as effectively as NTS and is better for a correctly specified TCM.

4.7 Discussion

This section will highlight the main contributions of this chapter and discuss any issues with the methods and future work that could result. We considered the problem of selecting multiple website elements which has considerable application in a number of areas of web commerce and services. The main contributions can be divided into modelling and solution methods.

The primary intention in modelling the problem was to gain insight into the key features of the problem and how it differed from existing similar problems. By doing this we could better understand whether existing solution methods were appropriate and if not develop new ones. A major objective of the model was to capture the interactions between elements in a way that satisfied the intuitive notion that element sets should be diverse. This was done in a concise manner by using a latent user state together with a range of click models. By analysing solutions to the click models a direct link can be made to element set diversity without introducing any objective other than maximising user clicks. In particular, it was found that a click model based on user click thresholds resulted in a high level of diversity and which avoided the undesirable redundancy found in sets chosen with other existing models.

The main purpose of the model is conceptual and it may seem too simple to translate to direct application. However this depends on the richness of the topic preference vector $\mathbf{q}$. In practice this could be very large and contain considerable information. The models and solutions methods were chosen to be practical at such a large scale. A possible limitation of the the model is that a user state is given by a single x. An extension would be to record the user state as a very sparse vector of equal length to $\mathbf{w}$. The click model would then be a function of these two vectors mapping to a click probability. The function need not be complicated e.g. the probit of the dot product. A similar model for single advert CTR prediction which has been implemented in commercial web search is given in Graepel et al. (2010).

The second modelling challenge was how to handle the process of learning the characteristics of the elements, given here by a weight vector $\mathbf{w}$. Using a Bayesian multi-armed bandit framework gave access to considerable existing research but also presented new challenges. The exact Bayesian updating scheme was not practical but in Section 4.4.2 we developed an heuristic method which was shown to work effectively and which will have wider application. This enabled the use of an underlying Beta-Bernoulli model which further linked with existing bandit work and led to solution methods for effectively learning element weights.

The solution method was given in two parts corresponding to the exploitation and exploration parts of the traditional multi-armed bandit. First a set choosing algorithm drew on existing theory using the submodularity of the click models to handle the combinatorial explosion in the number of available arm sets. The theory was tested in this setting using a simulation study and found to perform well, acting similarly to the optimal method not only in performance but also by choosing sets with similar diversity. Secondly, to solve the exploration part we gave a method by which existing bandit algorithms can easily be adapted to this problem. Our analysis concentrated on adapting Thompson Sampling. A proof was given that the proposed method selected all elements infinitely often under stated conditions. It demonstrated good performance in simulations.

We did not compare our approach with existing methods because none were designed for our problem formulation and make inappropriate assumptions for our specific model. A formal bound on finite time regret would be desirable but the difficulty of our problem with a stochastic latent state in addition to stochastic arms makes this impractical. In particular the approximate Bayesian scheme does not give guarantees of accurate convergence of weight beliefs so regret growth cannot be usefully bounded.

It was noted in Section 4.6.1 that exploration of weights will be slow as the problem is scaled up. This is unavoidable unless greater assumptions are made on the structure of the weights (e.g. assuming dependence between weights or elements). There are examples of this in bandit problems (e.g. Yue and Guestrin 2011) but the intention in this work is to use a model that is as general as possible, only adding in such assumptions if it is clear they are valid and necessary. It may be that exploration is not a problem in practice due to the high rate of observations in likely applications and by using priors that best represent existing knowledge of likely element CTRs.

A realistic extension of the problem would be to have control over the number of elements chosen at each time rather than have a fixed number. If there was some cost to each element then it would be necessary to estimate the value in immediate and longer term CTR of adding the element. A more realistic model of element ranking or position than the cascade model would be desirable in evaluating the benefit of adding extra elements.

Another extension would be to treat user preferences as a quantity that can be learnt over time. The most basic version would be to treat user preferences $\mathbf{q}$ as constant over time but not known exactly. It may be that there are fundamental identifiability issues with this problem. This would have important consequences for recommender type systems which have the dual unknowns of both user needs and the characteristics of items available for recommendation.

Finally, the methods given were not tested on a real data set as these were not available in a suitable form for this problem. Data from a seemingly similar setting such as that used in Li et al. (2011) is not usable as a single element is presented to the user whereas we require a user response to a *set* of elements and our approach is predicated on the assertion that the CTR of set depends on interactions between elements rather than a simple function of their individual CTRs. Many of the modelling assertions made in this chapter would not be difficult to test for companies able to carry out appropriate web experiments and this would give insight into the nature of their particular applications. For example, assessing the diversity of "good" element sets would help suggest which click model is most appropriate. A comparison of PCM and TCM as models for user behaviour would be of value in many applications.

Chapter 5

Practical Calculation of Gittins Indices

5.1 Introduction

The Gittins index (GI) is known to provide a method for a Bayes optimal solution to the multi-armed bandit problem (MAB) (Gittins 1979). In addition, Gittins indices (GIs) and their generalisation Whittle indices have been shown to provide strongly performing policies in many related problems even when not optimal (see Chapter 3). The breakthrough that GIs provided was one of computational tractability since optimal solution methods previously available had only been practical for very restricted range of small MAB problems. But despite this, and the rarity of other tractable optimal or near optimal solutions in problems of this type, GIs are often not used for many problems for which they would be well suited, including the MAB. Part of the reason is the perception that GIs are hard to compute in practice: "...the lookahead approaches [GI] become intractable in all but the simplest setting..." (May et al. 2012) and, "Logical and computational difficulties have prevented the widespread adoption of Gittins indices" (Scott 2010). In a very general setting there may be fundamental difficulties in computation, hence the value of reliable heuristics such as those developed in Chapter 3, but for common forms of the MAB with standard reward distributions calculation of the GI is very much tractable. However there is clearly a need to make the practice of calculation better understood and easier to carry out. Powell (2007) observed "Unfortunately, at the time of this writing, there do not exist easy to use software utilities for computing standard Gittins indices". To our knowledge this is still the case. Gittins et al. (2011) provides tables of GI values for some commonly occurring problem settings but these are limited in scope.

The difficulty in calculating GIs for general Bayes sequential decision problems motivated the investigation of heuristics reported in Chapter 3. In that work the MAB was used as a test for heuristics in part due to the existence of a known optimal policy, namely that based on the GI. To make this comparison it was necessary to obtain GI values for some common versions of the MAB, namely when rewards had Bernoulli or normal distributions. It was found that the difficulty of GI calculation, in these cases at least, was overstated. The purpose of this chapter is to give details of the method used to calculate GIs for the MAB with Bernoulli and normal rewards (respectively BMAB and NMAB). In doing so we will give a more general discussion, which will be applicable beyond the BMAB and NMAB, of the issues faced and how these may be overcome.

Probably the largest contribution of this work is the accompanying code that has been developed in the R programming language. This code has been made publicly available at `https://bitbucket.org/jedwards24/gittins`. By doing this, and by providing the settings we used with the code, it is our intention that the process of obtaining GI values will become easily reproducible. The run times reported on computational experiments were obtained using a laptop with an Intel Core i7-2640 2.80 GHz processor with 8GB RAM.

5.1.1 Problem Definition

A detailed formulation of the MAB and the GI are given in earlier chapters but a brief outline relevant to this chapter will now be given. For the purposes of notation we will assume rewards are from the exponential family as described in Section 3.2 in Chapter 3 but the methodology given in this chapter is appropriate for more general reward distributions.

In the MAB, at each decision time $t = 0, 1, 2, \ldots$ an action $a_t \in \{1, \ldots, k\}$ is taken. Action $a_t = a$ corresponds to choosing arm a at time t. Associated with each arm a is an unknown parameter θ_a. Our belief in the value of θ_a is given by $g(\theta_a \mid \Sigma_a, n_a)$ where Σ_a and n_a are known hyperparameters. After selection an outcome y is observed, where y has density $f(y \mid \theta_a)$, and our belief is updated to $g(\theta_a \mid \Sigma_a + y, n_a + 1)$.

For the BMAB we have $f(y \mid \theta_a) \sim Bern(\theta_a)$ with $g(\theta_a \mid \Sigma, n) \sim Beta(\Sigma_a, n_a - \Sigma_a)$ and for the standard version of the NMAB, $f(y \mid \theta_a) \sim N(\theta_a, 1)$ and $g(\theta_a \mid \Sigma_a, n_a) \sim N\left(\frac{\Sigma_a}{n_a}, \frac{1}{n_a}\right)$. The sections specific to the NMAB will consider a more general form of the NMAB which has extra parameters $\tau_a > 0$. This are the observation precision of the arms which are known and can be different for each arm. The τ_a alter the variance of rewards which are now $f(y \mid \theta_a) \sim N(\theta_a, 1/\tau_a)$ for arm a. As τ_a increases then the observed reward becomes a more reliable estimate of θ_a. In any material where τ_a is not mentioned it can be assumed that $\tau_a = 1$ for all arms.

After the observation a reward $\gamma^t y$ is received where $0 < \gamma < 1$ is the discount factor and $t = 0, 1, \ldots$ is the current time. For an infinite time horizon the total Bayes' reward is maximised by choosing an arm satisfying

$$\nu^{GI}(\Sigma_a, n_a, \gamma) = \max_{1 \leq b \leq k} \nu^{GI}(\Sigma_b, n_b, \gamma), \tag{5.1.1}$$

where ν^{GI} is the GI which will be defined later in this section. For a constant discount factor GI values are independent of time. For the NMAB where $\tau_a \neq 1$ for some arm a the GI will be denoted $\nu^{GI(\tau)}(\Sigma_a, n_a, \gamma, \tau_a)$. Note the subscripts a will usually be dropped in this chapter since we are only interested in a single arm for a given index calculation.

5.1.2 Use of Gittins Indices in Practice

GIs for the Bayesian MAB can, in general, only be calculated numerically and so it is more accurate to think of a GI value as only being known to lie within some range. For any given policy decision only the ordering of the arm GIs matters, not the absolute value, as we only need to find an arm that satisfies $\arg\max_a \nu^{GI}(\Sigma_a, n_a, \gamma)$. Therefore, for an optimal policy we only need to calculate GIs to sufficient accuracy such that the intervals containing ν^{GI} for competing arms do not overlap. However we do not seek such exact optimality as doing so will come with a computational cost unjustified by diminishing returns in reward. Instead we will give a simple, computationally practical approach that gives GIs to a good and bounded accuracy for almost all states. The suboptimality of this approach is limited and quantifiable because decisions involving arms that have very similar GI values will be rare and, more importantly, the cost of a suboptimal action is small when the GI of the suboptimal arm is close to that of the optimal arm: "…an approximate largest-index rule yields an approximate optimal policy" (Katehakis and Veinott Jr 1987). Glazebrook (1982) gives a bound for the lost reward of a suboptimal policy in terms of GIs.

Also, the accuracy of the quantity $\nu^{GI}(\Sigma_a, n_a, \gamma) - \nu^{GI}(\Sigma_b, n_b, \gamma)$ for any two arms a, b will be better than accuracy guarantees on individual GIs since, by using similar calculation methods for each arm, the approximation errors will be correlated. This will be indicated for any approximations where relevant in later sections.

GIs can be calculated either online, as they are needed, or offline where they are stored then retrieved when needed. Online calculation has the advantage that GIs need only be calculated for states that are visited which is beneficial when the state space is large and especially if it is continuous. If calculated online it is not necessary to do the full calculation for each arm at every time because only the GI of the arm selected will change and this need only be recalculated to sufficient accuracy to determine whether its new GI is below the GI of the second best arm at the previous time. Also it may be that fast to calculate bounds such as those given in Section 5.2 are sufficient to determine the next action. For the BMAB recalculation can often be avoided by using the stay-on-the-winner rule (Berry and Fristedt 1985) which guarantees that the optimal arm from the previous time remains optimal for the next if a success is observed.

However, for many applications, online calculation may be too slow to be used, at least to reasonable accuracy, and in this thesis only offline calculation was used. The rest of this chapter will focus on offline calculation although the methods will often be applicable to online use. When using GIs in simulations, as in Chapter 3, offline calculation is appropriate as the same states are repeated many times in different simulation runs and offline use avoids repeated calculations. Generally, offline calculation is simpler and will be efficient whenever online calculation is practical, the only possible exception being with large state spaces. Methods to handle this issue are discussed in Section 5.4.

5.1.3 General Approach

Various methods exist for calculating GIs (for a review see Chakravorty and Mahajan 2014 or Gittins et al. 2011) but we will use *calibration*. Calibration uses a bandit process with a *retirement option* (Whittle 1980) which is sometimes referred to as a one-armed bandit. The single arm in question is the one for which we wish to find the GI which we will call the *risky arm*. At each time we have the choice to *continue* to play this arm or instead choose an arm of known and stationary reward λ (the *safe arm*). Since there is nothing we can learn about the safe arm it's value does not change and once it is optimal to choose then it will continue to be optimal indefinitely (this is retirement). The GI of the risky arm is the value of λ for which, at the start of the one-armed bandit, we are indifferent between choosing the safe or risky arms. Hence we must find the expected reward available from (or *value* of) the one-armed bandit for a given λ with either initial action.

This can be done using *stochastic dynamic programming* to solve the value function for the one-armed bandit given λ and the risky arm's state given by Σ, n and γ:

$$V(\Sigma, n, \gamma, \lambda) = \max\left\{\frac{\Sigma}{n} + \gamma\mathbb{E}_Y\big[V(\Sigma + Y, n+1, \gamma, \lambda)\big]; \frac{\lambda}{1-\gamma}\right\}. \tag{5.1.2}$$

Note that in any state where it is optimal to choose the safe arm at any time then it remains optimal for all remaining times hence the value of choosing the safe arm is $\frac{\lambda}{1-\gamma}$ and can be found without recursion. However we do not need the absolute value so $V(\Sigma, n, \gamma, \lambda)$ will instead denote the relative value between the safe and risky arms and (5.1.2) is replaced by

$$V(\Sigma, n, \gamma, \lambda) = \max\left\{\frac{\Sigma}{n} - \lambda + \gamma\mathbb{E}_Y\big[V(\Sigma + Y, n+1, \gamma, \lambda)\big]; 0\right\}. \tag{5.1.3}$$

The Gittins index can then be defined as

$$\nu^{GI}(\Sigma, n, \gamma) = \min\{\lambda : V(\Sigma, n, \gamma, \lambda) = 0\}. \tag{5.1.4}$$

To find λ satisfying (5.1.4) a numerical method must be used. Observe that $V(\Sigma, n, \gamma, \lambda)$ decreases as λ increases while Σ, n and γ are fixed so we can progressively narrow an interval containing λ by repeatedly finding $V\left(\Sigma, n, \gamma, \hat{\lambda}\right)$ for appropriate $\hat{\lambda}$. The general method for a single state is given in Algorithm 3.

Algorithm 3 Calibration for Gittins indices

```
Input: Parameters $\Sigma$, $n$, $\gamma$. Initial bounds for $\nu^{GI}(\Sigma,\ n,\ \gamma)$ given by $u$, $l$. A required
  accuracy $\epsilon$.
  while $u - l > \epsilon$ do
    $\hat{\lambda} \leftarrow l + (u - l)/2$
    Calculate $V(\Sigma, n, \gamma, \hat{\lambda})$ as given in (5.1.3)
    if $V(\Sigma, n, \gamma, \hat{\lambda}) > 0$ then
      $l \leftarrow \hat{\lambda}$
    else
      $u \leftarrow \hat{\lambda}$
    end if
  end while
Output: An interval $[l, u]$ which contains $\nu^{GI}(\Sigma,\ n,\ \gamma)$ where $u - l < \epsilon$ .
```

The algorithm initialises an interval in which $\nu^{GI}(\Sigma,\ n,\ \gamma)$ is known to lie (methods for doing this will be given in Section 5.2). The interval is then reduced in size, using bisection and repeated calculation of the value function in (5.1.3), until it is sufficiently small for our purposes. We can use the mid-point of this interval for $\nu^{GI}(\Sigma,\ n,\ \gamma)$ which will be within the desired accuracy ϵ of the true value. Details of calculating the value function for the BMAB and NMAB will be given in Section 5.3.

Other interval reduction methods could be used but bisection is sufficient for our purposes. The number of calculations of $V(\Sigma, n, \gamma, \hat{\lambda})$ required is

$$N_V = \left\lceil \frac{\log\left(\frac{\epsilon}{u-l}\right)}{\log(0.5)} \right\rceil,$$

where u and l are respectively the initial upper and lower bounds for $\nu^{GI}(\Sigma,\ n,\ \gamma)$ given to the algorithm. Using $\epsilon = 0.001$ is sufficient to give a policy close enough to optimal for most purposes but $\epsilon = 0.0001$ can usually be obtained with $N_V \approx 15$.

Section 5.2 gives new and existing bounds for GIs that can be used to initialise the interval for calibration. Section 5.3 gives details of the calculation of the value functions for BMAB and NMAB. This allows us to calculate the GI for a single state. Section 5.4 describes how to extend the single state methods in a practical way to obtain GIs for the full state space of the MAB. This is made possible by using invariant transforms to reduce the dimension of the state space and by using monotonicity properties of GIs to handle continuous state spaces. A comparison will be made of two general approaches to finding GIs for a block of states. Section 5.5 will discuss some outstanding issues and the calculation of indices for the finite horizon MAB.

5.2 Bounds for Gittins Indices

In this section we give new and existing bounds for GIs. Bounds are needed to initialise an interval to calculate GIs and tighter bounds will reduce computation time. The bounds given will also be of general interest and use in the study and application of GIs.

The following theorem from Brezzi and Lai (2000) gives bounds for the MAB with general rewards.

Theorem 5.2.1. *For the MAB with general rewards under the condition that* $\int [\mathbb{E}_\theta(Y \mid \theta)]^2 \,\mathrm{d}g(\theta) < \infty$*,*

$$\mu \leq \nu^{GI} \leq \mu + \sigma \frac{\gamma}{1-\gamma}, \tag{5.2.1}$$

where μ *is the mean reward given the belief* $g(\theta)$ *and* σ *is the standard deviation of* $g(\theta)$*.*

For exponential family rewards $\mu = \Sigma/n$ and $\sigma = 1/\sqrt{n}$. For the BMAB a better bound is often given by $\nu^{GI} \leq 1$ since rewards are bounded. The upper bound in (5.2.1) can be very loose for larger γ so we now give new, tighter bounds that in general require numerical calculation but which are still fast to obtain for common reward distributions such those for the BMAB and NMAB.

5.2.1 Upper Bounds

Upper bounds are based on a variant of the MAB with *perfect information* where θ is assumed to be revealed after a single observation. Under this assumption the value function for the one-armed bandit is

$$V^{PI}\left(\Sigma, n, \gamma, \lambda\right) = \max\left\{\frac{\Sigma}{n} - \lambda + \frac{\gamma}{1-\gamma}\mathbb{E}_{\theta\sim g(\theta|\Sigma,n)}\big[\max(\theta-\lambda,0)\big]; 0\right\}, \tag{5.2.2}$$

and substituting this for $V(\cdot)$ in (5.1.4) gives the GI for the perfect information MAB.

Definition 5.2.2. *The index GI+ is the GI of the MAB where the unknown parameter θ becomes known after a single observation. Let Θ be the support of $g(\theta \mid \Sigma, n)$ and $V^{PI}(\Sigma, n, \gamma, \lambda)$ be the value function under perfect information as given in (5.2.2), then GI+ is given by*

$$\nu^{GI^+}(\Sigma, n, \gamma) = \min\{\lambda : V^{PI}(\Sigma, n, \gamma, \lambda) = 0\} \tag{5.2.3}$$

$$= \min\left\{\lambda : \frac{\lambda}{1-\gamma} = \frac{\Sigma}{n} + \frac{\gamma}{1-\gamma}\left[\int_{\min\Theta}^{\lambda} g(\theta \mid \Sigma, n)\lambda \,\mathrm{d}\theta + \int_{\lambda}^{\max\Theta} g(\theta \mid \Sigma, n)\theta \,\mathrm{d}\theta\right]\right\}. \tag{5.2.4}$$

The left and right hand sides of the equation in the curly braces in (5.2.4) are the expected reward of playing, respectively, the safe and risky arms over the remaining time horizon. Rewards after the next outcome partition depending on the value of θ with the risky arm being continued if $\theta > \lambda$ and retired otherwise. Since θ is now known the arm played will continue for the remaining time. Therefore to solve (5.2.3) only the first outcome need be considered which means that a full dynamic programming scheme is not needed. Note that for the NMAB GI+ is equivalent to the limit of $\nu^{GI(\tau)}(\Sigma, n, \gamma, \tau)$ as $\tau \to \infty$. The next proposition states that GI+ is an upper bound for GI.

Proposition 5.2.3. *$\nu^{GI}(\Sigma, n, \gamma) \leq \nu^{GI^+}(\Sigma, n, \gamma)$ for all Σ, n, γ.*

Proof. $V^{PI}(\cdot)$ maximises the value in a relaxed version of the one-armed bandit. Therefore $V(\Sigma, n, \gamma, \lambda) \leq V^{PI}(\Sigma, n, \gamma, \lambda)$ for any Σ, n, γ, λ and the result follows from the definitions of GI and GI+ in (5.1.4) and (5.2.3) respectively. □

Let $H = \gamma/(1-\gamma)$ and $\mu = \Sigma/n$ then substituting $G(\cdot)$, the CDF of $g(\theta \mid \Sigma, n)$, for the right hand integral in (5.2.4) and rearranging gives an alternative expression for GI+:

$$\nu^{GI^+} = \min \left\{ \lambda : \lambda = \frac{\mu + H \int_{\lambda}^{\max \Theta} g(\theta \mid \Sigma, n) \theta \, \mathrm{d}\theta}{1 + H[1 - G(\lambda)]} \right\}. \tag{5.2.5}$$

This leads to a new bound which is looser but which can be calculated directly without calibration.

Definition 5.2.4. *The index UBGI+ is*

$$\nu^{UBGI^+}(\Sigma, n, \gamma) = \frac{\mu + H \int_{\mu}^{\max \Theta} g(\theta \mid \Sigma, n) \theta \, \mathrm{d}\theta}{1 + H[1 - G(\mu + H\sigma)]}. \tag{5.2.6}$$

where $H = \gamma/(1-\gamma)$ and σ is the standard deviation of the belief $g(\theta \mid \Sigma, n)$.

Lemma 5.2.5. *$\nu^{GI^+}(\Sigma, n, \gamma) \leq \nu^{UBGI^+}(\Sigma, n, \gamma)$ for all Σ, n, γ.*

Proof. As $\nu^{GI^+}(\Sigma, n, \gamma)$ is a GI for a MAB it follows from Theorem 5.2.1 that $\mu \leq \nu^{GI^+}(\Sigma, n, \gamma) \leq \mu + H\sigma$. The result then follows directly from Definition 5.2.4 since ν^{UBGI^+} is found by substituting μ and $\mu + H\sigma$ appropriately for λ in (5.2.5). □

UBGI+ can be found directly without the use of interval reduction. Numerical methods are still required for the integral but these are fast for common distributions. A comparison of upper bounds is given for the NMAB in Figure 5.2.1 where UBBL is the upper bound from Theorem 5.2.1. It can be seen that GI+ is much tighter than the other bounds. A more appropriately scaled plot of GI+ will be given later in this section together with lower bounds.

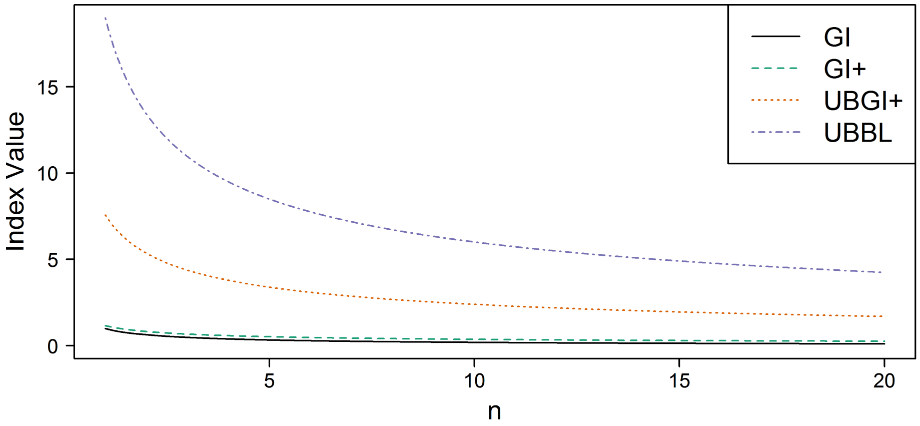

Figure 5.2.1: Comparison of Upper Bounds for GI for the NMAB with $\Sigma = \mu = 0$, $\gamma = 0.95$, $\tau = 1$ and $n = 1, \ldots, 20$.

5.2.2 Lower Bounds

The lower bound $\mu = \Sigma/n$ from Theorem 5.2.1 can be improved upon by using the Knowledge Gradient index (KGI) introduced in Section 3.4 in Chapter 3 the relevant details of which will be repeated here.

A value function $V^{KG}(\Sigma, n, \gamma, \lambda)$ which approximates $V(\Sigma, n, \gamma, \lambda)$ is found by imposing the constraint that whatever decision is made at the second stage of the dynamic program is final and so will apply for the remainder of the horizon:

$$\begin{aligned} &V^{KG}(\Sigma, n, \gamma, \lambda) \\ &= \max\left\{\frac{\Sigma}{n} - \lambda + \frac{\gamma}{1-\gamma}\mathbb{E}_Y\left[\max\left(\max\left(\frac{\Sigma+Y}{n+1}, \lambda\right) - \lambda; 0\right) \mid \Sigma, n\right]; 0\right\}. \end{aligned} \tag{5.2.7}$$

Note $V^{KG}(\Sigma, n, \gamma, \lambda)$ is decreasing in λ for any fixed Σ, n and γ. The KGI is then given by

$$\nu^{KGI}\left(\Sigma, n, \gamma\right) = \min\left\{\lambda : V^{KG}\left(\Sigma, n, \gamma, \lambda\right) = 0\right\}. \tag{5.2.8}$$

Let $p(y \mid \Sigma, n)$ and $P(y \mid \Sigma, n)$ be respectively the density and CDF of the predictive distribution of y found by integrating over θ. These will be abbreviated to $p(y)$ and

$P(y)$ in the following for brevity as conditioning is always on Σ and n. Unlike the GI^+ we do not observe the true θ and so the continuation threshold after the first stage depends on y. Let $\mu^+ = (\Sigma + y)/(n+1)$ be the mean of the posterior predictive given y, then the continuation threshold above which the risky arm is continued is $y^* = \lambda(n + 1 - \Sigma)$. An alternative statement of KGI is then

$$\nu^{KGI}(\Sigma, n, \gamma) = \min\left\{\lambda : \mu - \lambda + H\left[\int_{\min Y}^{y^*} p(y)\lambda \,\mathrm{d}y + \int_{y^*}^{\max Y} p(y)\mu^+ \,\mathrm{d}y - \lambda\right] = 0\right\} \tag{5.2.9}$$

$$\iff \min\left\{\lambda : \lambda = \frac{\mu + H\int_{y^*}^{\max Y} p(y)\mu^+ \,\mathrm{d}y}{1 + H(1 - P(y^*))}\right\} \tag{5.2.10}$$

where $H = \gamma/(1-\gamma)$. For the NMAB the corresponding index $\nu^{KGI(\tau)}$ is a function of τ where $y^* = (\lambda(n+\tau) - \Sigma)/\tau$ and $\mu^+ = (\Sigma + y\tau)/(n+\tau)$. The right hand side of the part of (5.2.10) in curly braces therefore depends on λ through y^* and interval reduction must be used.

Binary outcomes mean that the BMAB case for KGI can solved analytically,

$$\nu^{KGI}(\Sigma, n, \gamma) = \frac{\Sigma}{n + H\Sigma} + H\frac{\Sigma(\Sigma+1)}{(n+1)(n+H\Sigma)} = \frac{\mu + H\mu\left(\frac{\Sigma+1}{n+1}\right)}{1 + H\mu}, \tag{5.2.11}$$

where again $H = \gamma/(1-\gamma)$.

Proposition 5.2.6. *KGI satisfies* $\mu \leq \nu^{KGI}(\Sigma, n, \gamma, \lambda) \leq \nu^{GI}(\Sigma, n, \gamma, \lambda)$.

Proof. The left hand inequality follows directly from (5.2.7), which equals zero if and only if $\lambda \geq \Sigma/n$, and the definition of KGI in (5.2.8). The right hand inequality follows from the definition of KGI since $V^{KG}(\Sigma, n, \gamma)$ is an approximation of $V(\Sigma, n, \gamma)$ which finds a maximal solution to the dynamic program. □

5.2.3 Summary of Bounds

In summary we have $\mu \leq \nu^{KGI} \leq \nu^{GI} \leq \nu^{GI^+} \leq \nu^{UBGI^+}$ and $\nu^{GI^+} \leq \nu^{UBBL}$ where ν^{UBBL} is the upper bound from Theorem 5.2.1. The bounds μ and ν^{UBBL} have closed form and take negligible time to calculate. The calculation times for ν^{UBGI^+}, ν^{KGI} and ν^{GI^+} for a vector of 100 states are given in Table 5.2.1.

Bound	Time (s)
UBGI+	0.0012
GI+	0.0090
KGI	0.0025

Table 5.2.1: Computation times for bounds for 100 states in the NMAB ($\gamma = 1$, $\tau = 1$, $n = 1, \dots, 100$). An accuracy $\epsilon = 0.00005$ was used for the values of GI+ and KGI.

By using bisection for interval reduction, small tightening of initial bounds do not make a large saving in computation time but calculation of ν^{KGI} and ν^{GI^+} is far faster than finding $V(\Sigma, n, \gamma, \lambda)$ once so it usually worth using these bounds. Section 5.4 discusses how GI values from nearby states can provide a tighter initial range if they are available.

Figure 5.2.2 shows the two tightest bounds KGI and GI+ for the NMAB which can be seen to be a big improvement on UBBL (shown in Figure 5.2.1) and the lower bound $\mu = 0$. For smaller γ the GI values become closer to the KGI bound, moving closer to GI+ as γ increases. All the bounds shown become tighter as τ increases as the one step lookahead becomes a more appropriate approximation.

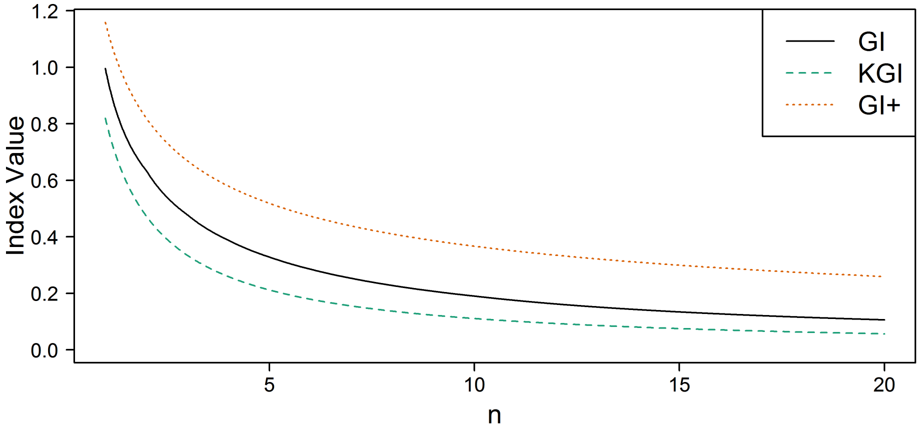

Figure 5.2.2: Comparison of bounds for GI for the NMAB with $\Sigma = \mu = 0$, $\gamma = 0.95$, $\tau = 1$ and $n = 1, \dots, 20$.

In Figure 5.2.3 KGI and GI+ are shown together with $\mu = \Sigma/n$ for the BMAB. The best alternative upper bound is 1.

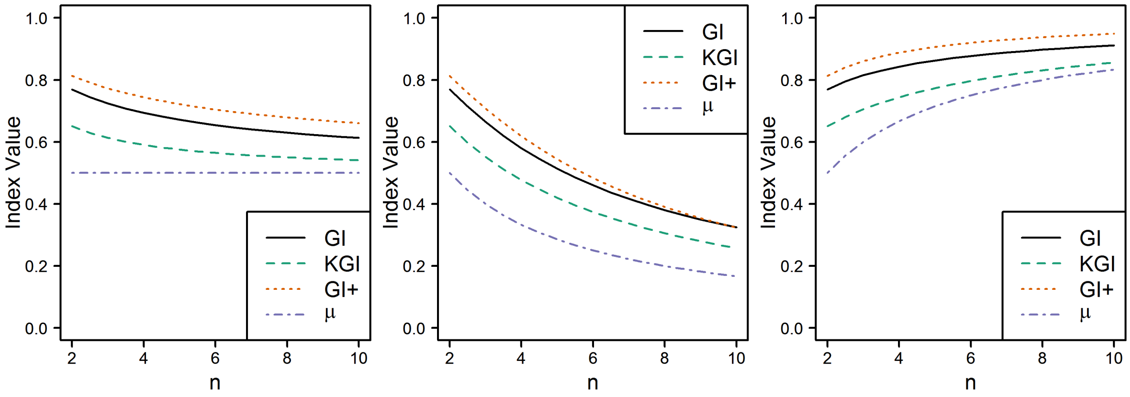

Figure 5.2.3: Comparison of bounds for GI for the BMAB with $\gamma = 0.95$, $n = 2, \dots, 10$ where $\mu = \Sigma/n$. The left plot shows $\Sigma = n/2$, the centre plot shows $\Sigma = 1$ and the right plot shows $\Sigma = n - 1$.

A side note of the work in this section is that $\nu^{GI^+} - \nu^{KGI}$ gives a bound on the approximation error for KGI. For the BMAB $\nu^{UBGI^+} - \nu^{KGI}$ gives an analytical bound on this error and which means we can easily bound the suboptimality of the KGI policy by using the result from Glazebrook (1982).

5.3 Value Function Calculation

This section will will give details of how the value function (5.1.3) can be calculated for the BMAB and NMAB. The emphasis for method choice and implementation, both here and in the general approach given in Section 5.1.3, is on practicality and general applicability. One obstacle to adoption of GIs as a practical solution method has been perceived complexity and it is not our aim to optimise for speed but to present a method that is easy to implement, with good demonstrable accuracy and which is practical for most situations. Much of methodology here is similar to that given in Chapter 8 of Gittins et al. (2011) but has been adapted to be more accessible and to give a more detailed and general implementation. Experimental accuracy and speeds for a range of settings for the calculation will also be given.

Each $V(\Sigma, n, \gamma, \hat{\lambda})$ is found with stochastic dynamic programming using the recursive equation (5.1.3). It is useful to distinguish between the states and times in the dynamic program used to calculate the value function, and states and times in the original MAB. We will use $(\tilde{n}, \tilde{\Sigma})$ to denote the states in the dynamic program and refer to the *stage* of the dynamic program rather than the time. Therefore the initial state at stage 0 is (n, Σ). The backward recursion process of dynamic programming cannot proceed for an infinite number of stages so a finite horizon approximation is used. An N is chosen and the values of states $\{(\Sigma_N, n_N, \gamma, \lambda) : n_N = n + N\}$ at stage N are calculated directly using some approximation. An obvious approximation is $\hat{V}(\Sigma_N, n_N, \gamma, \lambda) = \gamma^N \max(\Sigma_N/n_N - \lambda, 0)$ but this can be improved by using a similar idea to KGI so that

$$\hat{V}(\Sigma_N, n_N, \gamma, \lambda) = \frac{\gamma^N}{1-\gamma} \max(\Sigma_N/n_N - \lambda, 0). \tag{5.3.1}$$

States at stages up to stage N are then found recursively using (5.1.3). Discounting ensures that this index with a large N gives a very good approximation which can be made arbitrarily close to ν^{GI} by increasing N.

The size of the dynamic program and hence the computation required depends on N so we aim to use as small a value as possible that still gives the required accuracy. The remaining reward at stage N gives a bound on the approximation but is would lead to a too conservative value of N. An estimate of an appropriate value of N for a required accuracy can be found from testing convergence for a single state as shown in Table 5.3.1. The calculation time is also shown. A value of $N = 2000$ was used to find the ν^{GI} used in the comparison and this took 15 seconds to compute.

N	$\gamma = 0.9$		$\gamma = 0.99$	
	Error	RRN	Error	RRN
20	0.00827	1.21577	0.03738	81.79069
40	0.00086	0.14781	0.03531	66.89718
60	0.00010	0.01797	0.02825	54.71566
80	0.00001	0.00218	0.02227	44.75232
100	0	0.00027	0.01755	36.60323
200	0	0.00000	0.00557	13.39797
300	0	0.00000	0.00188	4.90409
400	0	0.00000	0.00066	1.79506
500	0	0.00000	0.00024	0.65705
600	0	0.00000	0.00008	0.24050
700	0	0.00000	0.00003	0.08803
800	0	0.00000	0.00001	0.03222
900	0	0.00000	0	0.01179
1000	0	0.00000	0	0.00432

Table 5.3.1: The error in GI from using a finite horizon approximation for the BMAB with $\Sigma = 1$, $n = 2$, accuracy $\epsilon = 0.000005$ and N, γ as given. RRN is the maximum reward remaining after stage N and is shown for comparison.

Convergence with the NMAB occurs at even smaller N as will be seen in a similar experiment in Section 5.3.2. The reason why low values of N give good accuracy is as follows. Let A be the risky arm and B be the safe arm. If B is the optimal action at stage N of the stopping problem then the value approximation in (5.3.1) is accurate. If A is optimal at stage N then it will have been chosen at each preceding stage.

Therefore its posterior distribution will be tight and Σ_N/n_N is a good estimate of its true expected reward. Hence it would likely have been continued if N had been much larger. This combined with discounting means there is little approximation error. It should also be noted that the errors will all be of the same sign and so will be largely cancelled out in the resulting index policy as long as the same N is used for all states.

5.3.1 Value Function Calculation - BMAB

The application of dynamic programming in the BMAB case for a single state is largely straightforward since the outcomes are binary and so the state space for the dynamic program is discrete. The value function (5.1.3) then becomes

$$\begin{aligned} &V(\Sigma, n, \gamma, \lambda) \\ &\qquad = \max\left\{-\lambda + \gamma\frac{\Sigma}{n}\big[1 + V(\Sigma+1, n+1, \gamma, \lambda)\big] + \gamma\left(1 - \frac{\Sigma}{n}\right)V(\Sigma, n+1, \gamma, \lambda); 0\right\}, \end{aligned}$$

with terminal states as given in (5.3.1). Apart from the use of a finite N, the calculation of $\nu^{GI}(\Sigma, n, \gamma)$ for the BMAB can therefore be done without approximation. The dynamic program state space is $\{(\tilde{\Sigma}, \tilde{n}) : \tilde{\Sigma} = \Sigma, \Sigma+1, \ldots, \Sigma+N, \tilde{n} = n, n+1, \ldots, n+N; \tilde{\Sigma} \leq \tilde{n}\}$, a total of $\frac{1}{2}(N+2)(N+1)$ states.

5.3.2 Value Function Calculation - NMAB

For the NMAB rewards are continuous in $\mathbb{R}$ rather than discrete and so solving the dynamic program to find $V(\Sigma, n, \gamma, \lambda)$ is not straightforward. We use a discretisation of the state space for the one-armed bandit which introduces extra levels of approximation compared to the BMAB.

The process evolves in two dimensions $\tilde{\Sigma}$ and $\tilde{n}$ but it is simpler to describe the following method using a reparameterisation from $(\tilde{\Sigma}, \tilde{n})$ to $(\tilde{\mu}, \tilde{n})$ where $\tilde{\mu} = \tilde{\Sigma}/\tilde{n}$ since we can then fix a range of $\tilde{\mu}$ into which the process is likely to evolve which is invariant to n. We need states to be countable so the $\tilde{\mu}$ dimension is discretised and bounded using two new parameters. The first ξ describes the extent of the state space and is the number of standard deviations $\sigma = \sqrt{1/n}$ of $g(\theta \mid \Sigma, n)$ from μ included in the $\tilde{\mu}$ dimension. The second δ controls the fineness of the discretisation. In addition we will restrict $\tilde{\mu} \geq \mu$ which will be justified shortly. Therefore Ω, the range for $\tilde{\mu}$ is

$$\Omega = \{\tilde{\mu} : \tilde{\mu} = \mu, \mu + \delta, \mu + 2\delta, \ldots, \mu + \left\lceil \frac{\xi\sigma}{\delta} \right\rceil \delta\}.$$

The $\tilde{n}$ dimension is discrete so the state space for the dynamic program is $\{(\tilde{\mu}, \tilde{n}) : \tilde{\mu} \in \Omega; \tilde{n} = n, n + \tau, n + 2\tau, \ldots, n + N\tau\}$. The total number of states is therefore $(N + 1)(\lceil \xi\sigma/\delta \rceil + 1) \approx N\xi\sigma/\delta$. Hence the state space size is decreasing in δ and n, and increasing in ξ and N.

The immediate reward of each state is given by $\tilde{\mu}$ as usual and the values of the states at stage N are found using (5.3.1) as with the BMAB. Denote $\pi(S, \mu^+, \tau)$ to be the transition probability from any state $S = (\mu, n)$ to another state $(\mu^+, n + \tau)$, $\mu^+ \in \Omega$. Then $\pi(S, \mu^+, \tau)$ is given by the probability that the posterior mean is in the interval $[\mu^+ - \delta/2, \mu^+ + \delta/2)$. Thus, for any state $S = (\tilde{\mu}, \tilde{n})$ we have

$$\pi(S, \tilde{\mu}^+, \tau) = P(y_u|\tilde{\mu}, \tilde{n}) - P(y_l|\tilde{\mu}, \tilde{n})$$

where $P(y \mid \tilde{\mu}, \tilde{n})$ is the predictive CDF and

$$y_l = \frac{(\tilde{\mu}^+ - \delta/2)(\tilde{n} + \tau) - n\tilde{\mu}}{\tau} \quad \text{and} \quad y_u = \frac{(\tilde{\mu}^+ + \delta/2)(\tilde{n} + \tau) - n\tilde{\mu}}{\tau}.$$

As $P(\cdot)$ is Gaussian the transition probabilities are quick to calculate. Then our value function approximation is

$$\begin{aligned} &V^*(\mu,n, \gamma, \lambda) \\ &\quad = \max \left\{ \mu - \lambda + \gamma \left[V^+ + V^- + \sum_{\mu^+ \in \Omega} \Big[\pi[(\mu, n), \mu^+, \tau] V(\mu^+, n + \tau, \gamma, \lambda) \Big] \right] ; 0 \right\} \end{aligned}$$

where V^+ and V^- are approximations to the expected value when μ updates to outside the range of Ω. To find V^+ and V^- let

$$y_U = \frac{(\max\Omega + \frac{\delta}{2})(\tilde{n}+\tau) - n\tilde{\mu}}{\tau} \quad \text{and} \quad y_L = \frac{(\min\Omega - \frac{\delta}{2})(\tilde{n}+\tau) - n\tilde{\mu}}{\tau}.$$

then

$$V^+ = (1 - P(y_U \mid \tilde{\mu}, \tilde{n}))V(\max\Omega, n+\tau, \gamma, \lambda)$$

and

$$V^- = P(y_L \mid \tilde{\mu}, \tilde{n})V(\min\Omega, n+\tau, \gamma, \lambda).$$

For ξ sufficiently high the approximation in V^+ will be small as the probability of θ being above $\max\Omega$ will be negligible. The approximation V^- does not result in any error for $\min\Omega = \mu - \delta/2$ by the following reasoning. For $\lambda = \nu^{GI(\tau)}(\mu, n, \gamma)$ we are indifferent between retirement and continuation in the starting state so $V(\mu, n, \gamma, \lambda) = 0$. Since $V(\mu, n, \gamma, \lambda)$ is non-increasing in n and non-decreasing in μ we have $V(\tilde{\mu}, \tilde{n}, \gamma, \lambda) = 0$ for any $\tilde{n} > n$, $\tilde{\mu} \leq \mu$ and so we will select the safe arm for the whole of this range and so the exact value of $\tilde{\mu} \leq \mu$ is unimportant. Since the use of V^- does not affect the monotonicity of $V^*(\cdot)$ with λ, setting $\min\Omega = \mu$ and using V^- will not change the ν^{GI} calculated.

To investigate the effect of the approximations N, δ and ξ on accuracy of $\nu^{GI(\tau)}$ a series of convergence tests were run. For each $\gamma \in \{0.9, 0.99\}$ and each $\tau \in \{0.1, 1, 10\}$ a benchmark value of $\nu^{GI(\tau)}(0, 1, \gamma, \tau)$ was calculated using $\xi = 8$, $\delta = 0.005$ and accuracy $\epsilon = 0.00005$, with $N = 80$ for $\gamma = 0.9$ and $N = 200$ for $\gamma = 0.99$. Then the values of N, δ and ξ were relaxed in turn. The difference $\nu^{GI} - \hat{\nu}^{GI}$ between the the benchmark GI value and the approximation are given in Tables 5.3.2 to 5.3.7 together with the calculation times.

N	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
10	0.00523	0.00123	0.00016	55	50	37
20	0.00054	0.00010	0.00004	109	94	69
30	0.00008	0	0	159	134	100
40	0	0	0	210	173	129
50	0	0	0	256	209	158
60	0	0	0	304	249	188

Table 5.3.2: Approximation error and run time of GI calculations with varying N and τ as shown and fixed $\gamma = 0.9$, $n = 1$, $\epsilon = 0.00005$, $\delta = 0.005$, $\xi = 8$.

δ	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
0.1	-0.00304	-0.00155	-0.00144	1.6	1.3	1.0
0.09	-0.00246	-0.00129	-0.00111	1.9	1.5	1.2
0.08	-0.00196	-0.00113	-0.00086	2.3	1.9	1.5
0.07	-0.00150	-0.00071	-0.00078	2.9	2.4	1.8
0.06	-0.00104	-0.00068	-0.00049	3.8	3.0	2.4
0.05	-0.00077	-0.00045	-0.00037	5.3	4.2	3.3
0.04	-0.00042	-0.00019	-0.00029	7.8	6.3	4.9
0.03	-0.00015	-0.00016	-0.00012	13	10	8.1
0.02	-0.00012	-0.00003	-0.00004	28	22	17
0.01	-0.00004	0	0	106	83	60

Table 5.3.3: Approximation error and run time of GI calculations with varying δ and τ as shown and fixed $\gamma = 0.9$, $n = 1$, $\epsilon = 0.00005$, $N = 80$ and $\xi = 8$.

ξ	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
3	0.00004	0.00081	0.00152	88	70	47
4	0	0	0.00004	148	115	81
5	0	0	0	223	180	122
6	0	0	0	311	255	171

Table 5.3.4: Approximation error and run time of GI calculations with varying ξ and τ as shown and fixed $\gamma = 0.9$, $n = 1$, $\epsilon = 0.00005$, $N = 80$ and $\delta = 0.005$.

N	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
20	0.04592	0.00839	0.00113	151	129	95
40	0.01328	0.00218	0.00031	286	238	176
60	0.00535	0.00086	0.00012	411	338	255
80	0.00249	0.00040	0.00004	549	456	333
100	0.00128	0.00022	0	678	559	394
120	0.00066	0.00012	0	800	663	468
140	0.00037	0.00006	0	922	766	544
160	0.00018	0.00003	0	1045	871	620
180	0.00007	0.00003	0	1143	973	696

Table 5.3.5: Approximation error and run time of GI calculations with varying N and τ as shown and fixed $\gamma = 0.99$, $n = 1$, $\epsilon = 0.00005$, $\delta = 0.005$, $\xi = 8$.

δ	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
0.1	-0.0153	-0.0074	-0.0021	4.8	4.4	3.1
0.09	-0.0128	-0.0065	-0.0018	5.5	5.1	3.6
0.08	-0.0102	-0.0059	-0.0023	6.8	6.2	4.5
0.07	-0.0080	-0.0049	-0.0020	8.4	7.7	5.5
0.06	-0.0066	-0.0039	-0.0017	11.0	10.0	7.3
0.05	-0.0048	-0.0031	-0.0013	15.1	13.8	10.0
0.04	-0.0030	-0.0023	-0.0011	23.3	20.4	14.8
0.03	-0.0018	-0.0015	-0.0007	39.1	32.9	24.6
0.02	-0.0007	-0.0007	-0.0005	81.5	69.1	52.3
0.01	-0.0001	-0.0002	-0.0002	310.7	266.3	196.6

Table 5.3.6: Approximation error and run time of GI calculations with varying δ and τ as shown and fixed $\gamma = 0.99$, $n = 1$, $\epsilon = 0.00005$, $N = 200$ and $\xi = 8$.

ξ	Approximation Error			Run Time (s)		
	$\tau = 0.1$	$\tau = 1$	$\tau = 10$	$\tau = 0.1$	$\tau = 1$	$\tau = 10$
3	0.00312	0.00966	0.01022	199	162	117
4	0.00007	0.00018	0.00020	335	276	201
5	0	0	0	504	420	310
6	0	0	0	701	593	441
7	0	0	0	926	795	624

Table 5.3.7: Approximation error and run time of GI calculations with varying ξ and τ as shown and fixed $\gamma = 0.99$, $n = 1$, $\epsilon = 0.00005$, $N = 200$ and $\delta = 0.005$.

The approximation due to N is smaller than for the BMAB with a value as small as $N = 30$ being sufficient for $\gamma = 0.9$. The N required for a given accuracy reduces with τ. Using $\xi = 5$ is sufficient in all cases. Speed generally decreases with τ which is due to narrower initial intervals. The choice of δ can have a large effect on the run time of the algorithm. It was found that $\delta = 0.01$ gives good accuracy but the guarantees are less clear than with N and ξ. The approximations from too small an N or ξ cause an underestimation in ν^{GI} while too high a δ causes an overestimation.

Improvements in the method are most likely to come from selecting δ appropriately and from improving the approximation δ causes. It is likely a more efficient discretisation will be found by varying δ across the $\tilde{\mu}$ dimension so that there is a higher concentration of states nearer μ. Another possible improvement is that lower values of δ needed to obtain good accuracy might not be needed at every stage of the calibration and it may be more efficient to determine the sign of $V(\Sigma, n, \gamma, \lambda)$ for some values of λ using a fast approximation only using lower δ if needed.

5.4 Computation Details for Multiple States

When GIs are calculated offline we theoretically need to find in advance the GI for each state that an arm in the MAB may visit. However this may not be possible as the state space can be infinite. This section will discuss how this issue can be addressed in practice and will give the states for which GIs are needed to create a GI-based policy for the BMAB and NMAB. It will also consider several methods of improving efficiency when going from single to multiple state GI calculations.

The MAB has an infinite time horizon but due to discounting rewards become very small for large times. Therefore GIs are only needed for states which can be reached in some suitable finite time T. If a large value of T is used then the states reached after T observations on a single arm will generally have tight belief distribution and so $\nu^{GI}(\Sigma, n, \gamma)$ will be close to Σ/n and will be easy to approximate, either by using the bounds given in Section 5.2 or by using an analytical approximation method such given by Brezzi and Lai (2002) which will be very good for high n. However, unless γ is close to one it will usually be feasible to obtain good GI calculations directly with better guarantees than other approximations.

If rewards are continuous, as in the NMAB then the MAB state space will also be continuous in the Σ dimension. Even when rewards are discrete we may want to find GIs for a continuous state space if priors are allowed to be parameterised over a continuous range.

The issue of continuous state spaces can be solved by using the following monotonicity properties of $\nu^{GI}(\cdot)$ to bound values for any states for which we don't have GIs by using the GIs of similar states. Theorem 3.2.1 in Chapter 3 states that $\nu^{GI}(c\Sigma, cn, \gamma)$ is decreasing in $c \in \mathbb{R}^+$ for any fixed Σ, n, γ and is increasing in Σ for any fixed c, n, γ. With this result $\nu^{GI}(\cdot)$ for a discrete grid of Σ and n can be used to bound and interpolate $\nu^{GI}(\cdot)$ for any interior state for a given γ. Yao (2006) gives monotonicity results specific to the NMAB: $\nu^{GI(\tau)}(\Sigma, n, \gamma, \tau)$ is non-decreasing in Σ and τ and non-increasing in n. From Kelly (1981), $\nu^{GI}(\Sigma, n, \gamma)$ is non-decreasing in γ which allows for a similar process of interpolation to obtain GIs for a continuous range of γ.

Any interpolation can be made arbitrarily accurate by using a finer grid of values. This may be impractical if the state space is not of low dimension. This is not a problem for the BMAB but would be for the NMAB if it were not for invariance properties that reduce the effective dimension. The details of the size of the state space for which we need GIs will be discussed in Sections 5.4.1 and 5.4.2 for the BMAB and NMAB respectively.

5.4.1 Multiple State Computation - BMAB

For the BMAB there are two dimensions, Σ and n, for each γ. Outcomes y are in $\{0, 1\}$ so for priors Σ_0, n_0 we need GIs for states

$$\{(\Sigma, n) : \Sigma = \Sigma_0, \Sigma_0 + 1, \ldots, \Sigma_0 + T, n = n_0, \ldots, n_0 + T, \Sigma \leq n\}. \tag{5.4.1}$$

If a set of GI values is needed for any possible prior (which hypothetically could take any positive values) then this could be done by finding GIs for the set of states in (5.4.1) for a reasonable grid of $\{(\Sigma_0, n_0) : \Sigma_0 \in (0, 1), n_0 \in (0, 2], \Sigma_0 < n_0\}$ and interpolating where needed. In practice to avoid searching a large lookup table of values online the interpolation is best done offline to create a two-dimensional matrix of values for each arm that has distinct priors.

An alternative to finding GIs one state at a time (the *state method*) is to use the method given in Section 8.4 of Gittins et al. (2011) for which MATLAB code is given (referred to here as the *block method*). The block method finds GI values for a block of states in one go by stepping through an increasing sequence of values of λ and assigning index values to states when the safe arm is first preferred to the risky arm. By doing this it reuses some value function calculations and can therefore be more efficient if used on a large number of states. However, there are several reasons why the state method may be preferred for general use.

1. Accuracy for the block method depends on the step size use for the λ sequence and so computation time scales linearly with accuracy. By using bisection the state method time scales logarithmically.
2. By using good initial bounds the state method only considers λ in a small subinterval of $[0, 1]$. These can be tightened further by bounds from GIs already calculated for similar states.
3. The state method is easy to parallelise which can drastically reduce the time needed.

The results of a speed comparison between methods can be seen in Table 5.4.1. This compares the block method with two implementations of the state method. Method "State B" initialises intervals for all states using KGI for a lower bound and GI+ for upper bound. "State A" also does this but then updates the intervals online as GI values are calculated. The values are calculated in ascending order of n for Σ_0 then repeating for each $\Sigma_0+1, \Sigma_0+2, \dots, \Sigma_0+T$. Lower and upper bounds for $\nu^{GI}(\Sigma, n, \gamma)$ are given respectively by $\nu^{GI}(\Sigma-1, n, \gamma)$ and $\nu^{GI}(\Sigma, n-1, \gamma)$ where these are available.

ϵ	T	N	Run Time (s)		
			Block	State A	State B
0.001	50	50	6.2	5.8	9.3
0.001	100	50	16	16	39
0.001	100	200	132	192	398
0.0001	50	50	62	13	12
0.0001	100	50	191	28	47
0.0001	100	200	1354	355	536

Table 5.4.1: A comparison of speeds of methods for finding GI for multiple states for the BMAB. State A uses online initialisation of intervals while State B intialises intervals offline. An accuracy $\epsilon = 0.0001$ is used and $\gamma = 0.99$ when $T = 200$ and $\gamma = 0.9$ otherwise. The state space is given by (5.4.1) with $\Sigma_0 = 1$, $n_0 = 2$.

The results clearly show that the online method “State A” is the faster of the two state methods. The block method can be faster for $\epsilon = 0.001$ especially for larger T but slows relatively as ϵ decreases.

If parallelisation is used then it is likely that there will be more states than processors. In this case states should be assigned to processors in a logical manner so that intervals can be initialised using GI of neighbouring states as they are found e.g. all states with the same Σ are sent to one processor which works through them in ascending n.

5.4.2 Multiple State Computation - NMAB

The NMAB takes continuous outcomes so could be much more challenging than the BMAB, especially if τ can take a range of values. Fortunately, Gittins et al. (2011) (p 217) gives invariance properties of $\nu^{GI(\tau)}$ for the NMAB which give:

$$\nu^{GI(\tau)}(\Sigma, n, \gamma, \tau) = \frac{\Sigma}{n} + \frac{1}{\sqrt{\tau}}\nu^{GI(\tau)}\left(0, \frac{n}{\tau}, \gamma, 1\right). \tag{5.4.2}$$

Therefore the dimensions have effectively been reduced to one for each γ. For a given γ, τ, T and prior n_0 we then require calculations of $\nu^{GI(\tau)}(0, n, \gamma, 1)$ for

$$n = \frac{n_0}{\tau}, \frac{n_0 + \tau}{\tau}, \ldots, \frac{n_0 + T\tau}{\tau} = \frac{n_0}{\tau}, \frac{n_0}{\tau} + 1, \ldots, \frac{n_0}{\tau} + T. \qquad (5.4.3)$$

So for $n_0 = 1$ we would need values for $n = 10, 11, \ldots, 10 + T$ for $\tau = 0.1$ and $n = 0.1, 1.1, \ldots, 0.1+T$ for $\tau = 10$. By using (5.4.2), we do not need to do calculations with $\tau \neq 1$. This is useful since the experiments in Section 5.3.2 showed that $\tau < 1$ required longer computation time. Note, though, that the $1/\sqrt{\tau}$ in (5.4.2) inflates errors when values are transformed for $\tau < 1$ so untransformed values will need to be calculated to a higher accuracy than if used directly. It will often not be necessary to calculate GI for every state as interpolation between the GIs of neighbouring states will be sufficiently accurate for larger n.

The remaining issue is the possible need to find $\nu^{GI(\tau)}(0, n/\tau, \gamma, 1)$ for $n/\tau < 1$. These states are challenging since n/τ can be arbitrarily close to zero if n is small or τ is large and $\nu^{GI(\tau)}(0, n/\tau, \gamma, 1)$ becomes large for small n/τ so using GI of similar as bounds is effective. At most one state per arm (the starting state) is of this nature. GI of states with small n/τ can be calculated most efficiently by observing from (5.4.2) that

$$\nu^{GI(\tau)}\left(0, \frac{1}{c}, \gamma, 1\right) = \sqrt{c}\nu^{GI(\tau)}(0, 1, \gamma, c).$$

Setting $c = \tau/n$ then gives

$$\nu^{GI(\tau)}\left(0, \frac{n}{\tau}, \gamma, 1\right) = \sqrt{\frac{\tau}{n}}\nu^{GI(\tau)}\left(0, 1, \gamma, \frac{\tau}{n}\right),$$

the right hand side of which is faster to calculate as can be seen from the run times in Section 5.3.2 when τ is large. In addition, the GI+ can be used to give the approximation

$$\nu^{GI(\tau)}(0, \frac{n}{\tau}, \gamma, 1) \approx \sqrt{\frac{\tau}{n}}\nu^{GI^+}(0, 1, \gamma)$$

for large τ/n. This also shows that $\nu^{GI(\tau)}(0, n, \gamma, 1)$ grows approximately linearly with $\sqrt{1/n}$ for small n. In practice this approximation is very good for even moderate values of $1/n$.

The same principles of parallelisation and online intialisation of intervals also apply to the NMAB. Computation is slower for the NMAB so finding a narrow intial interval is more important.

5.5 Discussion

In this chapter we have extended the work of Gittins et al. (2011) to give simple methods, with accompanying code, to easily calculate GIs for a extensive range of states for the BMAB and the NMAB.

The only area of these problems that could remain difficult is that for γ close to 1. Bounds are looser and so the initial interval for calibration is wider. The horizon N, and therefore the state space of the dynamic program, need to be large to guarantee good accuracy. In addition with higher γ MAB problems have longer effective time horizons before rewards become small and GI values will be needed for more states. Interpolation between GIs found for other γ is less accurate as γ increases so this useful approximation is less effective. Even here, though, the difficulties should not be overstated. GIs can still be calculated to reasonable accuracy and will produce a policy that is far closer to Bayes optimality than any alternative. The experiments in Section 5.3 indicate that accuracy can be much better than the remaining reward at stage N would suggest, especially for the NMAB, since the posterior in the one-armed bandit will become tight around θ for reasonably large N. A similar argument suggests that states with large n in a long horizon MAB will not be difficult to find. Therefore if computation resources are limited they should be concentrated on producing accurate GI values for low n. Quantifying these challenges more rigorously would be good questions for future research.

We did not discuss the calculation of GIs for more general reward distributions but many of the ideas given will apply there also. This will be of greatest difficulty if rewards are continuous and dimensions cannot be reduced using invariance results such as was done with the NMAB. Another issue will be if the CDF of the predictive cannot be calculated quickly. Even where computation of GIs is not tractable a numerical approximation using approximate dynamic programming paired with bounding methods such as those from Section 5.2 will usually produce a better policy than any simple analytical approximations.

The methods given in this chapter are also appropriate for the calculation of some forms of Whittle indices which are a generalisation of GIs applicable in a far wider range of bandit problems. We will now briefly discuss the tractability of the calculation of Whittle indices for a common variant of the MAB where the horizon is finite.

A finite horizon adds an extra variable s, the time remaining to the end of the horizon and we require a different set of Whittle indices $\nu^{WI}(\Sigma, n, \gamma, s)$ for each possible s. Each set of states is not as large as would be needed for GI since some values of n would not be possible for each s for given prior n_0. For each state, $\nu^{WI}(\Sigma, n, \gamma, s)$ can be calculated as for GI with the advantage that N does not cause any approximation and will often be relatively small. If Whittle index values for general priors and time horizons are needed then we will need to store a greater number of values relative to GI, but other than this, calculating Whittle Indices for the finite horizon MAB problem poses little extra difficulty compared to GIs. More on calculating indices for the finite horizon MAB can be found in Niño-Mora (2011).

Chapter 6

Conclusion

This section will summarise the contributions of this thesis, discuss what motivated the work and outline possible future work that leads from it. The work was concerned with Bayes sequential decision problems which display the exploration versus exploitation trade-off. This is a substantial problem class which, despite being well studied, has many open problems.

The original motivation was to study problems (beginning with variations of the multi-armed bandit) where the optimality and/or tractability of solutions based on Gittins indices (GIs) breaks down. These conditions are common : optimality can be lost if arms are correlated or if the problem horizon is finite while reward structures where Bayesian updating is no longer conjugate imply that simple heuristic policies would be needed to retain computational practicality of solutions.

The online knowledge gradient (KG) heuristic offered a possible solution to both of these issues. It has general application, is fast to calculate and because it is not an index policy it might be able to adapt to problem features, such as correlated arms, that an index policy cannot. Chapter 3 analysed the KG policy in detail and found several weaknesses, the foremost of which is that it takes actions which are dominated with respect to both exploration and exploitation. Variants were proposed which avoided these errors. These new policies include the knowledge gradient index (KGI), an index heuristic which deploys a KG approach to develop an approximation to the Gittins index.

The study has ramifications beyond the performance of a single heuristic though. KG represents a different approach to that of index methods. It can be seen as a simple approximation of a full dynamic programming solution, which is optimal very generally, whereas KGI is an approximation to the Gittins index which is optimal only in a restricted subset of problems. It might be expected therefore that the KG policy will outperform policies based on KGI and other index methods (including the Gittins index) on problems where index methods are known to be suboptimal. However this was not found to be the case with policy performance of KGI at least matching that of KG even on problems where KG does not take dominated actions. These problems included one where arms are correlated, a key problem motivating the use of KG. In addition to its unimpressive performance on this correlated multi-armed bandit problem, there is the added issue that KG in the correlated case requires considerably more computation than competing index heuristics, an issue that will reoccur in other problems. Overall the work did not yield any evidence that non-index heuristics have obvious advantages over index heuristics.

Although the intention in the work in Chapter 3 was to develop and test fast heuristic methods, an unexpected outcome was the robust effectiveness and tractability of policies based on Gittins or Whittle indices. As discussed in Chapter 5 it would be easy to take the message from recent literature that the calculation or use of GIs is not practical for most problems. This is hindering their adoption even for problems where they are known to provide an optimal solution. We have developed and made freely available software to calculate GIs for Bernoulli and normal rewards. The methodology used in this software and reports of its accuracy and speed is given in

Chapter 5. By doing this we hope to make the calculation and use of GIs accessible to a wide audience of practitioners and researchers.

The work in Chapters 3 and 5 motivates a number of outstanding research questions concerning the performance and tractability of index methods: (i) how well can index policies perform on problems for which they are not optimal, (ii) what is the extent of the tractability of calculation of GIs, (iii) if GIs can only be approximated how should this be done and, (iv) at what point do these approximations become inferior to alternative heuristics?

Two major problem classes in which to ask these questions would be contextual bandits and correlated bandits. Existing index approaches to contextual bandits (e.g. May et al. 2012) use indices that are *local*, that is they are based only on the current context. It is unclear how much return is sacrificed by making this assumption. Can indices be adapted to consider future contexts as well as the current one? The dependence of rewards on a changing context has similarities with restless bandits. Can the two problems be linked to make use of existing research? The problem in Chapter 4 had a context that took the challenging form of a probability distribution but there is still much research to be done on more basic forms of context.

Bandits with correlated beliefs can take many forms, an example of which was given in Chapter 3. Index methods which ignored the correlation in decisions but which incorporated it in belief updates were found to do well. How far from optimal is this approach? Are there problems where the optimal return (or a bound) can be found in order to make this comparison? Only positive correlations were explored in Chapter 3 but it is likely that negative correlations will make a more significant change to the problem.

One area where index methods can be optimal but the calculation of GIs might be difficult is the classical multi-armed bandit with non-standard reward distributions. This change has consequences for the state space size for which GIs are needed, the state space of the dynamic program using in calibration, and the nature of the Bayesian updating. A good example would be where reward distribution are mixtures which would occur naturally where rewards depend on some hyperparameters. These parameters could be common to all arms and be given at each time (contextual), unique to arms but vary over time (restless), common to two or more arms but constant and learnable (correlated) or simply constant, unique for each arm and learnable (closest to the classical multi-armed bandit).

The work in Chapter 4 also focused on an index method, Thompson sampling, although here the index is stochastic. This method has different characteristics than policies based on GIs. Thompson sampling together with other stochastic methods given in Chapter 2 and methods such as UCB, which are effective under regret-based criteria, were found to perform very poorly on the problems in Chapter 3 on discounted or finite horizon problems. The reasons (discussed next) why Thompson sampling was used for the multiple website elements problem illustrate when such methods are and are not appropriate.

The primary motivating application was search advertising for Google where adverts are presented alongside the results of a web search. The danger with using many methods that satisfy regret-type optimality criteria is that they sacrifice short and medium term reward for asymptotic learning guarantees. As discussed in Chapter 2 this can be risky if information is not exploitable in the long term for some reason. For example, a small web-based business in a rapidly changing area might not be able to plan that far ahead. Indeed if short term performance is poor the business might not exist to exploit the information it has gained either for financial or customer service reasons. Google on the other hand is very different. Although they are in a competitive area their established position as a dominant search engine gives them the ability to think long term. In addition, for customers, the important part of a search engine is the returned search results not the adverts. A poor selection adverts is unlikely to affect the customer experience and so will cost only short term revenue rather than affect the business long term. Thompson sampling is appropriate for this setting as it is known to be effective in exploring without having to define a time horizon.

This example illustrates how the both the value and cost of information in the system of interest can affect objectives and the consequences of using different policies. To formally study the differences in objectives will require investigation of problems where the system can change over time (e.g. restless bandits) so that the value of information decays over time.

Much of the challenge and contribution of Chapter 4 is in the development of an appropriate model for the problem. Intuitive and simple models were given which explained why it is undesirable to display similar elements as part of a set of elements. The models in turn give a method, based on maximising click-through-rate, of creating diversity in the element set. These models were based on a latent user state which gives a new, richer form of context for bandit problems. There are a number of barriers to implementing solutions for this model. The combinatorial explosion in the number of available sets was overcome by exploiting the submodularity of the click models and by recording learning about individual elements rather than sets. The latent state made exact Bayesian updating impractical but an approximate method was implemented which enabled the use of an underlying Beta-Bernoulli model which was shown to work well in simulations. By adapting the Beta-Bernoulli scheme familiar in bandit problems we were able to give a simple method by which a wide range of existing bandit algorithms could be used. This gives a flexible approach to a very general problem whereby methods can be chosen which are appropriate for the particular objectives of the problem.

Bibliography

R. Agrawal. Sample mean based index policies with $O(\log n)$ regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27(4):1054–1078, 1995.

R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *ACM Conference on Web Search and Data Mining (WSDM)*, pages 5–14. ACM, 2009.

S. Agrawal and N. Goyal. Further optimal regret bounds for Thompson sampling. In *AISTATS*, pages 99–107, 2013.

S. H. A. Ahmad and M. Liu. Multi-channel opportunistic access: A case of restless bandits with multiple plays. In *Allerton Conference*, pages 1361–1368. IEEE, 2009.

M. Asawa and D. Teneketzis. Multi-armed bandits with switching penalties. *IEEE Transactions on Automatic Control*, 41(3):328–348, 1996.

P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *The Journal of Machine Learning Research*, 3:397–422, 2002.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, 2002.

R. E. Bechhofer, T. J. Santner, and D. M. Goldsman. *Design and analysis of experiments for statistical selection, screening, and multiple comparisons.* John Wiley & Sons, New York, 1995.

R. Bellman. *Dynamic programming.* Princeton University Press, Princeton, NJ, 1957.

D. A. Berry and B. Fristedt. *Bandit Problems.* Chapman and Hall, London, UK, 1985.

D. P. Bertsekas. *Dynamic programming and optimal control.* Athena Scientific, Belmont, MA, 2012.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA, 1996.

D. Bertsimas and J. Niño-Mora. Conservation laws, extended polymatroids and multi-armed bandit problems; a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

D. Blackwell. Discrete dynamic programming. *The Annals of Mathematical Statistics*, 33(2):719–726, 1962.

L. Breiman. *Probability.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

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.

D. B. Brown and J. E. Smith. Optimal sequential exploration: Bandits, clairvoyants, and wildcats. *Operations research*, 61(3):644–665, 2013.

A. N. Burnetas and M. N. Katehakis. Optimal adaptive policies for sequential allocation problems. *Advances in Applied Mathematics*, 17(2):122–142, 1996.

O. Cappé and E. Moulines. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society. Series B*, 71(3):593–613, 2009.

O. Cappé, A. Garivier, O.-A. Maillard, R. Munos, G. Stoltz, et al. Kullback–Leibler upper confidence bounds for optimal sequential allocation. *The Annals of Statistics*, 41(3):1516–1541, 2013.

J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the ACM SIGIR*, pages 335–336. ACM, 1998.

N. Cesa-Bianchi and P. Fischer. Finite-time regret bounds for the multiarmed bandit problem. In *Proceedings of the 15th International Conference on Machine Learning*, pages 100–108. Morgan Kaufmann, San Francisco, CA, 1998.

N. Cesa-Bianchi and G. Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

J. Chakravorty and A. Mahajan. Multi-armed bandits, Gittins index, and its calculation. In *Methods and Applications of Statistics in Clinical Trials*, pages 416–435. John Wiley & Sons, Hoboken, NJ, 2014.

H. Chen and D. R. Karger. Less is more. In *ACM conference on information retrieval*, pages 429–436. ACM, 2006.

W. Chen, Y. Wang, and Y. Yuan. Combinatorial multi-armed bandit: General framework and applications. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 151–159, 2013.

S. E. Chick and N. Gans. Economic analysis of simulation selection problems. *Management Science*, 55(3):421–437, 2009.

W. Cowan, J. Honda, and M. N. Katehakis. Normal bandits of unknown means and variances: Asymptotic optimality, finite horizon regret bounds, and a solution to an open problem. *arXiv preprint*, 2015. arXiv:1504.05823.

N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the International Conference on Web Search and Web Data Mining*, pages 87–94. ACM, 2008.

V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. In *21st Annual COLT*, pages 355–366, 2008.

S. Dayanik, W. B. Powell, and K. Yamazaki. Index policies for discounted bandit problems with availability constraints. *Advances in Applied Probability*, 40(2):377–400, 2008.

N. B. Dimitrov, M. Kress, and Y. Nevo. Finding the needles in the haystack: efficient intelligence processing. *Journal of the Operational Research Society*, 67(6):801–812, 2015.

Z. Ding and I. O. Ryzhov. Optimal learning with non-Gaussian rewards. *Advances in Applied Probability*, 1(48):112–136, 2016.

J. Edwards, P. Fearnhead, and K. D. Glazebrook. On the identification and mitigation of weaknesses in the knowledge gradient policy for multi-armed bandits. *Probability in the Engineering and Informational Sciences*, in press. doi: 10.1017/S0269964816000279.

K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin. Turning down the noise in the blogosphere. In *ACM conference on knowledge discovery and data mining*, pages 289–298. ACM, 2009.

P. I. Frazier, W. B. Powell, and S. Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

P. I. Frazier, W. B. Powell, and S. Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4):599–613, 2009.

M. C. Fu. Optimization for simulation: Theory vs. practice. *INFORMS Journal on*

Computing, 14(3):192–215, 2002.

J. Ginebra and M. K. Clayton. Response surface bandits. *Journal of the Royal Statistical Society. Series B*, 57(4):771–784, 1995.

J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B*, 41:148–177, 1979.

J. C. Gittins and D. M. Jones. A dynamic allocation index for the sequential design of experiments. In J. Gani, editor, *Progress in Statistics*, pages 241–266. North-Holland, Amsterdam, NL, 1974.

J. C. Gittins, K. D. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices.* John Wiley & Sons, Chichester, UK, second edition, 2011.

K. D. Glazebrook. On randomized dynamic allocation indices for the sequential design of experiments. *Journal of the Royal Statistical Society. Series B*, 42(3):342–346, 1980.

K. D. Glazebrook. On the evaluation of suboptimal strategies for families of alternative bandit processes. *Journal of Applied Probability*, 19(3):716–722, 1982.

K. D. Glazebrook. Strategy evaluation for stochastic scheduling problems with order constraints. *Advances in Applied Probability*, 23(1):86–104, 1991.

T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 13–20, 2010.

S. S. Gupta and K. J. Miescke. Bayesian look ahead one-stage sampling allocations for selection of the best population. *Journal of Statistical Planning and Inference*, 54(2):229–244, 1996.

P. D. Hoff. *A first course in Bayesian statistical methods*. Springer Verlag, New York, 2009.

J. Honda and A. Takemura. Optimality of Thompson sampling for Gaussian bandits depends on priors. In *AISTATS*, pages 375–383, 2014.

N. Hurley and M. Zhang. Novelty and diversity in top-N recommendation–analysis and evaluation. *ACM Transactions on Internet Technology*, 10(4):1–29, 2011.

E. T. Jaynes and G. L. Bretthorst. *Probability theory: the logic of science*. Cambridge University Press, Cambridge, UK, 2003.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

T. Jun. A survey on the bandit problem with switching costs. *De Economist*, 152(4): 513–541, 2004.

L. P. Kaelbling. *Learning in embedded systems*. The MIT Press, Cambridge, MA, 1993.

M. N. Katehakis and H. Robbins. Sequential choice from several populations. *Proceedings of the National Academy of Sciences of the United States of America*, 92 (19):8584–8585, 1995.

M. N. Katehakis and A. F. Veinott Jr. The multi-armed bandit problem: decomposition and computation. *Mathematics of Operations Research*, 12(2):262–268, 1987.

E. Kaufmann, O. Cappé, and A. Garivier. On Bayesian upper confidence bounds for bandit problems. In *AISTATS*, pages 592–600, 2012.

F. P. Kelly. Contribution to the discussion of Gittins. *Journal of the Royal Statistical Society. Series B*, 41:167–8, 1979.

F. P. Kelly. Multi-armed bandits with discount factor near one: The Bernoulli case. *The Annals of Statistics*, 9(5):987–1001, 1981.

S. H. Kim and B. L. Nelson. Selecting the best system. In S. G. Henderson and B. L. Nelson, editors, *Handbooks in operations research and management science: simulation*, pages 501–534. Elsevier, 2006.

R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. *Machine learning*, 80(2-3):245–272, 2010.

N. Korda, E. Kaufmann, and R. Munos. Thompson sampling for 1-dimensional exponential family bandits. In *Advances in Neural Information Processing Systems*, pages 1448–1456. Curran Associates, Inc., 2013.

A. Krause and D. Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, volume 3. Cambridge University Press, Cambridge, UK, 2014.

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

T. Larsen, D. S. Leslie, E. J. Collins, and R. Bogacz. Posterior weighted reinforcement learning with state uncertainty. *Neural computation*, 22(5):1149–1179, 2010.

L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 297–306. ACM, 2011.

R. D. Luce. *Individual choice behavior*. John Wiley & Sons, Hoboken, NJ, 1959.

A. Mahajan and D. Teneketzis. Multi-armed bandit problems. In A. O. Hero, D. A. Castanon, D. Cochran, and K. Kastella, editors, *Foundations and Applications of Sensor Management*, pages 121–152. Springer, New York, 2008.

S. Mannor and J. N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *The Journal of Machine Learning Research*, 5:623–648, 2004.

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. Mellor. *Decision Making Using Thompson Sampling*. PhD thesis, University of Manchester, 2014.

G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

J. Niño-Mora. Computing a classic index for finite-horizon bandits. *INFORMS Journal on Computing*, 23(2):254, 2011.

D G. Pandelis and D. Teneketzis. On the optimality of the Gittins index rule for multi-armed bandits with multiple plays. *Mathematical Methods of Operations Research*, 50:449–461, 1999.

W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, Hoboken, NJ, 2007.

W. B. Powell and I. O. Ryzhov. *Optimal learning*. John Wiley & Sons, Hoboken, NJ, 2012.

M. L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Hoboken, NJ, second edition, 2005.

F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning (ICML)*, pages 784–791. ACM, 2008.

F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims. Redundancy, diversity and interdependent document relevance. *SIGIR Forum*, 43(2):46–52, 2009.

H. Raiffa and R. Schlaifer. *Applied statistical decision theory.* MIT Press, Cambridge, MA, 1968.

U. Rieder. Bayesian dynamic programming. *Advances in Applied Probability*, 7(2): 330–348, 1975.

H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

C. P. Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation.* Springer Verlag, New York, 2007.

P. Rusmevichientong and J. N. Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.

D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

I. O. Ryzhov and W. B. Powell. The value of information in multi-armed bandits with exponentially distributed rewards. In *Proceedings of the 2011 International Conference on Computational Science*, pages 1363–1372, 2011.

I. O. Ryzhov, P. I. Frazier, and W. B. Powell. On the robustness of a one-period look-ahead policy in multi-armed bandit problems. *Procedia Computer Science*, 1 (1):1635–1644, 2010.

I. O. Ryzhov, W. B. Powell, and P. I. Frazier. The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195, 2012.

S. L. Scott. A modern Bayesian look at the multiarmed bandit. *Applied Stochastic Models in Business and Industry*, 26:639–658, 2010.

M. Shaked and J. G. Shanthikumar. *Stochastic orders.* Springer, New York, 2007.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction.* MIT press, Cambridge, MA, 1998.

W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

J. N. Tsitsiklis. A short proof of the Gittins index theorem. *The Annals of Applied Probability*, 4(1):194–199, 1994.

M. P. Van Oyen, D. G. Pandelis, and D. Teneketzis. Optimality of index policies for stochastic scheduling with switching penalties. *Journal of Applied Probability*, 29 (4):957–966, 1992.

S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, pages 109–116. ACM, 2011.

A. Wald. *Statistical decision functions.* John Wiley & Sons, Hoboken, NJ, 1950.

A. Wald. *Sequential analysis.* John Wiley & Sons, New York, 1973.

C. J. C. H. Watkins. *Learning from delayed rewards.* PhD thesis, Cambridge University, UK, 1989.

R. R. Weber. On the Gittins index for multiarmed bandits. *The Annals of Applied Probability*, 2(4):1024–1033, 1992.

R. R. Weber and G. Weiss. On an index policy for restless bandits. *Journal of Applied Probability*, 27:637–648, 1990.

G. Weiss. Branching bandit processes. *Probability in the Engineering and Informational Sciences*, 2(3):269–278, 1988.

P. Whittle. Multi-armed bandits and the Gittins index. *Journal of the Royal Statistical Society. Series B*, 42(2):143–149, 1980.

P. Whittle. Arm-acquiring bandits. *The Annals of Probability*, 9(2):284–292, 1981.

P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25:287–298, 1988.

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.

Y. C. Yao. Some results on the Gittins index for a normal reward process. *Lecture Notes-Monograph Series*, 52:284–294, 2006.

Y. Yu. Structural properties of Bayesian bandits with exponential family distributions. *arXiv preprint*, 2011. arXiv:1103.3089v1.

Y. Yue and C. Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*, pages 2483–2491, 2011.

Y. Yue, J. Broder, R. Kleinberg, and T. Joachims. The K–armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515, 2010.