Decision Making Under Uncertainty and Reinforcement Learning

Christos Dimitrakakis Ronald Ortner

January 15, 2019

Contents

Chapter 1

Introduction

The purpose of this book is to collect the fundamental results for decision making under uncertainty in one place, much as the book by Puterman [1994] on Markov decision processes did for Markov decision process theory. In particular, the aim is to give a unified account of algorithms and theory for sequential decision making problems, including reinforcement learning. Starting from elementary statistical decision theory, we progress to the reinforcement learning problem and various solution methods. The end of the book focuses on the current state-of-the-art in models and approximation algorithms.

The problem of decision making under uncertainty can be broken down into two parts. First, how do we learn about the world? This involves both the problem of *modeling our initial uncertainty* about the world, and that of drawing *conclusions* from *evidence* and our initial belief. Secondly, given what we currently know about the world, how should we *decide* what to do, taking into account future events and observations that may change our conclusions?

Typically, this will involve creating long-term plans covering possible future eventualities. That is, when planning under uncertainty, we also need to take into account what possible future knowledge could be generated when implementing our plans. Intuitively, executing plans which involve trying out new things should give more information, but it is hard to tell whether this information will be beneficial. The choice between doing something which is already known to produce good results and experiment with something new is known as the exploration-exploitation dilemma, and it is at the root of the interaction between learning and planning.

1.1 Uncertainty and Probability

A lot of this book is grounded in the essential methods of probability, in particular using it to represent uncertainty. While probability is a simple mathematical construction, philosophically it has had at least three different meanings. In the classical sense, a probability distribution is a description for a truly random event. In the subjectivist sense, probability is merely an expression of our uncertainty, which is not necessarily due to randomness. Finally, in the algorithmic sense, probability is linked with how "simple" a program that can generate a particular output is.

In all cases, we are dealing with a set Ω of possible outcomes: the result of a random experiment, the underlying state of the world and the program output respectively. In all cases, we use probability to model our uncertainty over Ω.

Classical Probability

A *random experiment* is performed, with a given set Ω of possible outcomes. An example is the 2-slit experiment in physics, where a particle is generated which can go through either one of two slits. According to our current understanding of quantum mechanics, it is impossible to predict which slit the particle will go through. Herein, the set Ω consists of two possible events corresponding to the particle passing through one or the other slit.

In the 2-slit experiment, the probabilities of either event can be actually accurately calculated through quantum theory. However, which slit the particle

will go through is fundamentally unpredictable. Such quantum experiments are the only ones that are currently thought of as truly random (though some people disagree about that too). Any other procedure, such as tossing a coin or casting a die, is inherently deterministic and only *appears* random due to our difficulty in predicting the outcome. That is, modelling a coin toss as a random process is usually the best approximation we can make in practice, given our uncertainty about the complex dynamics involved. This gives rise to the concept of subjective probability as a general technique to model uncertainty.

Subjective Probability
Here Ω can conceptually not only describe the outcomes of some experiment, but also a set of possible *worlds* or realities. This set can be quite large and include anything imaginable. For example, it may include worlds where dragons are real. However, in practice one only cares about certain aspects of the world, such as whether in this world, you will win the lottery if you buy a ticket. We can interpret the probability of a world in Ω as our degree of belief that it corresponds to reality.

In such a setting there is an actual, true world $\omega^* \in \Omega$, which is simply unknown. This could have been set by Nature to an arbritrary value deterministically. The probability only reflects our lack of knowledge, rather than any inherent randomness about the selection of ω^*.

No matter which view we espouse, we must always take into account our uncertainty when making decisions. When the problem we are dealing with is sequential, we are taking actions, obtaining new observations, and then taking further actions. As we gather more information, we learn more about the world. However, the things we learn about depends on what actions we take. For example, if we always take the same route to work, then we learn how much time this route takes on different days and times of the week. However, we don't obtain information about the time other routes take. So, we potentially miss out on better choices than the one we follow usually. This phenomenon gives rise to the so-called exploration-exploitation trade-off.

1.2 The exploration-exploitation trade-off

Consider the problem of selecting a restaurant to go to during a vacation. The best restaurant you have found so far was *Les Epinards*. The food there is usually to your taste and satisfactory. However, a well-known recommendations website suggests that *King's Arm* is really good! It is tempting to try it out. But there is a risk involved. It may turn out to be much worse than Les Epinards, in which case you will regret going there. On the other hand, it could also be much better. What should you do?

It all depends on how much information you have about either restaurant, and how many more days you'll stay in town. If this is your last day, then it's probably a better idea to go to Les Epinards, unless you are expecting King's Arm to be significantly better. However, if you are going to stay there longer, trying out King's Arm is a good bet. If you are lucky, you will be getting much better food for the remaining time, while otherwise you will have missed only one good meal out of many, making the potential risk quite small.

Thus, one must decide whether to *exploit* knowledge about the world, to gain a *known* reward, or to *explore* the world to *learn* something new. This will potentially give you less reward immediately, but the knowledge itself can usually be put to use in the future.

This exploration-exploitation trade-off only arises when data collection is *interactive*. If we are simply given a set of data and asked to decide upon a course or action, but our decision does not affect the data we shall collect in the future, then things are much simpler. However, a lot of real-world human decision making as well as modern applications in data science involve such trade-offs. Decision theory offers precise mathematical models and algorithms for such problems.

1.3 Decision theory and reinforcement learning

Decision theory deals with the formalization and solution of decision problems. Given a number of alternatives, what would be the rational choice in a particular situation depending on one's goals and desires? In order to answer this question we need to develop a good concept of *rational behavior*. This will serve two purposes. Firstly, this can serve as an *explanation* for what animals and humans (should) do. Secondly, it should be *useful* for developing models and algorithms for automated decision making in complex tasks.

A particularly interesting problem in this setting is *reinforcement learning*. This problem arises when the environment is unknown, and the learner has to make decisions solely through interaction, which only gives limited *feedback*. Thus, the learning agent does not have access to detailed instructions on which task to perform, nor on how to do it. Instead, it performs *actions*, which affect the environment and obtains some observations (i.e. sensory input) and feedback, usually in form of *rewards* which correspond to the agent's desires. The learning problem is then formulated as the problem of learning how to act to maximize total reward. In biological systems, reward is intrinsically hardwired to signals associated with basic needs. In artificial systems, we can choose the reward signals so as to reinforce behaviour that achieves the designer's goals.

Reinforcement learning is a fundamental problem in artificial intelligence, since frequently we can tell robots, computers, or cars only what we would like them to achieve, but we do not know the best way to achieve it. We would like to simply give them a description of our goals and then let them explore the environment on their own to find a good solution. Since the world is (at least partially) unknown, the learner always has to deal with the exploration-exploitation trade-off.

Similarly, animals and humans also learn through imitation, exploration, and shaping their behavior according to reward signals to finally achieve their goals. In fact, it has been known since the 1990s that there is some connection between some reinforcement learning algorithms and mechanisms in the basal ganglia.[Yin and Knowlton, 2006, Barto, 1995, Schultz et al., 1997]

Decision theory is closely related to other fields, such as logic, statistics, game theory and optimization. Those fields have slightly different underlying objectives, even though they may share the same formalisms. In the field of *optimization*, we are not only interested in optimal planning in complex environments but also in how to make *robust* plans given some uncertainty about

the environment. *Artificial intelligence* research is concerned with modelling the environments and developing algorithms that are able to learn by interaction with the environment or from demonstration by teachers. *Economics* and *game theory* deal with the problem of modeling the behavior of rational agents and with designing mechanisms (such as markets) that will give incentives to agents to behave in a certain way.

Beyond pure research, there are also many applications connected to decision theory. Commercial applications arise e.g. in advertising where one wishes to model the preferences and decision making of individuals. Decision problems also arise in *security*. There are many decision problems, especially in cryptographic and biometric authentication, but also in detecting and responding to intrusions in networked computer systems. Finally, in the natural sciences, especially in *biology and medicine*, decision theory offers a way to automatically design and run experiments and to optimally construct clinical trials.

Outline

1. Subjective probability and utility: The notion of subjective probability; eliciting priors; the concept of utility; expected utility.

2. Decision problems: maximising expected utility; maximin utility; regret.

3. Estimation: Estimation as conditioning; families of distributions that are closed under conditioning; conjugate priors; concentration inequalities; PAC and high-probability bounds; Markov Chain Monte Carlo; ABC estimation.

4. Sequential sampling and optimal stopping: Sequential sampling problems; the cost of sampling; optimal stopping; martingales.

5. Reinforcement learning I - Markov decision processes Belief and information state; bandit problems; Markov decision processes; backwards induction; value iteration; policy iteration; temporal differences; linear programming

6. Reinforcement learning II – Stochastic and approximation algorithms: Sarsa; Q-learning; stochastic value iteration; TD(λ)

7. Reinforcement learning III – Function approximation features and the curse of dimensionality; approximate value iteration; approximate policy iteration; policy gradient

8. Reinforcement learning IV – Bayesian reinforcement learning: bounds on the utility; Thompson sampling; stochastic branch and bound; sparse sampling; partially observable MDPs.

9. Reinforcement learning V – Distribution-free reinforcement learning: stochastic and metric bandits; UCRL; (*) bounds for Thompson sampling.

B Probability refresher: measure theory; axiomatic definition of probability; conditional probability; Bayes' theorem; random variables; expectation

C Miscellaneous mathematical results.

1.4 Acknowledgements.

Many thanks go to all the students of the *Decision making under uncertainty* and *Advanced topics in reinforcement learning and decision making* class over the years, for bearing with early drafts of this book. A big "thank you" goes to Nikolaos Tziortziotis, whose code is used in some of the examples in the book. Finally, thanks to Aristide Tossou and Hannes Eriksson for proof-reading various chapters. Finally, a lot of the coded examples in the book were run using the *parallel* package by Tange [2011].

Chapter 2

Subjective probability and utility

2.1 Subjective probability

In order to make decisions, we need to be able to make predictions about the possible outcomes of each decision. Usually, we have *uncertainty* about what those outcomes are. This can be due to *stochasticity*, which is frequently used to model games of chance and inherently unpredictable physical phenomena. It can also be due to *partial information*, a characteristic of many natural problems. For example, it might be hard to guess at any one moment how much change you have in your wallet, whether you will be able to catch the next bus, or to remember where you left your keys.

In either case, this uncertainty can be expressed as a *subjective belief*. This does not have to correspond to reality. For example, some people believe, quite inaccurately, that if a coin comes up tails for a long time, it is quite likely to come up heads very soon. Or, you might quite happily believe your keys are in your pocket, only to realise that you left them at home as soon you arrive at the office.

In this book, we assume the view that subjective beliefs can be modelled as *probabilities*. This allows us to treat uncertainty due to stochasticity and due to partial information in a unified framework. In doing so, we shall treat each part of the problem as specifying a space of possible outcomes. What we wish to do is to find a *consistent way* of defining probabilities in the space of outcomes.

2.1.1 Relative likelihood

Let us start with the simple example of guessing whether a tossed coin will come up head, or tails. In this case the sample space Ω would correspond to every possible way the coin can land. Since we are only interested in predicting which face will be up, let $A \subset \Omega$ be all those cases where the coin comes up heads, and $B \subset \Omega$ be the set of tosses where it comes up tails. Here $A \cap B = \emptyset$, but there may be some other events such as the coin becoming lost, so it does not necessarily hold that $A \cup B = \Omega$. Nevertheless, we only care about whether A is more likely to occur than B. As said, this likelihood may be based only on subjective beliefs. We can express that via the concept of relative likelihood:

(The relative likelihood of two events A and B)

- If A is *more* likely than B, then we write $A \succ B$, or equivalently $B \prec A$.
- If A is *as likely* as B, then we write $A \asymp B$.

We also use $\succsim$ and $\precsim$ for *at least as likely as* and for *no more likely than*.

Let us now speak more generally about the case where we have defined an appropriate σ-field $\mathcal{F}$ on Ω. Then each element $A_i \in \mathcal{F}$ will be a subset of Ω. We now wish to define relative likelihood relations for the elements $A_i \in \mathcal{F}$.[1]

[1] More formally, we can define three classes: $C_\succ, C_\prec, C_\asymp \subset \mathcal{F}^2$ such that a pair $(A_i, A_j) \in C_R$ if an only if it satisfies the relation $A_i R A_j$, where $R \in \{\succ, \prec, \asymp\}$. These the three classes form a partition of $\mathcal{F}^2$ under the subjective probability assumptions we will introduce in the

As we would like to use the language of probability to talk about likelihoods, we need to define a probability measure that agrees with our given relations. A probability measure $P : \mathcal{F} \to [0, 1]$ is said to *agree* with a relation $A \precsim B$, if it has the property that $P(A) \leq P(B)$ if and only if $A \precsim B$, for all $A, B \in \mathcal{F}$. In general, there are many possible measures that can agree with a given relation, cf. Example 1 below. However, it could also be that a given relational structure is incompatible with any possible probability measure. We also consider the question under which assumptions a likelihood relation corresponds to a unique probability measure.

2.1.2 Subjective probability assumptions

We would like our beliefs to satisfy some intuitive properties about what statements we can make concerning the relative likelihood of events. As we will see, these assumptions are also necessary to guarantee the existence of a corresponding probability measure. First of all, it must always be possible to say whether one event is more likely than the other, i.e. our beliefs must be complete. Consequently, we are not allowed to claim ignorance.

Assumption 2.1.1 (SP1)**.** *For any pair of events $A, B \in \mathcal{F}$, one has either $A \succ B$, $A \prec B$, or $A \eqsim B$.*

Another important assumption is a principle of consistency: Informally, if we believe that every possible event A_i that leads to A is less likely than a unique corresponding event B_i that leads to an outcome B, then we should always conclude that A is less likely than B.

Assumption 2.1.2 (SP2)**.** *Let $A = A_1 \cup A_2$, $B = B_1 \cup B_2$ with $A_1 \cap A_2 = B_1 \cap B_2 = \emptyset$. If $A_i \precsim B_i$ for $i = 1, 2$ then $A \precsim B$.*

We also require the simple technical assumption that any event $A \in \mathcal{F}$ is at least as likely as the empty event $\emptyset$, which never happens.

Assumption 2.1.3 (SP3)**.** *For all A it holds that $\emptyset \precsim A$. Further, $\emptyset \prec \Omega$.*

As it turns out, these assumptions are sufficient for proving the following theorems [DeGroot, 1970]. The first theorem tells us that our belief must be consistent with respect to transitivity.

Theorem 2.1.1 (Transitivity)**.** *Under Assumptions 2.1.1, 2.1.2, and 2.1.3, for all events A, B, C: If $A \precsim B$ and $B \precsim C$, then $A \precsim C$.*

The second theorem says that if two events have a certain relation, then their negations have the converse relation.

Theorem 2.1.2 (Complement)**.** *For any A, B: $A \precsim B$ iff $A^{\complement} \succ B^{\complement}$.*

Finally, note that if $A \subset B$, then it must be the case that whenever A happens, B must happen and hence B must be at least as likely as A. This is demonstrated in the following theorem.

Theorem 2.1.3 (Fundamental property of relative likelihoods)**.** *If $A \subset B$ then $A \precsim B$. Furthermore, $\emptyset \precsim A \precsim \Omega$ for any event A.*

---

next section.

Since we are dealing with σ-fields, we need to introduce properties for infinite sequences of events. While these are not necessary if the field $\mathcal{F}$ is finite, it is good to include them for generality.

Assumption 2.1.4 (SP4)**.** *If $A_1 \supset A_2 \supset \cdots$ is a decreasing sequence of events in $\mathcal{F}$ and $B \in \mathcal{F}$ is such that $A_i \succsim B$ for all i, then $\bigcap_{i=1}^{\infty} A_i \succsim B$.*

As a consequence, we obtain the following dual theorem:

Theorem 2.1.4. *If $A_1 \subset A_2 \subset \cdots$ is an increasing sequence of events in $\mathcal{F}$ and $B \in \mathcal{F}$ is such that $A_i \precsim B$ for all i, then $\bigcup_{i=1}^{\infty} A_i \precsim B$.*

We are now able to state a theorem for the unions of infinite sequences of disjoint events.

Theorem 2.1.5. *If $(A_i)_{i=1}^{\infty}$ and $(B_i)_{i=1}^{\infty}$ are infinite sequences of disjoint events in $\mathcal{F}$ such that $A_i \precsim B_i$ for all i, then $\bigcup_{i=1}^{\infty} A_i \precsim \bigcup_{i=1}^{\infty} B_i$.*

The following theorem shows that if likelihood is induced by a probability measure P (that is, $A \succ B$ iff $P(A) > P(B)$, and $A \asymp B$ if $P(A) = P(B)$, so that P agrees with $\succsim$), it always satisfies the stipulated assumptions.

Theorem 2.1.6. *Let P be a probability measure over Ω. Then*

(i) $P(A) > P(B)$, $P(A) < P(B)$ or $P(A) = P(B)$ for all A, B.

(ii) Consider (possibly infinite) partitions $\{A_i\}_i$, $\{B_i\}_i$ of A, B, respectively. If $P(A_i) \leq P(B_i)$ for all i, then $P(A) \leq P(B)$.

(iii) For any A, $P(\emptyset) \leq P(A)$ and $P(\emptyset) < P(\Omega)$.

Proof. Part (i) is trivial, as $P : \mathcal{F} \to [0, 1]$. Part (ii) follows from $P(A) = P(\bigcup_i A_i) = \sum_i P(A_i) \leq \sum_i P(B_i) = P(B)$. Part (iii) follows from $P(\emptyset) = 0$, $P(A) \geq 0$, and $P(\Omega) = 1$. □

2.1.3 Assigning unique probabilities*

In many cases, and particularly when $\mathcal{F}$ is a finite field, there is a large number of probability distributions agreeing with our relative likelihoods. Choosing one specific probability over another does not seem easy. The following example underscores this ambiguity.

EXAMPLE 1. Consider $\mathcal{F} = \left\{\emptyset, A, A^{\complement}, \Omega\right\}$ and say $A \succ A^{\complement}$. Consequently, $P(A) > 1/2$. But this is insufficient for assigning a specific value to $P(A)$.

In some cases we would like to assign unique probabilities to events in order to facilitate computations.

This can be achieved by augmenting our set of events with random draws from a uniform distribution, defined below for intervals in $[0, 1]$. Intuitively, we may only be able to tell whether some event A is more likely than some other event B. However, we can create a new, uniformly distributed random variable x on $[0, 1]$ and ask ourselves, for each value $\alpha \in [0, 1]$ whether A more or less likely than the event $x > \alpha$. Since we need to compare both A and B with all such events, the distribution we'll obtain is unique. Essentially, we relate the

likelihoods of our two discrete events with the uniform distribution, in order to assign specific probabilities to them. Without further ado, here is the definition of the uniform distribution.

Definition 2.1.1 (Uniform distribution)**.** Let $\lambda(A)$ denote the length of any interval $A \subseteq [0,1]$. Then $x : \Omega \to [0,1]$ has a uniform distribution on $[0,1]$ if, for any subintervals A, B of $[0,1]$,

$$(x \in A) \precsim (x \in B) \quad \text{iff} \quad \lambda(A) \leq \lambda(B),$$

where $(x \in A)$ denotes the event that $x(\omega) \in A$. Then $(x \in A) \precsim (x \in B)$ means that ω is such that $x \in A$ is not more likely than $x \in B$.

This means that *any* larger interval is more likely than *any* smaller interval. Now we shall connect the uniform distribution to the original sample space Ω by assuming that there is some function with uniform distribution.

Assumption 2.1.5 (SP5)**.** *It is possible to construct a random variable* $x : \Omega \to [0,1]$ *with a uniform distribution in* $[0,1]$.

Constructing the probability distribution

We can now use the uniform distribution to create a unique probability measure that agrees with our likelihood relation. First, we have to map each event in Ω to an equivalent event in $[0,1]$.

Theorem 2.1.7 (Equivalent event)**.** *For any event* $A \in \mathcal{F}$*, there exists some* $\alpha \in [0,1]$ *such that* $A \eqsim (x \in [0,\alpha])$.

This means that we can now define the probability of an event A by matching it to a specific equivalent event on $[0,1]$.

Definition 2.1.2 (The probability of A)**.** Given any event A, define $P(A)$ to be the α with $A \eqsim (x \in [0,\alpha])$.

Hence

$$A \eqsim (x \in [0, P(A)]).$$

The above is sufficient to show the following theorem.

Theorem 2.1.8 (Relative likelihood and probability)**.** *If assumptions SP1-SP5 are satisfied, then the probability measure* P *defined above is unique. Furthermore, for any two events* A, B*,* $A \precsim B$ *iff* $P(A) \leq P(B)$.

2.1.4 Conditional likelihoods

So far we have only considered the problem of forming opinions about which events are more likely *a priori*. However, we also need to have a way to incorporate evidence which may adjust our opinions. For example, while we ordinarily may think that $A \precsim B$, we may have additional information D, given which we think the opposite is true. We can formalise this through the notion of conditional likelihoods.

Example 2. Say that A is the event that it rains in Gothenburg, Sweden tomorrow. We know that Gothenburg is quite rainy due to its oceanic climate, so we set $A \succsim A^{\complement}$. Now, let us try and incorporate some additional information. Let D denote the fact that good weather is forecast. I personally believe that $(A \mid D) \precsim (A^{\complement} \mid D)$, i.e. that good weather is more probable than rain, given the evidence of the weather forecast.

Conditional likelihoods

Define $(A \mid D) \precsim (B \mid D)$ to mean that B is at least as likely as A when it is known that D has occurred.

Assumption 2.1.6 (CP)**.** *For any events A, B, D,*

$$(A \mid D) \precsim (B \mid D) \quad \textit{iff} \quad A \cap D \precsim B \cap D.$$

Theorem 2.1.9. *If a likelihood relation $\precsim$ satisfies assumptions SP1 to SP5, as well as CP, then there exists a probability measure P such that: For any A, B, D such that $P(D) > 0$,*

$$(A \mid D) \precsim (B \mid D) \quad \textit{iff} \quad P(A \mid D) \leq P(B \mid D).$$

It turns out that there are very few ways that a conditional probability definition can satisfy all of our assumptions. One natural definition, indeed employed pretty much everywhere in probability theory, is the following.

Definition 2.1.3 (Conditional probability)**.**

$$P(A \mid D) \triangleq \frac{P(A \cap D)}{P(D)}. \tag{2.1.1}$$

This definition effectively answers the question of how much evidence for A we have, now that we have observed D. This is expressed as the ratio between the combined event $A \cap D$, also known as the joint probability of A and D, and the marginal probability of D itself. The intuition behind the definition becomes clearer once we rewrite it as $P(A \cap D) = P(A \mid D)P(D)$. Then conditional probability is effectively used as a way to factorise joint probabilities.

2.1.5 Probability elicitation

Probability elicitation is the problem of quantifying the subjective probabilities that a particular individual uses. One of the simplest, and most direct, methods, is to simply ask. However, because we cannot simply ask somebody to completely specify a probability distribution, we can ask for this distribution iteratively.

Example 3 (Temperature prediction). Let τ be the temperature tomorrow at noon in Gothenburg. What are your estimates?

Eliciting the prior / forming the subjective probability measure P

- Select temperature x_0 s.t. $(\tau \leq x_0) \eqsim (\tau > x_0)$.
- Select temperature x_1 s.t. $(\tau \leq x_1 \mid \tau \leq x_0) \eqsim (\tau > x_1 \mid \tau \leq x_0)$.

By repeating this procedure recursively we will slowly build the complete distribution, quantile by quantile.

Note that, necessarily, $P(\tau \leq x_0) = P(\tau > x_0) = p_0$. Since $P(\tau \leq x_0) + P(\tau > x_0) = P(\tau \leq x_0 \cup \tau > x_0) = P(\tau \in \mathbb{R}) = 1$, it follows that $p_0 = 1/2$. Similarly, $P(\tau \leq x_1 \mid \tau \leq x_0) = P(\tau > x_1 \mid \tau \leq x_0) = 1/4$.

EXERCISE 1. Propose another way to arrive at a prior probability distribution. For examples, define a procedure for eliciting a single probability distribution from a group of people without any interaction between the participants.

2.2 Updating beliefs: The theorem of Bayes

Although we always start with a particular belief, this belief must be adjusted when we receive new evidence. In probabilistic inference, the updated beliefs are simply the probability of future events conditioned on observed events. This idea is captured neatly by Bayes' theorem, which links the prior probability $P(A_i)$ of events to their posterior probability $P(A_i \mid B)$ given some event B and the probability $P(B \mid A_i)$ of observing the evidence B given that the events A_i are true.

Theorem 2.2.1 (Bayes' theorem)**.** *Let $A_1, A_2, \ldots$ be a (possibly infinite) sequence of disjoint events such that $\bigcup_{i=1}^{n} A_i = \Omega$ and $P(A_i) > 0$ for all i. Let B be another event with $P(B) > 0$. Then*

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{\sum_{j=1}^{n} P(B \mid A_j)P(A_j)}. \tag{2.2.1}$$

Proof. By definition, $P(A_i \mid B) = P(A_i \cap B)/P(B)$, and $P(A_i \cap B) = P(B \mid A_i)P(A_i)$, so:

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{P(B)}, \tag{2.2.2}$$

As $\bigcup_{j=1}^{n} A_j = \Omega$, we have $B = \bigcup_{j=1}^{n}(B \cap A_j)$. Since A_i are disjoint, so are $B \cap A_i$. As P is a probability, the union property and an application of (2.2.2) give

$$P(B) = P\left(\bigcup_{j=1}^{n}(B \cap A_j)\right) = \sum_{j=1}^{n} P(B \cap A_j) = \sum_{j=1}^{n} P(B \mid A_j)P(A_j).$$

□

A simple exercise in updating beliefs

EXAMPLE 4 (The weather forecast). Form a subjective probability for the probability that it rains.

A_1 : Rain.

A_2 : No rain.

First, choose $P(A_1)$ and set $P(A_2) = 1 - P(A_1)$. Now assume that there is a weather forecasting station that predicts *no rain* for tomorrow. However, you know the following fact about the station: on the days when it rains, half of the time the station had predicted it *was not* going to rain. On days when it doesn't rain, the station had said *no rain* 9 times out of 10.

Solution. Let B denote the event that the station predicts no rain. According to our information, $P(B \mid A_1) = 1/2$, i.e. whenever there is rain (A_1), the previous day's prediction said no rain (B). On the other hand, $P(B \mid A_2) = 0.9$. Combining these with Bayes rule, we obtain.

$$\begin{aligned} P(A_1 \mid B) &= \frac{P(B \mid A_1)P(A_1)}{P(B \mid A_1)P(A_1) + P(B \mid A_2)[1 - P(A_1)]} \\ &= \frac{1/2P(A_1)}{0.9 - 0.4P(A_1)}. \end{aligned}$$

□

2.3 Utility theory

While probability can be used to describe how likely an event is, utility can be used to describe how desirable it is. More concretely, our subjective probabilities are numerical representations of our beliefs and information. They can be taken to represent our "internal model" of the world. By analogy, our utilities are numerical representations of our tastes and preferences. Even if the consequences of our actions are not directly known to us, we assume that we act to maximise our utility, in some sense.

2.3.1 Rewards and preferences

Rewards

Consider that we have to choose a *reward* r from a set $\mathcal{R}$ of possible rewards. While the elements of $\mathcal{R}$ may be abritrary, we shall in general find that we prefer some rewards to others. In fact, some elements of $\mathcal{R}$ may not even be desirable. As an example, $\mathcal{R}$ might be a set of tickets to different musical events, or a set of financial commodities.

Preferences

Example 5 (Musical event tickets). We have a set of tickets $\mathcal{R}$, and we must choose the ticket $r \in \mathcal{R}$ we prefer best.

- Case 1: $\mathcal{R}$ are tickets to different music events at the same time, at equally good halls with equally good seats and the same price. Here preferences simply coincide with the preferences for a certain type of music or an artist.
- Case 2: $\mathcal{R}$ are tickets to different events at different times, at different quality halls with different quality seats and different prices. Here, preferences may depend on all the factors.

Example 6 (Route selection). We have a set of alternate routes and must pick one.

- $\mathcal{R}$ contains two routes, one short and one long, of the same quality.
- $\mathcal{R}$ contains two routes, one short and one long, but the long route is more scenic.

Preferences among rewards

We will treat preferences in a similar manner as we have treated probabilities. That is, we will define a linear ordering among possible rewards.

Let $a, b \in \mathcal{R}$ be two rewards. When we *prefer* a to b, we write $a \succ^* b$. Conversely, when we like a *less* than b we write $a \prec^* b$. If we like a *as much* as b, we write $a \eqsim^* b$. We also use $\succsim^*$ and $\precsim^*$ for *I like at least as much as* and for *I don't like any more than*, respectively. We make the following assumptions about the preference relations.

Assumption 2.3.1. *(i) For any $a, b \in \mathcal{R}$, one of the following holds: $a \succ^* b$, $a \prec^* b$, $a \eqsim^* b$.*

(ii) If $a, b, c \in \mathcal{R}$ are such that $a \precsim^* b$ and $b \precsim^* c$, then $a \precsim^* c$.

The first assumption means that we must always be able to decide between any two rewards. It may seem that it does not always hold in practice, since humans are frequently indecisive. However, without the second assumption, it is still possible to create preference relations that are cyclic.

Example 7 (Counter-example for transitive preferences). Consider vector rewards in $\mathcal{R} = \mathbb{R}^2$, with $r_i = (a_i, b_i)$, and some $\epsilon, \epsilon' > 0$. Our preference relation is:

- $r_i \succ^* r_j$ if $b_i \geq b_j + \epsilon'$.
- $r_i \succ^* r_j$ if $a_i \geq a_j$ and $|b_i - b_j| < \epsilon$.

This may correspond for example to an employer deciding to hire one of two employees, i, j, depending on their experience (a) or their school grades (b). Since grades are not very reliable, if two people have grades, then we prefer the one with the most experience. However, that may lead to a cycle. Consider a sequence of candidates $i = 1, \ldots, n$, such that each candidate satisfies $b_i = b_{i+1} + \delta$, with $\delta < \epsilon$ and $a_i > a_{i+1}$. Then clearly, we must always prefer r_i to r_{i+1}. However, if $\delta n > \epsilon$, we will prefer r_n to r_1.

2.3.2 Preferences among distributions

When we cannot select rewards directly

In most problems, we cannot choose the rewards directly. Rather, we must make some decision, and then obtain a reward depending on this decision. Since we may be uncertain about the outcome of a decision, we can specify our uncertainty regarding the rewards obtained by a decision in terms of a probability distribution.

Example 8 (Route selection). Assume that you have to pick between two routes P_1, P_2. Your preferences are such that shorter time routes are preferred over longer ones. For simplicity, let $\mathcal{R} = \{10, 15, 25, 30\}$ be the possible times we might take to reach your destination. Route P_1 takes 10 minutes when the road is clear, but 30 minutes when the traffic is heavy. The probability of heavy traffic on P_1 is 0.5. On the other hand, route P_2 takes 15 minutes when the road is clear, but 25 minutes when the traffic is heavy. The probability of heavy traffic on P_2 is 0.2.

Preferences among probability distributions

As seen in the previous example, we frequently have to define preferences between probability distributions, rather than over rewards. To represent our

preferences, we can use the same notation as before. Let P_1, P_2 be two distributions on $(R, \mathcal{F}_R)$. If we *prefer* P_1 to P_2, we write $P_1 \succ^* P_2$. If we like P_1 *less* than P_2, write $P_1 \prec^* P_2$. If we like P_1 *as much* as P_2, we write $P_1 \eqsim^* P_2$. Finally, we also use $\succsim^*$ and $\precsim^*$ do denote strict preference relations.

What would be a good principle for choosing between the two routes in Example 8? Clearly route P_1 gives both the lowest best-case time and the highest worst-case time. It thus appears as though both an extremely cautious person (who assumes the worst-case) and an extreme optimist (who assumes the best case) would say $P_2 \succ^* P_1$. However, the average time taken in P_1 is only 17 minutes versus 20 minutes for P_1. Thus, somebody that only took the average time into account would prefer P_1. In the following sections, we will develop one of the most fundamental methodologies for choices under uncertainty, based on the idea of utilities.

2.3.3 Utility

The concept of utility allows us to create a unifying framework, such that given a particular set of rewards and probability distributions on them, we can define preferences among distributions automatically. The first step is to define utility as a way to define a preference relation among rewards.

Definition 2.3.1 (Utility)**.** A utility function $U : \mathcal{R} \to \mathbb{R}$ is said to agree with the preference relation $\succsim^*$, if for all rewards $a, b \in \mathcal{R}$

$$a \succsim^* b \quad \text{iff} \quad U(a) \geq U(b). \tag{2.3.1}$$

The above definition is very similar to how we defined relative likelihood in terms of probability. For a given utility function, its expectation for a distribution over rewards is defined as follows:

Definition 2.3.2 (Expected utility)**.** Given a utility function U, the expected utility of a distribution P on $\mathcal{R}$ is:

$$\mathbb{E}_P(U) = \int_{\mathcal{R}} U(r) \, \mathrm{d}P(r) \tag{2.3.2}$$

We make the assumption that the utility function is such that the expected utility remains consistent with the preference relations between all probability distributions we are choosing between.

Assumption 2.3.2 (The expected utility hypothesis)**.** *Given a preference relation $\succsim^*$ over $\mathcal{R}$ and a corresponding utility function U, the utility of any probability measure P on $\mathcal{R}$ is equal to the expected utility of the reward under P. Consequently,*

$$P \succsim^* Q \quad \textit{iff} \quad \mathbb{E}_P(U) \geq \mathbb{E}_Q(U). \tag{2.3.3}$$

Example 9. Consider the following decision problem. You have the option of entering a lottery, for 1 currency unit (CU), that gives you a prize of 10 CU. The probability of winning is 0.01. This can be formalised by making it a choice between two probability distributions: P, where you do not enter the lottery and Q, which represents entering the lottery. Now we can calculate the expected utility for each choice. This is simply $\mathbb{E}(U \mid P) = \sum_r U(r)P(r)$ and $\mathbb{E}(U \mid Q) = \sum_r U(r)Q(r)$ respectively. Hence the utility of entering the lottery is -0.9, while it is 0 for not entering.

r	U(r)	P	Q
did not enter	0	1	0
paid 1 CU and lost	−1	0	0.99
paid 1 CU and won 10	9	0	0.01

Table 2.1: A simple gambling problem

Monetary rewards

Frequently, rewards come in the form of money. In general, it is assumed that people prefer to have more money than less money. However, the utility of additional money is not constant, i.e. 1,000 Euros are probably worth more to somebody with only 100 Euros in the bank than to somebody with 100,000 Euros. Hence, the utility of monetary rewards is generally assumed to be increasing, but not necessarily linear. Indeed, we would expect the utility of money to be concave. Nevertheless, we would in any case expect the behaviour of individuals to follow the tenets of expected utility theory. You should be able to verify this for following example for any increasing utility function U.

Example 10. Choose between the following two gambles:

1. The reward is 500,000 with certainty.
2. The reward is 2,500,000 with probability 0.10. It is 500,000 with probability 0.89, and 0 with probability 0.01.

Example 11. Choose between the following two gambles:

1. The reward is 500,000 with probability 0.11, or 0 with probability 0.89.
2. The reward is: 2,500,000 with probability 0.1, or 0 with probability 0.9.

Exercise 2. Show that if gamble 1 is preferred in the first example, gamble 1 must also be preferred in the second example, irrespective of the form of our utility function, under the expected utility hypothesis.

In practice, you may find that your preferences are not aligned with what this exercise suggests. This implies that either your decisions do not conform to the expected utility hypothesis, or that you are not internalising the given probabilities. We will explore this further in following example.

The St. Petersburg Paradox

The following simple example illustrates the fact that, internally, most humans do not seem to behave in ways that are not compatible with a linear utility for money. Ask yourself, or other classmates, how much money they would be willing to bet, in order to play the following game.

Example 12 (The St. Petersburg Paradox (Bernoulli, 1713)). In this game, we first pay k currency units, and then the *bank* tosses a fair coin repeatedly, until the coin comes up heads. Then the game ends and we obtain 2^n units, where n is the number of times the coin was thrown. So $n \in \{1, 2, \ldots, \infty\}$. The coin is *fair*, meaning that the probability of heads is always 1/2.

How many units k are you willing to pay, to play this game once?

As you can see below, the expected amount of money is infinite. First of all, the probability to stop at round n is 2^{-n}. Thus, the expected monetary gain of the game is

$$\sum_{n=1}^{\infty} 2^n 2^{-n} = \infty.$$

Were your utility function linear you'd be willing to pay any amount k to play, as the expected utility for playing the game is

$$\sum_{n=1}^{\infty} U(2^n - k)2^{-n} = \infty$$

for any finite k.

It would be safe to assume that very few readers would be prepared to pay any amount to play this game. One way to explain this is that the utility function used by the player is not necessarily linear. For example, if we also assume that the player has an initial capital C from which k has to be paid, we can consider a logarithmic utility function so that

$$\mathbb{E}\, U = \sum_{n=1}^{\infty} \ln(C + 2^n - k)2^{-n},$$

where C is our initial capital. In that case, for $C = 10,000$, the maximum bet is 14. For $C = 100$, the maximum bet is 6, while for $C = 10$, it is just 4.

There is another reason why one may not pay an abritrary amount to play this game. The player may not fully internalise the fact (or rather, the promise) that the coin is unbiased. Other explanations include whether you really believe that I can pay off an unbounded amount of money, or whether the sum only reaches up to some finite N. In the linear expected utility scenario, for a coin with probability p of coming heads, and sums only up to N, we have

$$\sum_{n=1}^{N} 2^n p^{n-1}(1-p) = 2(1-p)\frac{1-(2p)^N}{1-2p}.$$

For large N, it turns out that if $p = 0.45$, so slightly biased off heads, you should only expect about 10 dollars. But even if you believe the coin is fair, there is another possibility: if you think the *bank* only has a reserve of 1024 dollars, then again you should only bet up to 10 dollars. These are possible subjective beliefs that an individual might have that would influence their behaviour when dealing with a formally specified decision problem.

2.3.4 Measuring utility*

Since we cannot even rely on linear utility for money, we need to ask ourselves how we can measure the utility of different rewards. There are a number of ways, including trying to infer it from the actions of people. The simplest approach is to simply ask them to make even money bets. No matter what approach we use, however, we need to make some assumptions about the utility structure. This includes whether or not we should accept that the expected utility hypothesis holds for the observed human behaviour.

Experimental measurement of utility

EXAMPLE 13. We shall try and measure the utility of all monetary rewards in some interval $[a, b]$.

Let $\langle a, b\rangle$ denote a lottery ticket that yields a or b CU with equal probability. Consider the following sequence:

1. Find x_1 such that receiving x_1 CU with certainty is equivalent to receiving $\langle a, b\rangle$.
2. Find x_2 such that receiving x_2 CU with certainty is equivalent to receiving $\langle a, x_1\rangle$.
3. Find x_3 such that receiving x_3 CU with certainty is equivalent to receiving $\langle x_1, b\rangle$.
4. Find x_4 such that receiving x_4 CU with certainty is equivalent to receiving $\langle x_2, x_3\rangle$.

The above example algorithm allows us to measure the utility of money under the assumption that the expected utility hypothesis holds. However, if $x_1 \neq x_4$, then the preferences do not appear to meet the requirements of the expected utility hypothesis, which implies that $U(x_1) = U(x_4) = \frac{1}{2}(U(a) + U(b))$.

2.3.5 Convex and concave utility functions

As previously mentioned, utility functions of monetary rewards are not necessarily linear. In general, we'd say that a concave utility function implies risk aversion and a convex one risk taking. Intuitively, a risk averse person prefers a fixed amount of money to a random amount of money with the same expected value. A risk taker prefers to gamble. Let's start with the definition of a convex function.

Definition 2.3.3. A function $g : \Omega \to \mathbb{R}$, is convex on $A \subset \Omega$ if, for any points $x, y \in A$, and any $\alpha \in [0, 1]$:

$$\alpha g(x) + (1 - \alpha)g(y) \geq g\left(\alpha x + (1 - \alpha)y\right)$$

An important property of convex functions is that they are bounded from above by linear segments connecting their points. This property is formally given below.

Theorem 2.3.1 (Jensen's inequality)**.** *If g is convex on Ω and $x \in \Omega$ and P is a measure with $P(\Omega) = 1$ and $\mathbb{E}(x)$ and $\mathbb{E}[g(x)]$ exist, then:*

$$\mathbb{E}[g(x)] \geq g[\mathbb{E}(x)]. \tag{2.3.4}$$

EXAMPLE 14. If the utility function is convex, then we would prefer obtaining a random reward x rather than a fixed reward $y = \mathbb{E}(x)$. Thus, a convex utility function implies risk-taking. This is illustrated by Figure 2.1 which shows a linear function, x, a convex function, $e^x - 1$, and a concave function, $\ln(x + 1)$.

Definition 2.3.4. A function g is concave on Ω if, for any points $x, y \in \Omega$, and any $\alpha \in [0, 1]$:

$$\alpha g(x) + (1 - \alpha)g(y) \leq g[\alpha x + (1 - \alpha)y]$$

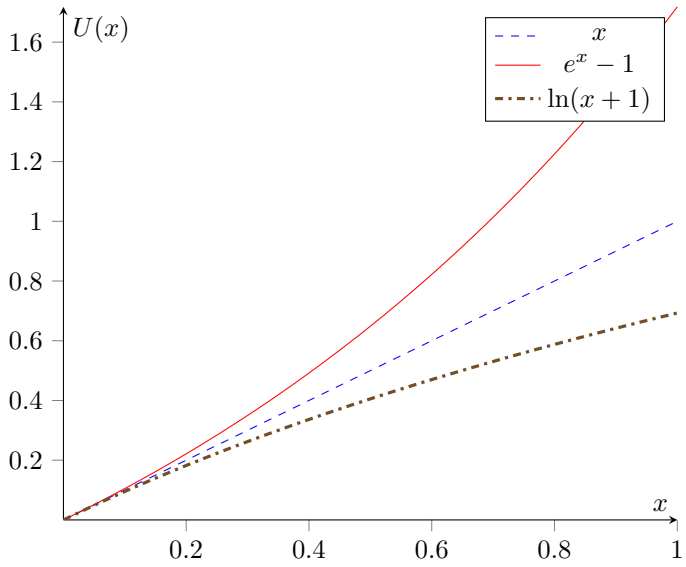


Figure 2.1: Linear, convex and concave functions

For concave functions, the inverse of Jensen's inequality holds (i.e. in the other direction). If the utility function is concave, then we choose a gamble giving a fixed reward $\mathbb{E}[x]$ rather than one giving a random reward x. Consequently, a concave utility function implies risk aversion. The act of buying insurance can be related to concavity of our utility function. Consider the following example, where we assume individuals are risk-averse, but insurance companies are risk-neutral.

EXAMPLE 15 (Insurance). Let x be the insurance cost, h our insurance cover, ϵ the probability of needing the cover, and U an increasing utility function (for monetary values). Then we are going to buy insurance if the utility of losing x with certainty is greater than the utility of losing $-h$ with probability ϵ.

$$U(-x) > \epsilon U(-h) + (1-\epsilon)U(0). \tag{2.3.5}$$

The company has a linear utility, and fixes the premium x high enough for

$$x > \epsilon h. \tag{2.3.6}$$

Consequently, we see from (2.3.6) that $U(-\epsilon h) \geq U(-x)$, as U is an increasing function. From (2.3.5) we obtain $U(-\epsilon h) > \epsilon U(-h) + (1-\epsilon)U(0)$. Now the $-\epsilon h$ term is the utility of our expected monetary loss, while the right hand side is our expected utility. Consequently if the inequality holds, our utility function is (at least locally) concave.

2.4 Exercises

EXERCISE 3. If $\mathcal{R}$ is our set of rewards, our utility function is $U : \mathcal{R} \to \mathbb{R}$ and $a \succ^* b$ iff $U(a) > U(b)$, then our preferences are transitive. Give an example of a utility function, not necessarily mapping to $\mathbb{R}$, and a binary relation $>$ such that transitivity can be violated. Back your example with a thought experiment.

EXERCISE 4. Assuming that U is increasing and absolutely continuous, consider the following experiment:

1. You specify an amount a, then observe random value Y.
2. If $Y \geq a$, you receive Y currency units.
3. If $Y < a$, you receive a random amount X with known distribution (independent of Y).

Show that we should choose a s.t. $U(a) = \mathbb{E}[U(X)]$.

EXERCISE 5. The usefulness of probability and utility.

- Would it be useful to separate randomness from uncertainty? What would be desirable properties of an alternative concept to probability?
- Give an example of how the expected utility assumption might be violated.

EXERCISE 6. Consider two urns, each containing red and blue balls. The first urn contains an equal number of red and blue balls. The second urn contains a *randomly* chosen proportion X of red balls, i.e. the probability of drawing a red ball from urn 2 is X.

1. Suppose that you were to select an urn, and then choose a random ball from that urn. If the ball is red, you win 1 CU, otherwise nothing. Show that: if your utility function is increasing with monetary gain, you should prefer urn 1 iff $\mathbb{E}(X) < \frac{1}{2}$.
2. Suppose that you were to select an urn, and then choose n random balls from that urn and that urn only. Each time you draw a red ball, you gain 1 CU. After you draw a ball, you put it back in the urn. Assume that the utility U is strictly concave and suppose that $\mathbb{E}(X) = \frac{1}{2}$. Show that you should always select balls from urn 1.

Hint: Show that for urn 2, $\mathbb{E}(U \mid x)$ is concave for $0 \leq x \leq 1$ (this can be done by showing $\frac{d^2}{dx^2}\mathbb{E}(U \mid x) < 0$). In fact,

$$\frac{d^2}{dx^2}\mathbb{E}(U \mid x) = n(n-1)\sum_{k=0}^{n-2}[U(k) - 2U(k+1) + U(k+2)]\binom{n-2}{k}x^k(1-x)^{n-2-k}.$$

Then apply Jensen's inequality.

EXERCISE 7. **Probability measures as a way to define likelihood relations.**

Show that a probability measure P on $(\Omega, \mathcal{F})$ satisfies the following: For any events $A, B \in \mathcal{F}$, one of the following holds: $P(A) > P(B)$, $P(B) > P(A)$ or $P(A) = P(B)$. If A_i, B_i are partitions of A, B such that for all $P(A_i) \leq P(B_i)$ for all i, then $P(A) \leq P(B)$. For any event A, $P(\emptyset) \leq P(A)$ and $P(\emptyset) < P(\Omega)$.

EXERCISE 8. **The definition of conditional probability**

Recall that $P(A \mid B) \triangleq \frac{P(A \cap B)}{P(B)}$ is only a definition. Give a plausible alternative that agrees with the basic properties of a probability measure. It helps if you see the conditional probability as a new probability measure $M_B(A) \triangleq P(A \mid B)$. The

properties are: (a) Null probability: $P(\emptyset \mid B) = 0$ (b) Total probability: $P(\Omega \mid B) = 1$ (c) Union of disjoint subsets: $P(A_1 \cup A_2 \mid B) = P(A_1 \mid B) + P(A_2 \mid B)$ (d) Conditional Probability: $P(A \mid D) \leq P(B \mid D)$ if and only if $P(A \cap D) \leq P(B \cap D)$.

“

Exercise 9 (30!). **Alternatives to the expected utility hypothesis** The expected utility hypothesis states that we prefer decision P over Q if and only if our expected utility under the distribution P is larger than that under Q, i.e. $\mathbb{E}_P(U) \geq \mathbb{E}_Q(U)$. Under what conditions do you think this is a reasonable hypothesis? Can you come up with a different rule for making decisions under uncertainty? Would it still satisfy the total order and transitivity properties of preference relations? In other words, could you still unambiguously say whether you prefer P to Q? If you had three choices, P, Q, W and you preferred P to Q and Q to W, would you always prefer P to W?

Exercise 10. **Rational Arthur-Merlin games.** You are Arthur, and you wish to pay Merlin to do a very difficult computation for you. More specifically, you perform a query $q \in Q$ and obtain an answer $r \in R$, from Merlin. However, there exists a unique correct answer $r^* = f(q)$. After he gives you the answer, you give Merlin a random amount of money m, depending on r, q such that $\mathbb{E}(m \mid r, q) = \sum_m mP(m \mid r, q)$ is maximised by the correct answer, i.e.

$$\mathbb{E}(m \mid r^*, q) > \mathbb{E}(m \mid r, q)$$

for any $r \neq r^*$. Assume that Merlin knows P and the function f.

Is this sufficient to incentivise Merlin to respond with the correct answer? If not, what other assumptions or knowledge do we require?

Exercise 11. Assume that you need to travel over the weekend. You wish to decide whether to take the train or take the car. Assume that the train and car trip cost exactly the same amount of money. The train trip takes 2 hours. If it does not rain, then the car trip takes 1.5 hour. However, if it rains the road becomes both more slippery and more crowded and so the average trip time is 2.5 hours. Assume that your utility function is equal to the negative amount of time spent travelling: $U(t) = -t$.

1. Let it be Friday. What is the expected utility of taking the car on Sunday? What is the expected utility of taking the train on Sunday? What is the Bayes-optimal decision, assuming you will travel on Sunday?
2. Let it be a rainy Saturday, i.e. that A holds. What is your posterior probability over the two weather stations, given that it has rained, i.e. $P(H_i \mid A)$? What is the new marginal probability of rain on Sunday, i.e. $P(B \mid A)$? What is now the expected utility of taking the car versus taking the train on Sunday? What is the Bayes-optimal decision?

Exercise 12. It is possible for the utility function to be nonlinear.

1. One example is $U(t) = 1/t$, which is a *convex* utility function. How would you interpret the utility in that case? Without performing the calculations, can you tell in advance whether your optimal decision can change? Verify your answer by calculating the expected utility of the two possible choices.
2. How would you model a problem where the objective involves arriving in time for a particular appointment?

Chapter 3

Decision problems

3.1 Introduction

In this chapter we describe how to formalise statistical decision problems. These involve making decisions whose utility depends on an unknown *state of the world*. In this setting, it is common to assume that the state of the world is a fundamental property that is not influenced by our decisions. However, we can calculate a probability distribution for the state of the world, using a prior belief and some data, and the data that we do obtain may depend on our decisions.

A classical application of this framework is parameter estimation. Therein, we stipulate the existence of a parameterised *law of nature*, and we wish to choose a best-guess set of parameters for the law through measurements and some prior information. An example would be determining the gravitational attraction constant from observations of planetary movements. These measurements are always obtained through experiments, the automatic design of which will be covered in later chapters.

The decisions we make will necessarily depend on both our prior belief and the data we obtain. In the last section of this chapter will examine how sensitive our decisions are to the prior, and how we can choose it so that our decisions are robust.

3.2 Rewards that depend on the outcome of an experiment

Consider the problem of choosing one of two different types of tickets in a raffle. Each type of ticket gives you the chance to win a different prize. The first is a bicycle and the second is a tea set. After n_i tickets are bought for the i-th prize, a number p_i is drawn uniformly from $\{1, \dots, n_i\}$ and the holder of that ticket wins that particular prize. Thus, the raffle guarantees that somebody will win either price. If most people opt for the bicycle, your chance of actually winning it by buying a single ticket is much smaller. However, if you prefer winning a bicycle to winning the tea set, it is not clear what choice you should make in the raffle. The above is the quintessential example for problems where the reward that we obtain depends not only on our decisions, but also in the outcome of an *experiment*.

This problem can be viewed more generally for scenarios where the reward you receive depends not only on your own choice, but also on some other, unknown fact in the world. This may be something completely uncontrollable, and hence you only can make an informed guess.

More formally, given a set of possible actions $\mathcal{A}$, we must make a decision $a \in \mathcal{A}$ *before* knowing the outcome ω of an experiment with outcomes in Ω. After the experiment is performed, we obtain a *reward* $r \in \mathcal{R}$ which depends on both the outcome ω of the experiment and our decision. As discussed in the previous chapter, our preferences for some rewards over others are determined by a *utility* function $U : \mathcal{R} \to \mathbb{R}$, such that we prefer r to r' if and only if $U(r) \geq U(r')$. Now, however, we cannot choose rewards directly. Another example, which will be used throughout this section, is the following.

Example 16 (Taking the umbrella). We must decide whether to take an umbrella to work. Our reward depends on whether we get wet and the amount of objects that we

carry. We would rather not get wet and not carry too many things, which can be made more precise by choosing an appropriate utility function. For example, we might put a value of -1 for carrying the umbrella and a value of -10 for getting wet. In this example, the only events of interest are whether it rains or not.

3.2.1 Formalisation of the problem setting

The elements we need to formulate the problem setting are a random variable, a decision variable, a reward function mapping the random and decision variable to a reward, and a utility function that says how much we prefer each reward.

Assumption 3.2.1 (Outcomes)**.** *There exists a probability measure P on $(\Omega, \mathcal{F}_\Omega)$ such that the probability of the random outcome ω being in $A \in \mathcal{F}_\Omega$ is*

$$\mathbb{P}(\omega \in A) = P(A). \tag{3.2.1}$$

The probability measure P is completely independent of any decision that we make.

Assumption 3.2.2 (Utilities)**.** *Given a set of rewards $\mathcal{R}$, our preferences satisfy Assumptions 2.1.1,2.1.2,2.1.3 , i.e. preferences are transitive, all rewards are comparable, and there exists a utility function U, measurable with respect to $\mathcal{F}_\mathcal{R}$ such that $U(r) \geq U(r')$ iff $r \succ^* r'$.*

Since the random outcome ω does not depend on our decision a, we must find a way to connect the two. This can be formalised via a reward function, so that the reward that we obtain (whether we get wet or not) depends on both our decision (to take the umbrella) and the random outcome (whether it rains).

Definition 3.2.1 (Reward function)**.** A reward function $\rho : \Omega \times \mathcal{A} \to \mathcal{R}$ defines the reward we obtain if we select $a \in \mathcal{A}$ and the experimental outcome is $\omega \in \Omega$:

$$r = \rho(\omega, a). \tag{3.2.2}$$

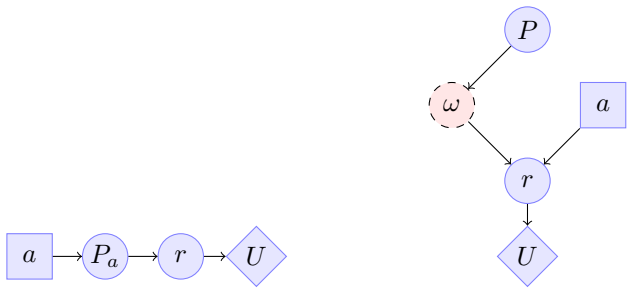
When we discussed the problem of choosing between distributions in Section 2.3.2, we had directly defined probability distributions on the set of rewards. We can now formulate our problem in that setting. First, we define a set of distributions $\{P_a \mid a \in \mathcal{A}\}$ on the reward space $(\mathcal{R}, \mathcal{F}_\mathcal{R})$, such that the decision a amounts to choosing a particular distribution P_a on the rewards.

Example 17 (Rock/Paper/Scissors). Consider a simple game of rock-paper-scissors, where your opponent plays a move at the same time as you, so that you cannot influence his move. The opponents moves are thus $\Omega = \{\omega_\mathrm{R}, \omega_\mathrm{P}, \omega_\mathrm{S}\}$.

You have studied your opponent for some time and you *believe* that he is most likely to play rock $P(\omega_\mathrm{R}) = 3/6$, somewhat likely to play paper $P(\omega_\mathrm{P}) = 2/6$, and less likely to play scissors: $P(\omega_\mathrm{S}) = 1/6$. Your decision set is your own moves: $\mathcal{A} = \{a_\mathrm{R}, a_\mathrm{P}, a_\mathrm{S}\}$, for rock, paper, scissors, respectively. The reward set is $\mathcal{R} = \{\mathrm{Win}, \mathrm{Draw}, \mathrm{Lose}\}$.

What is the probability of each reward, for each decision you make? Taking the example of a_R, we see that you win if the opponent plays scissors with probability $1/6$, you lose if the opponent plays paper ($2/6$), and you draw if he plays rock ($3/6$). Consequently, we can convert the outcome probabilities to reward probabilities for every decision:

$$\begin{aligned}
&P_{a_\mathrm{R}}(\mathrm{Win}) = 1/6, && P_{a_\mathrm{R}}(\mathrm{Draw}) = 3/6, && P_{a_\mathrm{R}}(\mathrm{Lose}) = 2/6\\
&P_{a_\mathrm{P}}(\mathrm{Win}) = 3/6, && P_{a_\mathrm{P}}(\mathrm{Draw}) = 2/6, && P_{a_\mathrm{P}}(\mathrm{Lose}) = 1/6\\
&P_{a_\mathrm{S}}(\mathrm{Win}) = 2/6, && P_{a_\mathrm{S}}(\mathrm{Draw}) = 1/6, && P_{a_\mathrm{S}}(\mathrm{Lose}) = 3/6.
\end{aligned}$$



(a) The combined decision problem

(b) The separated decision problem

Figure 3.1: Decision diagrams for the combined and separated formulation of the decision problem. Squares denote decision variables, diamonds denote utilities. All other variables are denoted by circles. Arrows denote the flow of dependency.

Of course, what you play depends on our own utility function. If we prefer winning over drawing or losing, we could for example have the utility function $U(\text{Win}) = 1$, $U(\text{Draw}) = 0$, $U(\text{Lose}) = -1$. Then, since $\mathbb{E}_a U = \sum_{\omega \in \Omega} U(\omega, a) P_a(\omega)$, we have

$$\begin{aligned} E_{a_\text{R}} U &= -1/6 \\ E_{a_\text{P}} U &= 2/6 \\ E_{a_\text{S}} U &= -1/6, \end{aligned}$$

so that based on your belief, choosing paper is best.

The above example illustrates that every decision that we make creates a corresponding probability distribution on rewards. While the outcome of the experiment is independent of the decision, the distribution of rewards is effectively chosen by our decision.

Expected utility

The expected utility of any decision $a \in \mathcal{A}$ under P is:

$$\mathbb{E}_{P_a}(U) = \int_{\mathcal{R}} U(r) \, \mathrm{d}P_a(r) = \int_{\Omega} U[\rho(\omega, a)] \, \mathrm{d}P(\omega) \tag{3.2.3}$$

From now on, we shall use the simple notation

$$U(P, a) \triangleq \mathbb{E}_{P_a} U \tag{3.2.4}$$

to denote the expected utility of a under distribution P.

Instead of viewing the decision as effectively choosing a distribution over rewards (Fig. 3.1(a)) we can separate the random part of the process from the deterministic part (Fig. 3.1(b)) by considering a measure P on some space of outcomes Ω, such that the reward depends on both a and the outcome $\omega \in \Omega$ through the reward function $\rho(\omega, a)$. The optimal decision is of course always the $a \in \mathcal{A}$ maximising $\mathbb{E}(U \mid P_a)$. However, this structure allows us to clearly distinguish the controllable from the random part of the rewards.

The probability measure induced by decisions
For every $a \in \mathcal{A}$, the function $\rho : \Omega \times \mathcal{A} \to \mathcal{R}$ induces a probability distribution P_a on $\mathcal{R}$. In fact, for any $B \in \mathcal{F}_{\mathcal{R}}$:

$$P_a(B) \triangleq \mathbb{P}(\rho(\omega, a) \in B) = P(\{\omega \mid \rho(\omega, a) \in B\}). \tag{3.2.5}$$

The above equation requires that the following technical assumption is satisfied. As usual, we employ the expected utility hypothesis (Assumption 2.3.2). Thus, we should choose the decision that results in the highest expected utility.

Assumption 3.2.3. *The sets $\{\omega \mid \rho(\omega, a) \in B\}$ must belong to $\mathcal{F}_\Omega$. That is, ρ must be $\mathcal{F}_\Omega$-measurable for any a.*

The dependency structure of this problem in either formulation can be visualised in the *decision diagram* shown in Figure 3.1.

Example 18 (Continuation of Example 16). You are going to work, and it might rain. The forecast said that the probability of rain (ω_1) was 20%. What do you do?

- a_1: Take the umbrella.
- a_2: Risk it!

The reward of a given outcome and decision combination, as well as the expected utility is given in Table 3.1.

$\rho(\omega, a)$	a_1	a_2
ω_1	dry, carrying umbrella	wet
ω_2	dry, carrying umbrella	dry
$U[\rho(\omega, a)]$	a_1	a_2
ω_1	-1	-10
ω_2	-1	1
$\mathbb{E}_P(U \mid a)$	-1	-1

Table 3.1: Rewards, utilities, expected utility for 20% probability of rain.

3.2.2 Decision diagrams

Decision diagrams are also known as *decision networks* or *influence diagrams*. Like the examples shown in Figure 3.1, they are used to show dependencies between different variables. In general, these include the following types of nodes:

- Choice nodes, denoted by squares. These are nodes whose values the decision maker can directly choose. Sometimes there is more than one decision maker involved.
- Value nodes, denoted by diamonds. These are the nodes that the decision maker is interested in influencing. The utility of the decision maker is always a function of the value nodes.

- Circle nodes are used to denote all other types of variables. These include deterministic, stochastic, known or unknown variables.

The nodes are connected via directed edges. These denote the dependencies between nodes. For example, in Figure 3.1(b), the reward is a function of both ω and a, i.e. $r = \rho(\omega, a)$, while ω depends only on the probability distribution P. Typically, there must be a path from a choice node to a value node, otherwise nothing the decision maker can do will influence its utility. Nodes belonging to or observed by different players will usually be denoted by different lines or colors. In Figure 3.1(b), ω, which is not observed, is shown in a lighter color.

3.2.3 Statistical estimation*

Statistical decision problems arise particularly often in *parameter estimation,* such as estimating the covariance matrix of a Gaussian random variable. In this setting, the unknown outcome of the experiment ω is called a *parameter*, while the set of outcomes Ω is called the *parameter space*. Classical statistical estimation involves selecting a single parameter value on the basis of observations. This requires us to specify a preference for different types of estimation errors, and is distinct from the standard Bayesian approach to estimation, which simply calculates a full distribution over all possible parameters.

A simple example is estimating the distribution of votes in an election from a small sample. Depending on whether we are interested in predicting the vote share of individual parties or the most likely winner of the election, we can use a distribution over vote shares (possibly estimated through standard Bayesian methodology) to decide on a share or the winner.

EXAMPLE 19 (Voting). Assume you wish to estimate the number of votes for different candidates in an election. The *unknown parameters* of the problem mainly include: the percentage of likely voters in the population, the probability that a likely voter is going to vote for each candidate. One simple way to estimate this is by polling.

Consider a nation with k political parties. Let $\omega = (\omega_1, \dots, \omega_k) \in [0,1]^k$ be the voting proportions for each party. We wish to make a guess $a \in [0,1]^k$. How should we guess, given a distribution $P(\omega)$? How should we select U and ρ? This depends on what our goal is, when we make the guess.

If we wish to give a reasonable estimate about the votes of all the k parties, we can use the squared error: First, set the error vector $r = (\omega_1 - a_1, \dots, \omega_k - a_k) \in [0,1]^k$. Then we set $U(r) \triangleq -\|r\|^2$, where $\|r\|^2 = \sum_i |\omega_i - a_i|^2$.

If on the other hand, we just want to predict the winner of the election, then the actual percentages of all individual parties are not important. In that case, we can set $r = 1$ if $\arg\max_i \omega_i = \arg\max_i a_i$ and 0 otherwise, and $U(r) = r$.

Losses and risks

In such problems, it is common to specify a loss instead of a utility. This is usually the negative utility:

Definition 3.2.2 (Loss)**.**

$$\ell(\omega, a) = -U[\rho(\omega, a)]. \tag{3.2.6}$$

Given the above, instead of the expected utility, we consider the expected loss, or risk.

Definition 3.2.3 (Risk)**.**

$$\kappa(P, a) = \int_{\Omega} \ell(\omega, a) \, \mathrm{d}P(\omega). \tag{3.2.7}$$

Of course, the optimal decision is a minimising κ.

3.3 Bayes decisions

The decision which maximises the expected utility under a particular distribution P, is called the *Bayes-optimal* decision, or simply the *Bayes decision*. The probability distribution P is supposed to reflect all our uncertainty about the problem.

Definition 3.3.1 (Bayes-optimal utility)**.** Consider an outcome (or parameter) space Ω, decision space $\mathcal{A}$, and a utility function $U : \Omega \times \mathcal{A} \to \mathbb{R}$. For any probability distribution P on Ω, the Bayes-optimal utility $U^*(P)$ is defined as the smallest upper bound on $U(P, a)$ for all decisions $a \in \mathcal{A}$. That is,

$$U^*(P) = \sup_{a \in \mathcal{A}} U(P, a). \tag{3.3.1}$$

The maximisation over decision is usually not easy. However, there exist a few cases where it is relatively simple. The first of those is when the utility function is the negative squared error.

Example 20 (Quadratic loss). Consider $\Omega = \mathbb{R}^k$ and $\mathcal{A} = \mathbb{R}^k$. The utility function that, for any point $\omega \in \mathbb{R}$, is defined as

$$U(\omega, a) = -\|\omega - a\|^2 \tag{3.3.2}$$

is called quadratic loss.

Quadratic loss is a very important special case of utility functions, as it is easy to calculate the optimal solution. This is illustrated by the following theorem.

Theorem 3.3.1. *Let P be a measure on Ω and $U : \Omega \times \mathcal{A} \to \mathbb{R}$ be the quadratic loss defined in Example 20. Then the decision*

$$a = \mathbb{E}_P(\omega), \tag{3.3.3}$$

maximises the expected utility $U(P, a)$, under the technical assumption that $\partial/\partial a |\omega - a|^2$ is measurable with respect to $\mathcal{F}_{\mathbb{R}}$.

Proof. We first write out the expected utility of a decision a.

$$U(P, a) = -\int_{\Omega} \|\omega - a\|^2 \, \mathrm{d}P(\omega).$$

We now take derivatives – due to the measurability assumption, we can swap the order of differentiation and integration:

$$\begin{aligned}\frac{\partial}{\partial a}\int_\Omega \|\omega - a\|^2 \,\mathrm{d}P(\omega) &= \int_\Omega \frac{\partial}{\partial a}\|\omega - a\|^2 \,\mathrm{d}P(\omega)\\ &= 2\int_\Omega (a-\omega)\,\mathrm{d}P(\omega)\\ &= 2\int_\Omega a\,\mathrm{d}P(\omega) - 2\int_\Omega \omega\,\mathrm{d}P(\omega)\\ &= 2a - 2\,\mathbb{E}(\omega).\end{aligned}$$

Setting the derivative equal to 0 and noting that the utility is concave, we see that the expected utility is maximised for $a = \mathbb{E}_P(\omega)$. □

Another simple example is the absolute error, where $U(\omega, a) = |\omega - a|$. The solution in this case differs significantly from the squared error. As can be seen from Figure 3.2(a), for absolute loss, the optimal decision is to choose the a that is closest to the most likely ω. Figure 3.2(b) illustrates the finding of Theorem 3.3.1.

3.3.1 Convexity of the Bayes-optimal utility*

Although finding the optimal decision for an arbitrary utility U and distribution P may be difficult, fortunately the Bayes-optimal utility has some nice properties which enable it to be approximated rather well. In particular, for any decision, the expected utility is linear with respect to our belief P. Consequently, the Bayes-optimal utility is convex with respect to P. This firstly implies that there is a unique "worst" distribution P, against which we cannot do very well. Secondly, we can approximate the Bayes-utility very well for all possible distributions by generalising from a small number of distributions. In order to define linearity and convexity, we first introduce the concept of a mixture of distributions.

mixture of distributions

Consider two probability measures P, Q on $(\Omega, \mathcal{F}_\Omega)$. These define two alternative distributions for ω. For any P, Q and $\alpha \in [0, 1]$, we define the *mixture of distributions*

$$Z_\alpha \triangleq \alpha P + (1-\alpha)Q \tag{3.3.4}$$

to mean the probability measure such that $Z_\alpha(A) = \alpha P(A) + (1-\alpha)Q(A)$ for any $A \in \mathcal{F}_\Omega$. For any fixed choice a, the expected utility varies linearly with α:

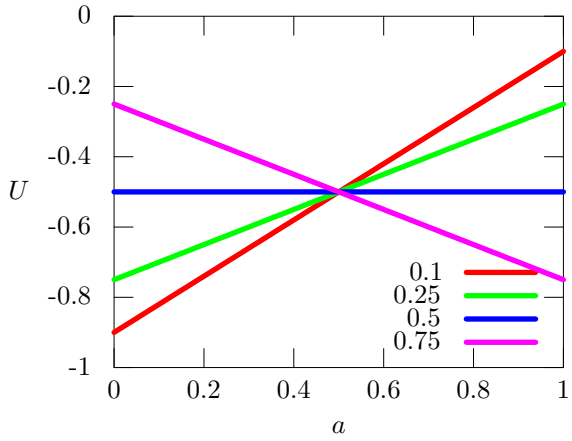
Remark 3.3.1 (Linearity of the expected utility). If Z_α is as defined in (3.3.4), then, for any $a \in \mathcal{A}$:

$$U(Z_\alpha, a) = \alpha U(P, a) + (1-\alpha)U(Q, a). \tag{3.3.5}$$

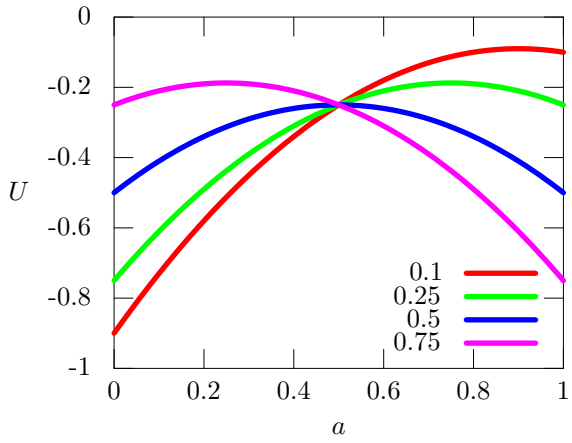
Proof.

$$\begin{aligned}U(Z_\alpha, a) &= \int_\Omega U(\omega, a)\,\mathrm{d}Z_\alpha(\omega)\\ &= \alpha\int_\Omega U(\omega, a)\,\mathrm{d}P(\omega) + (1-\alpha)\int_\Omega U(\omega, a)\,\mathrm{d}Q(\omega)\\ &= \alpha U(P, a) + (1-\alpha)U(Q, a).\end{aligned}$$

□



(a) Absolute error



(b) Quadratic error

Figure 3.2: Expected utility curves for different values of ω, as the decision a varies in $[0, 1]$.

However, if we consider Bayes-optimal decisions, this is no longer true, because the optimal decision depends on the distribution. In fact, the utility of Bayes-optimal decisions is convex, as the following theorem shows.

Theorem 3.3.2. *For probability measures P, Q on Ω and any $\alpha \in [0, 1]$,*

$$U^*[Z_\alpha] \leq \alpha U^*(P) + (1 - \alpha)U^*(Q), \tag{3.3.6}$$

where $Z_\alpha = \alpha P + (1 - \alpha)Q$.

Proof. From the definition of the expected utility (3.3.5), for any decision $a \in \mathcal{A}$,

$$U(Z_\alpha, a) = \alpha U(P, a) + (1 - \alpha)U(Q, a).$$

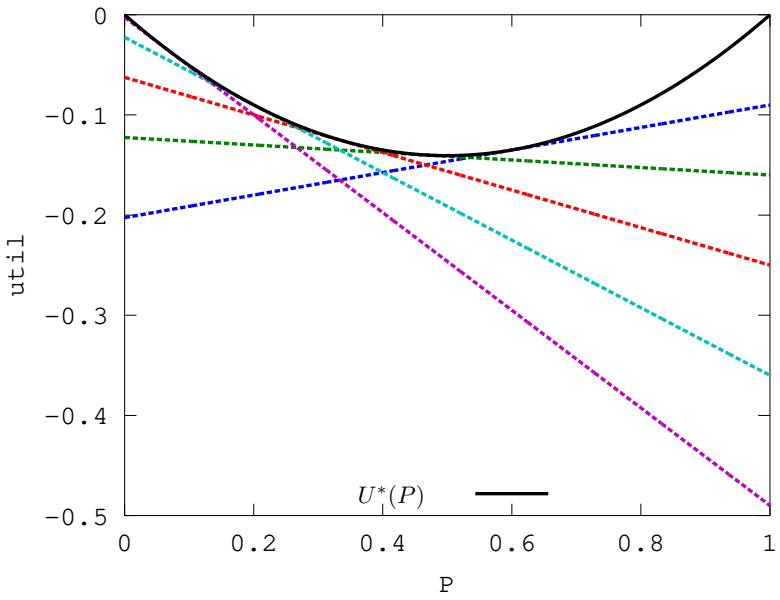


Figure 3.3: A strictly convex Bayes utility.

Hence, by definition (3.3.1) of the Bayes-utility:

$$\begin{aligned} U^*(Z_\alpha) &= \sup_{a \in \mathcal{A}} U(Z_\alpha, a) \\ &= \sup_{a \in \mathcal{A}} [\alpha U(P, a) + (1-\alpha) U(Q, a)]. \end{aligned}$$

As $\sup_x [f(x) + g(x)] \leq \sup_x f(x) + \sup_x g(x)$, we obtain:

$$\begin{aligned} U^*[Z_\alpha] &\leq \alpha \sup_{a \in \mathcal{A}} U(P, a) + (1-\alpha) \sup_{a \in \mathcal{A}} U(Q, a) \\ &= \alpha U^*(P) + (1-\alpha) U^*(Q). \end{aligned}$$

□

As we have proven, the expected utility is linear with respect to Z_α. Thus, for any fixed action a we obtain one of the lines in Fig. 3.3. Due to the theorem just proved, the Bayes-optimal utility is convex. Furthermore, the minimising decision for any Z_α is tangent to the Bayes-optimal utility at the point $(Z_\alpha, U^*(Z_\alpha))$. If we take a decision that is optimal with respect to some Z, but the distribution is in fact $Q \neq Z$, then we are not far from the optimal, Q, Z are close and U^* is smooth. Consequently, we can trivially lower bound the Bayes utility by examining any arbitrary finite set of decisions $\hat{\mathcal{A}}$:

$$U^*(P) \geq \max_{a \in \hat{\mathcal{A}}} U(P, a),$$

for any probability distribution P on Ω. In addition, we can upper-bound the Bayes utility as follows. Take any two distributions P_1, P_2 over Ω. Then, the following upper bound

$$U^*(\alpha P_1 + (1-\alpha) P_2) \leq \alpha U^*(P_1) + (1-\alpha) U^*(P_2)$$

holds due to convexity. The two bounds suggest an algorithm for successive approximation of the Bayes-optimal utilty, by looking for the largest gap between the lower and the upper bounds.

3.4 Statistical and strategic decision making

We do not need to be restricted to simply choosing one of a finite number of decisions. For example, we could choose a distribution over decisions. In addition, we may wish to consider other criteria than maximising expected utility / minimising risk.

Strategies Instead of choosing a specific decision, we could instead choose to randomise our decision somehow. In other words, instead of our choices being specific decisions, we can choose among distributions over decisions. For example, instead of choosing to eat lasanga or beef, we choose to throw a coin and eat lasagna if the coin comes heads and beef otherwise.

Definition 3.4.1 (Strategies)**.** A strategy σ is a probability measure on $\mathcal{A}$ such that $\sigma(A)$ is the probability that we select a decision $a \in A \subseteq \mathcal{A}$.

Interestingly, *for the type of problems that we have considered so far*, even if we expand our choices to the set of all possible probability measures on $\mathcal{A}$, there always is one decision (rather than a strategy) which is optimal. In the following we remove the reward function ρ from the decision problem, summarising everything with the utility function U for simplicity.

Theorem 3.4.1. *Consider any statistical decision problem with probability measure P on outcomes Ω and with utility function $U : \Omega \times \mathcal{A} \to \mathbb{R}$. Further let $a^* \in \mathcal{A}$ such that $U(P, a^*) \geq U(P, a)$ for all $a \in \mathcal{A}$. Then for any probability measure σ on $\mathcal{A}$,*

$$U(P, a^*) \geq U(P, \sigma).$$

Proof.

$$\begin{aligned} U(P, \sigma) &= \int_{\mathcal{A}} U(P, a) \, \mathrm{d}\sigma(a) \\ &\leq \int_{\mathcal{A}} U(P, a^*) \, \mathrm{d}\sigma(a) \\ &= U(P, a^*) \int_{\mathcal{A}} \mathrm{d}\sigma(a) = U(P, a^*) \end{aligned}$$

□

This theorem should be not be applied naively. It only states that if we know P then the expected utility of the best fixed/deterministic decision $a^* \in \mathcal{A}$ cannot be increased by randomising between decisions.

For example, it does not make sense to apply this theorem to cases where P itself is unknown. This can happen in two cases. The first is when P is chosen by somebody else, analogously to how we choose σ, and its value remains hidden to us. The second is when P is only known partially.

$U(\omega, a)$	a_1	a_2
ω_1	-1	0
ω_2	10	1
$\mathbb{E}(U \mid P, a)$	*4.5*	0.5
$\min_\omega U(\omega, a)$	-1	*0*

Table 3.2: Utility function, expected utility and maximin utility.

3.4.1 Alternative notions of optimality

There are some situations where maximising expected utility with respect to the distribution on outcomes is unnatural. Two simple examples—where, for simplicity, we consider utility functions $U : \Omega \times \mathcal{A} \to \mathbb{R}$ on outcomes and decisions directly—are the following.

Maximin/Minimax policies. These policies are useful when we have no information about ω. In that case, we may want to take a worst-case approach and select a^* that maximises the utility in the worst-case ω.

$$U_* = \max_a \min_\omega U(\omega, a) = \min_\omega U(\omega, a^*) \tag{maximin}$$

The maximin value of the problem can essentially be seen as how much utility we would be able to obtain, if we were to make a decision a first, and nature were to select an adversarial decision ω later. On the other hand, the minimax value is:

$$U^* = \min_\omega \max_a U(\omega, a) = \max_a U(\omega^*, a), \tag{minimax}$$

where $\omega^* \triangleq \arg\min_\omega \max_a U(\omega, a)$ is the worst-case choice nature could make, if we were to select our own decision a after its own choice was revealed to us.
To illustrate this, consider Table 3.2. Here, we see that a_1 maximises expected utility. However, under a worst-case assumption this is not the case, i.e. the maximin solution is a_2. Note that by definition

$$U^* \geq U(\omega^*, a^*) \geq U_*. \tag{3.4.1}$$

Maximin/minimax problems are a special case of problems in game theory, in particular two-player zero-sum games. The minimax problem can be seen as a game where the maximising player plays first, and the minimising player second. If $U^* = U_*$, then the game is said to have a value, which implies that if both players are playing optimal, then it doesn't matter which player moves first. More details about these types of problems will be given in Section 3.4.2.

Regret. Instead of calculating the expected utility for each possible decision, we could instead calculate how much utility we would have obtained if we had made the best decision in hindsight. Consider, for example the problem in Table 3.2. There the optimal action is either a_1 or a_2, depending on whether we accept the probability P over Ω, or adopt a worst-case approach. However, after we make a specific decision, we can always look at the best decision we could have made given the actual outcome ω, as shown in Table 3.3.

$L(\omega, a)$	a_1	a_2
ω_1	1	0
ω_2	0	9
$\mathbb{E}(L \mid P, a)$	*0.5*	4.5
$\max_\omega L(\omega, a)$	*1*	9

Table 3.3: Regret, in expectation and minimax.

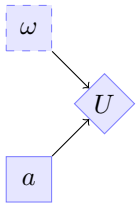


Figure 3.4: Simultaneous two-player stochastic game. The first player (nature) chooses ω, and the second player (the decision maker) chooses a. Then the second player obtains utility $U(\omega, a)$.

Definition 3.4.2 (Regret)**.** The regret of σ is how much we lose compared to the best decision in hindsight, that is,

$$L(\omega, a) \triangleq \max_{a'} U(\omega, a') - U(\omega, a). \tag{3.4.2}$$

The notion of regret is given in Table 3.3, which reuses Example 16. Here, the decision maker has a choice betwen two actions, while nature has a choice between two outcomes. We can see that the choice minimising regret either in expectation or in the minimax sense is a_1. This is in contrast to what occurs when we are considering utility. Given the regret of each action-outcome pair, we can now find the decision minimising expected regret $\mathbb{E}(L \mid P, a)$ and minimising maximum regret $\max_\omega L(\omega, a)$, analogously to expected utility and minimax utility. Interestingly, as Table 3.3 shows, in this setting we always prefer action a_2 to a_1, showing that the concept of regret results in quantitatively different decisions.

3.4.2 Solving minimax problems*

We now view minimax problems as two player games, where one player chooses a and the other player chooses ω. The decision diagram for this problem is given in Figure 3.4.2, where the dashed line indicates that, from the point of view of the decision maker, nature's choice is unobserved before she makes her own decision. A simultaneous two-player game is a game where both players act without knowing each other's decision. From the point of view of the player that chooses a, this is equivalent to assuming that ω is hidden, as shown in Figure 3.4.2. There are other variations of such games, however. For example, their moves may still be revealed after they have played. This is important in the case where the game is played *repeatedly*. However, what is usually revealed is not the belief ξ, which is something assumed to be internal to player one, but ω, the actual decision made by the first player. In other cases, we might have that U itself is not known, and we only observe $U(\omega, a)$ for the choices made.

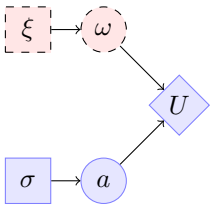


Figure 3.5: Simultaneous two-player stochastic game. The first player (nature) chooses ξ, and the second player (the decision maker) chooses σ. Then $\omega \sim \xi$ and $a \sim \sigma$ and the second player obtains utility $U(\omega, a)$.

Definition 3.4.3 (Strategy)**.** A strategy $\sigma \in \mathbb{\Delta}\mathcal{A}$, is a probability distribution over simple decisions $a \in \mathcal{A}$.

In this setting, by allowing the decision maker to select arbitrary strategies σ, we permit her to select arbitrary probability distributions over simple decisions, rather than fixing one decision.

Minimax utility, regret and loss If the decision maker knows the outcome, then the additional flexibility does not help. As we showed for the general case of a distribution over Ω, a simple decision is as good as any randomised strategy:

Remark 3.4.1. For each ω, there is some a such that:

$$U(\omega, a) \in \max_{\sigma \in \mathbb{\Delta}\mathcal{A}} U(\omega, \sigma). \tag{3.4.3}$$

What follows are some rather trivial remarks connecting regret with utility in various cases.

Remark 3.4.2.

$$L(\omega, \sigma) = \sum_a \sigma(a) L(w, a) \geq 0, \tag{3.4.4}$$

with equality iff σ is ω-optimal.

Proof.

$$\begin{aligned} L(w, \sigma) &= \max_{\sigma'} U(\omega, \sigma') - U(\omega, \sigma) = \max_{\sigma'} U(\omega, \sigma') - \sum_a \sigma(a) U(\omega, a) \\ &= \sum_a \sigma(a) \left(\max_{\sigma'} U(\omega, \sigma') - U(\omega, a) \right) \geq 0. \end{aligned}$$

The equality on optimality is obvious. □

Remark 3.4.3.

$$L(\omega, \sigma) = \max_a U(\omega, a) - U(\omega, \sigma). \tag{3.4.5}$$

Proof. As (3.4.3) shows, for any fixed ω, the best decision is always deterministic,

$$\sum_a \sigma(a) L(\omega, a) = \sum_a \sigma(a) [\max_{a' \in \mathcal{A}} U(\omega, a') - U(\omega, a)] = \max_{a' \in \mathcal{A}} U(\omega, a') - \sum_a \sigma(a) U(\omega, a).$$

□

U	ω_1	ω_2
a_1	1	-1
a_2	0	0

Table 3.4: Even-bet utility

Remark 3.4.4. $L(\omega, \sigma) = -U(\omega, \sigma)$ iff $\max_a U(\omega, a) = 0$.

Proof. If $\max_{\sigma'} U(\omega, \sigma') - U(\omega, \sigma) = -U(\omega, \sigma)$ then $\max_{\sigma'} U(\omega, \sigma') = \max_a U(\omega, a) = 0$. The converse follows trivially. □

EXAMPLE 21. (An even-money bet) For this problem, the maximum regret of a policy

L	ω_1	ω_2
a_1	0	1
a_2	1	0

Table 3.5: Even-bet regret

σ can be written as

$$\max_\omega L(\omega, \sigma) = \max_\omega \sum_a \sigma(a) L(\omega, a) = \max_\omega \sigma(a) \mathbb{I}\{\{\} a = a_i \wedge \omega \neq \omega_i) \cdot 1 \geq 1/2, \tag{3.4.6}$$

since $L(a, \omega) = 0$ when $a = a_i$ and $\omega \neq \omega_i$. In fact, equality is obtained iff $\sigma(a) = 1/2$, giving minimax regret $L^* = 1/2$.

3.4.3 Two-player games

Here we go into some more detail in the connections between minimax theory and the theory of two-player games. In particular, we extend the actions of nature to $\mathbb{\Delta}(\Omega)$, the distributions over Ω and our actions to distributions $\mathbb{\Delta}(\mathcal{A})$, the distributions over $\mathcal{A}$.

For two distributions σ, ξ on $\mathcal{A}$ and Ω, define our expected utility to be:

$$U(\xi, \sigma) \triangleq \sum_{\omega \in \Omega} \sum_{a \in \mathcal{A}} U(\omega, a) \xi(\omega) \sigma(a). \tag{3.4.7}$$

Then we define the maximin policy σ^* for which:

$$\min_\xi U(\xi, \sigma^*) = U_* \triangleq \max_\sigma \min_\xi U(\xi, \sigma), \tag{3.4.8}$$

The minimax prior ξ^* satisfies

$$\max_\sigma U(\xi^*, \sigma) = U^* \triangleq \min_\xi \max_\sigma U(\xi, \sigma), \tag{3.4.9}$$

where the solution exists as long as $\mathcal{A}, \Omega$ are finite, which we will assume in the following.

Expected regret
We can now define the expected regret for a given pair of distributions ξ, σ as

$$\begin{aligned} L(\xi, \sigma) &= \max_{\sigma'} \sum_{\omega} \xi(\omega) \left\{ U(\omega, \sigma') - U(\omega, \sigma) \right\} \\ &= \max_{\sigma'} U(\xi, \sigma') - U(\xi, \sigma). \end{aligned} \tag{3.4.10}$$

Not all minimax and maximin policies result in the same value. The following theorem gives a condition under which the game does have a value.

Theorem 3.4.2. *If there exist (perhaps singular) distributions ξ^*, σ^* and $C \in \mathbb{R}$ such that*

$$U(\xi^*, \sigma) \leq C \leq U(\xi, \sigma^*) \qquad \forall \xi, \sigma$$

then

$$U^* = U_* = U(\xi^*, \sigma^*) = C.$$

Proof. Since $C \leq U(\xi, \sigma^*)$ for all ξ we have

$$C \leq \min_{\xi} U(\xi, \sigma^*) \leq \max_{\sigma} \min_{\xi} U(\xi, \sigma) = U_*.$$

Similarly

$$C \geq \max_{\sigma} U(\xi^*, \sigma) \geq \min_{\xi} \max_{\sigma} U(\xi, \sigma) = U^*.$$

But then due to (3.4.1)

$$C \geq U^* \geq U_* \geq C.$$

□

One question is whether a solution exists, and if so we can find it. In fact, the type of games we have been looking at so far are called bilinear games. For these, a solution always exists and there are efficient methods for finding it.

Definition 3.4.4. A bilinear game is a tuple $(U, \Xi, \Sigma, \Omega, \mathcal{A})$ with $U : \Xi \times \Sigma \to \mathbb{R}$ such that all $\xi \in \Xi$ are arbitrary distributions on Ω and all $\sigma \in \Sigma$ are arbitrary distributions on $\mathcal{A}$:

$$U(\xi, \sigma) \triangleq \mathbb{E}(U \mid \xi, \sigma) = \sum_{\omega, a} U(\omega, a) \sigma(a) \xi(\omega).$$

Theorem 3.4.3. *For a bilinear game, $U^* = U_*$. In addition, the following three conditions are equivalent:*

1. *σ^* is maximin, ξ^* is minimax and $U^* = C$.*
2. *$U(\xi, \sigma^*) \geq C \geq U(\xi^*, \sigma)$ for all ξ, σ.*
3. *$U(\omega, \sigma^*) \geq C \geq U(\xi^*, a)$ for all ω, a.*

Linear programming formulation

While general games may be hard, bilinear games are easy, in the sense that minimax solutions can be found with well-known algorithms. One example is linear programming. The problem

$$\max_{\sigma} \min_{\xi} U(\xi, \sigma),$$

where ξ, σ are distributions over finite domains, can be converted to finding σ corresponding to the greatest lower bound $v_\sigma \in \mathbb{R}$ on the utility. Using matrix notation, set $\boldsymbol{U}$ to be the matrix such that $\boldsymbol{U}_{\omega,a} = U(\omega, a)$, $\boldsymbol{\pi}(a) = \sigma(a)$ and $\boldsymbol{\xi}(\omega) = \xi(\omega)$. Then the problem can be written as:

$$\max \left\{ v_\sigma \;\middle|\; (\boldsymbol{U}\boldsymbol{\pi})_j \geq v_\sigma \forall j,\ \sum_i \sigma_i = 1,\ \sigma_i \geq 0 \forall i \right\}.$$

Equivalently, we can find ξ with the least upper bound:

$$\min \left\{ v_\xi \;\middle|\; (\boldsymbol{\xi}^\top \boldsymbol{U})_i \leq v_\xi \forall i,\ \sum_j \xi_j = 1,\ \xi_j \geq 0 \forall j \right\},$$

where everything has been written in matrix form. In fact, one can show that $v_\xi = v_\sigma$, thus obtaining Theorem 3.4.3.

To understand the connection of two-person games with Bayesian decision theory, take a look at Figure 3.3, seeing the risk as negative expected utility, or as the opponent's gain. Each of the decision lines represents nature's gain as she chooses different prior distributions, while we keep our policy σ fixed. The bottom horizontal line that would be tangent to the Bayes-optimal utility curve would be minimax: if nature were to change priors, since the line is horizontal, it would not increase its gain. On the other hand, if we were to choose another tangent line, we would only increase nature's gain (and decrease our utility).

3.5 Decision problems with observations

So far we have only examined problems where the outcomes were drawn from some fixed distribution. This distribution constituted our subjective belief about what the unknown parameter is. Now, we examine the case where we can obtain some observations that depend on the unknown ω before we make our decision. These observations should give us more information about ω, before making a decision. Intuitively, we should be able to make decisions by simply considering the posterior distribution.

In this setting, we once more need to take some decision $a \in \mathcal{A}$ so as to maximise expected utility. As before, we have a prior distribution ξ on some parameter $\omega \in \Omega$, representing what we know about ω. Consequently, the expected utility of any fixed decision a is going to be $\mathbb{E}_\xi(U \mid a)$.

However, now we may obtain more infomation about ω before making a final decision. In particular, each ω corresponds to a *model* of the world P_ω, which is a probability distribution over the observation space $\mathcal{S}$, such that $P_\omega(X)$ is

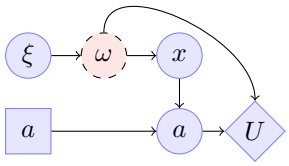


Figure 3.6: Statistical decision problem with observations

the probability that the observation is in $X \subset \mathcal{S}$. The set of parameters Ω thus defines a family of models:

$$\mathscr{P} \triangleq \{P_\omega \mid \omega \in \Omega\}. \tag{3.5.1}$$

Now, consider the case where we take an observation x from the true model P_{ω^*} before making a decision. We can represent the dependency of our decision on the observation by making our decision a function of x:

Definition 3.5.1 (policy)**.** A policy $\pi : \mathcal{S} \to \mathcal{A}$ maps from any observation to a decision.[1]

The expected utility of a policy π is:

$$U(\xi, \pi) \triangleq \mathbb{E}_\xi \{U[\omega, \pi(x)]\} = \int_\Omega \left(\int_\mathcal{S} U[\omega, \pi(x)] \, \mathrm{d}P_\omega(x) \right) \mathrm{d}\xi(\omega). \tag{3.5.2}$$

This is the standard Bayesian framework for decision making. It may be slightly more intuitive in some case to use the notation $\psi(x \mid \omega)$, in order to emphasize that this is a conditional distribution. However, there is no technical difference between the two notations.

When the set of policies includes all constant policies, then there is a policy π^* at least as good as the best fixed decision a^*. More formally:

Remark 3.5.1. Let Π denote a set of policies $\pi : \mathcal{S} \to \mathcal{A}$. If, $\forall a \in \mathcal{A}$ $\exists \pi \in \Pi$ such that $\pi(x) = a$ $\forall x \in \mathcal{S}$, then $\max_{\pi \in \Pi} \mathbb{E}_\xi(U \mid \pi) \geq \max_{a \in \mathcal{A}} \mathbb{E}_\xi(U \mid a)$.

Proof. The proof follows by setting Π_0 to be the set of constant policies. The result follows since $\Pi_0 \subset \Pi$. □

We conclude this section with a simple example, about deciding whether or not to go to a restaurant given expert opinions.

Example 22. Consider the problem of deciding whether or not to go to a particular restaurant. Let $\Omega = [0, 1]$ with $\omega = 0$ meaning the food is in general horrible and $\omega = 1$ meaning the restaurant is great. Let $x_1, \ldots, x_n$ be n expert opinions in $\mathcal{S} = \{0, 1\}$ about the restaurant. Under our model, the probability of observing $x_i = 1$ when the quality of the restaurant is ω is given by $P_\omega(1) = \omega$ and conversely $P_\omega(0) = 1 - \omega$. The probability of observing a particular[2] sequence x of length n is

$$P_\omega(x) = \omega^s (1 - \omega)^{n-s}$$

with $s = \sum_{i=1}^n x_i$.

[1] For that reason, policies are also sometimes called *decision functions* or *decision rules* in the literature.

[2] We obtain a different probability of observations under the binomial model, but the resulting posterior, and hence the policy, is the same.

Maximising utility when making observations

Statistical procedures based on the assumption that a distribution can be assigned to any parameter in a statistical decision problem, which we are considering here, are called *Bayesian statistical methods*. The scope of these methods has been the subject of much discussion in the statistical literature. See e.g. Savage [1972].

In the following, we shall look at different expressions for the expected utility. We shall overload the utility operator U for various cases: when the parameter is fixed, when the parameter is random, when the decision is fixed, and when the decision depends on the observation x and thus is random as well.

Expected utility of a fixed decision a with $\omega \sim \xi$

We first consider the expected utility of taking a fixed decision $a \in \mathcal{A}$, when $\mathbb{P}(\omega \in B) = \xi(B)$. This is the case we have dealt with so far.

$$U(\xi, a) \triangleq \mathbb{E}_\xi(U \mid a) = \int_\Omega U(\omega, a) \, \mathrm{d}\xi(\omega). \tag{3.5.3}$$

Expected utility of a policy π with fixed $\omega \in \Omega$

Now assume that ω is fixed, but instead of selecting a decision directly, we select a decision that depends on the random observation x, which is distributed according to P_ω on $\mathcal{S}$. We do this by defining a policy $\pi : \mathcal{S} \to \mathcal{A}$.

$$U(\omega, \pi) = \int_{\mathcal{S}} U(\omega, \pi(x)) \, \mathrm{d}P_\omega(x). \tag{3.5.4}$$

Expected utility of a policy π with $\omega \sim \xi$

Now we generalise to the case where ω is distributed with measure ξ. Note that the expectation of the previous expression (3.5.4) is by definition written as:

$$U(\xi, \pi) = \int_\Omega U(\omega, \pi) \, \mathrm{d}\xi(\omega), \quad U^*(\xi) \triangleq \sup_\pi U(\xi, \pi) = U(\xi, \pi^*). \tag{3.5.5}$$

Bayes decision rules

We wish to construct the Bayes decision rule, that is, the policy with maximal ξ-expected utility. However, doing so by examining all possible policies is hard, because (usually) there are many more policies than decisions. It is however, easy to find the Bayes decision for each possible observation. This is because it is usally possible to rewrite the expected utility of a policy in terms of the posterior distribution. While this is trivial to do when the outcome and observation spaces are finite, it can be extended to the general case as shown in the following theorem.

Theorem 3.5.1. *If U is non-negative or bounded, then we can reverse the integration order of*

$$U(\xi, \pi) = \mathbb{E}\left\{U[\omega, \pi(x)]\right\} = \int_\Omega \int_\mathcal{S} U[\omega, \pi(x)] \, \mathrm{d}P_\omega(x) \, \mathrm{d}\xi(\omega),$$

which is the normal form, to obtain the utility in extensive form, shown below:

$$U(\xi, \pi) = \int_\mathcal{S} \int_\Omega U[\omega, \pi(x)] \, \mathrm{d}\xi(\omega \mid x) \, \mathrm{d}P_\xi(x), \tag{3.5.6}$$

where $P_\xi(x) = \int_\Omega P_\omega(x) \, \mathrm{d}\xi(\omega)$.

Proof. To prove this when U is non-negative, we shall use Tonelli's theorem. First we need to construct an appropriate product measure. Let $p(x \mid \omega) \triangleq \frac{\mathrm{d}P_\omega(x)}{\mathrm{d}\nu(x)}$ be the Radon-Nikodym derivative of P_ω with respect to some dominating measure ν on $\mathcal{S}$. Similarly, let $p(\omega) \triangleq \frac{\mathrm{d}\xi(\omega)}{\mathrm{d}\mu(x)}$ be the corresponding derivative for ξ. Now, the utility can be written as:

$$\begin{aligned} U(\xi, \pi) &= \int_\Omega \int_\mathcal{S} U[\omega, \pi(x)] p(x \mid \omega) p(\omega) \, \mathrm{d}\nu(x) \, \mathrm{d}\mu(\omega) \\ &= \int_\Omega \int_\mathcal{S} h(\omega, x) \, \mathrm{d}\nu(x) \, \mathrm{d}\mu(\omega). \end{aligned}$$

Clearly, if U is non-negative, then so is $h(\omega, x) \triangleq U[\omega, \pi(x)] p(x \mid \omega) p(\omega)$. Then, Tonelli's theorem can be applied and:

$$\begin{aligned} U(\xi, \pi) &= \int_\mathcal{S} \int_\Omega h(\omega, x) \, \mathrm{d}\mu(\omega) \, \mathrm{d}\mu(x) \\ &= \int_\mathcal{S} \int_\Omega U[\omega, \pi(x)] p(x \mid \omega) p(\omega) \, \mathrm{d}\mu(\omega) \, \mathrm{d}\nu(x) \\ &= \int_\mathcal{S} \int_\Omega U[\omega, \pi(x)] p(\omega \mid x) \, \mathrm{d}\mu(\omega) p(x) \, \mathrm{d}\nu(x) \\ &= \int_\mathcal{S} \left[\int_\Omega U[\omega, \pi(x)] p(\omega \mid x) \, \mathrm{d}\mu(\omega) \right] \frac{\mathrm{d}P_\xi(x)}{\mathrm{d}\nu(x)} \, \mathrm{d}\nu(x) \quad = \int_\mathcal{S} \int_\Omega U[\omega, \pi(x)] \, \mathrm{d}\xi(\omega \mid x) \, \mathrm{d}P_\xi(x), \end{aligned}$$

□

We can construct an optimal policy π^* as follows. For any specific observed $x \in \mathcal{S}$, we set $\pi^*(x)$ to:

$$\pi^*(x) \triangleq \arg\max_{a \in \mathcal{A}} \mathbb{E}_\xi(U \mid x, a) = \arg\max_{a \in \mathcal{A}} \int_\Omega U(\omega, a) \, \mathrm{d}\xi(\omega \mid x).$$

So now we can plug π^* in the extensive form to obtain:

$$\int_\mathcal{S} \int_\Omega U[\omega, \pi^*(x)] \, \mathrm{d}\xi(\omega \mid x) \, \mathrm{d}P_\xi(x) = \int_\mathcal{S} \left\{ \max_a \int_\Omega U[\omega, a] \, \mathrm{d}\xi(\omega \mid x) \right\} \mathrm{d}P_\xi(x).$$

Consequently, there is no need to completely specify the policy before we have seen x. In particular, this would create problems when $\mathcal{S}$ is large.

Definition 3.5.2 (Prior distribution)**.** The distribution ξ is called the *prior distribution* of ω.

Definition 3.5.3 (Marginal distribution)**.** The distribution P_ξ is called the (prior) *marginal distribution* of x.

Definition 3.5.4 (Posterior distribution)**.** The conditional distribution $\xi(\cdot \mid x)$ is called the *posterior distribution* of ω.

Bayes decision rule.

The *optimal decision* given x, is the optimal decision with respect to the *posterior* $\xi(\omega \mid x)$. Thus, we do not need to pre-compute the complete Bayes-optimal decision rule.

3.5.1 Decision problems in classification.

Classification is the problem of deciding which class $y \in \mathcal{Y}$ some particular observation $x_t \in \mathcal{X}$ belongs to. From a decision-theoretic viewpoint, the problem can be seen at three different levels. In the first, we are given a classification model in terms of a probability distribution, and we simply we wish to classify optimally given the model. In the second, we are given a family of models, a prior distribution on the family, and a training data set, and we wish to classify optimally according to our belief. In the last form of the problem, we are given a set of *policies* $\pi : \mathcal{X} \to \mathcal{Y}$ and we must choose the one with highest expected performance. The two last classes of the problem are equivalent when the set of policies contains all Bayes decision rules for a specific model family.

Deciding the class given a probabilistic model

In the simple form of the problem, we are already given a classifier P that can calculate probabilities $P(y_t \mid x_t)$, and we simply must decide upon some class $a_t \in \mathcal{Y}$, so as to maximise a specific utility function. One standard utility function is the prediction accuracy

$$U_t \triangleq \mathbb{I}\{y_t = a_t\}.$$

The probability $P(y_t \mid x_t)$ is the posterior probability of the class given the observation x_t. If we wish to maximise expected utility, we can simply choose

$$a_t \in \arg\max_{a \in \mathcal{Y}} P(y_t = a \mid x_t).$$

This defines a particular, simple policy. In fact, for two-class problems with $\mathcal{Y} = \{0, 1\}$, such a rule can be often visualised as a *decision boundary* in $\mathcal{X}$, on whose one side we decide for class 0 and on whose other side for class 1.

decision boundary

Deciding the class given a model family

In the general form of the problem, we are given a *training* data set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, a set of *classification models* $\{P_\omega \mid \omega \in \Omega\}$, and a prior

distribution ξ on Ω. For each model, we can easily calculate $P_\omega(y_1, \dots, y_n \mid x_1, \dots, x_n)$. Consequently, we can calculate the posterior distribution

$$\xi(\omega \mid S) = \frac{P_\omega(y_1, \dots, y_n \mid x_1, \dots, x_n)\xi(\omega)}{\sum_{\omega' \in \Omega} P_{\omega'}(y_1, \dots, y_n \mid x_1, \dots, x_n)\xi(\omega')}$$

and the posterior marginal label probability

$$P_{\xi|S}(y_t \mid x_t) \triangleq P_\xi(y_t \mid x_t, S) = \sum_{\omega \in \Omega} P_\omega(y_t \mid x_t)\xi(\omega \mid S).$$

We can then construct the following simple policy:

$$a_t \in \arg\max_{a \in \mathcal{Y}} \sum_{\omega \in \Omega} P_\omega(y_t \mid x_t)\xi(\omega \mid S),$$

Bayes rule

known as *Bayes rule*.

The Bayes-optimal policy under parametrisation constraints*

In some cases, we are restricted to functionally simple policies, which do not contain any Bayes rules as defined above. For example, we might be limited to linear functions of x. Let $\pi : \mathcal{X} \to \mathcal{A}$ be such a rule and let Π be the set of allowed policies. Given a family of models and a set of training data, we wish to calculate the policy that maximises our expected utility. For a given ω, we can indeed calculate:

$$U(\omega, \pi) = \sum_{x,y} U(y, \pi(x)) P_\omega(y \mid x) P_\omega(x),$$

where we assume an i.i.d. model, i.e. $x_t \mid \omega \sim P_\omega(x)$ independently of previous observations. Note that to select the optimal rule $\pi \in \Pi$ we also need to know $P_\omega(x)$. For the case where ω is unknown and we have a posterior $\xi(\omega \mid S)$, the Bayesian framework is easily extensible:

$$U(\xi(\cdot \mid S), \pi) = \sum_{\omega} \xi(\omega \mid S) \sum_{x,y} U(y, \pi(x)) P_\omega(y \mid x) P_\omega(x).$$

This maximisation is not generally trivial. However, if our policy Π is parametrised, we can employ optimisation algorithms such as gradient ascent to find a maximum. In particular, it is true that if we sample $\omega \sim \xi(\cdot \mid S)$, then

$$\nabla_\pi U(\xi(\cdot \mid S), \pi) = \sum_{x,y} \nabla_\pi U(y, \pi(x)) P_\omega(y \mid x) P_\omega(x).$$

Fairness in classification problems*

Any policy, when applied to large-scale, real world problems, has certain externalities. This implies that considering only the decision maker's utility is not sufficient. One such issue is fairness.

This concerns desirable properties of policies applied to a population of individuals. For example, college admissions should be decided on variables that inform about merit, but fairness may also require taking into account the fact

that certain communities are inherently disadvantaged. At the same time, a person should not feel that another in a similar situation obtained an unfair advantage. All this must be taken into account while still caring about optimizing the decision maker's utility function. As another example, consider mortgage decisions: while lenders should take into account the creditworthiness of individuals in order to make a profit, society must ensure that they do not unduly discriminate against socially vulnerable groups.

Recent work in fairness for statistical decision making in the classification setting has considered two main notions of fairness. The first uses (conditional) *independence* constraints between a sensitive variable (such as ethnicity) and other variables, such as decisions made. The second type ensures that decisions are *meritocratic*, so that better individuals are favoured, but also smoothness,[3] in order to avoid elitism. While a thorough discussion of fairness is beyond the scope of this book, it is useful to note that some of these concepts are impossible to strictly achieve simultaneously, but may be approximately satisfied by careful design of the policy. The recent work by Dwork et al. [2012], Chouldechova [2016], Corbett-Davies et al. [2017], Kleinberg et al. [2016], Kilbertus et al. [2017], Dimitrakakis et al. [2017] goes much more deeply on this topic.

3.5.2 Calculating posteriors

Posterior distributions for multiple observations

We now consider how we can re-write the posterior distribution over Ω incrementally. Assume that we have a prior ξ on Ω. We then observe $x^n \triangleq x_1, \ldots, x_n$. For the observation probability, we write:

Observation probability given history x^{n-1} and parameter ω

$$P_\omega(x_n \mid x^{n-1}) = \frac{P_\omega(x^n)}{P_\omega(x^{n-1})}$$

Now we can write the posterior as follows:

Posterior recursion

$$\xi(\omega \mid x^n) = \frac{P_\omega(x^n)\xi(\omega)}{P_\xi(x^n)} = \frac{P_\omega(x_n \mid x^{n-1})\xi(\omega \mid x^{n-1})}{P_\xi(x_n \mid x^{n-1})}. \tag{3.5.7}$$

Here $P_\xi(\cdot \mid \cdot) = \int_\Omega P_\omega(\cdot \mid \cdot)\, \mathrm{d}\xi(\omega)$ is a marginal distribution.

Posterior distributions for multiple independent observations

Now we consider the case where, given the parameter ω, the next observation does not depend on the history: If $P_\omega(x_n \mid x^{n-1}) = P_\omega(x_n)$ then $P_\omega(x^n) = \prod_{k=1}^n P_\omega(x_k)$. Then:

[3] More precisely Lipschitz conditions on the policy

Posterior recursion with conditional independence

$$\xi_n(\omega) \triangleq \xi_0(\omega \mid x^n) = \frac{P_\omega(x^n)\xi_0(\omega)}{P_{\xi_0}(x_n)} \tag{3.5.8}$$

$$= \xi_{n-1}(\omega \mid x_n) = \frac{P_\omega(x_n)\xi_{n-1}(\omega)}{P_{\xi_{n-1}}(x_n)}, \tag{3.5.9}$$

where ξ_t is the belief at time t. Here $P_{\xi_n}(\cdot \mid \cdot) = \int_\Omega P_\omega(\cdot \mid \cdot)\,\mathrm{d}\xi_n(\omega)$ is the marginal distribution with respect to the n-th posterior.

Conditional independence allows us to write the posterior update as an identical recursion at each time t. We shall take advantage of that when we look at *conjugate prior* distributions in Chapter 4. For such models, the recursion involves a particularly simple parameter update.

3.6 Summary.

In this chapter, we introduced a general framework for making decisions $a \in \mathcal{A}$ whose optimality depends on an unknow outcome or parameter ω. We saw that, when our knowledge about $\omega \in \Omega$ is in terms of a probabiltiy distribution ξ on Ω, then the utility of the Bayes-optimal decision is convex with respect to ξ.

In some cases, observations $x \in \mathcal{X}$ may affect affect our belief, leading to a posterior $\xi(\cdot \mid x)$. This requires us to introduce the notion of a policy $\pi : \mathcal{X} \to \mathcal{A}$ mapping observations to decisions. While it is possible to construct a complete policy by computing $U(\xi, \pi)$ for all *policies* (normal form) and maximising, it is frequently simpler to just wait until we observe x and compute $U[\xi(\cdot \mid x), a]$ for all *decisions* (extensive form).

In minimax settings, we can consider a fixed but unknown parameter ω or a fixed but unknown prior ξ. This links statistical decision theory to game theory.

3.7 Exercises

The first part of this exercise set considers problems where we are simply given smoe distribution over Ω. In the second part, the distribution is a posterior distribtuion that depends on observations x.

3.7.1 Problems with no observations.

For the first part of exercises, we consider a set of worlds Ω and a decision set $\mathcal{A}$, as well as the following utility function $U : \Omega \times \mathcal{A} \to \mathbb{R}$:

$$U(\omega, a) = \operatorname{sinc}(\omega - a) \tag{3.7.1}$$

where $\operatorname{sinc}(x) = \sin(x)/x$. If ω is known and $\mathcal{A} = \Omega = \mathbb{R}$ then obviously the optimal decision is $a = \omega$, as $\operatorname{sinc}(x) \leq \operatorname{sinc}(0) = 1$. However, we consider the following case:

$$\Omega = \mathcal{A} = \{-2.5, \ldots, -0.5, 0, 0.5, \ldots, 2.5\}.$$

Exercise 13. Assume ω is drawn from ξ, with $\xi(\omega) = 1/11$ for all $\omega \in \Omega$, calculate and plot the expected utility $U(\xi, a) = \sum_\omega \xi(\omega) U(\omega, a)$ for each a. Report $\max_a U(\xi, a)$.

Exercise 14 (5). Assume $\omega \in \Omega$ is arbitrary (but deterministically selected). Calculate the utility $U(a) = \min_\omega U(\omega, a)$ for each a. Report $\max(U)$.

Exercise 15. Again assume $\omega \in \Omega$ is arbitrary (but deterministically selected). We now allow for stochastic policies π on $\mathcal{A}$. Then the expected utility is $U(\omega, \pi) = \sum_a U(\omega, a)\pi(a)$.

(a) Calculate and plot the expected utility when $\pi(a) = 1/11$ for all a, reporting values for all ω.

(b) Find

$$\max_\pi \min_\xi U(\xi, \pi).$$

Hint: Use the linear programming formulation, adding a constant to the utility matrix U so that all elements are non-negative.

Exercise 16. Consider the definition of rules that, for some $\epsilon > 0$, select a maximising

$$P\left(\left\{\omega \;\middle|\; U(\omega, a) > \sup_{d' \in \mathcal{A}} U(\omega, d') - \epsilon\right\}\right). \tag{3.7.2}$$

Prove that this is indeed a statistical decision problem, i.e. it corresponds to maximising the expectation of some utility function.

3.7.2 Problems with observations.

For this section, we consider a set of worlds Ω and a decision set $\mathcal{A}$, as well as the following utility function $U : \Omega \times \mathcal{A} \to \mathbb{R}$:

$$U(\omega, a) = -|\omega - a|^2. \tag{3.7.3}$$

In addition, we consider a family of distributions on a sample space $S = \{0, 1\}^n$,

$$\mathscr{F} \triangleq \{f_\omega \mid \omega \in \Omega\}, \tag{3.7.4}$$

such that f_ω is the binomial probability mass function with parameters ω (with the number of draws n being implied).

Consider the parameter set:

$$\Omega = \{0, 0.1, \dots, 0.9, 1\}. \tag{3.7.5}$$

Let ξ be the uniform distribution on Ω, such that $\xi(\omega) = 1/11$ for all $\omega \in \Omega$. Let the decision set be:

$$\mathcal{A} = [0, 1]. \tag{3.7.6}$$

EXERCISE 17. What is the decision a^* maximising $U(\xi, a) = \sum_\omega \xi(\omega) U(\omega, a)$ and what is $U(\xi, a^*)$?

EXERCISE 18. In the same setting, we now observe the sequence $x = (x_1, x_2, x_3) = (1, 0, 1)$.

1. Plot the posterior distribution $\xi(\omega \mid x)$ and compare it to the posterior we would obtain if our prior on ω was $\xi' = \mathcal{B}eta(2, 2)$.
2. Find the decision a^* maximising the *a posteriori* expected utility

$$\mathbb{E}_\xi(U \mid a, x) = \sum_\omega U(\omega, a) \xi(\omega \mid x).$$

3. Consider $n = 2$, i.e. $S = \{0, 1\}^2$. Calculate the Bayes-optimal expected utility in extensive form:

$$\mathbb{E}_\xi(U \mid \pi^*) = \sum_S \phi(x) \sum_\omega U[\omega, \pi^*(x)] \xi(\omega \mid x) = \sum_S \phi(x) \max_a \sum_\omega U[\omega, a] \xi(\omega \mid x), \tag{3.7.7}$$

where $\phi(x) = \sum_\omega f_\omega(x) \xi(\omega)$ is the prior marginal distribution of x and $\delta^* : S \to \mathcal{A}$ is the Bayes-optimal decision rule.

Hint: You can simplify the computational complexity somewhat, since you only need to calculate the probability of $\sum_t x_t$. This is not necessary to solve the problem though.

EXERCISE 19. In the same setting, we consider nature to be adversarial. Once more, we observe $x = (1, 0, 1)$. Assume that nature can choose a prior among a set of priors $\Xi = \{\xi_1, \xi_2\}$. Let $\xi_1(\omega) = 1/11$ and $\xi_2(\omega) = \omega/5.5$.

1. Calculate and plot our value for deterministic decisions a:

$$\min_{\xi \in \Xi} \mathbb{E}_\xi(U \mid a, x).$$

2. Find the minimax prior ξ^*

$$\min_{\xi \in \Xi} \max_{a \in \mathcal{A}} \mathbb{E}_\xi(U \mid a)$$

Hint: Apart from the adversarial prior selection, this is very similar to the previous exercise.

Chapter 4

Estimation

4.1 Introduction

In the previous unit, we have seen how to make optimal decisions with respect to a given utility function and belief. However, one important question is how a new belief can be calculated from observations and a prior belief. More generally, we wish to examine how much information we can obtain about an unknown parameter from observations, and how to bound our errors. Hence, while most of this chapter will focus on the Bayesian framework for estimating parameters, we shall also look at tools for making conclusions about the value of parameters without making specific assumptions about the data distribution, i.e. without providing specific prior information.

In the Bayesian setting, we calculate posterior distributions of parameters given data. The basic problem can be stated as follows. Let $\mathscr{P} \triangleq \{P_\omega \mid \omega \in \Omega\}$ be a family of probability measures on $(\mathcal{S}, \mathcal{F}_\mathcal{S})$ and ξ be our prior probability measure on $(\Omega, \mathcal{F}_\Omega)$. Given some data $x \sim P_{\omega^*}$, with $\omega^* \in \Omega$, how can we estimate ω^*? The Bayesian approach is to estimate the posterior distribution $\xi(\cdot \mid x)$, instead of guessing a single ω^*. In general, the posterior measure is a function $\xi(\cdot \mid x) : \mathcal{F}_\Omega \to [0, 1]$, with:

$$\xi(B \mid x) = \frac{\int_B P_\omega(x) \, \mathrm{d}\xi(\omega)}{\int_\Omega P_\omega(x) \, \mathrm{d}\xi(\omega)}. \tag{4.1.1}$$

The posterior distribution allows us to quantify our uncertainty about the unknown ω^*. This in turn enables us to take decisions that take uncertainty into account.

The first question we are concerned with in this chapter is how to calculate this posterior for any value of x in practice. If x is a complicated object, this may be computationally difficult. In fact, the posterior distribution can also be a complex function. However, there exist distribution families and priors such that this calculation is very easy, in the sense that the functional form of the posterior depends upon a small number of parameters. This happens when a summary of the data that contains all necessary information can be calculated easily. Formally, this is captured via the concept of a sufficient statistic.

4.2 Sufficient statistics

Sometimes we want to summarise the data we have observed. This can happen when the data is a long sequence of simple observations $x^t = (x_1, \ldots, x_t)$. It may also be useful to do so when we have a single observation x, such as a high-resolution image. For some applications, it may be sufficient to only calculate a really simple function of the data, such as the sample mean, defined below:

Definition 4.2.1 (Sample mean)**.** The sample mean $\bar{x}_t : \mathbb{R}^t \to \mathbb{R}$ of a sequence $x_k \in \mathbb{R}$ is defined as:

$$\bar{x}_t \triangleq \frac{1}{t} \sum_{k=1}^{t} x_k. \tag{4.2.1}$$

statistic

This summary, or any other function of the observations is called a *statistic*. In particular, we are interested in statistics that can replace all the complete original data in our calculations, without losing any information. Such statistics are called *sufficient*.

4.2.1 Sufficient statistics

We consider the standard probabilistic setting. Let $\mathcal{S}$ be a sample space and Ω be a parameter space defining a family of measures on $\mathcal{S}$:

$$\mathscr{P} = \{P_\omega \mid \omega \in \Omega\}.$$

In addition, we must also define an appropriate prior distribution ξ on the parameter space Ω. Now let us proceed to the definition of a sufficient statistic in the Bayesian sense.[1]

Definition 4.2.2. Let Ξ be a set of prior distributions on Ω, which indexes a family $\mathscr{P} = \{P_\omega \mid \omega \in \Omega\}$ of distributions on $\mathcal{S}$. A statistic $\phi : \mathcal{S} \to \mathcal{Z}$, where $\mathcal{Z}$ is a vector space[2] is a *sufficient statistic* for $\langle \mathscr{P}, \Xi \rangle$ if:

$$\xi(\cdot \mid x) = \xi(\cdot \mid x') \tag{4.2.2}$$

for any prior $\xi \in \Xi$ and any $x, x' \in \mathcal{S}$ such that $\phi(x) = \phi(x')$.

This simply states that the statistic is sufficient if, whenever we obtain the same value of the statistic for two different datasets x, x', then the resulting posterior distribution over the parameters is identical, no matter what the prior distribution. In other words, the value of the statistic is sufficient for computing the posterior. Interestingly, a sufficient statistic always implies the following factorisation for members of the family.

Theorem 4.2.1. *A statistic $\phi : \mathcal{S} \to \mathcal{Z}$ is sufficient $\langle \mathscr{P}, Bels \rangle$ iff there exist functions $u : \mathcal{S} \to (0, \infty)$, and $v : \mathcal{Z} \times \Omega \to [0, \infty)$ such that $\forall x \in \mathcal{S}, \omega \in \Omega$:*

$$P_\omega(x) = u(x)v[\phi(x), \omega], \qquad u > 0, v \geq 0. \tag{4.2.3}$$

Proof. The proof will be for the general case. The case when Ω is finite is technically simpler and is left as an exercise. Assume the existence of u, v. Then for any $B \in \mathcal{F}_\Omega$:

$$\begin{aligned} \xi(B \mid x) &= \frac{\int_B u(x)v[\phi(x), \omega] \, \mathrm{d}\xi(\omega)}{\int_\Omega u(x)v[\phi(x), \omega] \, \mathrm{d}\xi(\omega)} \\ &= \frac{\int_B v[\phi(x), \omega] \, \mathrm{d}\xi(\omega)}{\int_\Omega v[\phi(x), \omega] \, \mathrm{d}\xi(\omega)}. \end{aligned}$$

If $\phi(x) = \phi(x')$, then the above is also equal to $\xi(B \mid x')$, so $\xi(\cdot \mid x) = \xi(\cdot \mid x')$. Thus, ϕ satisfies the definition of a sufficient statistic.

Conversely, let ϕ be a sufficient statistic. Let μ be a dominating measure on $\mathcal{S}$ so that we can define the densities $p(\omega) \triangleq \frac{\mathrm{d}\xi(\omega)}{\mathrm{d}\mu(\omega)}$ and

$$p(\omega \mid x) \triangleq \frac{\mathrm{d}\xi(\omega \mid x)}{\mathrm{d}\mu(\omega)} = \frac{P_\omega(x)p(\omega)}{\int_\Omega P_\omega(x) \, \mathrm{d}\xi(\omega)}.$$

[1] There is an alternative definition, which replaces equality of posterior distributions with point-wise equality on the family members, i.e. $P_\omega(x) = P_\omega(x') \forall \omega$. This is a stronger definition, as it implies the Bayesian one we use here.

[2] Typically $\mathcal{Z} \subset \mathbb{R}^k$ for finite-dimensional statistics.

Consequently, we can write:

$$P_\omega(x) = \frac{p(\omega \mid x)}{p(\omega)} \int_\Omega P_\omega(x) \, \mathrm{d}\xi(\omega).$$

Since ϕ is sufficient, there is by definition some function $g : \mathcal{Z} \times \Omega \to [0, \infty)$ such that $p(\omega \mid x) = g[\phi(x), \omega]$. Consequently, we can factorise P_ω as:

$$P_\omega(x) = v[\phi(x), \omega] u(x),$$

where $u(x) = \int_\Omega P_\omega(x) \, \mathrm{d}\xi(\omega)$ and $v[\phi(x), \omega] = g[\phi(x), \omega] / \xi(\omega)$. □

In the factorisation of Theorem 4.2.1, u is the only factor that depends directly on x. Interestingly, it *does not appear* in the posterior calculation at all. So, the posterior only depends on x through the statistic.

Example 23. Suppose $x^t = (x_1, \ldots, x_t)$ is a random sample from a Bernoulli distribution with parameter ω. Then the joint probability is

$$P_\omega(x^t) = \prod_{k=1}^{t} P_\omega(x_k) = \omega^{s_t} (1 - \omega)^{t - s_t}$$

with $s_t = \sum_{k=1}^{t} x_k$ being the number of times 1 has been observed until time t. Then the statistic $\phi(x^t) = s_t$ satisfies (4.2.3) with $u(x) = 1$, while $P_\omega(x^t)$ only depends on the data through the statistic $s_t = \phi(x^t)$.

Another, example is when we have a *finite* set of models. Then the sufficient statistic is always a finite-dimensional vector.

Lemma 4.2.1. *Let a family $\mathscr{P} = \{P_\theta \mid \theta \in \Theta\}$, where each model P_θ is a probability measure on $\mathcal{X}$ and Θ contains n models. If $\boldsymbol{p} \in \mathbb{\Delta}^n$ is a vector representing our prior distribution, i.e. $\xi(\theta) = p_\theta$, then a sufficient statistic is the finite-dimensional vector $q_\theta = p_\theta P_\theta(x)$.*

Proof. Simply note that the posterior distribution in this case is

$$\xi(\theta \mid x) = \frac{q_\theta}{\sum_{\theta'} q_{\theta'}}.$$

Thus, all the information we need to compute the posterior is $\boldsymbol{q}$. Alternatively, we could also use a vector $\boldsymbol{w}$ with $w_\theta = \frac{q_\theta}{\sum_{\theta'} q_{\theta'}}$. □

In other words, when dealing with a finite set of models, it's always possible to maintain a finite dimensional sufficient statistic. This could simply be the actual posterior distribution, since that is also a finite-dimensional vector.

More generally, however, the prior and posterior distributions are functions (i.e. they have an infinite number of points and so cannot be represented as finite vectors). There are nevertheless still cases where we can compute posterior distributions efficiently.

4.2.2 Exponential families

Many well-known distributions such as the Gaussian, Bernoulli and Dirichlet distribution are members of the exponential family of distributions. All those distributions are factorisable in the manner shown below, while at the same time they have fixed-dimension sufficient statistics.

Definition 4.2.3. A distribution family $\mathscr{P} = \{P_\omega \mid \omega \in \Omega\}$ with P_ω being a probability function (or density) on the sample space $\mathcal{S}$, is said to be an *exponential family* if for any $x \in \mathcal{S}$, $\omega \in \Omega$:

$$P_\omega(x) = a(\omega)b(x)\exp\left[\sum_{i=1}^{k} g_i(\omega)h_i(x)\right]. \tag{4.2.4}$$

Informally, it is interesting to know that among families of distributions satisfying certain smoothness conditions, only exponential familes have a fixed-dimension sufficient statistic.

Because of this, exponential family distributions admit so-called parametric *conjugate* prior distribution families. These have the property that any posterior distribution calculated will remain within the conjugate family. Frequently, because of the simplicity of the statistic used, calculation of the conjugate posterior parameters is very simple.

4.3 Conjugate priors

In this section, we examine some well-known conjugate families. First, we give sufficient conditions for the of existence of conjugate family of priors for a given distribution family and statistic. While this section can be used as a reference, the reader may wish to initially only look at the first few example families.

The following remark gives sufficient conditions for the existence of a finite-dimensional sufficient statistic.

Remark 4.3.1. If a family $\mathscr{P}$ of distributions on $\mathcal{S}$ has a sufficient statistic $\phi : \mathcal{S} \to \mathcal{Z}$ of *fixed* dimension for any $x \in \mathcal{S}$, then there exists a conjugate family of priors $\Xi = \{\xi_\alpha \mid \alpha \in A\}$, where A is a set of possible parameters for the prior distribution, such that:

1. $P_\omega(x)$ is proportional to some $\xi_\alpha \in \Xi$:

$$\forall x \in S, \exists \xi_\alpha \in \Xi, c > 0 \ : \ \int_B P_\omega(x)\,\mathrm{d}\xi_\alpha(\omega) = c\xi_\alpha(B), \forall B \in \mathcal{F}_\Omega$$

2. The family is closed under multiplication:

$$\forall \xi_1, \xi_2 \in \Xi, \quad \exists \xi_\alpha \in \Xi, c > 0$$

such that:

$$\xi_\alpha = c\xi_1\xi_2.$$

While conjugate families exist for statistics with unbounded dimension, here we shall focus on finite-dimensional families. We will start wtih the simplest example, the Bernoulli-Beta pair.

4.3.1 Bernoulli-Beta conjugate pair

The Bernoulli-Beta conjugate pair of families is useful for problems where we wish to measure success rates of independent trials. First, we shall give details on the Bernoulli distribution. Then, we shall define the Beta distribution and describe its conjugate relation to the Bernoulli.

The Bernoulli distribution is a discrete distribution with outcomes taking values in $\{0, 1\}$. It is ideal for modelling the outcomes of independent random trials with fixed probability of success. The structure of the graphical model in

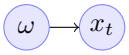


Figure 4.1: Bernoulli graphical model

Figure 4.1 shows the dependencies between the different variables of the model, which is explained below.

Definition 4.3.1 (Bernoulli distribution)**.** The Bernoulli distribution is discrete with outcomes $\mathcal{S} = \{0, 1\}$, parameter $\omega \in [0, 1]$, and probability function:

$$P_\omega(x_t = u) = \begin{cases} \omega, & u = 1 \\ 1 - \omega, & u = 0 \end{cases} = \omega^u (1 - \omega)^{1-u}.$$

If x_t is distributed according to a Bernoulli distribution with parameter ω, we write $x_t \sim \mathcal{Bern}(\omega)$.

The Bernoulli distribution can be extended to $\mathcal{S} = \{0, 1\}^n$ by modelling each outcome as independent. Then $P_\omega(x^n) = \prod_{t=1}^n P_\omega(x_t)$. This is the probability of observing the exact sequence x^t under the Bernoulli model. However, in many cases we are interested in the probability of observing the particular number of $1s$ and $0s$ and do not care about the actual order. In that case, we use what is called the binomial distribution.

bimomial coefficient

We first need a way to *count* the cases where, out of n trials, we have k positive outcomes. This is given by the *bimomial coefficient*, defined below:

$$\binom{n}{k} \triangleq \frac{\prod_{i=0}^{k-1}(n-i)}{k!}, \qquad k, n \in \mathbb{N}, \tag{4.3.1}$$

and $\binom{0}{k} = 1$. It follows that

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \qquad k, n \in \mathbb{N}, k \geq n. \tag{4.3.2}$$

Now we can define the binomial distribution in terms of the binomial coefficient. This is just a scaled product-Bernoulli distribution for multiple independent outcomes, but where we want to measure the probability of a particular number of 1s or 0s.

Definition 4.3.2 (Binomial Distribution)**.** Let us denote the total number of 1's observed until time t by $s_t = \sum_{k=1}^t x_k$. Then the probability that, k out

of t trials will be positive can be written in terms of the probability function $f(k \mid t, \omega)$ of the binomial distribution:

$$\mathbb{P}(s_t = k \mid \omega) = f(k \mid t, \omega) \triangleq \binom{t}{k} \omega^k (1-\omega)^{t-k}. \tag{4.3.3}$$

If s_t is drawn from a binomial distribution with parameters ω, t, we write $s_t \sim \mathcal{Binom}(\omega, t)$.

The Bernoulli is a distribution on a sequence of outcomes, while the binomial is a distribution on the total number of positive outcomes. This is why the binomial distribution includes the binomial coefficient, which basically counts the number of possible sequences of length t that have k positive outcomes.

Let us return to the Bernoulli distribution. If the ω parameter is known, then all the observations are independent of each other. However, this is not the case when ω is unknown. For example, let $\Omega = \{\omega_1, \omega_2\}$. Then

$$\mathbb{P}(x^t) = \sum_{\omega \in \Omega} \mathbb{P}(x^t \mid \omega)\, \mathbb{P}(\omega) = \sum_{\omega} \prod_{k=1}^{t} \mathbb{P}(x_k \mid \omega)\, \mathbb{P}(\omega) \neq \prod_{k=1}^{t} \mathbb{P}(x_k).$$

In general, however $\Omega = [0, 1]$. In that case, is there a prior distribution that could succinctly describe our uncertainty about the parameter? Indeed, there is, and it is called the *Beta distribution* This distribution is defined on the interval *Beta distribution*

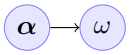


Figure 4.2: Beta graphical model

$[0, 1]$ has two parameters that determine the density of the observations. Because the Bernoulli distribution has a parameter in $[0, 1]$, the outcomes of the Beta can be used to specify a prior on the parameters of the Bernoulli distribution. Let us now call the distribution's outcomes ω and its parameter $\boldsymbol{\alpha}$. The dependencies between the parameters are described in the graphical model of Figure 4.2.

Definition 4.3.3 (Beta distribution)**.** The Beta distribution has outcomes $\omega \in \Omega = [0, 1]$ and parameters $\alpha_0, \alpha_1 > 0$, $\boldsymbol{\alpha} = (\alpha_1, \alpha_0)$. It is defined via its probability density function:

$$p(\omega \mid \boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0 + \alpha_1)}{\Gamma(\alpha_0)\Gamma(\alpha_1)} \omega^{\alpha_1 - 1} (1-\omega)^{\alpha_0 - 1}, \tag{4.3.4}$$

where Γ is the *gamma function*. If ω is distributed according to a Beta distribution with parameters α_1, α_0, we write: $\omega \sim \mathcal{Beta}(\alpha_1, \alpha_0)$. *gamma function*

A Beta distribution with parameter $\boldsymbol{\alpha}$ has expectation

$$\mathbb{E}(\omega \mid \boldsymbol{\alpha}) = \alpha_1 / (\alpha_0 + \alpha_1) //$$

and variance

$$\mathbb{V}(\omega \mid \boldsymbol{\alpha}) = \frac{\alpha_1 \alpha_0}{(\alpha_1 + \alpha_0)^2 (\alpha_1 + \alpha_0 + 1)}.$$

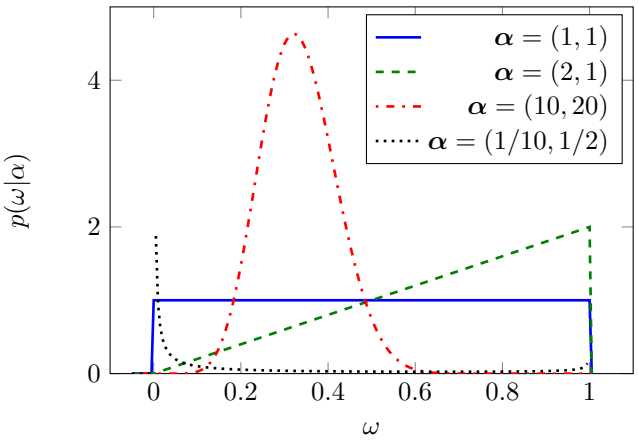


Figure 4.3: Four example Beta densities

Figure 4.3 shows the density of a Beta distribution for four different parameter vectors. When $\alpha_0 = \alpha_1 = 1$, the distribution is equivalent to a uniform one. The Beta distribution is useful for expressing probabilities of random variables in bounded intervals. In particular, since probabilities of events take values in $[0, 1]$, the Beta distribution is an excellent choice for expressing uncertainty about a probability.

Beta prior for Bernoulli distributions

We can encode our uncertainty about an unknown parameter of the Bernoulli distribution using a Beta distribution. The main idea is to assume that the Bernoulli parameter $\omega \in [0, 1]$ is unknown but fixed. We define a Beta prior distribution for ω to represent our uncertainty. This can be summarised by a parameter vector $\boldsymbol{\alpha}$ and we write $\xi_0(B) \triangleq \int_B p(\omega \mid \boldsymbol{\alpha})\, \mathrm{d}\omega$ for our prior distribution ξ_0. It is easy to see that in that case, the posterior probability is:

$$p(\omega \mid x^t, \boldsymbol{\alpha}) = \frac{\prod_{k=1}^t P_\omega(x_k) p(\omega \mid \boldsymbol{\alpha})}{\int_\Omega \prod_{k=1}^t P_\omega(x_k) p(\omega \mid \boldsymbol{\alpha})\, \mathrm{d}\omega} \propto \omega^{s_{t,1}+\alpha_1-1}(1-\omega)^{s_{t,0}+\alpha_0-1},$$

where $s_{t,1} = \sum_{k=1}^t x_k$ and $s_{t,0} = t - s_{t,1}$ is the total number of 1s and 0s respectively. As you can see, this again has the form of a Beta distribution.

Beta-Bernoulli model

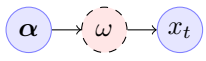


Figure 4.4: Beta-Bernoulli graphical model.

If ω is drawn from a Beta distribution with parameters α_1, α_0, and $x^t = x_1, \ldots, x_t$ is a sample drawn independently from a Bernoulli distribution

with parameter ω, i.e.:

$$\omega \sim \mathcal{Beta}(\alpha_1, \alpha_0) \qquad x^t \mid \omega \sim \mathcal{Bern}^t(\omega), \tag{4.3.5}$$

then the posterior distribution of ω given the sample the posterior distribution is also Beta:

$$\omega \mid x^t \sim \mathcal{Beta}(\alpha_1', \alpha_0'), \qquad \alpha_1' = \alpha_1 + \sum_{k=1}^{t} x_k, \quad \alpha_0' = \alpha_0 + t - \sum_{k=1}^{t} x_k \tag{4.3.6}$$

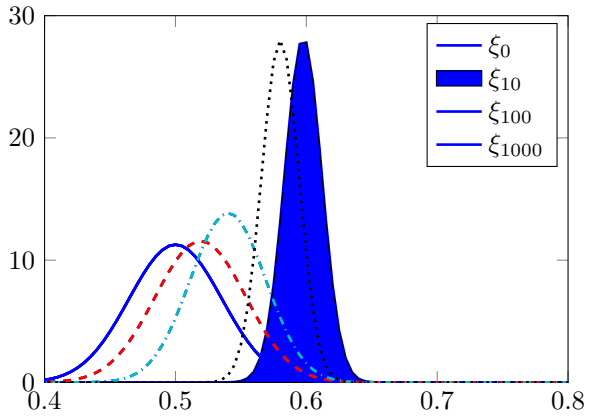


Figure 4.5: Changing beliefs as we observe tosses from a coin with probability $\omega = 0.6$ of heads.

Example 24. The parameter $\omega \in [0, 1]$ of a randomly selected coin can be modelled as a Beta distribution peaking around $1/2$. Usually one assumes that coins are fair. However, not all coins are exactly the same. Thus, it is possible that each coin deviates slightly from fairness. We can use a Beta distribution to model how likely (we think) different values ω of coin parameters are.

To demonstrate how belief changes, we perform the following simple experiment. Imagine a coin such that, when it is tossed, it has a probability 0.6 of coming heads every time it is tossed, independently of previous outcomes. Thus, the distribution of outcomes is a Bernoulli distribution with parameter $\omega = 0.6$.

We wish to form an accurate belief about how biased the coin is, under the assumption that the outcomes are Bernoulli with parameter ω. Our initial belief, ξ_0, is modelled as a Beta distribution on the parameter space $\Omega = [0, 1]$, with parameters $\alpha_0 = \alpha_1 = 100$. This places a strong prior on the coin being close to fair. However, we still allow for the possibility that the coin is biased.

Figure 4.5 shows a sequence of beliefs at times $0, 10, 100, 1000$ respectively, from a coin with bias $\omega = 0.6$. Due to the strength of our prior, after 10 observations, the situation has not changed much and the belief ξ_{10} is very close to the starting one. However, after 100 observations, our belief has now shifted towards 0.6, the true bias of the coin. After a total of 1000 observations, our belief is centered very close to 0.6, and is now much more concentrated, reflecting the fact that we are almost certain about the value of ω.

4.3.2 Conjugates for the normal distribution

The well-known normal distribution is also endowed with suitable conjugate priors. We first give the definition of the normal distribution, then consider the cases where we wish to estimate its mean, its variance, or both at the same time.

Definition 4.3.4 (Normal distribution)**.** The normal distribution is a continuous distribution, with outcomes in $\mathbb{R}$. It has two parameters, the mean $\omega \in \mathbb{R}$, and the variance $\sigma^2 \in \mathbb{R}^+$, or alternatively the precision $r \in \mathbb{R}^+$, where $\sigma^2 = r^{-1}$. It has the following probability density function:

$$f(x_t \mid \omega, r) = \sqrt{\frac{r}{2\pi}} \exp\left(-\frac{r}{2}(x_t - \omega)^2\right). \tag{4.3.7}$$

When x is distributed according to a normal distribution with parameters ω, r^{-1}, we write $x \sim \mathcal{N}(\omega, r^{-1})$. For a sample of size t, we write $x^t \sim \mathcal{N}^t(\omega, r^{-1})$. Independent samples satisfy the following independence condition

$$f(x^t \mid \omega, r) = \prod_{k=1}^{t} f(x_k \mid \omega, r) = \left(\frac{r}{\sqrt{2\pi}}\right)^t \exp\left(-\frac{r}{2}\sum_{k=1}^{t}(x_k - \omega)^2.\right) \tag{4.3.8}$$

The dependency graph in Figure 4.6 shows the dependencies between the parameters of a normal distribution and observations x_t. In this graph, only a

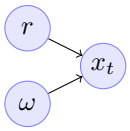


Figure 4.6: Normal graphical model

single sample x_t is shown, and it is implied that all x_t are independent of each other given r, ω.

Transformations of normal samples. The normal distribution is interesting mainly because many actual distributions turn out to be approximately normal. Further interesting properties of the normal distribution concern transformations of normal samples. For example, if x^n is drawn from a normal distribution with mean ω and precision r, then $\sum_{k=1}^{n} x_k \sim \mathcal{N}(n\omega, nr^{-1})$. Finally, if they are drawn from the *standard normal* distribution, i.e. $x_t \sim \mathcal{N}(0, 1)$, then $\sum_{k=1}^{n} x_t^2$ has a χ^2 *distribution* with n degrees of freedom.

standard normal
χ^2 *distribution*

Normal distribution with known precision, unknown mean

The simplest normal estimation problem occurs when we only need to estimate the mean, but we assume that the variance (or equivalently the precision) is known. For Bayesian estimation, it is convenient to assume that the mean ω is drawn from *another* normal distribution with known mean, as this results in a conjugate pair. Hence, we end up with a posterior normal distribution for the mean as well.

Normal-Normal conjugate pair

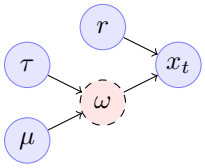


Figure 4.7: Normal with unknown mean, graphical model

If ω is drawn from a Normal distribution with parameters mean μ and precision τ, while $x_1, \ldots, x_n$ is a sample drawn independently from a Normal distribution with mean ω and precision r, i.e.

$$x^n \sim \mathcal{N}^n(\omega, r^{-1}), \qquad \omega \sim \mathcal{N}(\mu, \tau^{-1}),$$

then the posterior distribution of ω given the sample is also normal:

$$\omega \mid x^n \sim \mathcal{N}(\mu', 1/\tau'), \qquad \mu' = \frac{\tau\mu + nr\bar{x}_t}{\tau'}, \qquad \tau' = \tau + nr,$$

and $\bar{x}_n \triangleq \frac{1}{n}\sum_{k=1}^{n} x_k$.

In this case, our new estimate for the mean is shifted towards the empirical mean $\bar{x}_t$, and our precision increases linearly with the number of samples. Now we examine the case where we know the mean, but not the precision, of the normal distribution. This requires introducing another distribution as a prior for the value of the precision.

Normal with unknown precision and known mean

To model normal distributions with known mean, but unknown precision (or equivalently, unknown variance), we use the Gamma distribution to represent our uncertainty about the precision. The Gamma distribution itself is a two-parameter distribution, whose graphical model is shown in Figure 4.8. Since

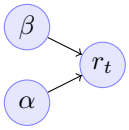


Figure 4.8: Gamma graphical model

.

the precision of the normal distribution is a positive parameter, the Gamma distribution only has support on $[0, \infty)$. Its two parameters determine the shape and scale of the distribution, as illustrated in Figure 4.9.

Definition 4.3.5 (Gamma distribution)**.** A random variable $r \sim \mathit{Gamma}(\alpha, \beta)$ is a random variable with outcomes in $[0, \infty)$, and probability density function:

$$f(r \mid \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} r^{\alpha-1} e^{-\beta r},$$

where $\alpha, \beta > 0$ and $\Gamma(\alpha) = \int_0^\infty u^{\alpha-1} e^{-u} \, \mathrm{d}u$ is the Gamma function (see also Appendix C.1.2).

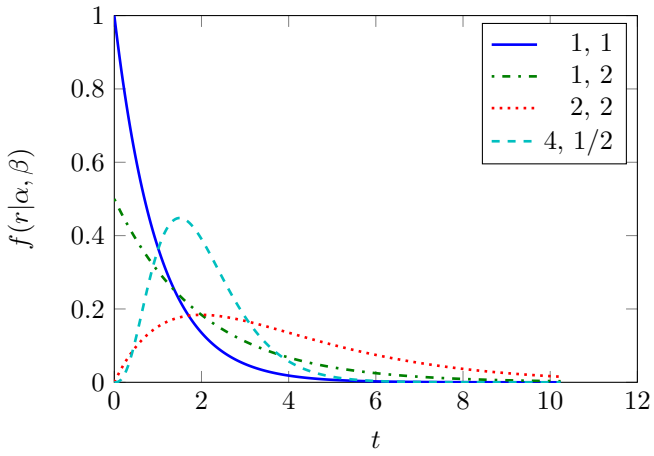


Figure 4.9: Example Gamma densities

A few example Gamma densities are shown in Figure 4.9. Some of those choices are special, as the Gamma distribution is a generalisation of a number of other standard distributions. For $\alpha = 1$, $\beta > 0$ one obtains an *exponential distribution* with parameter β and probability density function

exponential distribution

$$f(x \mid \beta) = \beta e^{-\beta x}, x > 0. \tag{4.3.9}$$

For $n \in \mathbb{N}$ and $\alpha = n/2$, $\beta = 1/2$ one obtains a χ^2 distribution with n degrees of freedom.

As already mentioned, the Gamma distribution is a natural choice for representing uncertainty about the accuracy of a normal distribution with known mean and unknown accuracy.

Normal-Gamma model

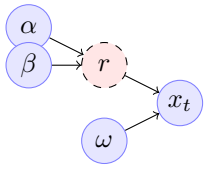


Figure 4.10: Normal-Gamma graphical model for normal distributions with unknown precision.

If r is drawn from a Gamma distribution with parameters α, β, while x^n is a sample drawn independently from a normal distribution with mean ω and precision r, i.e.

$$x^n \mid r \sim \mathcal{N}^n(\omega, 1/r), \qquad r \sim \mathit{Gamma}(\alpha, \beta)$$

then the posterior distribution of r given the sample is also Gamma:

$$r \mid x^n \sim \mathit{Gamma}(\alpha', \beta'), \qquad \alpha' = \alpha + \frac{n}{2}, \qquad \beta' = \beta + \frac{1}{2}\sum_{i=1}^{n}(x_i - \omega)^2$$

4.3.3 Normal with unknown precision and unknown mean

The more general problem is estimating a normal distribution when both the mean and the precision are unknown. In that case, we can use the same prior distributions for the mean and precision as when just one of them was unknown. The important thing to note is that the precision is independent of the mean, while the mean has a normal distribution given the precision.

Normal with unknown mean and precision

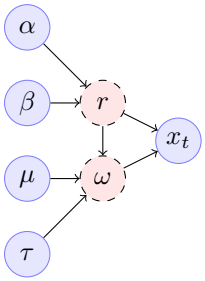


Figure 4.11: Graphical model for a normal distribution with unknown mean and precision, graphical model

Given a sample x^n from a normal distribution with unknown mean ω and precision r, whose prior joint distribution satisfies

$$\omega \mid r \sim \mathcal{N}(\mu, 1/(\tau)), \qquad r \sim \mathit{Gamma}(\alpha, \beta), \tag{4.3.10}$$

then the posterior distribution is

$$\omega \mid r, x^n \sim \mathcal{N}(\mu', 1/(\tau' r)), \qquad r \mid x^n \sim \mathit{Gamma}(\alpha', \beta'). \tag{4.3.11}$$

where

$$\mu' = \frac{\tau\mu + n\bar{x}}{\tau + n}, \qquad \tau' = \tau + n, \tag{4.3.12}$$

$$\alpha' = \alpha + \frac{n}{2}, \qquad \beta' = \beta + \frac{1}{2}\sum_{i=1}^{n}(x_i - \bar{x})_n^2 + \frac{\tau n(\bar{x} - \mu)^2}{2(\tau + n)}. \tag{4.3.13}$$

student t-distribution

In fact, while $\omega \mid r$ has normal distribution, the marginal distribution of ω is not normal. In fact, it can be shown that it has a *student t-distribution*. However, we are frequently interested in the marginal distribution of a set of observations x^n. This is also has a generalised student t-distribution, which is described below.

The marginal distribution of x. For a normal distribution with mean ω, precision r, we have

$$f(x \mid \omega, r) \propto r^{1/2} \exp\left(-\frac{r}{2}(\omega - x)^2\right).$$

For a prior $\omega|r \sim \mathcal{N}(\mu, 1/\tau r)$ and $r \sim \mathit{Gamma}(\alpha, \beta)$, as before, we have the following joint distribution for the mean and precision:

$$\xi(\omega, r) \propto r^{1/2} e^{-(\tau r/2)(\omega-\mu)^2} r^{\alpha-1} e^{-\beta r},$$

as $\xi(\omega, r) = \xi(\omega \mid r)\xi(r)$. Now we can write the marginal density of new observations as

$$\begin{aligned}
p_\xi(x) &= \int f(x \mid \omega, r)\, \mathrm{d}\xi(\omega, r) \\
&\propto \int_0^\infty \int_{-\infty}^\infty r^{1/2} e^{-\frac{r}{2}(\omega-x)^2} e^{-(\tau r/2)(\omega-\mu)^2} r^{\alpha-1} e^{-\beta r}\, \mathrm{d}\omega\, \mathrm{d}r \\
&= \int_0^\infty r^{\alpha-1/2} e^{-\beta r} \int_{-\infty}^\infty e^{-\frac{r}{2}(\omega-x)^2 - (\tau r/2)(\omega-\mu)^2}\, \mathrm{d}\omega\, \mathrm{d}r \\
&= \int_0^\infty r^{\alpha-1/2} e^{-\beta r} \left(\int_{-\infty}^\infty e^{-\frac{r}{2}\left[(\omega-x)^2 + \tau(\omega-\mu)^2\right]}\, \mathrm{d}\omega \right) \mathrm{d}r \\
&= \int_0^\infty r^{\alpha-1/2} e^{-\beta r} e^{-\frac{\tau r}{2(\tau+1)}(\mu-x)^2} \sqrt{\frac{2\pi}{r(1+\tau)}}\, \mathrm{d}r
\end{aligned}$$

4.3.4 Conjugates for multivariate distributions

The binomial distribution, as well as the normal distribution can be extended to multiple dimensions. Fortunately, multivariate extensions exist for their corresponding conjugate priors as well.

Multinomial-Dirichlet conjugates

The multinomial distribution is the extension of the binomial distribution to an arbitrary number of outcomes. Consider an outcome set $S = \{1, \ldots, K\}$.

This is a common model for independent random trials with a finite number of possible outcomes, such as repeated dice throws, multi-class classification problems, etc.

We now perform n trials, such that the outcome of each trial is independent of the rest. This is an extension of a sequence of n Bernoulli trials, but with a potentially larger set of possible outcomes in each trial.

The *multinomial* distribution gives the probability of obtaining outcome i exactly n_i times, given that we perform a total of n. The dependencies between the variables are given in Figure 4.12

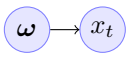


Figure 4.12: Multinomial graphical model.

Definition 4.3.6 (Multinomial distribution)**.** This is a discrete distribution with K outcomes $x_k \in S = \{1, \dots, K\}$. There is a vector parameter $\boldsymbol{\omega} \in \mathbb{R}^K$, with $\|\boldsymbol{\omega}\|_1 = 1$ and $\omega_i \geq 0$, with ω_i representing the probability of obtaining the i-th outcome. In other words, it is defined on the simplex $\mathbb{\Delta}^K$.[3] The outcomes are i.i.d., so that $\mathbb{P}(x_t = i \mid \boldsymbol{\omega}) = \omega_i$ for all i, t. Let us denote the number of times the i-th outcome was observed until time t by $n_{t,i} \triangleq \sum_{k=1}^{t} \mathbb{I}\{x_k = i\}$. Then the probability of obtaining a particular vector of outcome counts $\boldsymbol{n}_t = (n_{t,i})_{i=1}^K$ at time t is:

$$\mathbb{P}(\boldsymbol{n}_t \mid \boldsymbol{\omega}) = \frac{t!}{\prod_{i=1}^K n_{t,i}!} \prod_{i=1}^K \omega_i^{n_{t,i}}, \tag{4.3.14}$$

The Dirichlet distribution

The Dirichlet distribution is the multivariate extension of the Beta distribution. It has a vector parameter $\boldsymbol{\alpha}$ that determines the density of the observations, as shown in Figure 4.13.

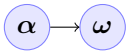


Figure 4.13: Dirichlet graphical model

Definition 4.3.7 (Dirichlet distribution)**.** The Dirichlet distribution is a continuous distribution with outcomes $\boldsymbol{\omega} \in \Omega = \mathbb{\Delta}^K$, i.e. $\|\boldsymbol{\omega}\|_1 = 1$ and $\omega_i \geq 0$ and parameter vector $\boldsymbol{\alpha} \in \mathbb{R}_+^K$. If

$$\boldsymbol{\omega} \sim \mathcal{D}ir(\boldsymbol{\alpha}),$$

then it is distributed according to the following probability density function:

$$f(\boldsymbol{\omega} \mid \boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^K \alpha_i)}{\prod_{i=1}^K \Gamma(\alpha_i)} \prod \omega_i^{\alpha_i - 1}., \tag{4.3.15}$$

[3] Due to these constraints, given $\omega_1, \dots, \omega_{K-1}$, the value of ω_K is fully determined from the remaining parameters.

The Dirichlet distribution is consequently a natural candidate for a prior on the multinomial distribution, as its support coincides with the parameter space of the latter. In fact, the Dirichlet distribution is conjugate to the multinomial distribution in the same way that the Beta distribution is conjugate to the Bernoulli/binomial distribution.

Multinomial distribution with unknown parameter.

Figure 4.14: Dirichlet-multinomial graphical model.

When the multinomial parameter $\boldsymbol{\omega}$ is unknown, we can assume it is generated from a Dirichlet distribution as shown in Figure 4.14. In particular, if we observe $x^t = (x_1, \ldots, x_t)$, and our prior is given by $\mathcal{D}ir(\boldsymbol{\alpha})$, so that our initial belief is $\xi_0(\boldsymbol{\omega}) \triangleq f(\boldsymbol{\omega} \mid \boldsymbol{\alpha})$, the resulting posterior after t observations is:

$$\xi_t(\boldsymbol{\omega}) \propto \prod_{i=1}^{K} \omega_i^{n_{t,i}+\alpha_i-1} \tag{4.3.16}$$

where $n_{t,i} = \sum_{k=1}^{t} \mathbb{I}\{x_k = i\}$.

Multivariate normal conjugate families

The last conjugate pair we shall discuss is that for multivariate normal distributions. Similarly to the extension of the Bernoulli distribution to the multinomial, and the corresponding extension of the Beta to the Dirichlet, the normal priors can be extended to the multivariate case. The prior of the mean becomes a multivariate normal distribution, while that of the precision becomes a Wishart distribution.

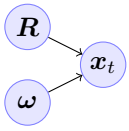


Figure 4.15: Multivariate normal graphical model

Definition 4.3.8 (Multivariate normal distribution)**.** The multivariate normal distribution is a continuous distribution, with outcome space $S = \mathbb{R}^K$, and the following parameters: the mean $\boldsymbol{\omega} \in \mathbb{R}^K$ and the precision[4] $\boldsymbol{R} \in \mathbb{R}^{K \times K}$, with $\boldsymbol{x}^\top \boldsymbol{R}\boldsymbol{x} > 0$ for any $\boldsymbol{x} \neq 0$, as shown in Figure 4.15. Its probability density function, where $|\boldsymbol{R}|$ denotes the *matrix determinant*, is:

matrix determinant

[4] In other words, the inverse of the covariance.

$$f(\boldsymbol{x}_t \mid \boldsymbol{\omega}, \boldsymbol{R}) = (2\pi)^{-K/2}|\boldsymbol{R}|^{1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x}_t - \boldsymbol{\omega}^\top \boldsymbol{R}(\boldsymbol{x}_t - \boldsymbol{\omega})\right). \quad (4.3.17)$$

Samples from the multivariate normal distribution are i.i.d. so that $f(\boldsymbol{x}^t \mid \boldsymbol{\omega}, \boldsymbol{R}) = \prod_{k=1}^t f(\boldsymbol{x}_k \mid \boldsymbol{\omega}, \boldsymbol{R})$.

First, we remind ourselves of the definition of a matrix trace:

Definition 4.3.9. The trace of a $n \times n$ square matrix A is

$$\text{trace}(A) \triangleq \sum_{i=1}^n a_{ii}.$$

Definition 4.3.10 (Wishart distribution)**.** The Wishart distribution is a *matrix distribution* on $\mathbb{R}^{K\times K}$ with $n > K-1$ degrees of freedom and precision matrix $\boldsymbol{T} \in \mathbb{R}^{K\times K}$. Its probability density function, for any positive $\boldsymbol{V} \in \mathbb{R}^{K\times K}$, is given by:

$$f(\boldsymbol{V} \mid n, \boldsymbol{T}) \propto |\boldsymbol{T}|^{n/2}|\boldsymbol{V}|^{(n-K-1)/2} e^{-\frac{1}{2}\text{trace}(\boldsymbol{T}\boldsymbol{V})}. \quad (4.3.18)$$

Construction of the Wishart distribution. Let $\boldsymbol{x}^n$ be drawn independently from a multivariate normal distribution with mean $\boldsymbol{\omega} \in \mathbb{R}^K$, and precision matrix $\boldsymbol{T} \in \mathbb{R}^{K,K}$, that is $\boldsymbol{x}^n \sim \mathcal{N}^n(\boldsymbol{\omega}, \boldsymbol{T}^{-1})$. Let $\bar{\boldsymbol{x}}_n$ be the empirical mean, and define the covariance matrix $\boldsymbol{S} = \sum_{i=1}^n (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_n)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_n)^\top$. Then $\boldsymbol{S}$ has a Wishart distribution with $n-1$ degrees of freedom and precision matrix $\boldsymbol{T}$ and we write $\boldsymbol{S} \sim \mathcal{W}ish(n-1, \boldsymbol{T})$.

Normal-Wishart conjugate prior

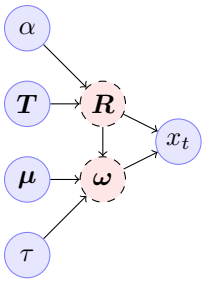


Figure 4.16: Normal-Wishart graphical model.

Theorem 4.3.1. *Given a sample $\boldsymbol{x}^n$ from a multivariate normal distribution in $\mathbb{R}^K$ with unknown mean $\boldsymbol{\omega} \in \mathbb{R}^K$ and precision $\boldsymbol{R} \in \mathbb{R}^{K\times K}$ whose joint prior distribution satisfies:*

$$\omega \mid \boldsymbol{R} \sim \mathcal{N}(\boldsymbol{\mu}, (\tau \boldsymbol{R})^{-1}), \qquad \boldsymbol{R} \sim \mathcal{W}ish(\alpha, \boldsymbol{T}), \quad (4.3.19)$$

with $\tau > 0$, $\alpha > K-1$, $\boldsymbol{T} > 0$, the posterior distribution is

$$\omega \mid \boldsymbol{R} \sim \mathcal{N}\left(\frac{\tau\boldsymbol{\mu} + n\bar{\boldsymbol{x}}}{\tau + n}, [(\tau+n)\boldsymbol{R}]^{-1}\right), \quad (4.3.20)$$

$$\boldsymbol{R} \sim \mathcal{W}ish\left(\alpha + n, \boldsymbol{T} + \boldsymbol{S} + \frac{\tau n}{\tau + n}(\boldsymbol{\mu} - \bar{\boldsymbol{x}})(\boldsymbol{\mu} - \bar{\boldsymbol{x}})^\top.\right), \quad (4.3.21)$$

where $\boldsymbol{S} = \sum_{i=1}^{n} (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top$.

4.4 Credible intervals

According to our current belief ξ, there is a certain subjective probability that the unknown parameter ω takes a certain value. However, we are not always interested in the precise probability distribution itself. Instead, we can use the complete distribution to describe an interval that we think contains the true value of the unknown parameter. In Bayesian parlance, this is called a credible interval.

Definition 4.4.1 (Credible interval)**.** Given some probability measure ξ on Ω representing our belief and some interval $A \subset \Omega$,

$$\xi(A) = \int_A \mathrm{d}\xi = \mathbb{P}(\omega \in A \mid \xi).$$

is our subjective belief that the unknown parameter ω is in A. If $\xi(A) = s$, then we say that A is an s-credible interval (or set), or an interval of size (or measure) s.

As an example, for prior distributions on $\mathbb{R}$, constructing an s-credible interval is usually done by finding $\omega_u, \omega_l \in \mathbb{R}$ such that

$$\xi([\omega_l, \omega_u]) = s.$$

Note that, *any* choice of A such that $\xi(A) = s$ is valid. However, typically the interval is chosen so as to exclude the tails (extremes) of the distribution and centered in the maximum. Figure 4.17 shows the 90% credible interval for

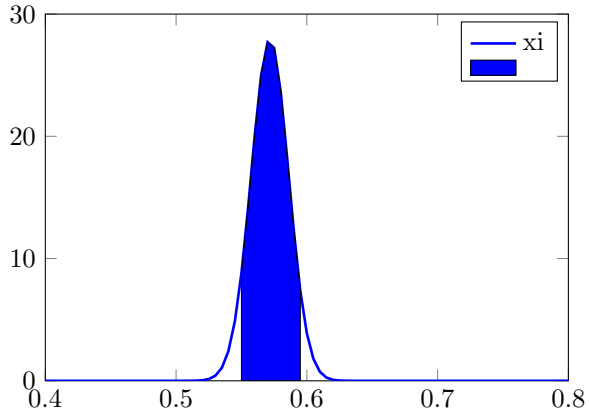


Figure 4.17: 90% credible interval after 1000 observations from a Bernoulli with $\omega = 0.6$.

the Bernoulli parameter. We see that the true parameter lies slightly outside it. (The measure of A under ξ is $\xi(A) = 0.9$.)

What is the probability that the true value of ω will be within a particular credible interval? This will depend on how well our prior ξ_0 matches the true distribution from which the parameter ω was drawn.

Reliability of credible intervals

Assume ϕ, ξ_0 are probability measures on the parameter set Ω, where our prior belief is ξ_0 and ϕ is the actual distribution of $\omega \in \Omega$. Each ω defines a measure P_ω on the observation set $\mathcal{S}$. We would liek to construct a credible interval $A_t \subset \Omega$ (which is a random variable $A_t : S^t \to \mathcal{F}_\Omega$) such that it has measure $s = \xi_t(A_t)$ for all t. Finally, let $Q \triangleq \int_\omega P_\omega \, \mathrm{d}\phi(\omega)$ be the marginal distribution on $\mathcal{S}$. Then the probability that the credible interval A_t will not include ω is

$$Q\left(\left\{x^t \in S^t \;\middle|\; \omega \notin A_t\right\}\right).$$

The main question is how this failure probability relates to s, t and ξ_0. So, let us design and conduct experiment for examining how often a typical credible interval includes the parameter we are interested in. In order to do so, we will have Nature draw the parameter from some arbitrary distibution ϕ, which may differ from our own assumed prior distribution ξ_0.

Experimental testing of a credible interval

Given a probability family $\mathscr{P} = \{P_\omega \mid w \in \Omega\}$.
Nature chooses distribution ϕ over Ω.
We choose another distribution ξ_0 over Ω.
for $k = 1, \ldots, n$ **do**
  Draw $\omega_k \sim \phi$.
  Draw $x^T \mid \omega_k \sim P_{\omega_k}$.
  for $t = 1, \ldots, T$ **do**
    Calculate $\xi_t(\cdot) = \xi_0(\cdot \mid x^t)$ for all t.
    Calculate A_t, for all t with $\xi_t(A_t) = 0.5$.
    Check failure: $\epsilon_{t,k} = \mathbb{I}\{\omega_k \notin A_t\}$
  end for
end for
Average over all k: $\epsilon_t = \frac{1}{n} \sum_{k=1}^{n} \epsilon_{t,k}$.

We performed this experiment for $n = 1000$ trials and for $T = 100$ observations per trial. Figure 4.18 illustrates what happens when $\phi = \xi_0$. We see that the credible interval is always centered around our initial mean guess and that it is quite tight. Figure 4.19 shows the failure rate the credible interval A_t around our estimated mean did not match the actual value of ω_k. Since the measure of our interval A_t is always $\xi_t(A_t) = 1/2$, we expect our error probability to be $1/2$, and this is borne out by the experimental results.

On the other hand, Figure 4.20 illustrates what happens when $\phi \neq \xi_0$. In fact in that case, $\phi(\omega) = \delta(\omega - 0.6)$, so that $\omega_k = 0.6$ for all trials k. We see that the credible interval is always centered around our initial mean guess and that it is always quite tight. Figure 4.21 shows the average number of failures. We see that initially, due to the fact that our prior is different from the distribution from

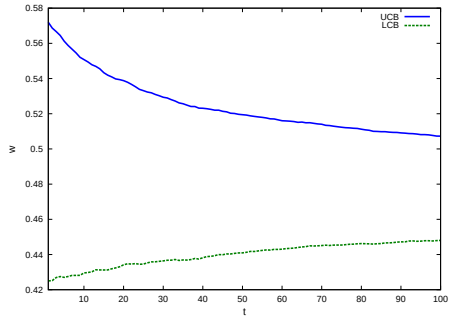


Figure 4.18: 50% credible intervals for a prior $\mathcal{B}eta(10, 10)$, matching the distribution of ω.

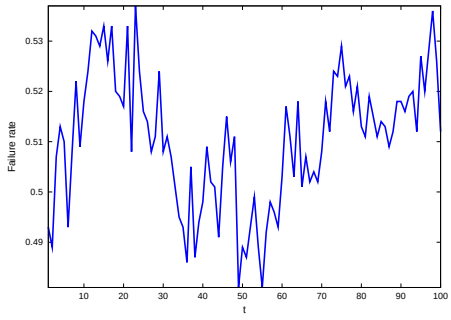


Figure 4.19: Failure rate of 50% CI for a prior $\mathcal{B}eta(10, 10)$, matching the distribution of ω.

which the ω_k are selected, we make many more mistakes. However, eventually, our prior is swamped by the data and our error rate converges to 50%.

4.5 Concentration inequalities

While Bayesian ideas are useful, as they allow us to express our subjective beliefs about a particular unknown quantity, they nevertheless are difficult to employ when we have no good intuition about what prior to use. One way to overcome this difficulty is by looking at the Bayesian estimation problem as a minimax game between us and nature, as seen in the previous chapter. In this case, we assume that nature chooses the prior distribution in the worst possible way. However, even in that case, we must select a family of distributions and priors.

This section will examine what guarantees we can give about any calculation we make from observations, if we make only very minimal assumptions about the distribution generating these observations. The results are fundamental, in the sense that they rely on a very general phenomenon, called *concentration of measure*. As a consequence, they are much stronger than results such as the central limit theorem (which is not treated in this textbook). However, here we shall focus on their most common applications.

It is interesting to consider the case calculating a sample mean, as given in Definition 4.2.1. We have seen that, for the Beta-Bernoulli conjugate prior, it

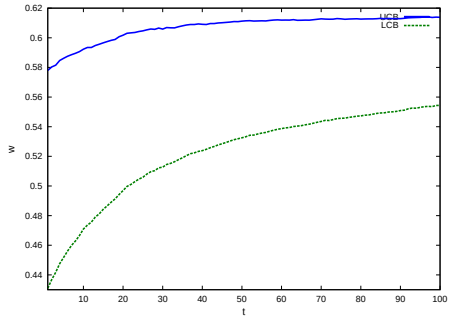


Figure 4.20: 50% credible intervals for a prior $\mathcal{Beta}(10, 10)$, when $\omega = 0.6$.

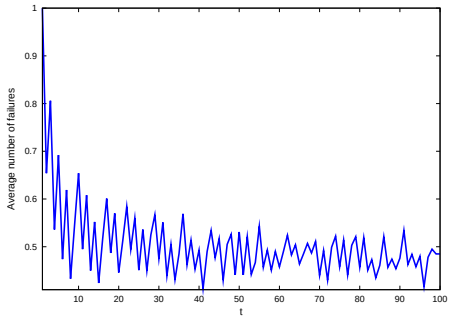


Figure 4.21: Failure rate of 50% CI for a prior $\mathcal{Beta}(10, 10)$, when $\omega = 0.6$.

is a simple enough matter to calculate a posterior distribution. From that, we can obtain a credible interval on the expected value of the unknown Bernoulli distribution. However, we would like to do the same for arbitrary distributions on $[0, 1]$, rather than just the Bernoulli. We shall now give an overview of a set of tools that can be used to do this.

Theorem 4.5.1 (Markov's inequality)**.** *If $X \sim P$ with P a distribution on $[0, \infty)$, then:*

$$\mathbb{P}(X \geq t) \leq \frac{\mathbb{E}\, X}{t}, \tag{4.5.1}$$

where $\mathbb{P}(X \geq t) = P(\{x \mid x \geq t\})$.

Proof. The expectation of X is:

$$\begin{aligned}
\mathbb{E}\, X &= \int_0^\infty x \, \mathrm{d}P(x) \\
&= \int_0^u x \, \mathrm{d}P(x) + \int_u^\infty x \, \mathrm{d}P(x) \\
&\geq 0 + \int_u^\infty u \, \mathrm{d}P(x) \\
&= uP(\{x \mid x \geq y\}) = u\, \mathbb{P}(X \geq u).
\end{aligned}$$

□

Consequently, if $\bar{x}_t$ is the empirical mean after t observations, for a random variable X with expectation $\mathbb{E}\, X = \mu$, we can use Markov's inequality to show that $\mathbb{P}(|\bar{x}_t - \mu| \geq \epsilon) \leq \mathbb{E}\,|\bar{x}_t - p|/\epsilon$. For $X \in [0,1]$, we obtain the bound

$$\mathbb{P}(|\bar{x}_t - \mu| \geq \epsilon) \leq 1/\epsilon.$$

Unfortunately this bound does not improve for a larger number of observations t. However, we can get significantly better bounds through various transformations. For monotonic f,

$$\mathbb{P}(X \geq t) = \mathbb{P}(f(X) \geq f(t)) \tag{4.5.2}$$

as $\{x \mid x \geq t\} = \{x \mid f(x) \geq f(t)\}$. Thus, we can apply Markov's inequality as a building block in other inequalities. The first of those is Chebyshev's inequality.

Theorem 4.5.2 (Chebyshev inequality)**.** *Let X be a random variable with expectation $\mu = \mathbb{E}\, X$ and variance $\sigma^2 = \mathbb{V}\, X$. Then, for all $k > 0$:*

$$\mathbb{P}(|X - \mu| \geq k\sigma) \leq k^{-2}. \tag{4.5.3}$$

Proof.

$$\begin{aligned}\mathbb{P}(|X-\mu| \geq k\sigma) = \mathbb{P}\left(\frac{|X-\mu|}{k\sigma} \geq 1\right) &\overset{(4.5.2)}{=} \mathbb{P}\left(\frac{|X-\mu|^2}{k^2\sigma^2} \geq 1\right) \\ &\overset{(4.5.1)}{\leq} \mathbb{E}\left(\frac{(X-\mu)^2}{k^2\sigma^2}\right) = \frac{\mathbb{E}(X-\mu)^2}{k^2\sigma^2} = k^{-2}.\end{aligned}$$

□

We can now apply (4.5.3) to our sample mean estimator in order to obtain a t-dependent bound on the probability that the sample mean is more than ϵ-far away from the actual mean.

Example 25 (Application to sample mean). It is easy to show that the sample mean has expectation μ and variance σ_x^2/t, where $\sigma_x^2 = \mathbb{V}\, x$. Consequently:

$$\mathbb{P}\left(|\bar{x}_t - \mu| \geq k\sigma_x/\sqrt{t}\right) \leq k^{-2}.$$

Setting $\epsilon = k\sigma_x/\sqrt{t}$ we get $k = \epsilon\sqrt{t}/\sigma_x$ and hence

$$\mathbb{P}(|\bar{x}_t - \mu| \geq \epsilon) \leq \frac{\sigma_x^2}{\epsilon^2 t}.$$

4.5.1 Chernoff-Hoeffding bounds

The previous inequality can be quite loose. In fact, one can prove tighter bounds for the estimation of an expected value. All these bounds rest upon a different application of the Markov inequality, due to Chernoff.

Main idea of Chernoff bounds.
Let $S_t = \sum_{k=1}^{t} X_k$, with $X_k \sim P$ independently, i.e. $X^t \sim P^t$. By definition, from Markov's inequality we obtain in turn, for any $\theta > 0$

$$\mathbb{P}(S_t \geq u) = \mathbb{P}(e^{\theta S_t} \geq e^{\theta u}) \leq e^{-\theta u}\, \mathbb{E}\, e^{\theta S_t} = e^{-\theta u} \prod_k \mathbb{E}\, e^{\theta X_k}, \qquad \text{for } x \in [a,b]. \tag{4.5.4}$$

Theorem 4.5.3. *Hoeffding inequality (Hoeffding [1963], Theorem 2) Let $x_k \sim P_k$ with $x_k \in [a_k, b_k]$ with $\mathbb{E}\, X_k = \mu_k$. Then*

$$\mathbb{P}\left(\bar{x}_t - \mu \geq \epsilon\right) \leq \exp\left(-\frac{2t^2\epsilon^2}{\sum_{k=1}^t (b_k - a_k)^2}\right), \tag{4.5.5}$$

where $\bar{x}_t = \frac{1}{t}\sum_{k=1}^t x_k$ and $\mu = \frac{1}{t}\sum_{k=1}^t \mu_k$.

Proof. Use (4.5.4), setting $X_k = x_k - \mu_k$ so that $S_t = t(\bar{x}_t - \mu)$ and $u = t\epsilon$. Then:

$$\mathbb{P}(\bar{x}_t - \mu \geq \epsilon) = \mathbb{P}(S_t \geq u) \leq e^{-\theta u} \prod_{k=1}^t \mathbb{E}\, e^{\theta X_k} = e^{-\theta t\epsilon} \prod_{k=1}^t \mathbb{E}\, e^{\theta(x_k - \mu)}. \tag{4.5.6}$$

Applying Jensen's inequality directly to the expectation does not help. However, we can use convexity in another way. Let $f(x)$ be the linear upper bound on $e^{\theta x}$ on the interval $[a, b]$, i.e.

$$f(x) = \frac{b-x}{b-a}e^{\theta a} + \frac{x-a}{b-a}e^{\theta b} \geq e^{\theta x}.$$

Then obviously $\mathbb{E}\, e^{\theta x} \leq \mathbb{E}\, f(x)$ for $x \in [a, b]$. Applying this to the expectation term (4.5.6) above we get,

$$e^{\theta(x_k - \mu_k)} \leq \frac{e^{-\theta\mu_k}}{b_k - a_k}\left\{(b_k - \mu_k)e^{\theta a_k} + (\mu_k - a_k)e^{\theta b_k}\right\}.$$

Taking derivatives and computing the Taylor expansion, we get

$$\begin{aligned}\mathbb{E}\, e^{\theta(x_k - \mu_k)} &\leq e^{\frac{1}{8}\theta^2(b_k - a_k)^2} \\ \mathbb{P}(\bar{x}_t - \mu \geq \epsilon) &\leq e^{-\theta t\epsilon + \frac{1}{8}\theta^2 \sum_{k=1}^t (b_k - a_k)^2}.\end{aligned}$$

This is minimised for $\theta = 4t\epsilon / \sum_{k=1}^t (b_k - a_k)^2$ and we obtain the required result. □

We can apply this inequality directly to the sample mean example, for $x_k \in [0, 1]$, to obtain

$$\mathbb{P}\left(|\bar{x}_t - \mu| \geq \epsilon\right) \leq 2e^{-2t\epsilon^2}.$$

4.6 Approximate Bayesian approaches

Unfortunately, being able to exactly calculate posterior distributions is only possible in special cases. In this section, we give a brief overview of some classic methods for approximate Bayesian inference. The first, Monte-Carlo methods, rely on stochastic approximations of the posterior distributions, where at least the likelihood function is computable. The second, approximate Bayesian computation, extends Monte Carlo methods to the case where the probability function is incomputable or not available at all. In the third which includes, variational Bayes methods, we replace distributions with an analytic approximation. Finally, in empirical Bayes methods, some parameters are replaced by an empirical estimate.

4.6.1 Monte-Carlo inference

Monte-Carlo inference has been a cornerstone of approximate Bayesian statistics ever since computing power was sufficient for such methods to become practical. Let us begin with a simple example, that of estimating expectations.

Estimating expectations.
Let $f : \mathcal{S} \to [0, 1]$ and P a measure on $\mathcal{S}$. Then

$$\mathbb{E}_P f = \int_{\mathcal{S}} f(x) \, \mathrm{d}P(x). \tag{4.6.1}$$

Estimating expectations is relatively easy, as long as we can generate samples from P. Then, we can our error in estimating its expectation by using the Hoeffding bound.

Corollary 4.6.1. *Let $\hat{f}_n = \frac{1}{n} \sum_t f(x_t)$ with $x_t \sim P$ and $f : \mathcal{S} \to [0, 1]$. Then:*

$$P\left(\left\{x^n \in S^n \;\middle|\; |\hat{f}_n - \mathbb{E}\, f| \geq \epsilon\right\}\right) \leq 2e^{-2n\epsilon^2}. \tag{4.6.2}$$

This technique is simple and fast. However, we frequently cannot sample from P, but only from some alternative distribution Q. Then it is hard to bound our error.

Another interesting application of this technique is the calculation of posterior distributions.

Example 26 (Calculation of posterior distributions). Assume a probability family $\mathscr{P} = \{P_\omega \mid \omega \in \Omega\}$ and a prior distribution ξ on Ω such that we can draw $\omega \sim \xi$. The posterior distribution can be written according to (4.1.1). The nominator can be written as

$$\int_B P_\omega(x) \, \mathrm{d}\xi(\omega) = \int_\Omega \mathbb{I}\{\omega \in B\} \, P_\omega(x) \, \mathrm{d}\xi(\omega) = \mathbb{E}_\xi \left[\mathbb{I}\{\omega \in B\} \, P_\omega(x)\right]. \tag{4.6.3}$$

Similarly, the denominator can be written as $\mathbb{E}_\xi[P_\omega(x)]$. If P_ω is bounded, then the errors can be bounded too.

An extension of this approach involves Markov chain Monte-Carlo (MCMC) methods. These are sequential sampling procedures, where data is sampled iteratively. At the k-th iteration, we obtain a sample $x^{(k)} \sim Q_k$, where Q_k depends on the previous sample drawn, $x^{(k-1)}$. Although under mild conditions $Q_k \to P$, there is no easy way to determine *a priori* when the procedure has converged. For more details see for example [Casella et al., 1999].

4.6.2 Approximate Bayesian Computation

The main problem we wish to solve in approximate Bayesian computation (ABC) is how to weigh the evidence we have for or against different models. The assumption is that we have a family of models $\{M_\omega \mid \omega \in \Omega\}$, from which we can generate data. However, there is no easy way to calculate the probability of any model having generated the data. On the other hand, like in the standard

Bayesian setting, we start with a prior ξ over Ω, and given some data $x \in \mathcal{W}$ we wish to calculate the posterior $\xi(\omega \mid x)$. ABC methods generally rely on what is called an *approximate statistic*, in order to weigh the relative likelihood of models for the data.

An approximate statistic $\phi : \mathcal{X} \to \mathcal{S}$ maps the data to some lower dimensional space. Then it is possible to compare different data points in terms of how similar their statistics are. For this, we also define some distance $D : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$.

ABC methods are useful in two specific situations. The first is when the family of models that we consider has an intractable likelihood. This means that calculating $M_\omega(x)$ is prohibitely expensive. The second is in some applications which admit a class of *parametrised simulators*, which have no probabilistic description. Then, one reasonable approach is to find the best simulator in the class, and then apply it to the actual problem.

The simplest algorithm in this context is rejection sampling (Alg 1). Here, we repeatedly sample a model from the prior distribution, and then generate data $\hat{x}$ from the model. If the sampled data is ϵ-close to the original data in terms of the statistic, we accept the sample as an approximate posterior sample.

For an overview of ABC methods see [Csilléry et al., 2010, Marin et al., 2011]. Early ABC methods were developed developed for applications, such as econometric modelling [e.g. Geweke, 1999], where detailed simulators were available, but no useful analytical probabilistic models. ABC methods have also been used for inference in dynamical systems [e.g Toni et al., 2009], the reinforcement learning problem [Dimitrakakis and Tziortziotis, 2013, 2014].

Algorithm 1 ABC Rejection Sampling from $\xi(\omega \mid x)$.

1: **input** prior ξ, data x, generative model family $\{M_\omega \mid \omega \in \Omega\}$, statistic ϕ, error bound ϵ.
2: **repeat**
3: $\quad \hat{\omega} \sim \xi$
4: $\quad \hat{x} \sim M_{\hat{\omega}}$.
5: **until** $D[\phi(x), \phi(\hat{x})] \leq \epsilon$
6: Return $\hat{\omega}$.

4.6.3 Analytic approximations of the posterior.

Another type of approximation involves substituting complex distributions with members from a simpler family. For example, one could replace a multimodal posterior distribution $\xi(\omega \mid x)$ with a Gaussian. However, a more principled approximation. would involve selecting a distribution that is the closest with respect to some divergence or distance, in this case the KL divergence. In particular, we would like to approximate the target distribution $\xi(\omega \mid x)$ with some other distribution $Q_\theta(\omega)$ in a family $\{Q_\theta \mid \theta \in \Theta\}$. While a number of distances such as the total variation or Wasserstein distance. However, the most popular algorithms employ the KL divergence

$$D\left(Q \parallel P\right) \triangleq \int_\Omega \ln \frac{\mathrm{d}Q}{\mathrm{d}P} \, \mathrm{d}Q, \tag{4.6.4}$$

where one term is the target posterior distribution and the other the approximation. As the KL divergence is asymmetric, its use results in two distinct

approximation methods: variational Bayes and expectation propagation.

Variational approximation. In this formulation, we wish to minimise the KL divergence

$$D\left(Q_\theta \parallel \xi_{|x}\right) = \int_\Omega \ln \frac{\mathrm{d}Q_\theta}{\mathrm{d}\xi_{|x}} \,\mathrm{d}Q_\theta, \tag{4.6.5}$$

where $\xi_{|x}$ is shorthand for the distribution $\xi(\omega \mid x)$. An efficient method for minimising this divergence is rewriting it as follows:

$$\begin{aligned} D\left(Q_\theta \parallel \xi_{|x}\right) &= -\int_\Omega \ln \frac{\mathrm{d}\xi_{|x}}{\mathrm{d}Q_\theta} \,\mathrm{d}Q_\theta \\ &= -\int_\Omega \ln \frac{\mathrm{d}\xi_x}{\mathrm{d}Q_\theta} \,\mathrm{d}Q_\theta + \ln \xi(x), \end{aligned}$$

where ξ_x is shorthand for the joint distribution $\xi(\omega, x)$ for a fixed value of x. As the latter term does not depend on θ, we can find the best element of the family by the following optimisation:

$$\max_{\theta \in \Theta} \int_\Omega \ln \frac{\mathrm{d}\xi_x}{\mathrm{d}Q_\theta} \,\mathrm{d}Q_\theta, \tag{4.6.6}$$

where the term we are maximising can also be seen as a lower bound on the marginal log likeliihood.

Expectation propagation. The other direction requires us to minimise the divergence

$$D\left(P \parallel Q\right) = \int_\Omega \ln \frac{\mathrm{d}P}{\mathrm{d}Q} \,\mathrm{d}P.$$

An algorithm for achieving this in the case of data terms that are independent given the parameter is expectation propagation [Minka, 2001a]. There, the approximation has a factored form and is iteratively updated, with each term minimising the KL divergence while keeping the remaining terms fixed.

4.6.4 Maximum Likelihood and Empirical Bayes methods.

When a full posterior distribution is not necessary, some parameter may be estimated point-wise. One simple such approach is maximum likelihood. In the simplest case, we replace the posterior distribution $\xi(\theta \mid x)$ with a point estimate corresponding to the parameter value that maximises the likelihood:

$$\theta^*_{\mathrm{ML}} \in \arg\max_\theta P_\theta(x). \tag{4.6.7}$$

Alternatively, the *maximum a posteriori* parameter maybe obtained:

$$\theta^*_{\mathrm{MAP}} \in \arg\max_\theta \xi(\theta \mid x). \tag{4.6.8}$$

In the latter case, even though we cannot compute the full function $\xi(\theta \mid x)$, we can still maximise (perhaps locally) for θ.

More generally, there might be some parameters ϕ for which we *can* compute a posterior distribution. Then we can still use the same approaches, maximising one of:

$$P_\theta(x) = \int P_{\theta,\phi}(x)\,\mathrm{d}\xi(\phi \mid x) \tag{4.6.9}$$

$$\xi(\theta \mid x) = \int_\Phi \xi(\theta \mid \phi, x)\,\mathrm{d}\xi(\phi \mid x) \tag{4.6.10}$$

Empirical Bayes methods, pioneered by Robbins [1955], some parameters are replaced by an empirical estimate, not necessary corresponding to the maximum likelihood. These methods are quite diverse Laird and Louis [1987], Lwin and Maritz [1989], Robbins [1964, 1955], Deely and Lindley [1981] and unfortunately beyond the scope of this book.

4.7 Exercises.

EXERCISE 20. Consider a set of $n = 2$ weather stations. Your prior belief that the i-th station is correct is $P(H_i) = 1/n$. You assume that only one station is the correct one, i.e. that $P(H_i \cap H_j) = 0$ for any two different stations $i \neq j$.

Now assume that the stations are making the following predictions for the next two days. Let A denote the event of rain on Saturday. Let B denote the event of rain on Sunday. The first station predicts that there will be rain on Saturday with probability 10%, i.e. $P(A \mid H_1) = 0.1$ and rain on Sunday with probability 50%, .e. $P(B \mid H_1) = 0.5$. The second station that there will be rain with probability on Saturday with probability 20% and on Sunday with probability 20% again, i.e. $P(A \mid H_2) = 0.2$ and $P(B \mid H_2) = 0.2$.

1. What is the marginal probability of rain on Saturday, $P(A)$?
2. What about on Sunday, $P(B)$?

EXERCISE 21. Consider the general case of n stations. At the end of each day t, you observe the i-th station's predictions for the probability of rain the next day: $x_{t+1,i} \triangleq P(y_{t+1} \mid H_i, y_1, \ldots, y_t)$. Let $\boldsymbol{x}_t = (x_{t,1}, \ldots, x_{t,n})$ denote the vector of probabilities. You also observe y_t, whether it rained or not that day.

Finally, at time $t = 0$, you have a prior belief $p_{0,i} = P(H_i)$ that the i-th station is correct.

1. Write a program, such that, at each time-step t and given weather history $y_1, \ldots, y_t$ and the predictions of all stations until that time: $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \boldsymbol{x}_{t+1}$, it returns:

   - The posterior probability that each station is correct:

$$p_{t,i} \triangleq P(H_i \mid y_1, \ldots, y_t) \tag{4.7.1}$$

$$= \frac{P(y_t \mid H_i, y_1, \ldots, y_{t-1}) P(H_i \mid y_1, \ldots, y_{t-1})}{P(y_{t+1} \mid y_1, \ldots, y_t)} \tag{4.7.2}$$

$$= \frac{x_{t,i} p_{t-1,i}}{\boldsymbol{x}_t^\top \boldsymbol{p}_{t-1}}, \tag{4.7.3}$$

     where $\boldsymbol{p}_{t-1} = (p_{t-1,1}, \ldots, p_{t-1,n})$ is the vector of priro probabilities.

   - The marginal probability of rain the following day:

$$P(y_{t+1} \mid y_1, \ldots, y_t) = \sum_i P(y_{t+1} \mid H_i, y_1, \ldots, y_t) P(H_i \mid y_1, \ldots, y_t) \tag{4.7.4}$$

$$= \boldsymbol{x}_{t+1}^\top \boldsymbol{p}_t \tag{4.7.5}$$

   - Generate weather data so that it agrees with the predictions of one of the stations, i.e. so that the probability of rain is what that station says. How fast does your posterior converge?
   - Now generate weather data arbitrarily, so that there is no "right" weather station. How does the posterior behave over time?

4.7.1 A medical conundrum

Many patients arriving at an emergency room, suffer from chest pain. This may indicate acute coronary syndrome (ACS). Patients suffering from ACS that go untreated may die with probability 2% in the next few days. Successful diagnosis

results lowers the short-term mortality rate to 0.2%. Consequently, a prompt diagnosis is essential.[5]

Statistics of patients. Approximately 50% of patients presenting with chest pain turn out to suffer from ACS (either acute myocardial infraction or unstable angina pectoris). Approximately 10% suffer from lung cancer. Of ACS sufferers in general, ⅔ are smokers and ⅓ non-smokers. Only ¼ of non-ACS sufferers are smokers. In addition, 90% of lung cancer patients are smokers. Only ¼ of non-cancer patients are smokers.

Assumption 4.7.1. *A patient may suffer from none, either or both conditions!*

Assumption 4.7.2. *When the smoking history of the patient is known, the development of cancer or ACS are independent.*

Tests

One can perform an ECG to test for ACS. An ECG test has *sensitivity* of 66.6% (i.e. it correctly detects ⅔ of all patients that suffer from ACS), and a *specificity* of 75% (i.e. ¼ of patients that do not have ACS, still test positive).

An X-ray can diagnose lung cancer with a sensitivity of 90% and a specificity of 90%.

Assumption 4.7.3. *Repeated applications of a test produce the same result for the same patient, i.e. that randomness is only due to patient variability.*

Assumption 4.7.4. *The existence of lung cancer* does not *affect the probability that the ECG will be positive. Conversely, the existence of ACS* does not *affect the probability that the X-ray will be positive.*

The main problem we want to solve, is how to perform experiments or tests, so as to

- diagnose the patient
- use as few resources as possible.
- make sure the patient lives

This is a problem in *experiment design*. We start from the simplest case, and look at a couple of example where we only observe the results of some tests. We then examine the case where we can select which tests to perform.

Exercise 22. In this exercise, we only worry about making inferences from different tests results.

1. What does the above description imply about the dependencies between the patient condition, smoking and test results? Draw a belief network for the above problem, with the following events (i.e. variables that can be either true or false)
   - A: ACS
   - C: Lung cancer.

[5]The figures given are not really accurate, as they are liberally adapted from different studies.

- S: Smoking
- E: Positive ECG result.
- X: Positive X-ray result.

2. What is the probability that the patient suffers from ACS if $S = \texttt{true}$?
3. What is the probability that the patient suffers from ACS if the ECG result is negative?
4. What is the probability that the patient suffers from ACS if the X-ray result is negative and the patient is a smoker?

4.7.2 The famous medium

EXERCISE 23. Abdul Alhazred [Lovecraft, 1938] claims that he is *psychic* and can *always predict a coin toss*. Let A denote the event that AA is psychic and let $P(A) = 2^{-16}$ be your prior belief that AA is a psychic. Let us now make a set of bets about Abdul. In the following, make the following assumptions

ASSUMPTION 4.7.5. *All experiments are conducted with a fair coin, whose probability of coming heads at the k-th toss is $P(H_k) = 1/2$ and where H_k is independent of $H_{k-1}, \ldots, H_1$.*

ASSUMPTION 4.7.6. *If AA is a psychic, then he can perfectly predict the coin tosses. Thus, if B_k is the event that he predicts correctly, then $P(B_k \mid A, H_k) = P(B_k \mid A) = 1$.*

1. Let B_k denote the event that AA predicts the k-th coin toss correctly. What are the dependencies between A, H_k, B_k? Draw a Bayesian network to represent them. For simplicity, take $k = 1, 2$ only.
2. What is the marginal probability $P(B_1)$? What is the marginal probability $P(B_2)$?
3. Say that AA predicts the first coin toss correctly, i.e. B_1 holds. What is then the marginal probability $P(B_2 \mid B_1)$?

Hint: The only important events for the calculations are whether AA predicts correctly or not.

EXERCISE 24. Instead of assuming that Abdul is either a perfect psychic or a fraud, we can relax our assumption. Let A denote the event that Abdul is psychic, but that he can only predict some proportion $\theta \in [0, 1]$ of tosses (he could also be a *bad* psychic). We shall model our uncertainty about θ with a Beta distribution $\xi(\theta)$ with parameters $(2, 1)$, $\mathcal{Beta}(2, 1)$, which places higher probability on all values of θ closer to 1. So now our prior marginal distribution looks like this:

$$P(B) = P(B \mid A)P(A) + P(B \mid \neg A)P(\neg A) \tag{4.7.6}$$

$$P(B \mid A) = \int_0^1 P(B \mid \theta)\xi(\theta)d\theta = \int_0^1 \theta\xi(\theta)d\theta = \mathbb{E}_\xi\, \theta \tag{4.7.7}$$

$$P(B \mid \neg A) = \frac{1}{2}. \tag{4.7.8}$$

1. What is the resulting posterior distribution $P(B_2 \mid B_1)$ of AA predicting the next toss correctly after he has predicted one correctly? Write this in terms of the resulting $P(A \mid B_1)$ and the resulting Beta posterior parameters, given that $P(A) = 2^{-16}$.
2. Write an expression for the marginal probability $P(B_{n+1} \mid B^n)$.

Chapter 5

Sequential sampling

5.1 Gains from sequential sampling

So far, we have mainly considered decision problems where the sample size was fixed. However, frequently the sample size can also be part of the decision. Since normally larger sample sizes give us more information, in this case the decision problem is only interesting when obtaining new samples has a cost. Consider the following example.

EXAMPLE 27. Consider that you have 100 produced items and you want to determine whether there are fewer than 10 faulty items among them. If testing has some cost, it pays off to think about whether it is possible to do without testing all 100 items. Indeed, this is possible by the following simple online testing scheme: You test one item after another until you either have discovered 10 faulty items or 91 good items. In either case you have the correct answer at considerably lower cost than when testing all items.

A sequential sample from some unknown distribution P is generated as follows. First, let us fix notation and assume that each new sample x_i we obtain belongs in some alphabet $\mathcal{X}$, so that at time t, we have observed $x_1, \ldots, x_t \in \mathcal{X}^t$. It is also convenient to define the set of all sequences in the alphabet $\mathcal{X}$ as $\mathcal{X}^* \triangleq \bigcup_{t=0}^{\infty}$. The distribution P defines a probability on $\mathcal{X}^*$ so that x_{t+1} may depend on the previous samples $x_1, \ldots, x_t$ in an arbitrary manner. At any time t, we can either *stop sampling* or obtain one *more* observation x_{t+1}. A sample obtained in this way is called a *sequential sample*. More formally, we give the following definition:

stopping function

Definition 5.1.1 (Sequential sampling)**.** A sequential sampling procedure on a probability space[1] $(\mathcal{X}^*, \mathfrak{B}(\mathcal{X}^*), P)$ involves a *stopping function* $\pi_s : \mathcal{X}^* \to \{0, 1\}$, such that we stop sampling at time t if and only if $\pi_s(x^t) = 1$, otherwise we obtain a new sample $x_{t+1} \mid x^t \sim P(\cdot \mid x^t)$.

Thus, the sample obtained depends both on P and the sampling procedure π_s. In our setting, we don't just want to sample sequentially, but also to take some action after sampling is complete. For that reason, we can generalise the above definition to sequential decision procedures.

Definition 5.1.2 (Sequential decision procedure)**.** A sequential decision procedure $\pi = (\pi_s, \pi_d)$ is tuple composed of

1. a stopping rule $\pi_s : \mathcal{X}^* \to \{0, 1\}$ and
2. a decision rule $\pi_d : \mathcal{X}^* \to \mathcal{A}$.

The stopping rule π_s specifies whether, at any given time, we should stop and make a decision in $\mathcal{A}$ or take one more sample. That is, stop if

$$\pi_s(x^t) = 1,$$

otherwise observe x_{t+1}. Once we have stopped (i.e. $\pi_s(x^t) = 1$), we choose the decision

$$\pi_d(x^t).$$

[1] This is simply a sample space and associated algebra, together with a probability measure. See Appendix B for a complete definition.

Deterministic stopping rules If the stopping rule π_s is deterministic, then for any t, there exists some *stopping set* $B_t \subset \mathcal{X}^t$ such that *stopping set*

$$\pi_s(x^t) = \begin{cases} 1, & \text{if } if x^t \in B_t \\ 0, & \text{if } x^t \notin B_t. \end{cases} \tag{5.1.1}$$

As with any Bayesian decision problem, it is sufficient to consider only deterministic decision rules.

We are interested in sequential sampling problems especially when there is a reason for us to stop sampling early enough, like the case when we incur a cost with each sample we take. A detailed example is given in the following section.

5.1.1 An example: sampling with costs

We once more consider problems where we have some observations $x_1, x_2, \ldots$, with $x_t \in \mathcal{X}$, which are drawn from some distribution with parameter $\theta \in \Theta$, or more precisely from a family $\mathscr{P} = \{P_\theta \mid \theta \in \Theta\}$, such that each $(\mathcal{X}, \mathfrak{B}(\mathcal{X}), P_\theta)$ is a probability space for all $\theta \in \Theta$. Since we take repeated observations, the probability of a sequence $x^n = x_1, \ldots, x_n$ under an i.i.d. model θ is $P_\theta^n(x^n)$. We have a prior probability measure ξ on $\mathfrak{B}(\Theta)$ for the unknown parameter, and we wish to take an action $a \in \mathcal{A}$ that maximises the expected utility according to a utility function $u : \Theta \times \mathcal{A} \to \mathbb{R}$.

In the classical case, we obtain a complete sample of fixed size n, $x^n = (x_1, \ldots, x_n)$ and calculate a posterior measure $\xi(\cdot \mid x^n)$. We then take the decision maximising the expected utility according to our posterior. Now consider the case of sampling with costs, such that a sample of size n results in a cost of cn. For that reason we define a new utility function U which depends on the number of observations we have.

Samples with costs

$$U(\theta, a, x^n) = u(\theta, a) - cn, \tag{5.1.2}$$

$$\mathbb{E}_\xi(U \mid a, x^n) = \int_\Theta u(\theta, a)\, \mathrm{d}\xi(\theta \mid x^n) - cn. \tag{5.1.3}$$

In the remainder of this section, we shall consider the following simple decision problem, where we need to make a decision the value of an unknown parameter. As we get more data, we have a better chance of discovering the right parameter. However, there is always a small chance of getting no information.

Example 28. Consider the following decision problem, where the goal is to distinguish between two possible hypotheses θ_1, θ_2, with corresponding decisions a_1, a_2. We have three possible observations $\{1, 2, 3\}$, with $1, 2$ being more likely under the first and second hypothesis, respectively. However, the third observation gives us no information about the hypothesis, as its probability is the same under θ_1 and θ_2. In this problem γ is the probability that we obtain an uninformative sample.

- Parameters: $\Theta = \{\theta_1, \theta_2\}$.
- Decisions: $\mathcal{A} = \{a_1, a_2\}$.
- Observation distribution $f_i(k) = \mathbb{P}_{\theta_i}(x_t = k)$ for all t with

$$f_1(1) = 1 - \gamma, \qquad f_1(2) = 0, \qquad f_1(3) = \gamma, \tag{5.1.4}$$
$$f_2(1) = 0, \qquad f_2(2) = 1 - \gamma, \qquad f_2(3) = \gamma. \tag{5.1.5}$$

- Local utility: $u(\theta_i, a_j) = 0$, for $i = j$ and $b < 0$ otherwise.
- Prior: $P_\xi(\theta = \theta_1) = \xi = 1 - P_\xi(\theta = \theta_2)$.
- Observation cost per example: c.

At any step t, you have the option of continuing for one more step, or stopping and taking an action in $\mathcal{A}$. The question is what is the policy for sampling and selecting an action that maximises expected utility?

In this problem, it is immediately possible to distinguish θ_1 from θ_2 when you observe $x_t = 1$ or $x_t = 2$. However, the values $x_t = 3$ provide no information. Hence, the utility of stopping only depends on. So, the expected utility of stopping if you have only observed 3s after t steps is $\xi b - ct$. In fact, if your posterior parameter after t steps is ξ_t, then the expected utility of stopping is $b \min\{\xi_t, 1 - \xi_t\} - ct$. In general, you should expect ξ_t to approach 0 or 1 with high probability, and hence taking more samples is better. However, if we pay utility $-c$ for each additional sample, there is a point of diminishing returns, after which it will not be worthwhile to take any more samples.

value

We first investigate the setting where the number of observations is fixed. In particular, the *value* of the optimal procedure taking n observation is defined to be the expected utility that maximises the *a posteriori* utility given x^n, i.e.

$$V(n) = \sum_{x^n} P_\xi^n(x^n) \max_a \mathbb{E}_\xi(U \mid x^n, a),$$

where $P_\xi^n = \int_\Theta P_\theta^n \, \mathrm{d}\xi(\theta)$ is the marginal distribution over n observations. For this specific example, it is easy to calculate the value of the procedure that takes n observations, by noting the following facts.

(a) The probability of observing $x_t = 3$ for all $t = 1, \ldots, n$ is γ^n. Then we must rely on our prior ξ to make a decision.

(b) If we observe any other sequence, we know the value of θ.

Consequently, the total value $V(n)$ of the optimal procedure taking n observations is

$$V(n) = \xi b \gamma^n - cn. \tag{5.1.6}$$

Based on this, we now want to find the optimal number of samples n. Since V is a smooth function, an approximate maximiser can be found by viewing n as a continuous variable.[2] Taking derivatives, we get

$$n^* = \left[\log \frac{c}{\xi b \log \gamma}\right] \frac{1}{\log \gamma} \tag{5.1.7}$$

$$V(n^*) = \frac{c}{\log \gamma}\left[1 + \log \frac{c}{\xi b \log \gamma}\right] \tag{5.1.8}$$

[2] In the end, we can find the optimal maximiser by looking at the nearest two integers to the value found.

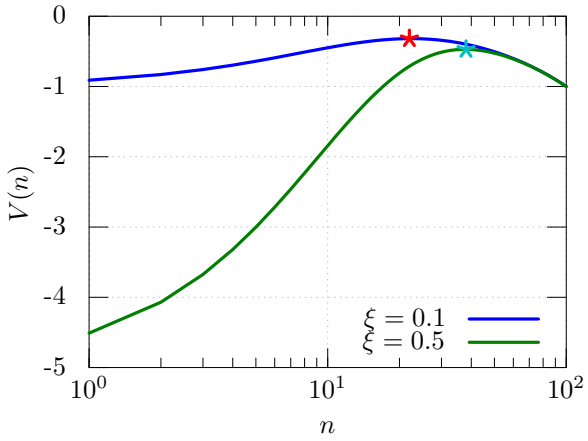


Figure 5.1: Illustration of P1, the procedure taking a fixed number of samples n. The value of taking exactly n observations under two different beliefs, for $\gamma = 0.9$, $b = -10$, $c = 10^{-2}$.

The results of applying this procedure are illustrated in Figure 5.1. Here we can see that, for two different choices of priors, the optimal number of samples is different. In both cases, there is a clear choice for how many samples to take, when we must fix the number of samples before seeing any data.

However, we may *not* be constrained to fix the number of samples a priori. As illustrated in Example 27, many times it is a good idea to adaptively decide when to stop taking samples. This is illustrated by the following *sequential* procedure. In this one, since we already know that there is an optimal *a priori* number of steps n^*, we can choose to look at all possible stopping times for that are smaller or equal to n^*.

P2. A sequential procedure stopping after at most n^* steps.

- If $t < n^*$, use the stopping rule $\pi_s(x^t) = 1.$ iff $x_t = 3$.
- In other words, stop as soon as you observe a 3, or until you reach $t = n^*$.
- Our posterior after stopping is, just $\xi(\theta \mid x^n)$, where both x^n and the number of observations n are random.

Since the probability of $x_t = 3$ is always the same for both θ_1 and θ_2, we have:

$$\mathbb{E}_\xi(n) = \mathbb{E}(n \mid \theta = \theta_1) = \mathbb{E}(n \mid \theta = \theta_2) < n^*$$

We can calculate the expected number of steps as follows:

$$\mathbb{E}_\xi(n \mid n \leq n^*) = \mathbb{E}_\xi(n \mid \theta = \theta_1) = \sum_{t=1}^{n^*} t\, \mathbb{P}_\xi(n = t \mid \theta = \theta_1) \tag{5.1.9}$$

$$= \sum_{t=1}^{n^*-1} t\gamma^{t-1}(1-\gamma) + n^*\gamma^{n^*-1} = \frac{1-\gamma^{n^*}}{1-\gamma}, \tag{5.1.10}$$

from the *geometric series* (see equation C.1.4). Consequently, the value of this procedure is

$$\begin{aligned}\bar{V}(n^*) &= \mathbb{E}_\xi(U \mid n = n^*)\, \mathbb{P}_\xi(n = n^*) + \mathbb{E}_\xi(U \mid n < n^*)\, \mathbb{P}_\xi(n < n^*) \\ &= \xi b \gamma^{n^*} - c\, \mathbb{E}_\xi(n)\end{aligned}$$

and from the definition of n^*:

$$\bar{V}(n^*) = \frac{c}{\gamma - 1} + \frac{c}{\log \gamma}\left[1 + \frac{c}{\xi b(1-\gamma)}\right]. \tag{5.1.11}$$

As you can see, there is a non-zero probability that $n = n^*$, at which time we will have not resolved the true value of θ. In that case, we are still not better off than at the very beginning of the procedure, when we had no observations. If our utility is linear with the number of steps, it thus makes sense that we should still continue. For that reason, we should consider *unbounded procedures*.

unbounded procedures

The unbounded procedure for our example is simply this to use the stopping rule $\pi_s(x^t) = 1$ iff $x_t \neq 3$. Since we only obtain information whenever $x_t \neq 3$, and that information is enough to fully decide θ, once we observe $x_t \neq 3$, we can make a decision that has value 0, as we can guess correctly. So, the value of the unbounded sequential procedure is just $V^* = -c\, \mathbb{E}_\xi(n)$.

$$\mathbb{E}_\xi(n) = \sum_{t=1}^{\infty} t\, \mathbb{P}_\xi(n = t) = \sum_{t=1}^{\infty} t\gamma^{t-1}(1-\gamma) = \frac{1}{1-\gamma}, \tag{5.1.12}$$

again using the formula for the geometric series.

In the given example, it is clear that bounded procedures are (in expectation) better than fixed-sampling procedures, as seen in Figure 5.2. In turn, the unbounded procedure is (in expectation) better than the bounded procedure. Of course, an unbounded procedure may end up costing much more than taking a decision without observing any data, as it disregards the amount spent to time t. This relates to the economic idea of *sunk costs*: since our utility is additive in terms of the cost, our optimal decision now should not be dependent on previously accrued costs.

5.2 Optimal sequential sampling procedures

We now turn our attention to the general case. While it is easy to define the optimal stopping rule and decision in this simple example, how can actually do the same thing for *arbitrary* problems? The following section characterises optimal sequential sampling procedures and gives an algorithm for constructing them.

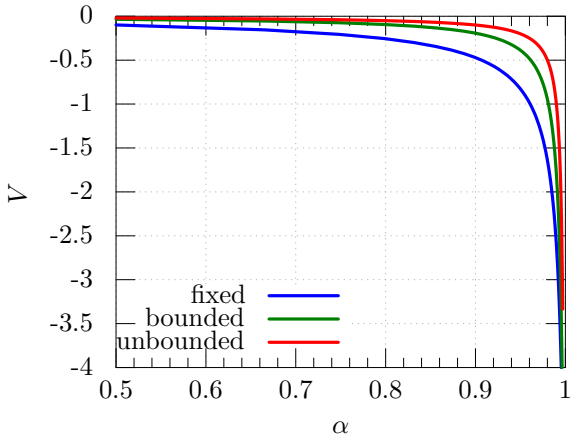


Figure 5.2: The value of three strategies for $\xi = 1/2$, $b = -10$, $c = 10^{-2}$ and varying γ. Higher values of γ imply a longer time before the true θ is known.

Once more, consider a distribution family $\mathscr{P} = \{P_\theta \mid \theta \in \Theta\}$ and a prior ξ over $\mathfrak{B}(\Theta)$. For a decision set $\mathcal{A}$, a utility function $U : \Theta \times \mathcal{A} \to \mathbb{R}$, and a sampling cost c, the utility of a sequential decision procedure is the local utility at the end of the procedure, minus the sampling cost. In expectation, this can be written as

$$U(\xi, \pi) = \mathbb{E}_\xi \left\{ u[\theta, \pi(x^n)] - nc. \right\} \tag{5.2.1}$$

Here the cost is inside the expectation, since the number of samples we take is random. Summing over all the possible stopping times n, and taking $B_n \subset \mathcal{X}^*$ as the set of observations for which we stop, we have:

$$U(\xi, \pi) = \sum_{n=1}^{\infty} \int_{B_n} \mathbb{E}_\xi[U(\theta, \pi(x^n)) \mid x^n] \, \mathrm{d}P_\xi(x^n) - \sum_{n=1}^{\infty} P_\xi(B_n) nc \tag{5.2.2}$$

$$\sum_{n=1}^{\infty} \int_{B_n} \left\{ \int_\Theta U[\theta, \pi(x^n)] \, \mathrm{d}\xi(\theta \mid x^n) \right\} \mathrm{d}P_\xi(x^n) - \sum_{n=1}^{\infty} P_\xi(B_n) nc \tag{5.2.3}$$

where P_ξ is the marginal distribution under ξ. Although it may seem difficult to evaluate this, it can be done by a simple dynamic programming technique called *backwards induction*. We first give the algorithm for the case of bounded procedures (i.e. procedures that must stop after a particular time) and later for unbounded ones.

Definition 5.2.1 (Bounded sequential decision procedure)**.** A sequential decision procedure δ is *bounded* if there is a positive integer T such that $\mathbb{P}_\xi(n \leq T) = 1$. Similarly, the procedure is T-bounded if it is bounded for a specific T.

We can analyse such a procedure by recursively analysing procedures of larger T, starting from the final point of the process and working our way backwards. Consider a π that is T-bounded. Then we know that we shall take at most T samples. If the process ends at stage T, we will have observed some

sequence x^T, which gives rise to a posterior $\xi(\theta \mid x^T)$. Since we *must* stop at T, we must choose a maximising expected utility at that stage:

$$\mathbb{E}_\xi[U \mid x^T, a] = \int_\Theta U(\theta, a) \, \mathrm{d}\xi(\theta \mid x^T)$$

Since need not take another sample, the respective value (maximal expected utility) of that stage is:

$$V^0[\xi(\cdot|x^T)] \triangleq \max_{a \in \mathcal{A}} U(\xi(\cdot \mid x^T), a)$$

where we introduce the notation V^n to denote the expected utility, given that we are stopping after at most n steps.

More generally, we need to consider the effect on subsequent decisions. Consider the following simple two-stage problem as an example. Let $\mathcal{X} = \{0, 1\}$ and the prior ξ on the θ parameter of $\mathcal{Bern}(\theta)$. We wish to either decide immediately on a parameter θ, or take one more observation, at cost c, before deciding. The problem we consider has two stages, as illustrated in Figure 5.3. In this exam-

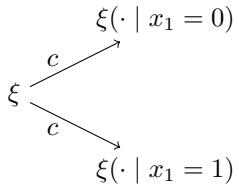


Figure 5.3: An example of a sequential decision problem with two stages. The initial belief is ξ and there are two possible subsequent beliefs, depending on whether we observe $x_t = 0$ or $x_t = 1$. At each stage we pay c.

ple, we begin with a prior ξ at the first stage. There are two possible outcomes for the **second stage**.

1. If we observe $x_1 = 0$ then our value is $V^0[\xi(\cdot \mid x_1 = 0)]$.
2. If we observe $x_1 = 1$ then our value is $V^0[\xi(\cdot \mid x_1 = 1)]$.

At the first stage, we can:

1. Stop with value $V^0(\xi)$.
2. Pay a sampling cost c for value: $V^0[\xi(\cdot \mid x_1)]$, with $P_\xi(x_1) = \int_\Theta P_\theta(x_1) \, \mathrm{d}\xi(w)$.

So the expected value of continuing for one more step is

$$V^1(\xi) \triangleq \int_{\mathcal{X}} V^0[\xi(\cdot \mid x_1)] \, \mathrm{d}P_\xi(x_1).$$

Thus, the overall value for this problem is:

$$\max \left\{ V^0(\xi), \sum_{x_1=0}^{1} V^0[\xi(\cdot \mid x_1)] P_\xi(x_1) - c. \right\}$$

The above is simply the maximum of the value of stopping immediately (V^0), and the value of continuing for at most one more step (V^1). This procedure can be applied recursively for multi-stage problems, as explained below.

5.2.1 Multi-stage problems

For simplicity, we use ξ_n to denote a posterior $\xi(\cdot \mid x^n)$, omitting the specific value of x^n. For any specific ξ_n, there is a range of given possible next beliefs ξ_{n+1}, depending on what the value of the next observation x_n is. This is illustrated in Figure 5.4, by extension from the previous two-stage example. The

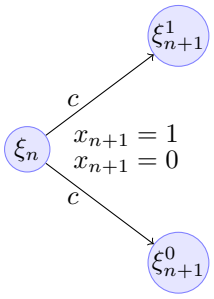


Figure 5.4: A partial view of the multi-stage process.

value of the process can be calculated as follows, more generally:

$$V^0(\xi_t) = \sup_{a \in \mathcal{A}} \int_\Theta u(\theta, a) \, \mathrm{d}\xi_t(\theta) \qquad \text{(Immediate value)}$$

$$\xi_n(\cdot) \triangleq \xi(\cdot \mid x^n) \qquad \text{(posterior)}$$

$$\mathbb{E}_{\xi_n} V^0(\xi_{n+1}) = \int_{\mathcal{X}^n} V^0[\xi_n(\cdot \mid x_n)] \, \mathrm{d}\xi_n(x_n) \qquad \text{(Next-step value)}$$

$$V^1(\xi_n) = \max \left\{ V^0(\xi_n), \mathbb{E}_{\xi_n} V^0(\xi_{n+1}) - c \right\} \qquad \text{(Optimal value)}$$

The immediate value is the expected value if we stop immediately at time t. The next-step value is the expected value of the next step, ignoring the cost. Finally, the optimal value at the n-th step is just the maximum of the value of stopping immediately and the next-step value. We can generalise this procedure over all steps $1, 2, \ldots, T$, to obtain a general procedure.

5.2.2 Backwards induction for bounded procedures

The main idea expressed in the previous section is to start from the last stage of our decision problem, where the utility is known, and then move backwards. At each stage, we know the probability of reaching different points in the next stage, as well as their values. Consequently, we can compute the value of any point in the current stage as well. This idea is formalised below, via the algorithm of backwards induction.

Theorem 5.2.1 (Backwards induction)**.** *The utility of a T-bounded optimal procedure with prior ξ_0 is $V^T(\xi_0)$ and is given by the recursion:*

$$V^{j+1}(\xi_n) = \max \left\{ V^0(\xi_n), \mathbb{E}_{\xi_n} V^j(\xi_{n+1}) - c \right\} \tag{5.2.4}$$

for every belief ξ_n in the set of beliefs that arise from the prior ξ_0, with $j = T - n$.

The proof of this theorem follows by induction. However, we shall prove a more general version in Chapter 6. Equation 5.2.4 essentially gives a recursive calculation of the value of the T-bounded optimal procedure. To evaluate it, we first need to calculate all possible beliefs $\xi_1, \ldots, \xi_T$. For each belief ξ_T, we calculate $V^0(\xi_T)$. We then move backwards, and calculate $V^0(\xi_{T-1})$ and $V^1(\xi_{T-1})$. Proceeding backwards, for $n = T-1, T-2, \ldots, 1$, we calculate $V^{T+1}(\xi_n)$ for all beliefs ξ_n with $j = T - n$. The value of the procedure also determines the optimal sampling strategy, as shown by the following theorem.

Theorem 5.2.2. *The optimal T-bounded procedure stops at time t if the value of stopping at t is better than that of continuing, i.e. if*

$$V^0(\xi_t) \geq V^{T-t}(\xi_t).$$

This procedure chooses a maximising $\mathbb{E}_{\xi_t} U(\theta, a)$, otherwise takes one more sample.

Finally, longer procedures (i.e. procedures that allow for stopping later) are always better than shortest ones, as shown by the following theorem.

Theorem 5.2.3. *For any probability measure ξ on Θ,*

$$V^n(\xi) \leq V^{n+1}(\xi). \tag{5.2.5}$$

That is, the procedure that stops after at most n steps is never better than the procedure that stops after at most $n+1$ time steps. To obtain an intuition of why this is the case, consider the example of Section 5.1.1. In that example, if we have a sequence of 3s, then we obtain no information. Consequently, when we compare the value of a plan taking at most n samples with that of a plan taking at most $n+1$ samples, we see that the latter plan is better for the event where we obtain n 3s, but has the same value for all other events.

5.2.3 Unbounded sequential decision procedures

Given the monotonicity of the value of bounded procedures (5.2.5), one may well ask what is the value of unbounded procedures, i.e. procedures that may never stop sampling. The value of an unbounded sampling and decision procedure π under prior ξ is

$$U(\xi, \pi) = \int_{\mathcal{X}^*} \left\{ V^0[\xi(\cdot \mid x^n)] - cn \right\} \mathrm{d}P_\xi^\pi(x^n) = \mathbb{E}_\xi^\pi \left\{ V^0[\xi(\cdot \mid x^n)] - cn \right\},$$

where $P_\xi^\pi(x^n)$ is the probability that we observe samples x^n and stop under the marginal distribution defined by ξ and π, while n is the random number of samples taken by π. As before, this is random because the observations x are random; π itself can be deterministic.

Definition 5.2.2 (Regular procedure)**.** Given a procedure π, l $B_{>k}(\pi) \subset \mathcal{X}^*$ be the set of sequences such that π takes more than k samples. Then π is regular if $U(\xi, \pi) \geq V^0(\xi)$ and if, for all $n \in \mathbb{N}$, and for all $x^n \in B_{>n}(\pi)$

$$U[\xi(\cdot \mid x^n), \pi] \geq V^0[\xi(\cdot \mid x^n)] - cn, \tag{5.2.6}$$

i.e. the expected utility given for any sample that starts with x^n where we don't stop, is greater than that of stopping at n.

In other words, if π specifies that at least one observation should be taken, then the value of π is greater than the value of choosing a decision without any observation. Furthermore, whenever π specifies that another observation should be taken, the expected value of continuing must be larger than the value of stopping. If the procedure is *not* regular, then there may be stages where the procedure specifies that sampling should be continued, though the value may not increased by doing so.

Theorem 5.2.4. *If π is not regular, then there exists a regular π' such that $U(\xi, \pi') \geq U(\xi, \pi)$.*

Proof. First, consider the case that π is not regular because $U(\xi, \pi) \leq V^0(\xi)$. Then π' can be the regular procedure which chooses $a \in \mathcal{A}$ without any observations.

Now consider the case that $U(\xi, \pi) > V^0(\xi)$ and that π specifies at least one sample should be taken. Let π' be the procedure which stops as soon as the observed x^n does not satisfy (5.2.6).

If π stops, then both sides of (5.2.6) are equal, as the value of stopping immediately is at least as high as that of continuing. Consequently, π' stops no later than π for any x^n. Finally, let

$$B_k(\pi) = \{x \in \mathcal{X}^* \mid n = k\} \tag{5.2.7}$$

be the set of observations such that exactly k samples are taken by rule π and

$$B_{\leq k}(\pi) = \{x \in \mathcal{X}^* \mid n \leq k\} \tag{5.2.8}$$

be the set of observations such that at most k samples are taken by rule π. Then

$$\begin{aligned} U(\xi, \pi') &= \sum_{k=1}^{\infty} \int_{B_k(\pi')} \{V^0[\xi(\cdot \mid x^k) - ck]\} \, \mathrm{d}P_\xi(x^k) \\ &\geq \sum_{k=1}^{\infty} \int_{B_k(\pi')} U[\xi(\cdot \mid x^n, \pi)] \, \mathrm{d}P_\xi(x^k) \\ &= \sum_{k=1}^{\infty} \mathbb{E}_\xi^\pi \{U \mid B_k(\pi')\} P_\xi(B_k(\pi')) = E_\xi^\pi U = U(\xi, \pi). \end{aligned}$$

□

5.2.4 The sequential probability ratio test

Sometimes we wish to collect just enough data in order to be able to confirm or disprove a particular hypothesis. More specifically, we have a set of parameters Θ, and we need to pick the right one. However, rather than simply using an existing set of data, we are collecting data sequentially, and we need to decide when to stop and select a model. In this case, each one of our decisions a_i corresponds to choosing the model θ_i, and we have a utility function that favours our picking the correct model. As before, data collection has some cost, which we must balance against the expected utility of picking a parameter.

As an illustration, consider a problem where we must decide for one out of two possible parameters θ_1, θ_2. At each step, we can either take another sample from the unknown $P_\theta(x_t)$, or decide for one or the other of the parameters.

EXAMPLE 29 (A two-point sequential decision problem.). Consider a problem where there are two parameters and two final actions which select one of the two parameter values, such that:

- Observations $x_t \in \mathcal{X}$
- Distribution family: $\mathscr{P} = \{P_\theta \mid \theta \in \Theta\}$
- Probability space $(\mathcal{X}^*, \mathfrak{B}(\mathcal{X}^*), P_\theta)$.
- Parameter set $\Theta = \{\theta_1, \theta_2\}$.
- Action set $\mathcal{A} = \{a_1, a_2\}$.
- Prior $\xi = \mathbb{P}(\theta = \theta_1)$.
- Sampling cost $c > 0$.

The actions we take upon stopping can be interpreted as guessing the parameter. When we guess wrong, we suffer a cost, as seen in the following table:

$U(\theta, d)$	a_1	a_2
θ_1	0	λ_1
θ_2	λ_2	0

Table 5.1: The local utility function, with $\lambda_1, \lambda_2 < 0$

As will be the case for all our sequential decision problems, we only need to consider our current belief ξ, and its possible evolution, when making a decision. To obtain some intuition about this procedure, we are going to analyse this problem by examining what the optimal decision is under all possible beliefs ξ.

Under some belief ξ, the immediate value (i.e. the value we obtain if we stop immediately), is simply:

$$V^0(\xi) = \max\{\lambda_1 \xi, \lambda_2(1-\xi)\}. \tag{5.2.9}$$

The worst-case immediate value, i.e. the minimum, is attained when both terms are equal. Consequently, setting $\lambda_1 \xi = \lambda_2(1-\xi)$, gives $\xi = \lambda_2/(\lambda_1 + \lambda_2)$. Intuitively, this is the worst-case belief, as the uncertainty it induces leaves us unable to choose between either hypothesis. Replacing in (5.2.9) gives a lower bound for the value for any belief.

$$V^0(\xi) \geq \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}.$$

Let $\mathit{\Pi}$ denote the set of procedures π which take at least one observation and define:

$$V'(\xi) = \sup_{\pi \in \mathit{\Pi}} U(\xi, \pi). \tag{5.2.10}$$

Then the ξ-expected utility $V^*(\xi)$ must satisfy

$$V^*(\xi) = \max\left\{V^0(\xi), V'(\xi)\right\}. \tag{5.2.11}$$

As we showed in Section 3.3.1, V' is a convex function of ξ. Now let

$$\Xi_0 \triangleq \left\{\xi \;\middle|\; V^0(\xi) \geq V'(\xi)\right\} \tag{5.2.12}$$

be the set of priors where it is optimal to terminate sampling. It follows that $\Xi \setminus \Xi_0$, the set of priors where we must not terminate sampling, is a convex set.

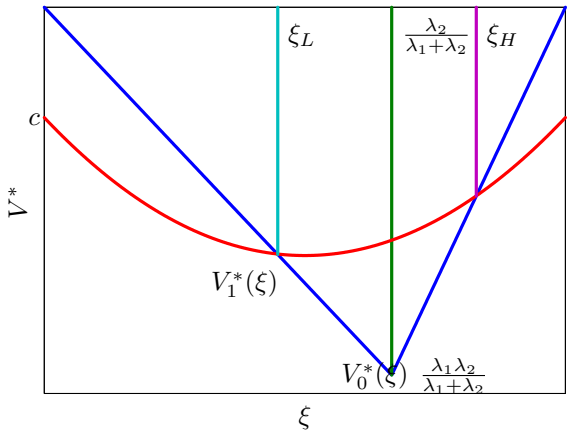


Figure 5.5: The value of the optimal continuation V' versus stopping V^0.

Figure 5.5 illustrates the above arguments, by plotting the immediate value against the optimal continuation after taking one more sample. For the worst-case belief, we must always continue sampling. When we are absolutely certain about the model, then it's always better to stop immediately. There are two points where the curves intersect. Together, these define three subsets of beliefs: On the left, if $\xi < \xi_L$, we decide for one parameter θ_0. On the right, if $\xi > \xi_H$, we decide for the other parameter, θ_1. Otherwise, we continue sampling. This is the main idea of the sequential probability ratio test, explained below.

The sequential probability ratio test (SPRT)

Figure 5.5 offers a graphical illustration of when it is better ot take one more sample in this setting. In particular, if $\xi \in (\xi_L, \xi_T)$, then it is optimal to take at least one more sample. Otherwise, it is optimal to make an immediate decision with value $\rho_0(\xi)$.

This has a nice interpretation as a standard tool in statistics: the *sequential probability ratio test.* First note that our posterior at time t can be written as

$$\xi_t = \frac{\xi P_{\theta_1}(x^t)}{\xi P_{\theta_1}(x^t) + (1-\xi)P_{\theta_2}(x^t)}.$$

Then, for any posterior, the optimal procedure is:

- If $\xi_L < \xi_t < \xi_T$, take one more sample.
- If $x_L \geq \xi_t$, stop and choose a_2.
- If $x_T \leq \xi_t$, stop and choose a_1.

We can now restate the optimal procedure in terms of a probability ratio, i.e. we should always take another observation as long as

$$\frac{\xi(1-\xi_T)}{(1-\xi)\xi_T} < \frac{P_{\theta_2}(x^t)}{P_{\theta_1}(x^t)} < \frac{\xi(1-\xi_L)}{(1-\xi)\xi_L}.$$

If the first inequality is violated, we choose a_1. If the second inequality is violated, we choose a_2. So, there is an equivalence between SPRT and optimal sampling procedures, when the optimal policy is to continue sampling whenever our belief is within a specific interval.

5.2.5 Wald's theorem

An important tool in the analysis of SPRT as well as other procedures that stop at random times is the following theorem by Wald.

Theorem 5.2.5 (Wald's theorem)**.** *Let $z_1, z_2, \dots$ be a sequence of i.i.d. random variables with measure G, such that $\mathbb{E}\, z_i = m$ for all i. Then for any sequential procedure with $\mathbb{E}\, n < \infty$:*

$$\mathbb{E} \sum_{i+1}^{n} z_i = m\, \mathbb{E}\, n. \tag{5.2.13}$$

Proof.

$$\begin{aligned}
\mathbb{E} \sum_{i=1}^{n} z_i &= \sum_{k=1}^{\infty} \int_{B_k} \sum_{i=1}^{k} z_i \, \mathrm{d}G^k(z^k) \\
&= \sum_{k=1}^{\infty} \sum_{i=1}^{k} \int_{B_k} z_i \, \mathrm{d}G^k(z^k). \\
&= \sum_{i=1}^{\infty} \sum_{k=i}^{\infty} \int_{B_k} z_i \, \mathrm{d}G^k(z^k) \\
&= \sum_{i=1}^{\infty} \int_{B_{\geq i}} z_i \, \mathrm{d}G^i(z^i) \\
&= \sum_{i=1}^{\infty} \mathbb{E}(z_i)\, \mathbb{P}(n \geq i) = m\, \mathbb{E}\, n.
\end{aligned}$$

□

We now consider an application of this theorem to the SPRT. Let $z_i = \log \frac{P_{\theta_2}(x_i)}{P_{\theta_1}(x_i)}$. Consider the equivalent formulation of the SPRT which uses

$$a < \sum_{i=1}^{n} z_i < b$$

as the test. Using Wald's theorem and the previous properties and assuming $c \approx 0$, we obtain the following approximately optimal values for a, b:

$$a \approx \log c - \log \frac{I_1 \lambda_2 (1-\xi)}{\xi} \qquad b \approx \log \frac{1}{c} - \log \frac{I_2 \lambda_1 \xi}{1-\xi}, \tag{5.2.14}$$

where $I_1 = -\mathbb{E}(z \mid \theta = \theta_1)$ and $I_2 = \mathbb{E}(z \mid \theta = \theta_2)$ is the *information*, better known as the *KL divergence*. If the cost c is very small, then the information terms vanish and we can approximate the values by $\log c$ and $\log \frac{1}{c}$.

5.3 Martingales

Martingales are a fundamentally important concept in the analysis of stochastic processes where the expectation at time $t+1$ only depends on the state of the process at time t.

An example of a martingale sequence is when x_t is the amount of money you have at a given time, and where at each time-step t you are making a gamble such that you lose or gain 1 currency unit with equal probability. Then, at any step t, it holds that $\mathbb{E}(x_{t+1} \mid x_t) = x_t$. This concept can be generalised to two random processes x_t and y_t, which are dependent.

Definition 5.3.1. Let $x^n \in \mathcal{S}^n$ be a sequence of observations with distribution P_n, and $y_n : \mathcal{S}^n \to \mathbb{R}$ be a random variable. Then the sequence $\{y_n\}$ is a *martingale with respect to* $\{x_n\}$ if for all n the expectation

$$\mathbb{E}(y_n) = \int_{\mathcal{S}^n} y_n(x^n) \, \mathrm{d}P_n(x^n) \tag{5.3.1}$$

exists and

$$\mathbb{E}(y_{n+1} \mid x^n) = y_n \tag{5.3.2}$$

holds with probability 1. If $\{y_n\}$ is a martingale with respect to itself, i.e. $y_i(x) = x$, then we call it simply a *martingale*.

It is also useful to consider the following generalisations of martingale sequences

Definition 5.3.2. A sequence $\{y_n\}$ is a super-martingale if $\mathbb{E}(y_{n+1} \mid x^n) \leq y_n$ and a sub-martingale if $\mathbb{E}(y_{n+1} \mid x^n) \geq y_n$, w.p. 1.

At a first glance, it might appear that martingales are not very frequently encountered, apart from some niche applications. However, we can always construct a martingale from any sequence of random variables as follows.

Definition 5.3.3 (Doob martingale)**.** Consider a function $f : \mathcal{S}^m \to \mathbb{R}$ and some associated random variables $x^m \triangleq x_1, \dots x_m$. Then, for any $n \leq m$, assuming the expectation $\mathbb{E}(f \mid x^n) = \int_{\mathcal{S}^{m-n}} f(x^m) \, \mathrm{d}\,\mathbb{P}(x_{n+1}, \dots, x_m \mid x^n)$ exists, we can construct the random variable

$$y_n(x^n) = \mathbb{E}[f \mid x^n].$$

Then $\mathbb{E}(y_{n+1} \mid x^n) = y_n$, and so y_n is a martingale sequence with respect to x_n.

Another interesting type of martginale sequence are martingale *difference* sequences. They are particularly important as they are related to some useful concentration bounds.

Definition 5.3.4. A sequence $\{y_n\}$ is a *martingale difference sequence* with respect to $\{x_n\}$ if

$$\mathbb{E}(y_{n+1} \mid x^n) = 0 \qquad \text{with probability 1.} \tag{5.3.3}$$

For bounded difference sequences, the following well-known concentration bound holds.

Theorem 5.3.1. *Let b_k be a random variable depending on x^{k-1} and $\{y_k\}$ be a martingale difference sequence with respect to the $\{x_k\}$, such that $y_k \in [b_k, b_k + c_k]$ w.p. 1, Then, defining $s_k \triangleq \sum_{i=1}^{k} y_i$, it holds that:*

$$\mathbb{P}(s_n \geq t) \leq \exp\left(\frac{-2t^2}{\sum_{i=1}^{n} c_i^2}\right). \tag{5.3.4}$$

This allows us to bound the probability that the difference sequence deviates from zero. Since there are only few problems where the default random variables are difference sequences, use of this theorem is most common by defining a new random variable sequence that is a difference sequence.

5.4 Markov processes

A more general type of sequence of random variables than martingales are Markov processes. Informally speaking, a Markov process is a sequence of variables $\{x_n\}$ such that the next value x_{t+1} only depends on the current value x_t.

Definition 5.4.1 (Markov Process)**.** Let $(\mathcal{S}, \mathfrak{B}(\mathcal{S}))$ be a measurable space. If $\{x_n\}$ is a sequence of random variables $x_n : \mathcal{S} \to \mathcal{X}$ such that

$$\mathbb{P}(x_t \in A \mid x_{t-1}, \ldots, x_1) = \mathbb{P}(x_t \in A \mid x_{t-1}), \qquad \forall A \in \mathfrak{B}(\mathcal{X}), \tag{5.4.1}$$

i.e. x_t is independent of $x_{t-2}, \ldots$ given x_{t-1}, then $\{x_n\}$ is a Markov process, and x_t is called the *state* of the Markov process at time t. If $\mathbb{P}(x_t \in A \mid x_{t-1} = x) = \tau(A \mid x)$ where $\tau : \mathfrak{B}(\mathcal{S}) \times \mathcal{S} \to [0, 1]$, is the *transition kernel*, then $\{x_n\}$ is a *stationary Markov process*

stationary Markov process

Note that is the sequence of posterior parameters obtained in Bayesian inference is a Markov process.

5.5 Exercises.

EXERCISE 25. Consider a stationary Markov process with state space S and whose transition kernel is a matrix $\boldsymbol{\tau}$. At time t, we are at state $x_t = s$ and we can either, 1: Terminate and receive reward $b(s)$, or 2: Pay $c(s)$ and continue to a random state x_{t+1} from the distribution $\boldsymbol{\tau}(z' \mid z)$.

Assuming $b, c > 0$ and $\boldsymbol{\tau}$ are known, design a backwards induction algorithm that optimises for the utility function

$$U(x_1, \dots, x_T) = b(x_T) - \sum_{t=1}^{T-1} c(x_t).$$

Finally, show that the expected utility of the optimal policy starting from any state must be bounded.

EXERCISE 26. Consider the problem of classification with features $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$, where each label costs $c > 0$. Assume a Bayesian model with some parameter space Θ on which we have a prior distribution ξ_0. Let ξ_t be the posterior distribution after t examples $(x_1, y_1), \dots, (x_t, y_t)$.

Let our expected utility be the expected accuracy (i.e. the marginal probability of correctly guessing the right label over all possible models) of the Bayes-optimal classifier $\pi : \mathcal{X} \to \mathcal{Y}$ minus the cost paid:

$$\mathbb{E}_t(U) \triangleq \max_{\pi} \int_{\Theta} \int_{\mathcal{X}} P_\theta(\pi(x) \mid x) \, \mathrm{d}P_\theta(x) \, \mathrm{d}\xi_t(\theta) - ct$$

Show that the Bayes-optimal classification accuracy after t observations can be rewritten as

$$\int_{\Theta} \int_{\mathcal{X}} \max_{y \in \mathcal{Y}} P_\theta(y \mid x) \, \mathrm{d}\xi_t(\theta \mid x) \, \mathrm{d}\, \mathbb{P}_t(x), -$$

where $\mathbb{P}_t$ and $\mathbb{E}_t$ denote marginal distributions under the belief ξ_t.

Write the expression for the expected gain in accuracy when obtaining one more sample and label.

Implement the above for a model family of your choice. Two simple options are the following. The first is a finite model family composed of two different classifiers $P_\theta(y \mid x)$. The second is the family of discrete classifier models with a Dirichlet product prior, i.e. where $\mathcal{X} = \{1, \dots, n\}$, and each different $x \in \mathcal{X}$ corresponds to a different multinomial distribution over $\mathcal{Y}$. In both cases, you can assume a common (and known) data distribution $P(x)$, in which case $\xi_t(\theta \mid x) = \xi_t(\theta)$.

Figure 5.6 shows the performance for a family of discrete classifier models with $|\mathcal{X}| = 4$. It shows the **expected** classifier performance (based on the posterior marginal), the **actual** performance on a small test set, as well as the cumulative **predicted** performance gain. As you can see, even though the expected performance gain is zero in some cases, cumulatively it reaches the actual performance of the classifier. You should be able to produce a similar figure for your own setup.

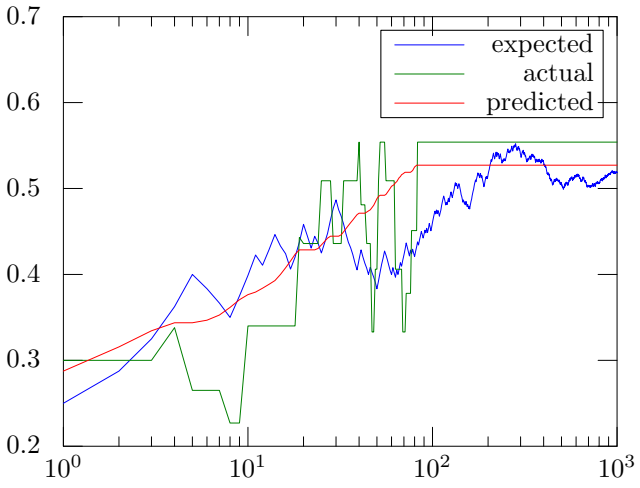


Figure 5.6: Illustrative results for an implementation of Exercise 26 on a discrete classifier model.

Chapter 6

Experiment design and Markov decision processes

6.1 Introduction

This unit describes the very general formalism of Markov decision processes (MDPs) for formalising problems in sequential decision making. Thus a *Markov decision process* can be used to model stochastic path problems, stopping problems, reinforcement learning problems, experiment design problems, and control problems.

experimental design

We begin by taking a look at the problem of *experimental design*. One instance of this problem occurs when considering how to best allocate treatments with unknown efficacy to patients in an adaptive manner, so that the best treatment is found, or so as to maximise the number of patients that are treated successfully. The problem, originally considered by Chernoff [1959, 1966], informally can be stated as follows.

We have a number of treatments of unknown efficacy, i.e. some of them work better than the others. We observe patients one at a time. When a new patient arrives, we must choose which treatment to administer. Afterwards, we observe whether the patient improves or not. Given that the treatment effects are initially unknown, how can we maximise the number of cured patients? Alternatively, how can we discover the best treatment? The two different problems are formalised below.

Adaptive treatment allocation

EXAMPLE 30. Consider k treatments to be administered to T volunteers. To each volunteer only a single treatment can be assigned. At the t-th trial, we treat one volunteer with some treatment $a_t \in \{1, \dots, k\}$. We then obtain a reward $r_t = 1$ if the patient is healed and 0 otherwise. We wish to choose actions maximising the utility $U = \sum_t r_t$. This would correspond to maximising the number of patients that get healed over time.

Adaptive hypothesis testing

EXAMPLE 31. An alternative goal would be to do a *clinical trial*, in order to find the best possible treatment. For simplicity, consider the problem of trying to find out whether a particular treatment is better or not than a placebo. We are given a hypothesis set Ω, with each $\omega \in \Omega$ corresponding to different models for the effect of the treatment and the placebo. Since we don't know what is the right model, we place a prior ξ_0 on Ω. We can perform T experiments, after which we must make decide whether or not the treatment is significantly better than the placebo. To model this, we define a decision set $\mathcal{D} = \{d_0, d_1\}$ and a utility function $U : \mathcal{D} \times \Omega \to \mathbb{R}$ which models the effect of each decision d given different versions of reality ω. One hypothesis $\omega \in \Omega$ is true. To discover the truth, we can choose from a set of k possible experiments to be performed over T trials. At the t-th trial, we choose experiment $a_t \in \{1, \dots, k\}$ and observe outcome $x_t \in \mathcal{X}$, with $x_t \sim P_\omega$ drawn from the true hypothesis. Our posterior is

$$\xi_t(\omega) \triangleq \xi_0(\omega \mid a_1, \dots, a_t, x_1, \dots, x_t).$$

The reward is $r_t = 0$ for $t < T$ and

$$r_T = \max_{d \in D} \mathbb{E}_{\xi_T}(U \mid d).$$

Our utility can again be expressed as a sum over individual rewards, $U = \sum_{t=1}^{T} r_t$.

Both formalizations correspond to so-called *bandit problems* which we take a closer look at in the following section.

6.2 Bandit problems

The simplest bandit problem is the stochastic n-armed bandit. We are faced with n different one-armed bandit machines, such as those found in casinos. In this problem, at time t, you have to choose one *action* (i.e. a machine) $a_t \in \mathcal{A} = \{1, \dots, n\}$. In this setting, each time t you play a machine, you receive a reward r_t, with fixed expected value $\omega_i = \mathbb{E}(r_t \mid a_t = i)$. Unfortunately, you do not know the ω_i, and consequently the best arm is also unknown. How do you then choose arms so as to maximise the total expected reward?

Definition 6.2.1 (The stochastic n-armed bandit problem.)**.** This is the problem of selecting a sequence of actions $a_t \in \mathcal{A}$, with $\mathcal{A} = \{1, \dots, n\}$, so as to maximise expected utility, where the utility is

$$U = \sum_{t=0}^{T-1} \gamma^t r_t,$$

where $T \in (0, \infty]$ is the horizon and $\gamma \in (0, 1]$ is a *discount factor*. The reward r_t is stochastic, and only depends on the current action, with expectation $\mathbb{E}(r_t \mid a_t = i) = \omega_i$. *discount factor*

In order to select the actions, we must specify some *policy* or decision rule. Such a rule can only depend on the sequence of previously taken actions and observed rewards. Usually, the policy $\pi : \mathcal{A}^* \times \mathbb{R}^* \to \mathcal{A}$ is a deterministic mapping from the space of all sequences of actions and rewards to actions. That is, for every observation and action history $a_1, r_1, \dots, a_{t-1}, r_{t-1}$ it suggests a single action a_t. More generally, it could also be a stochastic policy, that specifies a mapping to action distributions. We use the notation *policy*

$$\pi(a_t \mid a^{t-1}, r^{t-1}) \tag{6.2.1}$$

for stochastic history-dependent bandit policies, i.e. the probability of actions a_t given the history until time t.

How can we solve bandit problems? One idea is to apply the Bayesian decision-theoretic framework we have developed earlier to maximise utility in expectation. More specifically, given the horizon $T \in (0, \infty]$ and the discount factor $\gamma \in (0, 1]$, we define our utility from time t to be:

$$U_t \triangleq \sum_{k=1}^{T-t} \gamma^k r_{t+k}. \tag{6.2.2}$$

To apply the decision theoretic framework, we need to define a suitable family of probability measures $\mathscr{P}$, indexed by parameter $\omega \in \Omega$ describing the reward distribution of each bandit, together with a prior distribution ξ on Ω. Since ω is unknown, we cannot maximise the expected utility with respect to it. However, we can always maximise expected utility with respect to our belief ξ. That is, we replace the ill-defined problem of maximising utility in an unknown model with that of maximising expected utility given a distribution over possible models. The problem can be written in a simple form as

$$\max_\pi \mathbb{E}^\pi_\xi U_t = \max_\pi \int_\Omega \mathbb{E}^\pi_\omega U_t \, \mathrm{d}\xi(\omega). \tag{6.2.3}$$

The following figure summarises the statement of the bandit problem in the Bayesian setting.

Decision-theoretic statement of the bandit problem

- Let $\mathcal{A}$ be the set of arms.
- Define a family of distributions $\mathscr{P} = \{P_{\omega,i} \mid \omega \in \Omega, i \in \mathcal{A}\}$ on $\mathbb{R}$.
- Assume the i.i.d model $r_t \mid \omega, a_t = i \sim P_{\omega,i}$.
- Define prior ξ on Ω.
- Select a policy $\pi : \mathcal{A}^* \times \mathbb{R}^* \to \mathcal{A}$ maximising

$$\mathbb{E}_\xi^\pi U = \mathbb{E}_\xi^\pi \sum_{t=0}^{T-1} \gamma^t r_t$$

There are two main difficulties with this approach. The first is specifying the family and the prior distribution: this is effectively part of the problem formulation and can severely influence the solution. The second is calculating the policy that maximises expected utility given a prior and family. The first problem can be resolved by either specifying a subjective prior distribution, or by selecting a prior distribution that has good worst-case guarantees. The second problem is hard to solve, because in general, such policies are history dependent and the set of all possible histories is exponential in the horizon T.

6.2.1 An example: Bernoulli bandits

As a simple illustration, consider the case when the reward for choosing one of the n actions is either 0 or 1, with some fixed, yet unknown probability depending on the chosen action. This can be modelled in the standard Bayesian framework using the Beta-Bernoulli conjugate prior. More specifically, we can formalise the problem as follows.

Consider n Bernoulli distributions with unknown parameters ω_i $(i = 1, \ldots, n)$ such that

$$r_t \mid a_t = i \sim \mathit{Bern}(\omega_i), \qquad \mathbb{E}(r_t \mid a_t = i) = \omega_i. \tag{6.2.4}$$

Each Bernoulli distribution thus corresponds to the distribution of rewards obtained from each bandit that we can play. In order to apply the statistical decision theoretic framework, we have to quantify our uncertainty about the parameters ω in terms of a probability distribution.

We model our belief for each bandit's parameter ω_i as a Beta distribution $\mathit{Beta}(\alpha_i, \beta_i)$, with density $f(\omega \mid \alpha_i, \beta_i)$ so that

$$\xi(\omega_1, \ldots, \omega_n) = \prod_{i=1}^{n} f(\omega_i \mid \alpha_i, \beta_i).$$

Recall that the posterior of a Beta prior is also a Beta. Let

$$N_{t,i} \triangleq \sum_{k=1}^{t} \mathbb{I}\{a_k = i\}$$

be the number of times we played arm i and

$$\hat{r}_{t,i} \triangleq \frac{1}{N_{t,i}} \sum_{k=1}^{t} r_t \, \mathbb{I}\{a_k = i\}$$

be the *empirical reward* of arm i at time t. We can set $\hat{r}_{t,i} = 0$ when $N_{t,i} = 0$. Then, the posterior distribution for the parameter of arm i is

$$\xi_t = \mathcal{Beta}(\alpha_i + N_{t,i}\hat{r}_{t,i} \ , \ \beta_i + N_{t,i}(1 - \hat{r}_{t,i})).$$

Since $r_t \in \{0, 1\}$, the possible states of our belief given some prior are $\mathbb{N}^{2n}$.

To be able to evaluate a policy, we need to be able to predict the expected utility we obtain. This only depends on our current belief, and the state of our belief corresponds to the state of the bandit problem. This means that everything we know about the problem at time t can be summarised by ξ_t. For Bernoulli bandits, a sufficient statistic for our belief is the number of times we played each bandit and the total reward from each bandit. Thus, our state at time t is entirely described by our priors α, β (the initial state) and the vectors

belief state

$$N_t = (N_{t,1}, \ldots, N_{t,n}) \tag{6.2.5}$$

$$\hat{r}_t = (\hat{r}_{t,1}, \ldots, \hat{r}_{t,n}). \tag{6.2.6}$$

At any time t, we can calculate the probability of observing $r_t = 1$ if we pull arm i as:

$$\xi_t(r_t = 1 \mid a_t = i) = \frac{\alpha_i + N_{t,i}\hat{r}_{t,i}}{\alpha_i + \beta_i + N_{t,i}}.$$

So, not only we can predict the immediate reward based on our current belief, but we can also predict all next possible beliefs: the next state is well-defined and depends only on the current state and observation. As we shall see later, this type of decision problem can be modelled as a Markov decision process (Definition 6.3.1). For now, we shall more generally (and precisely) define the bandit process itself.

6.2.2 Decision-theoretic bandit process

The basic view of the bandit process is to consider only the decision maker's actions a_t, obtained rewards r_t and the latent parameter ω, as shown in Figure 6.2(a). With this basic framework, we can now define the general decision-theoretic bandit process, which also includes the states of belief ξ_t of the decision maker.

Definition 6.2.2. Let $\mathcal{A}$ be a set of actions, not necessarily finite. Let Ω be a set of possible parameter values, indexing a family of probability measures $\mathscr{P} = \{P_{\omega,a} \mid \omega \in \Omega, a \in \mathcal{A}\}$. There is some $\omega \in \Omega$ such that, whenever we take action $a_t = a$, we observe reward $r_t \in \mathcal{R} \subset \mathbb{R}$ with probability measure:

$$P_{\omega,a}(R) \triangleq \mathbb{P}_\omega(r_t \in R \mid a_t = a), \qquad R \subseteq \mathbb{R}. \tag{6.2.7}$$

Let ξ_1 be a prior distribution on Ω and let the posterior distributions be defined as

$$\xi_{t+1}(B) \propto \int_B P_{\omega,a_t}(r_t)\,\mathrm{d}\xi_t(\omega). \tag{6.2.8}$$

The next belief is random, since it depends on the random quantity r_t. In fact, the probability of the next reward lying in some set R if $a_t = a$ is given by the following marginal distribution:

$$P_{\xi_t,a}(R) \triangleq \int_\Omega P_{\omega,a}(R)\,\mathrm{d}\xi_t(\omega). \tag{6.2.9}$$

Finally, as ξ_{t+1} deterministically depends on ξ_t, a_t, r_t, the probability of obtaining a particular next belief is the same as the probability of obtaining the corresponding rewards leading to the next belief. In more detail, we can write:

$$\mathbb{P}(\xi_{t+1} = \xi \mid \xi_t, a_t) = \int_{\mathcal{R}} \mathbb{I}\left\{\xi_t(\cdot \mid a_t, r_t = r) = \xi\right\}\,\mathrm{d}P_{\xi_t,a}(r). \tag{6.2.10}$$

In practice, although multiple reward sequences may lead to the same beliefs, we frequently ignore that possibility for simplicity. Then the process becomes a tree. A solution to the problem of which action to select is given by a backwards induction algorithm similar to the one given in Section 5.2.2:

$$U^*(\xi_t) = \max_{a_t} \mathbb{E}(r_t \mid \xi_t, a_t) + \sum_{\xi_{t+1}} \mathbb{P}(\xi_{t+1} \mid \xi_t, a_t) U^*(\xi_{t+1}). \tag{6.2.11}$$

backwards induction

The above equation is the *backwards induction* algorithm for bandits. If you look at this structure, you can see that the next belief only depends on the current belief, action and reward, i.e. it satisfies the Markov property, as seen in Figure 6.1.

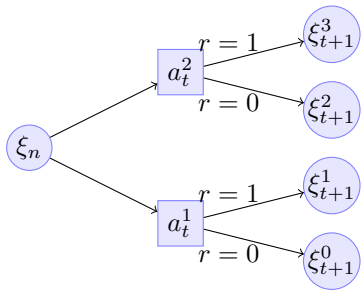
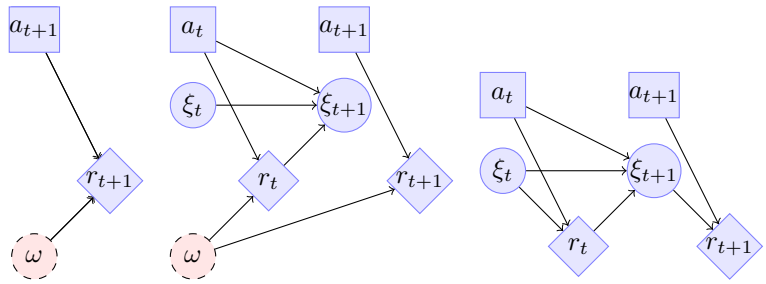


Figure 6.1: A partial view of the multi-stage process. Here, the probability that we obtain $r = 1$ if we take action $a_t = i$ is simply $P_{\xi_t,i}(\{1\})$.

Consequently, a decision-theoretic bandit process can be modelled more generally as a Markov decision process, explained in the following section. It turns out that backwards induction, as well as other efficient algorithms, can provide optimal solutions for Markov decision processes.

In reality, the reward depends only on the action and the unknown ω, as can be seen in Figure 6.2(a). This is the point of view of an external observer.



(a) The basic process (b) The full process (c) The lifted process

Figure 6.2: Three views of the bandit process. Figure 6.2(a) shows the basic bandit process, from the view of an external observer. The decision maker selects a_t, while the parameter ω of the process is hidden. It then obtains reward r_t. The process repeats for $t = 1, \ldots, T$. The decision-theoretic bandit process is shown in Figures 6.2(b) and 6.2(c). While ω is not known, at each time step t we maintain a belief ξ_t on Ω. The reward distribution is then defined through our belief. In Figure 6.2(b), we can see the complete process, where the dependency on ω is clear. In Figure 6.2(c), we marginalise out ω and obtain a model where the transitions only depend on the current belief and action.

If we want to add the decision maker's internal belief to the graph, we obtain Figure 6.2(b). However, from the point of view of the decision maker, the distribution of ω only depends on his current belief. Consequently, the distribution of rewards also only depends on the current belief, as we can marginalise over ω. This gives rise to the decision-theoretic bandit process shown in Figure 6.2(c). In the following section, we shall consider Markov decision processes more generally.

6.3 Markov decision processes and reinforcement learning

The bandit setting is one of the simplest instances of reinforcement learning problems. Informally, speaking, these are problems of learning how to act in an unknown environment, only through interaction with the environment and limited reinforcement signals. The learning agent interacts with the environment through actions and observations, and obtains rewards.

For example, we can consider a rat running through a maze, which from time to time finds a piece of cheese, the reward. The goal of the agent is usually to maximise some measure of the total reward. In summary, we can state the problem as follows.

The reinforcement learning problem.
The reinforcement learning problem is the problem of *learning* how to act in an *unknown* environment, only by *interaction* and *reinforcement*.

Generally, we assume that the environment μ that we are acting in has an

underlying state $s_t \in \mathcal{S}$, which changes in discrete time steps t. At each step, the agent obtains an observation $x_t \in \mathcal{X}$ and chooses an action $a_t \in \mathcal{A}$. We usually assume that the environment is such that its next state s_{t+1} only depends on its current state s_t and the last action taken by the agent, a_t. In addition, the agent observes a reward signal r_t, and its goal is to maximise the total reward during its lifetime.

When the environment μ is unknown, this is hard even in seemingly simple settings, like n-armed bandits, where the underlying state never changes. In many real-world applications, the problem is even harder, as the state often is not directly observed. Instead, we may have to rely on the observables x_t, which give only partial information about the true underlying state s_t.

Reinforcement learning problems typically fall into one of the following three groups: (1) Markov decision processes (MDPs), where the state s_t is observed directly, i.e., $x_t = s_t$; (2) Partially observable MDPs (POMDPs), where the state is hidden, i.e., x_t is only probabilistically dependent on the state; and (3) stochastic Markov games, where the next state also depends on the move of other agents. While all of these problem *descriptions* are different, in the Bayesian setting, they all can be reformulated as MDPs by constructing an appropriate belief state, similarly to how we did it for the decision theoretic formulation of the bandit problem.

In this chapter, we shall confine our attention to Markov decision processes. Hence, we shall not discuss the existence of other agents, or the case where we cannot observe the state directly.

transition distribution
reward distribution

Definition 6.3.1 (Markov Decision Process)**.** A Markov decision process μ is a tuple $\mu = \langle \mathcal{S}, \mathcal{A}, \mathscr{P}, \mathscr{R} \rangle$, where $\mathcal{S}$ is the *state space* and $\mathcal{A}$ is the *action space*. The *transition distribution* $\mathscr{P} = \{P(\cdot \mid s, a) \mid s \in \mathcal{S}, a \in \mathcal{A}\}$ is a collection of probability measures on $\mathcal{S}$, indexed in $\mathcal{S} \times \mathcal{A}$ and the *reward distribution* $\mathscr{R} = \{\rho(\cdot \mid s, a) \mid s \in \mathcal{S}, a \in \mathcal{A}\}$ is a collection of probability measures on $\mathbb{R}$, such that:

$$P(S \mid s, a) = \mathbb{P}_\mu(s_{t+1} \in S \mid s_t = s, a_t = a) \tag{6.3.1}$$
$$\rho(R \mid s, a) = \mathbb{P}_\mu(r_t \in R \mid s_t = s, a_t = a). \tag{6.3.2}$$

For simplicity, we shall also use

$$r_\mu(s, a) = \mathbb{E}_\mu(r_{t+1} \mid s_t = s, a_t = a), \tag{6.3.3}$$

for the expected reward.

Of course, the transition and reward distributions are different for different environments μ. For that reason, we shall usually subscript the relevant probabilities and expectations with μ, unless the MDP is clear from the context.

Markov property of the reward and state distribution

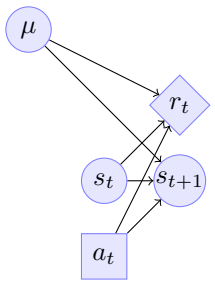


Figure 6.3: The structure of a Markov decision process.

$$\mathbb{P}_\mu(s_{t+1} \in S \mid s_1, a_1, \ldots, s_t, a_t) = \mathbb{P}_\mu(s_{t+1} \in S \mid s_t, a_t), \quad \text{(Transition distribution)}$$

$$\mathbb{P}_\mu(r_t \in R \mid s_1, a_1, \ldots, s_t, a_t) = \mathbb{P}_\mu(r_t \in R \mid s_t, a_t), \quad \text{(Reward distribution)}$$

where $S \subset \mathcal{S}$ and $R \subset \mathcal{R}$ are reward and state subsets respectively.

Dependencies of rewards. Sometimes it is more convenient to have rewards that depend on the next state as well, i.e.

$$r_\mu(s, a, s') = \mathbb{E}_\mu(r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'), \tag{6.3.4}$$

though this is complicates the notation considerably since now the reward is obtained on the next time step. However, we can always replace this with the expected reward for a given state-action pair:

$$r_\mu(s, a) = \mathbb{E}_\mu(r_{t+1} \mid s_t = s, a_t = s) = \sum_{s' \in \mathcal{S}} P_\mu(s' \mid s, a) r_\mu(s, a, s') \tag{6.3.5}$$

In fact, it is notationally more convenient to have rewards that only depend on the current state:

$$r_\mu(s) = \mathbb{E}_\mu(r_t \mid s_t = s). \tag{6.3.6}$$

For simplicity, we shall mainly consider the latter case.

Policies. The actions are taken through a policy π which selects them depending on the observed history. One can think of a policy as implemented through an algorithm or an embodied agent, who is interested in maximising expected utility.

The policy π

The policy defines a conditional distribution on actions given the history:

$$\mathbb{P}^\pi(a_t \mid s_t, \ldots, s_1, a_{t-1}, \ldots, a_1) \quad \text{(history-dependent policy)}$$

$$\mathbb{P}^\pi(a_t \mid s_t) \quad \text{(Markov policy)}$$

policy

The *policy* π is otherwise known as a *decision function*. In general, the policy can be history-dependent. In certain cases, however, there are optimal policies that are Markov. This is for example the case with additive utility functions. In paticular, the utility function maps from the sequence of all possible rewards to a real number $U : \mathcal{R}^* \to \mathbb{R}$, given below:

Definition 6.3.2 (Additive utility)**.** The utility here has two parameters: the horizon T, after which the agent is no longer interested in rewards, and a discount factor $\gamma \in (0, 1]$, which discounts future rewards. The utility function $U : \mathcal{R}^* \to \mathbb{R}$ is defined as

$$U(r_0, r_1, \ldots, r_T) = \sum_{k=0}^{T} \gamma^k r_k. \tag{6.3.7}$$

It is convenient to give a special name to the utility starting from time t, i.e. the sum of rewards from that time on:

$$U_t \triangleq \sum_{k=0}^{T-t} \gamma^k r_{t+k}. \tag{6.3.8}$$

At any time t, the agent wants to to find a policy π *maximising* the *expected total future reward*

$$\mathbb{E}^\pi_\mu U_t = \mathbb{E}^\pi_\mu \sum_{k=0}^{T-t} \gamma^k r_{t+k}. \tag{expected utility}$$

This is so far identical to the expected utility framework we have seen so far, with the only difference that now the reward space is a sequence of numerical rewards and that we are acting within a dynamical system with state space $\mathcal{S}$. In fact, it is a good idea to think about the *value* of different states of the system under certain policies, in the same way that one thinks about how good different positions are in chess.

6.3.1 Value functions

A value function represents the expected utility of a given state, or state-and-action pair for a specific policy. They are really useful as shorthand notation and as the basis of algorithm development. The most basic of those is the state value function.

State value function

$$V^\pi_{\mu,t}(s) \triangleq \mathbb{E}^\pi_\mu(U_t \mid s_t = s) \tag{6.3.9}$$

The state value function for a particular policy π can be interpreted as how much utility you should expect if you follow the policy starting from state s at time t, for the particular MDP μ.

State-action value function

$$Q^{\pi}_{\mu,t}(s,a) \triangleq \mathbb{E}^{\pi}_{\mu}(U_t \mid s_t = s, a_t = a) \tag{6.3.10}$$

The state-action value function for a particular policy π can be interpreted as how much utility you should expect if you play action a, at state s at time t, and then follow the policy π, for the particular MDP μ.

It is also useful to define the optimal policy and optimal value functions for a given MDP. In the following, a star indicates optimal quantities. The *optimal policy* π^*

$$\pi^*(\mu) : V^{\pi^*(\mu)}_{t,\mu}(s) \geq V^{\pi}_{t,\mu}(s) \quad \forall \pi, t, s \tag{6.3.11}$$

dominates all other policies π everywhere in $\mathcal{S}$.

The *optimal value function* V^*

$$V^*_{t,\mu}(s) \triangleq V^{\pi^*(\mu)}_{t,\mu}(s), \quad Q^*_{t,\mu}(s) \triangleq Q^{\pi^*(\mu)}_{t,\mu}(s,a). \tag{6.3.12}$$

is the value function of the optimal policy π^*.

Finding the optimal policy when μ is known

When the MDP μ is known, the expected utility of any policy can be calculated. Therefore, one could find the optimal policy by brute force, i.e. by calculating the utility of every possible policy. This might be as reasonable strategy if the number of policies is small. However, there are many better appr. First, there are iterative/offline methods where an optimal policy is found for all states of the MDP. These either try to estimate the optimal value function directly, or try to iteratively improve a policy until it is optimal. The second type of methods tries to find an optimal policy online. That is, the optimal actions are estimated only for states which can be visited in the future starting from the current state. However, the same main ideas are used in all of these algorithms.

6.4 Finite horizon, undiscounted problems

The conceptually simplest type of problems are finite horizon problems where $T < \infty$ and $\gamma = 1$. The first thing we shall try to do is to evaluate a given policy for a given MDP. There are a number of algorithms that can achieve this.

6.4.1 Policy evaluation

Here we are interested in the problem of determining the value function of a policy π (for $\gamma = 1, T < \infty$). All the algorithms we shall consider can be

recovered from the following recursion. Noting that $U_{t+1} = \sum_{k=1}^{T-t} r_{t+k}$ we have:

$$V^\pi_{\mu,t}(s) \triangleq \mathbb{E}^\pi_\mu(U_t \mid s_t = s) \tag{6.4.1}$$
$$= \sum_{k=0}^{T-t} \mathbb{E}^\pi_\mu(r_{t+k} \mid s_t = s) \tag{6.4.2}$$
$$= \mathbb{E}^\pi_\mu(r_t \mid s_t = s) + \mathbb{E}^\pi_\mu(U_{t+1} \mid s_t = s) \tag{6.4.3}$$
$$= \mathbb{E}^\pi_\mu(r_t \mid s_t = s) + \sum_{i \in \mathcal{S}} V^\pi_{\mu,t+1}(i)\, \mathbb{P}^\pi_\mu(s_{t+1} = i | s_t = s). \tag{6.4.4}$$

Note that the last term can be calculated easily through marginalisation.

$$\mathbb{P}^\pi_\mu(s_{t+1} = i | s_t = s) = \sum_{a \in \mathcal{A}} \mathbb{P}_\mu(s_{t+1} = i | s_t = s, a_t = a)\, \mathbb{P}^\pi(a_t = a | s_t = s).$$

This derivation directly gives a number of *policy evaluation algorithms*.

Direct policy evaluation Direct policy evaluation is based on (6.4.2), which can be implemented by Algorithm 2. One needs to *marginalise out* all possible state sequences to obtain the expected reward given the state at time $t + k$ giving the following:

$$\mathbb{E}^\pi_\mu(r_{t+k} \mid s_t = s) = \sum_{s_{t+1}, \ldots, s_{t+k} \in \mathcal{S}^k} \mathbb{E}^\pi_\mu(r_{t+k} \mid s_{t+k})\, \mathbb{P}^\pi_\mu(s_{t+1}, \ldots, s_{t+1} \mid s_t).$$

By using the Markov property, we calculate the probability of reaching any state from any other state at different times, and then add up the expected reward we would get in that state under our policy. Then $\hat{V}_t(s) = V^\pi_{\mu,t}(s)$ by definition.

Unfortunately it is not a very good idea to use direct policy evaluation. The most efficient implementation involves calculating $P(s_t \mid s_0)$ recursively for every state. This would result in a total of $|\mathcal{S}|^3 T$ operations. Monte-Carlo evaluations should be considerably cheaper, especially when the transition structure is sparse.

Algorithm 2 Direct policy evaluation

1: **for** $s \in \mathcal{S}$ **do**
2: **for** $t = 0, \ldots, T$ **do**
3:
$$\hat{V}_t(s) = \sum_{k=t}^{T} \sum_{j \in \mathcal{S}} \mathbb{P}^\pi_\mu(s_k = j \mid s_t = s)\, \mathbb{E}^\pi_\mu(r_k \mid s_k = j).$$
4: **end for**
5: **end for**

6.4.2 Monte-Carlo policy evaluation

Another conceptually simple algorithm is Monte-Carlo policy evaluation shown as Algorithm 3. The idea is that instead of summing over all possible states to be visited, we just draw states from the Markov chain defined jointly by the policy and the Markov decision process. Unlike direct policy evaluation

the algorithm needs a parameter K, the number of trajectories to generate. Nevertheless, this is a very useful method, employed within a number of more complex algorithms.

Algorithm 3 Monte-Carlo policy evaluation

for $s \in \mathcal{S}$ **do**
  for $k = 0, \dots, K$ **do**
    Choose initial state s_1.
    for $t = 1, \dots, T$ **do**
      $a_t \sim \pi(a_t \mid s_t)$ // Take action
      Observe reward r_t and next state s_{t+1}.
      Set $r_{t,k} = r_t$.
    end for
    Save total reward:

$$\hat{V}_k(s) = \sum_{t=1}^{T} r_{t,k}.$$

  end for
  Calculate estimate:

$$\hat{V}(s) = \frac{1}{K} \sum_{k=1}^{K} \hat{V}_k(s).$$

end for

Remark 6.4.1. The estimate $\hat{V}$ of the Monte Carlo evaluation algorithm satisfies

$$\|V - \hat{V}\|_\infty \leq \sqrt{\frac{\ln(2|\mathcal{S}|/\delta)}{2K}} \qquad \text{with probability } 1 - \delta$$

Proof. From Hoeffding's inequality (4.5.5) we have for any state s that

$$\mathbb{P}\left(|\hat{V}(s) - V(s)| \geq \sqrt{\frac{\ln(2|\mathcal{S}|/\delta)}{2K}} \right) \leq \delta/|\mathcal{S}|.$$

Consequently, using a union bound of the form $P(A_1 \cup A_2 \cup \ldots \cup A_n) \leq \sum_i P(A_i)$ gives the required result. □

The main advantage of Monte-Carlo policy evaluation is that it can be used in very general settings. It can be used not only in Markovian environments such as MDPs, but also in partially observable and multi-agent settings.

6.4.3 Backwards induction policy evaluation

Finally, the backwards induction algorithm shown as Algorithm 4 is similar to the backwards induction algorithm we saw for sequential sampling and bandit problems. However, here we are only evaluating a policy rather than finding the optimal one. This algorithm is slightly less generally applicable than the Monte-Carlo method because it makes Markovian assumptions. The Monte-Carlo algorithm, can be used for environments that with a non-Markovian variable s_t.

Algorithm 4 Backwards induction policy evaluation

For each state $s \in S$, for $t = 1, \ldots, T-1$:

$$\hat{V}_t(s) = r^\pi_\mu(s) + \sum_{j \in S} \mathbb{P}^\pi_\mu(s_{t+1} = j \mid s_t = s)\hat{V}_{t+1}(j), \tag{6.4.5}$$

with $\hat{V}_T(s) = r^\pi_\mu(s)$.

Theorem 6.4.1. *The backwards induction algorithm gives estimates $\hat{V}_t(s)$ satisfying*

$$\hat{V}_t(s) = V^\pi_{\mu,t}(s) \tag{6.4.6}$$

Proof. For $t = T-1$, the result is obvious. We can prove the remainder by induction. Let (6.4.6) hold for all $t \geq n+1$. Now we prove that it holds for n. Note that from the recursion (6.4.5) we have:

$$\begin{aligned}
\hat{V}_t(s) &= r_\mu(s) + \sum_{j \in S} \mathbb{P}_{\mu,\pi}(s_{t+1} = j \mid s_t = s)\hat{V}_{t+1}(j) \\
&= r(s) + \sum_{j \in S} \mathbb{P}_{\mu,\pi}(s_{t+1} = j \mid s_t = s)V^\pi_{\mu,t+1}(j) \\
&= r(s) + \mathbb{E}_{\mu,\pi}(U_{t+1} \mid s_t = s) \\
&= \mathbb{E}_{\mu,\pi}(U_t \mid s_t = s) = V^\pi_{\mu,t}(s),
\end{aligned}$$

where the second equality is by the induction hypothesis, the third and fourth equalities are by the definition of the utility, and the last by definition of $V^\pi_{\mu,t}$. □

6.4.4 Backwards induction policy optimisation

Backwards induction as given in algorithm 5 is the first non-naive algorithm for finding an optimal policy for the sequential problems with T stages. It is basically identical to the backwards induction algorithm we saw in Chapter 5, which was for the very simple sequential sampling problem, as well as the backwards induction algorithm for the decision-theoretic bandit problem.

Algorithm 5 Finite-horizon backwards induction

Input μ, set $\mathcal{S}_T$ of states reachable within T steps.
Initialise $V_T(s) := \max_a r(s,a)$, for all $s \in \mathcal{S}_T$.
for $n = T-1, T-2, \ldots, 1$ **do**
  for $s \in \mathcal{S}_n$ **do**
    $\pi_n(s) = \arg\max_a r(s,a) + \sum_{s' \in \mathcal{S}_{n+1}} P_\mu(s' \mid s, a)V_{n+1}(s')$
    $V_n(s) = r(s,a) + \sum_{s' \in \mathcal{S}_{n+1}} P_\mu(s' \mid s, \pi_n(s))V_{n+1}(s')$
  end for
end for
Return $\pi = (\pi_n)_{n=1}^T$.

Theorem 6.4.2. *For T-horizon problems, backwards induction is optimal, i.e.*

$$V_n(s) = V^*_{\mu,n}(s) \tag{6.4.7}$$

Proof. Note that the proof below also holds for $r(s,a) = r(s)$. First we show that $V_t \geq V_t^*$. For $n = T$ we evidently have $V_T(s) = \max_a r(s,a) = V_{\mu,T}^*(s)$. Now assume that for $n \geq t+1$, (6.4.7) holds. Then it also holds for $n = t$, since for any policy π'

$$\begin{aligned} V_t(s) &= \max_a \left\{ r(s,a) + \sum_{j \in \mathcal{S}} P_\mu(j \mid s,a) V_{t+1}(j) \right\} \\ &\geq \max_a \left\{ r(s,a) + \sum_{j \in \mathcal{S}} P_\mu(j \mid s,a) V_{\mu,t+1}^*(j) \right\} \quad \text{(by induction assumption)} \\ &\geq \max_a \left\{ r(s,a) + \sum_{j \in \mathcal{S}} P_\mu(j \mid s,a) V_{\mu,t+1}^{\pi'}(j) \right\} \\ &\geq V_t^{\pi'}(s). \end{aligned}$$

This holds for any policy π', including $\pi' = \pi$, the policy returned by backwards induction. Then:

$$V_{\mu,t}^*(s) \geq V_{\mu,t}^{\pi}(s) = V_t(s) \geq V_{\mu,t}^*(s).$$

□

Remark 6.4.2. A similar theorem can be proven for arbitrary $\mathcal{S}$. This requires using sup instead of max and proving the existence of a π' that is arbitrary-close in value to V^*. For details, see [Puterman, 1994].

6.5 Infinite-horizon

When problems have no fixed horizon, they usually can be modelled as infinite horizon problems, sometimes with help of a *terminating state*, whose visit terminates the problem, or discounted rewards, which indicate that we care less about rewards further in the future. When reward discounting is exponential, these problems can be seen as undiscounted problems with random and geometrically distributed horizon. For problems with no discounting and no termination states there are some complications in the definition of optimal policy. However, we defer discussion of such problems to Chapter 10.

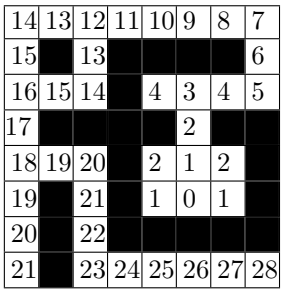
6.5.1 Examples

We begin with some examples, which will help elucidate the concept of terminating states and infinite horizon. The first is shortest path problems, where the aim is to find the shortest path to a particular goal. Although the process terminates when the goal is reached, not all policies may be able to reach the goal, and so the process may never terminate.

Shortest-path problems

We shall consider two types of shortest path problems, deterministic and stochastic. Although conceptually very different, both problems have essentially the same complexity.

Consider an agent moving in a maze, aiming to get to some terminating goal state X. That is, when reaching this state, the agent cannot move anymore, and receives a reward of 0. In general, the agent can move deterministically in the four cardinal directions, and receives a negative reward at each time step. Consequently, the optimal policy is to move to X as quickly as possible.

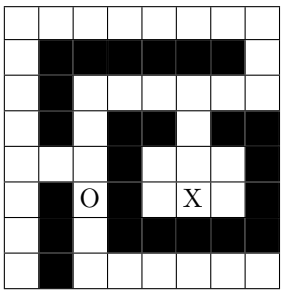


Properties

- $\gamma = 1$, $T \to \infty$.
- $r_t = -1$ unless $s_t = X$, in which case $r_t = 0$.
- $\mathbb{P}_\mu(s_{t+1} = X | s_t = X) = 1$.
- $\mathcal{A} = \{\text{North}, \text{South}, \text{East}, \text{West}\}$
- Transitions are deterministic and walls block.

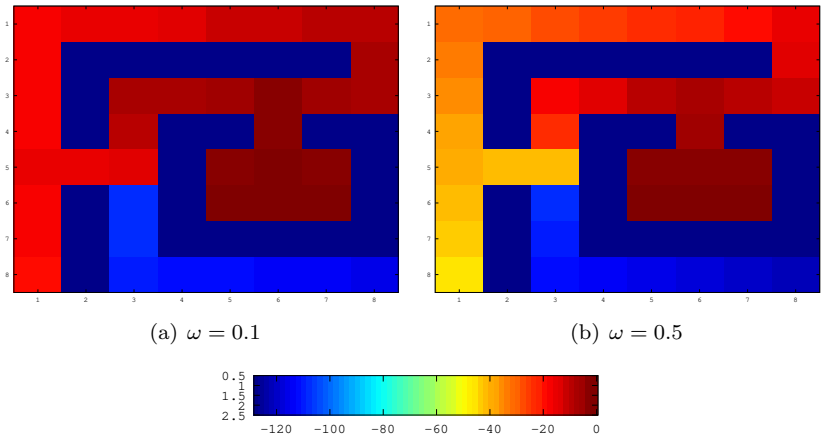
Solving the shortest path problem can be done simply by looking at the distance of any point to X. Then the reward obtained by the optimal policy starting from any point, is simply the negative distance. The optimal policy simply moves to the state with the smallest distance to X.

Stochastic shortest path problem with a pit Now assume the shortest path problem with stochastic dynamics. That is, at each time-step there is a small probability ω that move to a random direction. In addition, there is a pit O, that is a terminating state with a reward of -100.



Properties

- $\gamma = 1$, $T \to \infty$.
- $r_t = -1$, but $r_t = 0$ at X and -100 at O and episode ends.
- $\mathbb{P}_\mu(s_{t+1} = X | s_t = X) = 1$.
- $\mathcal{A} = \{\text{North}, \text{South}, \text{East}, \text{West}\}$
- Moves to a random direction with probability ω. Walls block.



(a) $\omega = 0.1$ (b) $\omega = 0.5$

(c) value

Figure 6.4: Pit maze solutions for two values of ω.

Randomness changes the solution significantly in this environment. When ω is relatively small, it is worthwhile (in expectation) for the agent to pass past the pit, even though there is a risk of falling in and getting a reward of -100. In the example given, even starting from the third row, the agent prefers taking the short-cut. For high enough ω, the optimal policy avoids approaching the pit. Still, the agent prefers jumping in the pit, than being trapped at the bottom of the maze forever.

Continuing problems

Finally, many problems have no natural terminating state, but are continuing *ad infinitum*. Frequently, we model those problems using a utility that discounts future rewards exponentially. This way, we can guarantee that the utility is bounded. In addition, exponential discounting also has some economical sense. This is partially because of the effects of inflation, and partially because money now may be more useful than money in the future. Both these effects diminish the value of money over time. As an example, consider the following inventory management problem.

Example 32 (Inventory management). There are K storage locations, and each location i can store n_i items. At each time-step there is a probability ϕ_i that a client tries to buy an item from location i, where $\sum_i \phi_i \leq 1$. If there is an item available, when this happens, you gain reward 1. There are two types of actions, one for ordering a certain number u units of stock, paying $c(u)$. Further one may move u units of stock from one location i to another location j, paying $\psi_{ij}(u)$.

An easy special case is when $K = 1$, and we assume that deliveries happen once every m timesteps, and each time-step a client arrives with probability ϕ. Then the state set $\mathcal{S} = \{0, 1, \ldots, n\}$ corresponds to the number of items we have, the action set $\mathcal{A} = \{0, 1, \ldots, n\}$ to the number of items we may order. The transition probabilities are given by $P(s'|s, a) = \binom{m}{d}\phi^d(1-\phi)^{m-d}$, where $d = s + a - s'$, for $s + a \leq n$.

6.5.2 Markov chain theory for discounted problems

Here we consider MDPs with infinite horizon and discounted rewards. We shall consider undiscounted rewards only in Chapter 10. Our utility in this case is the discounted total reward:

$$U_t = \lim_{T\to\infty} \sum_{k=t}^{T} \gamma^k r_k, \qquad \gamma \in (0,1)$$

For simplicity, in the following we assume that rewards only depend on the current state instead of both state and action. It can easily be verified that results still hold in the latter case. More importantly, we also assume that the state and action spaces $\mathcal{S}, \mathcal{A}$ are finite, and that the transition kernel of the MDP is time-invariant. This allows us to use the following simplified vector notation:

- $\boldsymbol{v}^\pi = (\mathbb{E}^\pi(U_t \mid s_t = s))_{s\in\mathcal{S}}$ is a vector in $\mathbb{R}^{|\mathcal{S}|}$ representing the value of policy π.
- Sometimes we will use $p(j|s,a)$ as a shorthand for $\mathbb{P}_\mu(s_{t+1} = j \mid s_t = s, a_t = a)$.
- $\boldsymbol{P}_{\mu,\pi}$ is a transition matrix in $\mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ for policy π, such that

$$\boldsymbol{P}_{\mu,\pi}(i,j) = \sum_a p(j \mid i,a)\, \mathbb{P}^\pi(a \mid i).$$

- $\boldsymbol{r}$ is a reward vector in $\mathbb{R}^{|\mathcal{S}|}$.
- The space of value functions $\mathcal{V}$ is a Banach space (i.e., a complete, normed vector space) equipped with the norm

$$\|\boldsymbol{v}\| = \sup\{|\boldsymbol{v}(s)| \mid s \in \mathcal{S}\}$$

For infinite-horizon discounted MDPs, stationary policies are sufficient. This can be proven by induction, using arguments similar to other proofs given here. For a detailed set of proofs, see Puterman [1994].

Definition 6.5.1. A policy π is stationary if $\pi(a_t \mid s_t) = \pi(a_n \mid s_n)$ for all n, t.

We now present a set of important results that link Markov decision processes to linear algebra.

Remark 6.5.1. We can use the Markov chain kernel $\boldsymbol{P}$ to write the expected reward vector as

$$\boldsymbol{v}^\pi = \sum_{t=0}^{\infty} \gamma^t \boldsymbol{P}_{\mu,\pi}^t \boldsymbol{r} \tag{6.5.1}$$

Proof.

$$\begin{aligned} V^\pi(s) &= \mathbb{E}\left(\sum_{t=0}^{\infty} \gamma^t r_t \;\middle|\; s_0 = s\right) \\ &= \sum_{t=0}^{\infty} \gamma^t \,\mathbb{E}(r_t | s_0 = s) \\ &= \sum_{t=0}^{\infty} \gamma^t \sum_{i \in \mathcal{S}} \mathbb{P}(s_t = i \mid s_0 = s)\, \mathbb{E}(r_t \mid s_t = i). \end{aligned}$$

Since for any distribution vector $\boldsymbol{p}$ over $\mathcal{S}$, we have $\mathbb{E}_{\boldsymbol{p}}\, r_t = \boldsymbol{p}^\top \boldsymbol{r}$, the result follows. □

It is possible to show that the expected discounted total reward of a policy is equal to the expected undiscounted total reward with a geometrically distributed horizon (see exercise 28. As a corollary, it follows a Markov decision process with discounting is equivalent with one where there is no discounting, but a stopping probability $(1 - \gamma)$ at every step.

The value of a particular policy can be expressed as a linear equation. This is an important result, as it has led to a number of successful algorithms that employ linear theory.

Theorem 6.5.1. *For any stationary policy π, $\boldsymbol{v}^\pi$ is the unique solution of*

$$\boldsymbol{v} = \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi} \boldsymbol{v}. \tag{6.5.2}$$

In addition, the solution is:

$$\boldsymbol{v}^\pi = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})^{-1} \boldsymbol{r}, \tag{6.5.3}$$

where $\boldsymbol{I}$ is the identity matrix.

To prove this we will need the following important theorem.

Theorem 6.5.2. *For any bounded linear transformation $\boldsymbol{A} : S \to S$ on a normed linear space S (i.e., there is $c < \infty$ s.t. $\|\boldsymbol{A}x\| := \sup_i \sum_j a_{i,j} \leq c\|x\|$ for all $x \in S$ with* spectral radius $\sigma(\boldsymbol{A}) \triangleq \lim_{n\to\infty} \|\boldsymbol{A}^n\|^{1/n} < 1$*), $\boldsymbol{A}^{-1}$ exists and is given by* *spectral radius*

$$\boldsymbol{A}^{-1} = \lim_{T\to\infty} \sum_{n=0}^{T} (\boldsymbol{I} - \boldsymbol{A})^n. \tag{6.5.4}$$

Proof of Theorem 6.5.1. First note that by manipulating the infinite sum in Remark 6.5.1, one obtains $\boldsymbol{r} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{v}^\pi$. Since $\|\gamma \boldsymbol{P}_{\mu,\pi}\| < 1 \cdot \|\boldsymbol{P}_{\mu_\pi}\| = 1$, the inverse

$$(\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})^{-1} = \lim_{n\to\infty} \sum_{t=0}^{n} (\gamma \boldsymbol{P}_{\mu,\pi})^t$$

exists by Theorem 6.5.2. It follows that

$$\boldsymbol{v} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})^{-1} \boldsymbol{r} = \sum_{t=0}^{\infty} \gamma^t \boldsymbol{P}_{\mu,\pi}^t \boldsymbol{r} = \boldsymbol{v}^\pi,$$

where the last step is by Remark 6.5.1 again. □

It is important to note that the matrix $\boldsymbol{X} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})^{-1}$ can be seen as the expected number of discounted cumulative visits to each state s, starting from state s' and following policy π. More specifically, the entries of the matrix are:

$$x(s, s') = \mathbb{E}_\mu^\pi \left\{ \sum_{t=0}^{\infty} \gamma^t \, \mathbb{P}_\mu^\pi (s_t = s' \mid s_t = s) \right\}. \tag{6.5.5}$$

This interpretation is quite useful, as many algorithms rely on an estimation of $\boldsymbol{X}$ for approximating value functions.

6.5.3 Optimality equations

Let us now look at the backwards induction algorithms in terms of operators. We introduce the operator of a policy, which is the one-step backwards induction operation for a fixed policy, and the Bellman operator, which is the equivalent operator for the optimal policy. If a value function is optimal, then it satisfies the Bellman optimality equation.

Definition 6.5.2 (Policy and Bellman operator)**.** The linear operator of a policy π is:

$$\mathscr{L}_\pi \boldsymbol{v} \triangleq \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v} \tag{6.5.6}$$

Sby contract The (non-linear) Bellman operator in the space of value functions $\mathcal{V}$ is defined as:

$$\mathscr{L} \boldsymbol{v} \triangleq \sup_\pi \{ \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v} \}, \qquad \boldsymbol{v} \in \mathcal{V} \tag{6.5.7}$$

We now show that the Bellman operator satisfies the following monotonicity properties with respect to an arbitrary value vector $\boldsymbol{v}$.

Theorem 6.5.3. *Let $\boldsymbol{v}^* \triangleq \sup_\pi \boldsymbol{v}^\pi$. Then for any bounded $\boldsymbol{r}$, it holds that for $\boldsymbol{v} \in \mathcal{V}$:*

(1) If $\boldsymbol{v} \geq \mathscr{L} \boldsymbol{v}$, then $\boldsymbol{v} \geq \boldsymbol{v}^*$.

(2) If $\boldsymbol{v} \leq \mathscr{L} \boldsymbol{v}$, then $\boldsymbol{v} \leq \boldsymbol{v}^*$.

(3) If $\boldsymbol{v} = \mathscr{L} \boldsymbol{v}$, then $\boldsymbol{v}$ is unique and $\boldsymbol{v} = \sup_\pi \boldsymbol{v}^\pi$. Therefore, $\boldsymbol{v} = \mathscr{L} \boldsymbol{v}$ is called the Bellman optimality equation.

Proof. We first prove (1). A simple proof by induction over n shows that for any π

$$\boldsymbol{v} \geq \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v} \geq \sum_{k=0}^{n-1} \gamma^k \boldsymbol{P}_\pi^k \boldsymbol{r} + \gamma^n \boldsymbol{P}_\pi^n \boldsymbol{v}.$$

Since $\boldsymbol{v}^\pi = \sum_{t=0}^{\infty} \gamma^t \boldsymbol{P}_\pi^t \boldsymbol{r}$ it follows that

$$\boldsymbol{v} - \boldsymbol{v}^\pi \geq \gamma^n \boldsymbol{P}_\pi^n \boldsymbol{v} - \sum_{k=n}^{\infty} \gamma^k \boldsymbol{P}_\pi^k \boldsymbol{r}.$$

The first-term on the right-hand side can be bounded by arbitrary $\epsilon/2$ for large enough n. Also note that

$$\sum_{k=n}^{\infty} \gamma^k \boldsymbol{P}_\pi^k \boldsymbol{r} \geq -\frac{\gamma^n \boldsymbol{e}}{1-\gamma},$$

with $\boldsymbol{e}$ being a unit vector, so this can be bounded by $\epsilon/2$ as well. So for any $\pi, \epsilon > 0$:

$$\boldsymbol{v} \geq \boldsymbol{v}^\pi - \epsilon,$$

so

$$\boldsymbol{v} \geq \sup_\pi \boldsymbol{v}^\pi.$$

An equivalent argument shows that

$$\boldsymbol{v} \leq \boldsymbol{v}^\pi + \epsilon,$$

proving (2). Putting together (1) and (2) gives (3). □

We eventually want show that repeated application of the Bellman operator converges to the optimal value. As a preparation, we need the following theorem.

Theorem 6.5.4 (Banach Fixed-Point theorem)**.** *Suppose $\mathcal{S}$ is a Banach space (i.e. a complete normed linear space) and $T : \mathcal{S} \to \mathcal{S}$ is a contraction mapping (i.e. $\exists \gamma \in [0, 1)$ s.t. $\|Tu - Tv\| \leq \gamma \|u - v\|$ for all $u, v \in \mathcal{S}$). Then*

- *there is a unique $u^* \in U$ s.t. $Tu^* = u^*$, and*
- *for any $u^0 \in \mathcal{S}$ the sequence $\{u^n\}$:*

$$u^{n+1} = Tu^n = T^{n+1}u^0$$

converges to u^*.

Proof. For any $m \geq 1$

$$\begin{aligned}\|u^{n+m} - u^n\| &\leq \sum_{k=0}^{m-1} \|u^{n+k+1} - u^{n+k}\| = \sum_{k=0}^{m-1} \|T^{n+k}u^1 - T^{n+k}u^0\| \\ &\leq \sum_{k=0}^{m-1} \gamma^{n+k} \|u^1 - u^0\| = \frac{\gamma^n(1-\gamma^m)}{1-\gamma} \|u^1 - u^0\|.\end{aligned}$$

□

Theorem 6.5.5. *For $\gamma \in [0, 1)$ the Bellman operator $\mathscr{L}$ is a contraction mapping in $\mathcal{V}$.*

Proof. Let $\boldsymbol{v}, \boldsymbol{v}' \in \mathcal{V}$. Consider $s \in \mathcal{S}$ such that $\mathscr{L}\boldsymbol{v}(s) \geq \mathscr{L}\boldsymbol{v}'(s)$, and let

$$a_s^* \in \arg\max_{a \in \mathcal{A}} \left\{ r(s) + \sum_{j \in \mathcal{S}} \gamma p_\mu(j \mid s, a) \boldsymbol{v}(j) \right\}.$$

Using the fact that a_s^* is optimal for $\boldsymbol{v}$, but not necessarily for $\boldsymbol{v}'$, we have:

$$\begin{aligned}0 &\leq \mathscr{L}\boldsymbol{v}(s) - \mathscr{L}\boldsymbol{v}'(s) \leq \sum_{j\in S} \gamma p(j \mid s, a_s^*)\boldsymbol{v}(j) - \sum_{j\in S} \gamma p(j \mid s, a_s^*)\boldsymbol{v}'(j) \\ &= \gamma \sum_{j\in S} p(j \mid s, a_s^*)[\boldsymbol{v}(j) - \boldsymbol{v}'(j)] \\ &\leq \gamma \sum_{j\in S} p(j \mid s, a_s^*)\|\boldsymbol{v} - \boldsymbol{v}'\| = \gamma\|\boldsymbol{v} - \boldsymbol{v}'\|.\end{aligned}$$

Repeating the argument for s such that $\mathscr{L}\boldsymbol{v}(s) \leq \mathscr{L}\boldsymbol{v}'(s)$, we obtain

$$|\mathscr{L}\boldsymbol{v}(s) - \mathscr{L}\boldsymbol{v}'(s)| \leq \gamma\|\boldsymbol{v} - \boldsymbol{v}'\|.$$

Taking the supremum over all possible s, the required result follows. □

It is easy to show the same result for the $\mathscr{L}_\pi$ operator, as a corollary to this theorem.

Theorem 6.5.6. *For discrete $\mathcal{S}$, bounded $\boldsymbol{r}$, and $\gamma \in [0, 1)$*

(i) there is a unique $\boldsymbol{v}^* \in \mathcal{V}$ such that $\mathscr{L}\boldsymbol{v}^* = \boldsymbol{v}^*$ and such that $\boldsymbol{v}^* = V_\mu^*$,

(ii) for any stationary policy π, there is a unique $\boldsymbol{v} \in \mathcal{V}$ such that $\mathscr{L}_\pi \boldsymbol{v} = \boldsymbol{v}$ and $\boldsymbol{v} = V_\mu^\pi$.

Proof. As the Bellman operator $\mathscr{L}$ is a contraction by Theorem 6.5.5, application of the fixed-point Theorem 6.5.4 shows that there is a unique $\boldsymbol{v}^* \in \mathcal{V}$ such that $\mathscr{L}\boldsymbol{v}^* = \boldsymbol{v}^*$. This is also the optimal value function due to Theorem 6.5.5.The second part of the theorem follows from the first part when considering only a single policy π (which then is optimal). □

6.5.4 MDP Algorithms

Let us now look at three basic algorithms for solving a known Markov decision process. The first, *value iteration*, is a simple extension of the backwards induction algorithm to the infinite horizon case.

Value iteration

In this version of the algorithm, we assume that rewards are dependent only on the state. An algorithm for the case where reward only depends on the state can be obtained by replacing $r(s, a)$ with $r(s)$.

Algorithm 6 Value iteration

Input μ, $\mathcal{S}$.
Initialise $\boldsymbol{v}_0 \in \mathcal{V}$.
for $n = 1, 2, \ldots$ **do**
  for $s \in \mathcal{S}_n$ **do**
    $\pi_n(s) = \arg\max_{a\in\mathcal{A}} \left\{ r(s, a) + \gamma \sum_{s'\in\mathcal{S}} P_\mu(s' \mid s, a)\boldsymbol{v}_{n-1}(s') \right\}$
    $\boldsymbol{v}_n(s) = r(s, \pi_n(s)) + \gamma \sum_{s'\in\mathcal{S}} P_\mu(s' \mid s, \pi_n(s))\boldsymbol{v}_{n-1}(s')$
  end for
  break if `termination-condition` is met
end for
Return π_n, V_n.

The value iteration algortihm is a direct extension of the backwards induction algorithm for an infinite horizon. However, since we know that stationary policies are optimal, we do not need to maintain the values and actions for all time steps. At each step, we can merely keep the previous value $\boldsymbol{v}_{n-1}$. However, since there is an infinite number of steps, we need to know whether the algorithm converges to the optimal value, and what is the error we make at a particular iteration.

Theorem 6.5.7. *The value iteration algorithm satisfies*

- $\lim_{n\to\infty} \|\boldsymbol{v}_n - \boldsymbol{v}^*\| = 0$.
- *For each $\epsilon > 0$ there exists $N_\epsilon < \infty$ such that for all $n \geq N_\epsilon$*

$$\|\boldsymbol{v}_{n+1} - \boldsymbol{v}_n\| \leq \epsilon(1-\gamma)/2\gamma. \tag{6.5.8}$$

- *For $n \geq N_\epsilon$ the policy π_ϵ that takes action*

$$\arg\max_a r(s,a) + \gamma \sum_j p(j|s,a)\boldsymbol{v}_n(s')$$

  is ϵ-optimal, i.e. $V_\mu^{\pi_\epsilon}(s) \geq V_\mu^*(s) - \epsilon$ for all states s.
- $\|\boldsymbol{v}_{n+1} - \boldsymbol{v}^*\| < \epsilon/2$ *for $n \geq N_\epsilon$.*

Proof. The first two statements follow from the fixed-point Theorem 6.5.4. Now note that

$$\|V_\mu^{\pi_\epsilon} - \boldsymbol{v}^*\| \leq \|V_\mu^{\pi_\epsilon} - \boldsymbol{v}_n\| + \|\boldsymbol{v}_n - \boldsymbol{v}^*\|$$

We can bound these two terms easily:

$$\begin{aligned}
\left\|V_\mu^{\pi_\epsilon} - \boldsymbol{v}_{n+1}\right\| &= \left\|\mathscr{L}_{\pi_\epsilon} V_\mu^{\pi_\epsilon} - \boldsymbol{v}_{n+1}\right\| && \text{(by definition of } \mathscr{L}_{\pi_\epsilon}) \\
&\leq \left\|\mathscr{L}_{\pi_\epsilon} V_\mu^{\pi_\epsilon} - \mathscr{L}\boldsymbol{v}_{n+1}\right\| + \|\mathscr{L}\boldsymbol{v}_{n+1} - \boldsymbol{v}_{n+1}\| && \text{(triangle)} \\
&= \left\|\mathscr{L}_{\pi_\epsilon} V_\mu^{\pi_\epsilon} - \mathscr{L}_{\pi_\epsilon}\boldsymbol{v}_{n+1}\right\| + \|\mathscr{L}\boldsymbol{v}_{n+1} - \mathscr{L}\boldsymbol{v}_n\| && \text{(by definition)} \\
&\leq \gamma \left\|V_\mu^{\pi_\epsilon} - \boldsymbol{v}_{n+1}\right\| + \gamma \left\|\boldsymbol{v}_{n+1} - \boldsymbol{v}_n\right\|. && \text{(by contraction)}
\end{aligned}$$

An analogous argument gives the same bound for the second term $\|\boldsymbol{v}_n - \boldsymbol{v}^*\|$. Then, rearranging we obtain

$$\|V^{\pi_\epsilon} - \boldsymbol{v}_{n+1}\| \leq \frac{\gamma}{1-\gamma}\|\boldsymbol{v}_{n+1} - \boldsymbol{v}_n\|, \qquad \|\boldsymbol{v}_{n+1} - \boldsymbol{v}^*\| \leq \frac{\gamma}{1-\gamma}\|\boldsymbol{v}_{n+1} - \boldsymbol{v}_n\|,$$

and the third and fourth statements follow from the second statement. □

The *termination condition* of value iteration has been left unspecified. However, the theorem above shows that if we terminate when (6.5.8) is true, then our error will be bounded by ϵ. However, better termination conditions can be obtained. *termination condition*

Now let us prove how fast value iteration converges.

Theorem 6.5.8 (Value iteration monotonicity)**.** *Let $\mathcal{V}$ be the set of value vectors with Bellman operator $\mathscr{L}$. Then:*

1. *Let $\boldsymbol{v}, \boldsymbol{v}' \in \mathcal{V}$ with $\boldsymbol{v}' \geq \boldsymbol{v}$. Then $\mathscr{L}\boldsymbol{v}' \geq \mathscr{L}\boldsymbol{v}$.*

2. *Let $\boldsymbol{v}_{n+1} = \mathscr{L}\boldsymbol{v}_n$. If there is an N s.t. $\mathscr{L}\boldsymbol{v}_N \leq \boldsymbol{v}_N$, then $\mathscr{L}\boldsymbol{v}_{N+k} \leq \boldsymbol{v}_{N+k}$ for all $k \geq 0$ and similarly for $\geq$.*

Proof. Let $\pi \in \arg\max_\pi \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{v}$. Then

$$\mathscr{L}\boldsymbol{v} = \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{v} \leq \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{v}' \leq \max_{\pi'} \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi'}\boldsymbol{v}',$$

where the first inequality is due to the fact that $\boldsymbol{P}\boldsymbol{v} \geq \boldsymbol{P}\boldsymbol{v}'$ for any $\boldsymbol{P}$. For the second part,

$$\mathscr{L}\boldsymbol{v}_{N+k} = \boldsymbol{v}_{N+k+1} = \mathscr{L}^k \mathscr{L}\boldsymbol{v}_N \leq \mathscr{L}^k \boldsymbol{v}_N = \boldsymbol{v}_{N+k}.$$

since $\mathscr{L}\boldsymbol{v}_N \leq \boldsymbol{v}_N$ by assumption and consequently $\mathscr{L}^k\mathscr{L}\boldsymbol{v}_N \leq \mathscr{L}^k\boldsymbol{v}_N$ by part one of the theorem. □

Thus, value iteration converges monotonically to V_μ^* if the initial value $\boldsymbol{v}_0 \leq \boldsymbol{v}'$ for all $\boldsymbol{v}'$. If $r \geq 0$, it is sufficient to set $\boldsymbol{v}_0 = \boldsymbol{0}$. Then $\boldsymbol{v}_n$ is always a lower bound on the optimal value function.

Theorem 6.5.9. *Value iteration converges with error in $O(\gamma^n)$ More specifically, for $r \in [0,1]$ and $\boldsymbol{v}_0 = \boldsymbol{0}$,*

$$\|\boldsymbol{v}_n - V_\mu^*\| \leq \frac{\gamma^n}{1-\gamma}, \qquad \|V_\mu^{\pi_n} - V_\mu^*\| \leq \frac{2\gamma^n}{1-\gamma}.$$

Proof. The first part follows from the contraction property (Theorem 6.5.5):

$$\|\boldsymbol{v}_{n+1} - \boldsymbol{v}^*\| = \|\mathscr{L}\boldsymbol{v}_n - \mathscr{L}\boldsymbol{v}^*\| \leq \gamma\|\boldsymbol{v}_n - \boldsymbol{v}^*\|. \tag{6.5.9}$$

Now divide by γ^n to obtain the final result.

□

Although value iteration converges exponentially fast, the convergence is dominated by the discount factor γ. When γ is very close to one, convergence can be extremely slow. In fact, Tseng [1990] showed that the number of iterations are on the order of $1/(1-\gamma)$, for bounded accuracy of the input data. The overall complexity is $\tilde{O}(|\mathcal{S}|^2|\mathcal{A}|L(1-\gamma)^{-1}$, omitting logarithmic factors, where L is the total number of bits used to represent the input.[1]

Policy iteration

Unlike value iteration, *policy iteration* attempts to iteratively improve a given policy, rather than a value function. At each iteration, it calculates the value of the current policy and then calculates the policy that is greedy with respect to this value function. For finite MDPs, the policy evaluation step can be performed with either linear algebra or backwards induction, while the policy improvement step is trivial. The algorithm described below can be extended to the case when the reward also depends on the action, by replacing $\boldsymbol{r}$ with the policy-dependent reward vector $\boldsymbol{r}_\pi$.

[1] Thus the result is *weakly* polynomial complexity, due to the dependence on the input size description.

Algorithm 7 Policy iteration

Input μ, $\mathcal{S}$.
Initialise $\boldsymbol{v}_0$.
for $n = 1, 2, \ldots$ **do**
  $\pi_{n+1} = \arg\max_\pi \{\boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v}_n\}$ `// policy improvement`
  $\boldsymbol{v}_{n+1} = V_\mu^{\pi_{n+1}}$ `// policy evaluation`
  break if $\pi_{n+1} = \pi_n$.
end for
Return $\pi_n, \boldsymbol{v}_n$.

The following theorem describes an important property of policy iteration, namely that the policies generated are monotonically improving.

Theorem 6.5.10. *Let $\boldsymbol{v}_n, \boldsymbol{v}_{n+1}$ be the value vectors generated by policy iteration. Then $\boldsymbol{v}_n \leq \boldsymbol{v}_{n+1}$.*

Proof. From the policy improvement step

$$\boldsymbol{r} + \gamma \boldsymbol{P}_{\pi_{n+1}} \boldsymbol{v}_n \geq \boldsymbol{r} + \gamma \boldsymbol{P}_{\pi_n} \boldsymbol{v}_n, = \boldsymbol{v}_n$$

where the equality is due to the policy evaluation step for π_n. Rearranging, we get $\boldsymbol{r} \geq (\boldsymbol{I} - \gamma \boldsymbol{P}_{\pi_{n+1}})\boldsymbol{v}_n$ and hence

$$(\boldsymbol{I} - \gamma \boldsymbol{P}_{\pi_{n+1}})^{-1} \boldsymbol{r} \geq \boldsymbol{v}_n,$$

noting that the inverse is positive. Since the left side equals $\boldsymbol{v}_{n+1}$ by the policy evaluation step for π_{n+1}, the theorem follows. □

We can use the fact that the policies are monotonically improving to show that policy iteration will terminate after a finite number of steps.

Corollary 6.5.1. *If $\mathcal{S}, \mathcal{A}$ are finite, then policy iteration terminates after a finite number of iterations and returns an optimal policy.*

Proof. There is only a finite number of policies, and since policies in policy iteration are monotonically improving, the algorithm must stop after finitely many iterations. Finally, the last iteration satisfies

$$\boldsymbol{v}_n = \max_\pi \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v}_n. \tag{6.5.10}$$

Thus $\boldsymbol{v}_n$ solves the optimality equation. □

However, it is easy to see that the number of policies is $|\mathcal{A}|^{|\mathcal{S}|}$, thus the above corollary only guarantees exponential-time convergence in the number of states. However, it is also known that the complexity of policy iteration is strongly polynomial Ye [2011], for any fixed γ, with the number of iterations required being $\frac{|\mathcal{S}|^2(|\mathcal{A}|-1)}{1-\gamma} \cdot \ln\left(\frac{|\mathcal{S}|^2}{1-\gamma}\right)$.

Policy iteration seems to have very different behaviour from value iteration. In fact, one can obtain families of algorithms that lie at the extreme ends of the spectrum between policy iteration and value iteration. The first member of this family is modified policy iteration, and the second member is temporal difference policy iteration.

Modified policy iteration

The astute reader will have noticed that it may be not necessary to fully evaluate the improved policy. In fact, we can take advantage of that to speed up policy iteration. Thus, a simple variant of policy iteration involves doing only a k-step update for the policy evaluation step. For $k = 1$, the algorithm becomes identical to value iteration, while for $k \to \infty$ the algorithm is equivalent to policy iteration, as $\boldsymbol{v}_n = V^{\pi_n}$.

Algorithm 8 Modified policy iteration

Input μ, $\mathcal{S}$.
Initialise $\boldsymbol{v}_0$.
for $n = 1, 2, \ldots$ **do**
  $\pi_n = \arg\max_\pi \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v}_{n-1}$ `// policy improvement`
  $\boldsymbol{v}_n = \mathscr{L}^k_{\pi_n} \boldsymbol{v}_{n-1}$ `// partial policy evaluation`
  break if $\pi_n = \pi_{n+1}$.
end for
Return $\pi_n, \boldsymbol{v}_n$.

Modified policy iteration can perform much better than either pure value iteration or pure policy iteration.

A geometric view

It is perhaps interesting to see the problem from a geometric perspective. This also gives rise to the so-called "temporal-difference" set of algorithms. First, we define the difference operator, which is the difference between a value function vector $\boldsymbol{v}$ and its transformation via the Bellman operator.

difference operator

Definition 6.5.3. The *difference operator* is defined as

$$\mathscr{B}\boldsymbol{v} \triangleq \max_\pi \{\boldsymbol{r} + (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})\boldsymbol{v}\} = \mathscr{L}\boldsymbol{v} - \boldsymbol{v}. \tag{6.5.11}$$

Essentially, it is the change in the value function vector when we apply the Bellman operator. Thus the Bellman optimality equation can be rewritten as

$$\mathscr{B}\boldsymbol{v} = \boldsymbol{0}. \tag{6.5.12}$$

Now let us define the set of greedy policies with respect to a value vector $\boldsymbol{v} \in \mathcal{V}$ to be:

$$\Pi_{\boldsymbol{v}} \triangleq \arg\max_{\pi \in \Pi} \{\boldsymbol{r} + (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})\boldsymbol{v}\}.$$

We can now show the following inequality between the two different value function vectors.

Theorem 6.5.11. *For any $\boldsymbol{v}, \boldsymbol{v}' \in \mathcal{V}$ and $\pi \in \Pi_{\boldsymbol{v}}$*

$$\mathscr{B}\boldsymbol{v}' \geq \mathscr{B}\boldsymbol{v} + (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})(\boldsymbol{v}' - \boldsymbol{v}). \tag{6.5.13}$$

Proof. By definition, $\mathscr{B}\boldsymbol{v}' \geq \boldsymbol{r} + (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})\boldsymbol{v}'$, while $\mathscr{B}\boldsymbol{v} = \boldsymbol{r} + (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})\boldsymbol{v}$. Subtracting the latter from the former gives the result. □

Equation (6.5.13) is similar to the convexity of the Bayes-optimal utility (3.3.6). Geometrically, we can see from a look at Figure 6.5, that applying the Bellman operator on value function always improves it, yet may have a negative effect on the other value function. If the number of policies is finite, then the figure is also a good illustration of the policy iteration algorithm, where each value function improvement results in a new point on the horizontal axis, and the choice of the best improvement (highest line) for that point. In fact, we can write the policy iteration algorithm in terms of the difference operator.

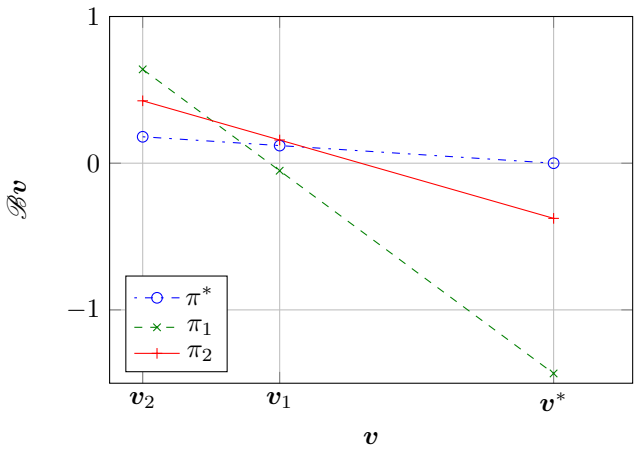


Figure 6.5: The difference operator. The graph shows the effect of the operator for the optimal value function $\boldsymbol{v}^*$, and two arbitrary value functions, $\boldsymbol{v}_1, \boldsymbol{v}_2$. Each line is the improvement effected by the greedy policy π^*, π_1, π_2 with respect to each value function $\boldsymbol{v}^*, \boldsymbol{v}_1, \boldsymbol{v}_2$.

Theorem 6.5.12. *Let $\{\boldsymbol{v}_n\}$ be the sequence of value vectors obtained from policy iteration. Then for any $\pi \in \Pi_{\boldsymbol{v}_n}$,*

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n - (\gamma \boldsymbol{P}_\pi - \boldsymbol{I})^{-1} \mathscr{B}\boldsymbol{v}_n. \tag{6.5.14}$$

Proof. By definition, we have for $\pi \in \Pi_{\boldsymbol{v}_n}$

$$\begin{aligned} \boldsymbol{v}_{n+1} &= (\boldsymbol{I} - \gamma \boldsymbol{P}_\pi)^{-1} \boldsymbol{r} - \boldsymbol{v}_n + \boldsymbol{v}_n \\ &= (\boldsymbol{I} - \gamma \boldsymbol{P}_\pi)^{-1} [\boldsymbol{r} - (\boldsymbol{I} - \gamma \boldsymbol{P}_\pi) \boldsymbol{v}_n] + \boldsymbol{v}_n. \end{aligned}$$

Since $\boldsymbol{r} - (\boldsymbol{I} - \gamma \boldsymbol{P}_\pi)\boldsymbol{v}_n = \mathscr{B}\boldsymbol{v}_n$ the claim follows. □

Temporal-Difference Policy Iteration

In *temporal-difference policy iteration*, similarly to the modified policy iteration algorithm, we replace the next-step value with an approximation $\boldsymbol{v}_n$ of the n-th policy's value. Informally, this approximation is chosen so as to reduce the discrepancy of our value function over time.

At the n-th iteration of the algorithm, we use a policy improvement step to obtain the next policy π_{n+1} given our current approximation $\boldsymbol{v}_n$:

$$\mathscr{L}_{\pi_{n+1}} \boldsymbol{v}_n = \mathscr{L}\boldsymbol{v}_n. \tag{6.5.15}$$

To update the value from $\boldsymbol{v}_n$ to $\boldsymbol{v}_{n+1}$ we rely on the *temporal difference error*, *temporal difference er* defined as:

$$d_n(i,j) = [\boldsymbol{r}(i) + \gamma \boldsymbol{v}_n(j)] - \boldsymbol{v}_n(i). \tag{6.5.16}$$

This can be seen as the difference in the estimate when we move from state i to state j. In fact, it is easy to see that, if our value function estimate satisfies $\boldsymbol{v} = V^{\pi_n}$, then the expected error should be zero, as:

$$\sum_{j \in \mathcal{S}} d_n(i,j) p(j \mid i, \pi_n(i)) = \sum_{j \in \mathcal{S}} [\boldsymbol{r}(i) + \gamma \boldsymbol{v}_n(j)] p(j \mid i, \pi_n(i)) - \boldsymbol{v}_n(i).$$

Note the similarity to the difference operator in modified policy iteration. The idea of the temporal-difference policy iteration is to use adjust the current value $\boldsymbol{v}_n$, using the temporal differences mixed over an infinite number of steps:

$$\boldsymbol{\tau}_n(i) = \sum_{t=0}^{\infty} \mathbb{E}_{\pi_n} \left[(\gamma\lambda)^t d_n(s_t, s_{t+1}) \mid s_0 = i \right], \tag{6.5.17}$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \boldsymbol{\tau}_n. \tag{6.5.18}$$

Here the λ parameter is a simple way to mix together the different temporal difference errors. If $\lambda \to 1$, our error will be dominated by the terms far in the future, while if $\lambda \to 0$, our error $\boldsymbol{\tau}_n$, will be dominated by the short-term discrepancies in our value function. In the end, we shall adjust our value function in the direction of this error.

Putting all of those steps together, we obtain the following algorithm:

Algorithm 9 Temporal-Difference Policy Iteration

Input μ, $\mathcal{S}, \lambda$.
Initialise $\boldsymbol{v}_0$.
for $n = 0, 1, 2, \ldots$ **do**
  $\pi_{n+1} = \arg\max_\pi \boldsymbol{r} + \gamma \boldsymbol{P}_\pi \boldsymbol{v}_n$ `// policy improvement`
  $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \boldsymbol{\tau}_n$ `// temporal difference update.`
  break if $\pi_{n+1} = \pi_n$.
end for
Return $\pi_n, \boldsymbol{v}_n$.

In fact, $\boldsymbol{v}_{n+1}$ is the unique fixed point of the following equation:

$$\mathscr{D}_n \boldsymbol{v} \triangleq (1-\lambda) \mathscr{L}_{\pi_{n+1}} \boldsymbol{v}_n + \lambda \mathscr{L}_{\pi_{n+1}} \boldsymbol{v}. \tag{6.5.19}$$

That is, if we repeatedly apply the above operator to some vector $\boldsymbol{v}$, then at some point we shall obtain a fixed point $\boldsymbol{v}^* = \mathscr{D}_n \boldsymbol{v}^*$. It is interesting to see what happens at the two extreme choices of λ in this case. For $\lambda = 1$, this becomes identical to standard policy iteration, as the fixed point satisfies $\boldsymbol{v}^* = \mathscr{L}_{\pi_{n+1}} \boldsymbol{v}^*$, so then $\boldsymbol{v}^*$ must be the value of policy π_{n+1}. For $\lambda = 0$, one obtains standard value iteration, as the fixed point is reached under one step and is simply $\boldsymbol{v}^* = \mathscr{L}_{\pi_{n+1}} \boldsymbol{v}_n$, i.e. the approximate value of the one-step greedy policy. In other words, the new value vector is moved only partially towards the direction of the Bellman update, depending on how we choose λ.

Linear programming

Perhaps surprisingly, we can also solve Markov decision processes through linear programming. The main idea is to reformulate the maximisation problem as a linear optimisation problem with linear constraints. The first step in our procedure is to recall that there is an easy way to determine whether a particular $\boldsymbol{v}$ is an upper bound on the optimal value function $\boldsymbol{v}^*$, since if

$$\boldsymbol{v} \geq \mathscr{L}\boldsymbol{v}$$

then $\boldsymbol{v} \geq \boldsymbol{v}^*$. In order to transform this into a linear program, we must first define a scalar function to minimise. We can do this by selecting some arbitrary distribution on the states $\boldsymbol{y} \in \Delta^{|\mathcal{S}|}$. Then we can write the following linear program.

Primal linear program

$$\min_{\boldsymbol{v}} \boldsymbol{y}^\top \boldsymbol{v},$$

such that

$$\boldsymbol{v}(s) - \gamma \boldsymbol{p}_{s,a}^\top \boldsymbol{v} \geq r(s,a), \qquad \forall a \in \mathcal{A}, s \in \mathcal{S},$$

where we use $\boldsymbol{p}_{s,a}$ to denote the vector of next state probabilities $p(j \mid s, a)$.

Note that the inequality condition is equivalent to $\boldsymbol{v} \geq \mathscr{L}\boldsymbol{v}$. Consequently, the problem is to find the smallest $\boldsymbol{v}$ that satisfies this inequality. When $\mathcal{A}, \mathcal{S}$ are finite, it is easy to see that this will be the optimal value function and the Bellman equation is satisfied.

It also pays to look at the dual linear program, which is in terms of a maximisation. This time, instead of finding the minimal upper bound on the value function, we find the maximal cumulative discounted state-action visits $x(s,a)$ that are consistent with the transition kernel of the process.

Dual linear program

$$\max_x \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} r(s,a) x(s,a)$$

such that $x \in \mathbb{R}_+^{|\mathcal{S} \times \mathcal{A}|}$ and

$$\sum_{a \in \mathcal{A}} x(j,a) - \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \gamma p(j \mid s,a) x(s,a) = y(j) \qquad \forall j \in \mathcal{S}.$$

with $\boldsymbol{y} \in \Delta^{|\mathcal{S}|}$.

In this case, x can be interpreted as the discounted sum of state-action visits, as proved by the following theorem.

Theorem 6.5.13. *For any policy π,*

$$x_\pi(s,a) = \mathbb{E}_{\pi,\mu}\left\{\sum \gamma^n \mathbb{I}\left\{s_t = s, a_t = a \mid s_0 \sim y\right\}\right\}$$

is a feasible solution to the dual problem. On the other hand, if x is a feasible solution to the dual problem then $\sum_a x(s,a) > 0$. Finally, if we define the strategy

$$\pi(a \mid s) = \frac{x(s,a)}{\sum_{a' \in \mathcal{A}} x(s,a')}$$

then $x_\pi = x$ is a feasible solution.

The equality condition ensures that x is consistent with the transition kernel of the Markov decision process. Consequently, the program can be seen as search among all possible cumulative state-action distributions to find the one giving the highest total reward.

6.6 Optimality Criteria

In all previous cases, we assumed a specific discount rate, or a finite horizon for our problem. This section will give an overview of different optimality criteria when there is no discounting and the horizon is infinite, and compare them to the ones already discussed in this chapter.

As mentioned previously, the following two views of discounted reward processes are equivalent.

Infinite horizon, discounted
Discount factor γ such that

$$U_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \qquad \Rightarrow \mathbb{E}\, U_t = \sum_{k=0}^{\infty} \gamma^k \,\mathbb{E}\, r_{t+k} \tag{6.6.1}$$

Geometric horizon, undiscounted
At each step t, the process terminates with probability $1-\gamma$:

$$U_t^T = \sum_{k=0}^{T-t} r_{t+k}, \quad T \sim \mathit{Geom}(1-\gamma) \quad \Rightarrow \mathbb{E}\, U_t = \sum_{k=0}^{\infty} \gamma^k \,\mathbb{E}\, r_{t+k} \tag{6.6.2}$$

$$V_\gamma^\pi(s) \triangleq \mathbb{E}(U_t \mid s_t = s)$$

The expected total reward criterion

$$V_t^{\pi,T} \triangleq \mathbb{E}_\pi\, U_t^T, \qquad V^\pi \triangleq \lim_{T\to\infty} V^{\pi,T} \tag{6.6.3}$$

Dealing with the limit

- Consider μ s.t. the limit exists $\forall\pi$.

$$V_+^\pi(s) \triangleq \mathbb{E}_\pi\left(\sum_{t=1}^{\infty} r_t^+ \;\middle|\; s_t = s\right), \quad V_-^\pi(s) \triangleq \mathbb{E}_\pi\left(\sum_{t=1}^{\infty} r_t^- \;\middle|\; s_t = s\right) \tag{6.6.4}$$

$$r_t^+ \triangleq \max\{-r, 0\}, \qquad r_t^- \triangleq \max\{r, 0\}. \tag{6.6.5}$$

- Consider μ s.t. $\exists\pi^*$ for which V^{π^*} exists and

$$\lim_{T\to\infty} V^{\pi^*,T} = V^{\pi^*} \geq \limsup_{T\to\infty} V^{\pi,T}.$$

- Use optimality criteria sensitive to the divergence rate.

The average reward (gain) criterion

Definition 6.6.1. The gain g of a policy π starting from state s is the expected average reward the policy obtains when starting from that state.

$$g^\pi(s) \triangleq \lim_{T\to\infty} \frac{1}{T} V^{\pi,T}(s) \tag{6.6.6}$$

$$g_+^\pi(s) \triangleq \limsup_{T\to\infty} \frac{1}{T} V^{\pi,T}(s), \qquad g_-^\pi(s) \triangleq \liminf_{T\to\infty} \frac{1}{T} V^{\pi,T}(s) \tag{6.6.7}$$

If $\lim_{T\to\infty} \mathbb{E}(r_T \mid s_0 = s)$ exists then it equals $g^\pi(s)$.

Let Π be the set of all history-dependent, randomised policies. Using our overloaded symbols, we have that

Definition 6.6.2. π^* is *total reward optimal* if

$$V^{\pi^*}(s) \geq V^\pi(s) \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

Definition 6.6.3. π^* is *discount optimal* for $\gamma \in [0, 1)$ if

$$V_\gamma^{\pi^*}(s) \geq V_\gamma^\pi(s) \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

Definition 6.6.4. π^* is *gain optimal* if

$$g^{\pi^*}(s) \geq g^\pi(s) \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

Overtaking optimality π^* is *overtaking optimal* if

$$\liminf_{T\to\infty} \left[V^{\pi^*,T}(s) - V^{\pi,T}(s)\right] \geq 0 \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

However, no overtaking optimal policy may exist.

π^* is *average-overtaking optimal* if

$$\liminf_{T\to\infty} \frac{1}{T}\left[V^{\pi^*,T}(s) - V^{\pi}_{+}(s)\right] \geq 0 \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

Sensitive discount optimality π^* is *n-discount optimal* for $n \in \{-1, 0, 1, \ldots\}$ if

$$\liminf_{\gamma\uparrow 1} (1-\gamma)^{-n}\left[V^{\pi^*}_{\gamma}(s) - V^{\pi}_{\gamma}(s)\right] \geq 0 \qquad \forall s \in \mathcal{S}, \pi \in \Pi.$$

A policy is *Blackwell optimal* if $\forall s$, $\exists \gamma^*(s)$ such that

$$V^{\pi^*}_{\gamma}(s) - V^{\pi}_{\gamma}(s) \geq 0, \qquad \forall \pi \in \Pi, \gamma^*(s)\gamma\gamma < 1.$$

Lemma 6.6.1. *If a policy is m-discount optimal then it is n-discount optimal for all $n \leq m$.*

Lemma 6.6.2. *Gain optimality is equivalent to -1-discount optimality.*

The different optimality criteria summarised here are treated in detail in Puterman [1994] Chapter 5.

6.7 Summary

Markov decision processes can represent shortest path problems, stopping problems, experiment design problems, multi-armed bandit problems and reinforcement learning problems.

Bandit problems are the simplest type of Markov decision process, since they have a fixed, never-changing state. However, to solve them, one can construct a Markov decision processes in belief space, within a Bayesian framework. It is then possible to apply backwards induction to find the optimal policy.

Backwards induction is applicable more generally to arbitrary Markov decision processes. For the case of infinite-horizon problems, it is referred to as value iteration, as it converges to a fixed point. It is tractable when either the state space $\mathcal{S}$ or the horizon T are small (finite).

When the horizon is infinite, policy iteration can also be used to find optimal policies. It is different from value iteration in that at every step, it fully evaluates a policy before the improvement step, while value iteration only performs a partial evaluation. In fact, at the n-th iteration, value iteration has calculated the value of an n-step policy.

We can arbitrarily mix between the two extremes of policy iteration and value iteration in two ways. Firstly, we can perform a k-step partial evaluation. When $k = 1$, we obtain value iteration, and when $k \to \infty$, we obtain policy iteration. The generalised algorithm is called modified policy iteration. Secondly, we can perform adjust our value function by using a temporal difference error of values in future time steps. Again, we can mix liberally between policy iteration

and value iteration by focusing on errors far in the future (policy iteration) or on short-term errors (value iteration).

Finally, it is possible to solve MDPs through linear programming. This is done by reformulating the problem as a linear optimisation with constraints. In the primal formulation, we attempt to find a minimal upper bound on the optimal value function. In the dual formulation, our goal is to find a distribution on state-action visitations that maximises expected utility and is consistent with the MDP model.

6.8 Further reading

See the last chapter of [DeGroot, 1970] for further information on the MDP formulation of bandit problems in the decision theoretic setting. This was explored in more detail in Duff's PhD thesis [Duff, 2002]. When the number of (information) states in the bandit problem is finite, Gittins [1989] has proven that it is possible to formulate simple index policies. However, this is not generally applicable. Easily computable, near-optimal heuristic strategies for bandit problems will be given in Chapter 10. The decision-theoretic solution to the unknown MDP problem will be given in Chapter 9.

Further theoretical background on Markov decision processes, including many of the theorems in Section 6.5, can be found in [Puterman, 1994]. Chapter 2 of Bertsekas and Tsitsiklis [1996] gives a quick overview of MDP theory from the operator perspective. The introductory reinforcement learning book of Sutton and Barto [1998] also explains the basic Markov decision process framework.

6.9 Exercises

6.9.1 Medical diagnosis

EXERCISE 27 (Continuation of exercise 22). Now consider the case where you have the choice between tests to perform First, you observe S, whether or not the patient is a smoker. Then, you select a test to make: $d_1 \in \{\texttt{X-ray}, \texttt{ECG}\}$. Finally, you decide whether or not to treat for ASC: $d_2 \in \{\texttt{heart treatment}, \texttt{no treatment}\}$. An untreated ASC patient may die with probability 2%, while a treated one with probability 0.2%. Treating a non-ASC patient result in death with probability 0.1%.

1. Draw a decision diagram, where:
   - S is an observed random variable taking values in $\{0, 1\}$.
   - A is an hidden variable taking values in $\{0, 1\}$.
   - C is an hidden variable taking values in $\{0, 1\}$.
   - d_1 is a choice variable, taking values in $\{\texttt{X-ray}, \texttt{ECG}\}$.
   - r_1 is a result variable, taking values in $\{0, 1\}$, corresponding to negative and positive tests results.
   - d_2 is a choice variable, which depends on the test results, d_1 and on S.
   - r_2 is a result variable, taking values in $\{0, 1\}$ corresponding to the patient dying (0), or living (1).
2. Let $d_1 = \texttt{X-ray}$, and assume the patient suffers from ACS, i.e. $A = 1$. How is the posterior distributed?
3. What is the optimal decision rule for this problem?

6.9.2 Markov Decision Process theory

EXERCISE 28 (30). Show that the expected discounted total reward of any given policy is equal to the expected undiscounted total reward with a finite, but random horizon T. In particular, let T be distributed according to a geometric distribution on $\{1, 2, \ldots\}$ with parameter $1 - \gamma$. Then show that:

$$\mathbb{E} \lim_{T \to \infty} \sum_{k=0}^{T} \gamma^k r_k = \mathbb{E} \left(\sum_{k=0}^{T} r_k \;\middle|\; T \sim \mathit{Geom}(1 - \gamma) \right).$$

6.9.3 Automatic algorithm selection

Consider the problem of selecting algorithms for finding solutions to a sequence of problems. Assume you have n algorithms to choose from. At time t, you get a task and choose the i-th algorithm. Assume that the algorithms are randomised, so that the i-th algorithm will find a solution with some unknown probability. Our aim is to maximise the expected total number of solutions found. Consider the following specific cases of this problem:

EXERCISE 29 (120). In this case, we assume that the probability that the i-th algorithm successfully solves the t-th task is always p_i. Furthermore, tasks are in no way distinguishable from each other. In each case, assume that $p_i \in \{0.1, \ldots, 0.9\}$ and a prior distribution $\xi_i(p_i) = 1/9$ for all i, with a complete belief $\xi(\boldsymbol{p}) = \prod_i \xi_i(p_i)$, and formulate the problem as a decision-theoretic n-armed bandit problem with reward

at time t being $r_t = 1$ if the task is solved and $r_t = 0$ if the problem is not solved. Whether or not the task at time t is solved or not, at the next time-step we go to the next problem. Our aim is to find a policy π mapping from the history of observations to selection of algorithms such that we maximise the total reward to time T in expectation

$$\mathbb{E}_{\xi,\pi} U_0^T = E_{\xi,\pi} \sum_{t=1}^{T} r_t.$$

1. Characterise the essential difference between maximising U_0^0, U_0^1, U_0^2?
2. For $n = 3$, calculate the maximum expected utility

   $$\max_{\pi} \mathbb{E}_{\xi,\pi} U_0^T$$

   using backwards induction for $T \in \{0, 1, 2, 3, 4\}$ and report the expected utility in each case. *Hint: Use the decision-theoretic bandit formulation to dynamically construct a Markov decision process which you can solve with backwards induction. See also the extensive decision rule utility from exercise set 3.*
3. Now utilise the backwards induction algorithm developed in the previous step in a problem where we receive a sequence of N tasks to solve and our utility is

   $$U_0^N = \sum_{t=1}^{N} r_t$$

   At each step $t \leq N$, find the optimal action by calculating $\mathbb{E}_{\xi,\pi} U_t^{t+T}$ for $T \in \{0, 1, 2, 3, 4\}$ take it. *Hint: At each step you can update your prior distribution using the same routine you use to update your prior distribution. You only need consider $T < N - t$.*
4. Develop a simple heuristic algorithm of your choice and compare its utility with the utility of the backwards induction. Perform 10^3 simulations, each experiment running for $N = 10^3$ time-steps and average the results. How does the performance improve? *Hint: If the program runs too slowly go only up to $T = 3$*

6.9.4 Scheduling

You are controlling a small processing network that is part of a big CPU farm. You in fact control a set of n processing nodes. At time t, you may be given a job of class $x_t \in X$ to execute. Assume these are identically and independently drawn such that $\mathbb{P}(x_t = k) = p_k$ for all t, k. With some probability p_0, you are not given a job to execute at the next step. If you do have a new job, then you can either:

(a) Ignore the job

(b) Send the job to some node i. If the node is already active, then the previous job is lost.

Not all the nodes and jobs are equal. Some nodes are better at processing certain types of jobs. If the i-th node is running a job of type $k \in X$, then it has a probability of finishing it within that time step equal to $\phi_{i,k} \in [0, 1]$. Then the node becomes free, and can accept a new job.

For this problem, assume that there are $n = 3$ nodes and $k = 2$ types of jobs and that the completion probabilities are given by the following matrix:

$$\boldsymbol{\Phi} = \begin{bmatrix} 0.3 & 0.1 \\ 0.2 & 0.2 \\ 0.1 & 0.3 \end{bmatrix}. \tag{6.9.1}$$

Also, we set $p_0 = 0.1, p_1 = 0.4, p_2 = 0.5$ to be the probabilities of not getting any job, and the probabilities of the two job types respectively. We wish to find the policy maximising the expected total reward given the MDP model π:

$$\mathbb{E}_{\mu,\pi} \sum_{t=0}^{\infty} \gamma^t r_t, \tag{6.9.2}$$

with $\gamma = 0.9$ and where we get a reward of 1 every time a job is completed.

More precisely, at each time step t, the following events happen:

1. A new job x_t appears
2. Each node either continues processing, or completes its current job and becomes free. You get a reward r_t equal to the number of nodes that complete their jobs within this step.
3. You decide whether to ignore the new job or add it to one of the nodes. If you add a job, then it immediately starts running for the duration of the time step. (If the job queue is empty then you cannot add a job to a node, obviously)

Exercise 30 (180). Solve the following problems:

1. Identify the state and action space of this problem and formulate it as a Markov decision process. *Hint: Use independence of the nodes to construct the MDP parameters.*
2. Solve the problem using value iteration, using the stopping criterion indicated in theorem 15, equation (5.5), in Chapter VII, with $\epsilon = 0.1$. Indicate the number of iterations needed to stop.
3. Solve the problem using policy iteration. Indicate the number of iterations needed to stop. *Hint: You can either modify the value iteration algorithm to perform policy evaluation, using the same epsilon, or you can use the linear formulation. If you use the latter, take care with the inverse!*
4. Now consider an alternative version of the problem, where we suffer a penalty of 0.1 (i.e. we get a negative reward) for each time-step that each node is busy. Are the solutions different?
5. Finally consider a version of the problem, where we suffer a penalty of 10 (i.e. we get a negative reward) each time we cancel an executing job. Are the solutions different?
6. Plot the value function for the optimal policy in each setting.

Hint: To verify that your algorithms work, test them first on a smaller MDP with known solutions. For example, `http://webdocs.cs.ualberta.ca/~sutton/book/ebook/node35.html`

6.9.5 General questions

EXERCISE 31 (20!). 1. What in your view is the fundamental advantages and disadvantages of modelling problems as Markov decision processes?

2. Is the algorithm selection problem of Exercise 29 solvable with policy iteration? If so, how? What are the fundamental similarities and differences between the decision-theoretic finite-horizon bandit setting of exercise 1 and the infinite-horizon MDP settings of exercise 2?

Chapter 7

Simulation-based algorithms

7.1 Introduction

In this chapter, we consider the general problem of reinforcement learning in dynamic environments. Up to now, we have only examined a solution method for bandit problems, which are only a special case. The Bayesian decision-theoretic solution is to *reduce* the bandit problem to a *Markov decision process* which can then be solved with backwards induction.

We also have seen that Markov decision processes can be used to *describe environments* in more general reinforcement learning problems. When our knowledge of the MDP describing these problems is perfect, then we can employ a number of standard algorithms to find the optimal policy. However, in the actual reinforcement learning problem, the model of the environment is *unknown*. However, as we shall see later, both of these ideas can be combined to solve the general reinforcement learning problem.

The main focus of this chapter is how to simultaneously learn about the underlying process and act to maximise utility in an *approximate* way. This can be done through approximate dynamic programming, where we replace the actual unknown dynamics of the Markov decision process with estimates. The estimates can be improved by drawing samples from the environment, either by acting within the real environment or using a simulator. In both cases we end up with a number of algorithms that can be used for reinforcement learning. Although may not be performing as well as the Bayes-optimal solution, these have a low enough computational complexity that they are worth investigating in practice.

It is important to note that the algorithms in this chapter can be quite far from optimal. They may converge eventually to an optimal policy, but they may not accumulate a lot of reward while still learning. In that sense, they are not solving the full reinforcement learning problem because their *online* performance can be quite low.

For simplicity, we shall first return to the example of bandit problems. As before, we have n actions corresponding to probability distributions P_i on the real numbers $\{P_i \mid i = 1, \dots, n\}$ and our aim is to maximise to total reward (in expectation). Had we known the distribution, we could simply always the maximising action, as the expected reward of the i-th action can be easily calculated from P_i and the reward only depends on our current action.

As the P_i are unknown, we must use a history-dependent policy. In the remainder of this section, we shall examine algorithms which asymptotically convergence to the optimal policy (which, in the case of bandits corresponds to pulling always pulling the best arm), but for which we cannot always guarantee a good initial behaviour.

7.1.1 The Robbins-Monro approximation

In this setting, we wish to to replace the actual Markov decision process in which we are acting, with an estimate that will eventually converge to the true process. At the same time, we shall be taking actions which are nearly-optimal with respect to the estimate.

To approximate the process, we shall use the general idea of a Robbins-Monro stochastic approximation [Robbins and Monro, 1951]. This entails maintaining a *point estimate* of the parameter we want to approximate and perform *random*

steps that on average move towards the solution, in a way to be made more precise later. The stochastic approximation actually defines a large class of procedures, and it contains stochastic gradient descent as a special case.

Algorithm 10 Robbins-Monro bandit algorithm

1: **input** Step-sizes $(\alpha_t)_t$, initial estimates $(\mu_{i,0})_i$, policy π.
2: **for** $t = 1, \ldots, T$ **do**
3: Take action $a_t = i$ with probability $\pi(i \mid a_1, \ldots, a_{t-1}, r_1, \ldots, r_{t-1})$.
4: Observe reward r_t.
5: $\mu_{t,i} = \alpha_{i,t} r_t + (1 - \alpha_{i,t})\mu_{i,t-1}$ `// estimation step`
6: $\mu_{t,i} = \mu_{j,t-1}$ for $j \neq i$.
7: **end for**
8: **return** $\boldsymbol{\mu_T}$

An bandit algorithm that uses a Robbins-Monro approximation is given in Algorithm 10. The input is a particular policy π, which defines probability distribution over the next actions given the observed history, a set of initial estates $\mu_{i,0}$ for the bandit means, and a sequence of step sizes α.

The algorithm can be separated in two parts. Taking actions according to the policy (step 3) and the observation of rewards with an update of the estimated values (steps 4-6). The policy itself is an input to the algorithm, but it will in practice only depend on $\mu_{t,i}$ and t; we shall discuss appropriate policies later. Regarding the estimation itself, note that only the estimate for the arm which we have drawn is updated. As we shall see later, this particular update rule chosen in this case be seen as trying to minimise the expected squared error between the estimated reward, and the random reward obtained by each bandit. Consequently, the variance of the reward of each bandit plays an important role.

The step-sizes α must obey certain constraints in order for the algorithm to work, in particular it must decay neither too slowly, nor too fast. There is one particular choice, for which our estimates are in fact the mean estimate of the expected value of the reward for each action i, which is a natural choice if the bandits are stationary.

The other question is what policy to use to take actions. We must take all actions often enough, so that we have good estimates for the expected reward of every bandit. One simple way to do it is to play the apparently best bandit most of the time, but to sometimes select bandits randomly. This is called ϵ-greedy action selection. This ensures that all actions are tried a sufficient number of times.

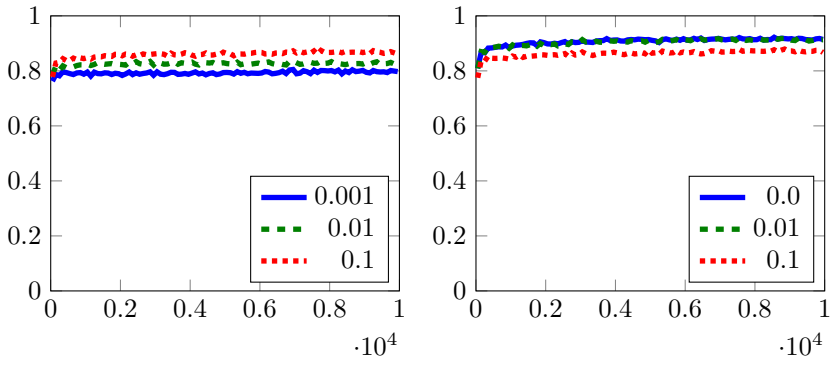
Definition 7.1.1 (ϵ-greedy policy)**.**

$$\hat{\pi}^*_\epsilon \triangleq (1 - \epsilon_t)\hat{\pi}^*_t + \epsilon_t \mathcal{U}nif(\mathcal{A}), \tag{7.1.1}$$

$$\hat{\pi}^*_t(i) = \mathbb{I}\left\{i \in \hat{\mathcal{A}}^*_t\right\} / |\hat{\mathcal{A}}^*_t|, \qquad \hat{\mathcal{A}}^*_t = \arg\max_{i \in \mathcal{A}} \mu_{t,i} \tag{7.1.2}$$

This is formally defined in Definition 7.1.1. We allow the randomness of the policy to depend on t. This is because, as our estimates converge to the true values, we wish to reduce randomness so as to converge to the optimal policy.

The main two parameters of the algorithm are the amount of randomness in the ϵ-greedy action selection and the step-size α in the estimation. Both of them have a significant effect in the performance of the algorithm. Although we could vary them with time, it is perhaps instructive to look at what happens for fixed values of ϵ, α. Figures 7.1 show the average reward obtained, if we keep the step size α or the randomness ϵ fixed, respectively, with initial estimates $\mu_{0,i} = 0$.



(a) fixed ϵ (b) fixed α

Figure 7.1: For the case of fixed $\epsilon_t = 0.1$, the step size is $\alpha \in \{0.01, 0.1, 0.5\}$. For the case of fixed α, the exploration rate is

For a fixed ϵ, we find that larger values of α tend to give a better result eventually, while smaller values have a better initial performance. This is a natural trade-off, since large α appears to "learn" fast, but it also "forgets" quickly. That is, for a large α, our estimates mostly depend upon the last few rewards observed.

Things are not so clear-cut for the choice of ϵ. We see that the choice of $\epsilon = 0$, is significantly worse than $\epsilon = 0.1$. So, that appears to suggest that there is an optimal level of exploration. How should that be determined? Ideally, we should be able to to use the decision-theoretic solution seen earlier, but perhaps a good heuristic way of choosing ϵ may be good enough.

7.1.2 The theory of the approximation

Here we quickly review some basic results of stochastic approximation theory. Complete proofs can be found in Bertsekas and Tsitsiklis [1996]. The main question here is whether our estimates converge to the right values, and whether the complete algorithm itself converges to a optimal policy. We are generally not interested in how much reward we obtain during the optimisation process, but only on asymptotic convergence.

We first consider the core problem of stochastic approximation itself. In particular, we shall cast the approximation problem as a minimisation problem, i.e. we shall define a function f such that, if μ_t is our estimate of μ, then f is minimised at $f(\mu_t)$. Then, given the function $f : \mathbb{R}^n \to \mathbb{R}$, we wish to develop an algorithm that generates a sequences of values μ_t which converges to some

μ^* that is a local minimum, or a stationary point for f. For strictly convex f, this would also be a global minimum.

In particular, we examine algorithms which maintain estimates μ_t over time, with the update equation:

$$\mu_{t+1} = \mu_t + \alpha_t z_{t+1}. \tag{7.1.3}$$

Here μ_t is our estimate, α_t is a step-size and z_t is a direction. In addition, we use $h_t \triangleq \{\mu_t, z_t, \alpha_t, \ldots\}$ to denote the complete history of the algorithm.

The above algorithm can be shown to converge to a stationary point of f under certain assumptions. Sufficient conditions include continuity and smoothness properties of f and the update direction z. In particular, we shall assume the following about the function f that we wish to minimise.

Assumption 7.1.1. *Assume a function $f : \mathbb{R}^n \to \mathbb{R}$ such that:*

(i) $f(x) \geq 0$ *for all* $x \in \mathbb{R}^n$.

*(ii) (*Lipschitz derivative*) f is continuously differentiable (i.e. the derivative ∇f exists and is continuous) and $\exists L > 0$ such that:*

$$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|, \qquad \forall\ x, y \in \mathbb{R}^n$$

*(iii) (*Pseudo-gradient*) $\exists c > 0$ such that:*

$$c \|\nabla f(\mu_t)\|^2 \leq -\nabla f(\mu_t)^\top \mathbb{E}(z_{t+1} \mid h_t), \qquad \forall\ t.$$

(iv) $\exists K_1, K_2 > 0$ such that

$$\mathbb{E}(\|z_{t+1}\|^2 \mid h_t) \leq K_1 + K_2 \|\nabla f(\mu_t)\|^2$$

Condition (ii) is a very basic condition for convergence. It basically ensures that the function is well-behaved, so that gradient-following methods can easily find the minimum. Condition (iii) combines two assumptions in one. Firstly, that expected direction of update always decreases cost, and secondly that the squared norm of the gradient is not too large relative to the size of the update. Finally, condition (iv) ensures that update is bounded in expectation relative to the gradient. One can see how putting together the last two conditions ensures that the expected direction of our update is correct, and that its norm is bounded.

Theorem 7.1.1. *For the algorithm*

$$\mu_{t+1} = \mu_t + \alpha_t z_{t+1},$$

where $\alpha_t \geq 0$ satisfy

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2 < \infty, \tag{7.1.4}$$

and under Assumption 7.1.1, with probability 1:

1. *The sequence* $\{f(\mu_t)\}$ *converges.*
2. $\lim_{t\to\infty} \nabla f(\mu_t) = 0$.
3. *Every limit point* μ^* *of* μ_t *satisfies* $\nabla f(\mu^*) = 0$.

The above conditions are not necessary conditions. Alternative sufficient conditions relying on contraction properties are discussed in detail in Bertsekas and Tsitsiklis [1996]. The following example illustrates the impact of the choice of step size schedule on convergence.

Estimating the mean of a Gaussian distribution.
Consider a sequence of observations x_t, sampled from a Gaussian distribution with mean $1/2$ and variance 1, in other words $x_t \sim \mathcal{N}(0.5, 1)$. We compare three different step-size schedules, with update direction:

$$z_{t+1} = x_{t+1} - \mu_t.$$

The first one, $\alpha_t = 1/t$, satisfies both assumptions. The second one, $\alpha_t = 1/\sqrt{t}$, reduces too slowly, and the third one, $\alpha_t = t^{-3/2}$, approaches zero too fast.

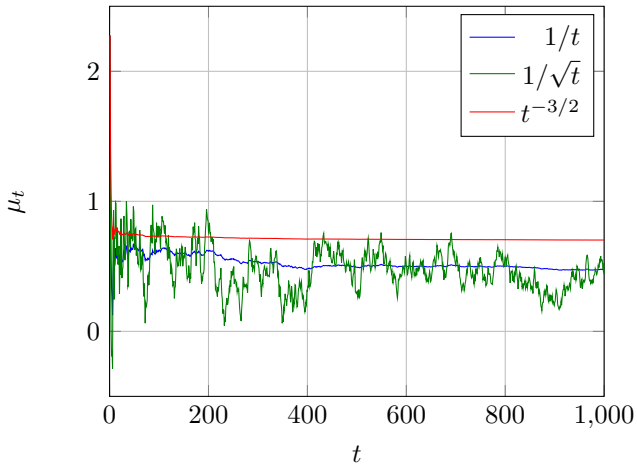


Figure 7.2: Estimation of the expectation of $x_t \sim \mathcal{N}(0.5, 1)$ using use three step-size schedules.

Figure 7.2 demonstrates the convergence, or lack thereof, of our estimates μ_t to the expected value. In fact, the schedule $t^{-3/2}$ converges to a value quite far away from the expected value, while the slow schedule $1/\sqrt{t}$ oscillates.

EXAMPLE 33 (Robbins-Monroe conditions for Bernoulli bandits.). Let us now consider the conditions for convergence of the estimates of the bandit algorithm we examined before. Firstly, the function that we wish to minimise relates to the difference between our own estimates and the actual expected reward of the bandit arms. For that reason, we can write the function that we wish to approximate

7.2 Dynamic problems

It is possible to extend the ideas outlined in the previous section to dynamic settings. We simply need to have a policy that is greedy with respect to our estimates, and a way to update our estimates so that they converge to the actual Markov decision process we are acting in. However, the dynamic setting presents one essential difference. Our policy now affects which sequences of states we observe, while before it only affected the rewards. While in the bandit problem we could freely select an arm to pull, we might no longer be able to go to an arbitrary state.[1] Otherwise, the algorithmic structure remains the same and is described below.

Algorithm 11 Generic reinforcement learning algorithm

1: **input** Update-rule $f : \Theta \times \mathcal{S}^2 \times \mathcal{A} \times \mathscr{R} \to \Theta$, initial parameters $\theta_0 \in \Theta$, policy $\pi : \mathcal{S} \times \Theta \to \mathbb{\Delta}(\mathcal{A})$.
2: **for** $t = 1, \dots, T$ **do**
3: $a_t \sim \pi(\cdot \mid \theta_t, s_t)$ `// take action`
4: Observe reward r_{t+1}, state s_{t+1}.
5: $\theta_{t+1} = f(\theta_t, s_t, a_t, r_{t+1}, s_{t+1})$ `// update estimate`
6: **end for**

What should we estimate? For example, θ_t could be describing a posterior distribution over MDPs, or a distribution over parameters. What policy should we use? For example, we could try and use the Bayes-optimal policy with respect to θ, or some heuristic policy.

Example 34 (The chain task). The chain task has two actions and five states, as shown in Fig. 7.3. The reward in the leftmost state is 0.2 and 1.0 in the rightmost state, and zero otherwise. The first action (dashed, blue) takes you to the right, while the second action (solid, red) takes you to the first state. However, there is a probability 0.2 with which the actions have the opposite effects. The value function of the chain task for a discount factor $\gamma = 0.95$ is shown in Table 7.1.

The chain task is a very simple, but well-known task, used to test the efficacy of reinforcement learning algorithms. In particular, it is useful for analysing how algorithms solve the exploration-exploitation trade-off, since in the short run simply moving to the leftmost state is advantageous. For a long enough horizon or large enough discount factor, algorithms should be incentivised to more fully explore the state space. A variant of this task, with action-dependent rewards (but otherwise equivalent) was used by [Dearden et al., 1998].

[1] This actually depends on what the exact setting is. If the environment is a simulation, then we could try and start from an arbitrary state, but in the reinforcement learning setting this is not the case.

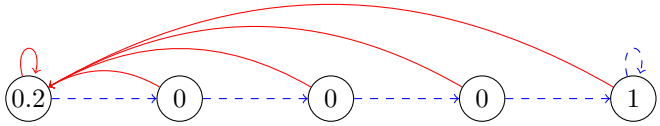


Figure 7.3: The chain task

s	s_1	s_2	s_3	s_4	s_5
$V^*(s)$	7.6324	7.8714	8.4490	9.2090	10.209
$Q^*(s,1)$	7.4962	7.4060	7.5504	7.7404	8.7404
$Q^*(s,2)$	7.6324	7.8714	8.4490	9.2090	10.2090

Table 7.1: The chain task's value function for $\gamma = 0.95$

7.2.1 Monte-Carlo policy evaluation and iteration

reset action
simulation

To make things as easy as possible, let us assume that we have a way to start the environment from any arbitrary state. That would be the case if the environment had a *reset action*, or if we were simply running an accurate *simulation*.

We shall begin with simplest possible problem, that of estimating the expected utility of each state for a specific policy. This can be performed with Monte-Carlo policy evaluation. In the standard setting, we can the value function for every state by approximating the expectation with the sum of rewards obtained over multiple trajectories starting from each state. The k-th trajectory starts from some initial state $s_0 = s$ and the next states are sampled as follows

$$a_t^{(k)} \sim \pi(a_t \mid h_t).r_t^{(k)} \quad \sim \mathbb{P}_\mu(r_t \mid s_t^{(k)}, a_t^{(k)}) s_{t+1}^{(k)} \sim \mathbb{P}_\mu(s_{t+1} \mid s_t^{(k)}, a_t^{(k)}). \quad (7.2.1)$$

Then the value function satisfies

$$V_\mu^\pi(s) \triangleq \mathbb{E}_\mu^\pi(U \mid s_1 = s) \approx \frac{1}{K} \sum_{k=1}^{K} \sum_{t=1}^{T} r_t^{(k)},$$

where $r_t^{(k)}$ is the sequence of rewards obtained from the k-th trajectory.

Algorithm 12 Stochastic policy evaluation

```
input Initial parameters v_0, Markov policy π.
for s ∈ S do
    s_1 = s.
    for k = 1, ..., K do
        Run policy π for T steps.
        Observe utility U_k = Σ_t r_t.
        Update estimate v_{k+1}(s) = v_k(s) + α_k(U_k − v_k(s))
    end for
end for
return v_K
```

For $\alpha_k = 1/k$ and iterating over all $\mathcal{S}$, this is the same as Monte-Carlo policy evaluation.

7.2.2 Monte Carlo updates

Note that $s_1, \dots, s_T$ contains $s_k, \dots, s_T$.

This suggests that we could update the value of all encountered states, as we also have the utility starting from each state. We call this algorithm

Algorithm 13 Every-visit Monte-Carlo update

input Initial parameters $\boldsymbol{v}_k$, trajectory $s_1, \dots, s_T$, rewards $r_1, \dots, r_T$ visit counts n.
for $t = 1, \dots, T$ **do**
  $U_t = \sum_{t=1}^{T} r_t$.
  $n_t(s_t) = n_{t-1}(s_t) + 1$
  $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_{n_t(s_t)}(s_t)(U_t - \boldsymbol{v}_t(s_t))$
  $n_t(s) = n_{t-1}(s)$, $\boldsymbol{v}_t(s) = \boldsymbol{v}_{t-1}(s)$ $\forall s \neq s_t$.
end for
return $\boldsymbol{v}_K$

For a proper Monte-Carlo estimate, when the environment is stationary $\alpha_{n_t(s_t)}(s_t) = 1/n_t(s_t)$. Nevertheless, this type of estimate can be biased, as can be seen by the following example.

EXAMPLE 35. Consider a two-state chain with $\mathbb{P}(s_{t+1} = 1 \mid s_t = 0) = \delta$ and $\mathbb{P}(s_{t+1} = 1 \mid s_t = 1) = 1$, and reward $r(1) = 1$, $r(0) = 0$. Then the every-visit estimate is biased.

Let us consider the discounted setting. Then value of the second state is $1/(1-\gamma)$ and the value of the first state is $\sum_k (\delta\gamma)^k = 1/(1-\delta\gamma)$. Consider the every-visit Monte-Carlo update. The update is going to be proportional to the number of steps you spend in that state.

In order to avoid the bias, we must instead look at only the first visit to every state. This eliminates the dependence between states and is called the first visit Monte-Carlo update .

Algorithm 14 First-visit Monte-Carlo update

input Initial parameters $\boldsymbol{v}_1$, trajectory $s_1, \dots, s_T$, rewards $r_1, \dots, r_T$, visit counts n.
for $t = 1, \dots, T$ **do**
  $U_t = \sum_{t=1}^{T} r_t$.
  $n_t(s_t) = n_{t-1}(s_t) + 1$
  $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_{n_t(s_t)}(s_t)(U_t - \boldsymbol{v}_t(s_t))$ if $n_t(s_t) = 1$.
  $n_t(s) = n_{t-1}(s)$, $\boldsymbol{v}_t(s) = \boldsymbol{v}_{t-1}(s)$ otherwise
end for
return $\boldsymbol{v}_{T+1}$

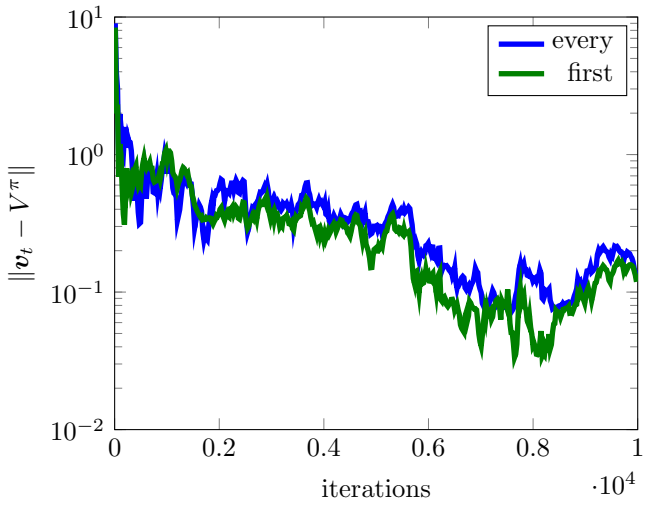


Figure 7.4: Error as the number of iterations n increases, for first and every visit Monte Carlo estimation.

7.2.3 Approximate policy iteration

A well-known algorithm for getting an optimal policy is policy iteration, Algorithm 7 in Section 6.5.4. This consists of estimating the value of a particular policy, and then trying to get an improved policy using this value. We can still apply the same principle for the case where we cannot exactly evaluate a policy. This is called approximate policy iteration. Unfortunately, approximate policy iteration does not necessarily converge without strong conditions on each approximation step.

Algorithm 15 Approximate policy iteration

1: **input** Initial parameters $\boldsymbol{v}_0$, inital Markov policy π_0, estimator f.
2: **for** $i = 1, \dots, N$ **do**
3: Get estimate $\boldsymbol{v}_i = f(\boldsymbol{v}_{i-1}, \pi_{i-1})$.
4: Calculate new policy $\pi_i = \arg\max_\pi \mathscr{L}\boldsymbol{v}_i$.
5: **end for**

7.2.4 Temporal difference methods

The main idea of temporal differences is to use partial samples of the utility and replace the remaining sample from time t with an estimate of the expected utility after time t. Since there maybe no particular reason to choose a specific t, frequently an exponential distribution t's is used.

Let us first look at the usual update when we have the complete utility sample U_k. The full stochastic update is of the form:

$$\boldsymbol{v}_{k+1}(s) = \boldsymbol{v}_k(s) + \alpha(U_k - \boldsymbol{v}_k(s)),$$

Using the *temporal difference error* $d(s_t, s_{t+1}) = \boldsymbol{v}(s_t) - [\boldsymbol{r}(s_t) + \gamma \boldsymbol{v}(s_{t+1})]$, we obtain the update:

$$\boldsymbol{v}_{k+1}(s) = \boldsymbol{v}_k(s) + \alpha \sum_t \gamma^t d_t, \qquad d_t \triangleq d(s_t, s_{t+1}) \tag{7.2.2}$$

Stochastic, incremental, update:

$$\boldsymbol{v}_{t+1}(s) = \boldsymbol{v}_t(s) + \alpha \gamma^t d_t. \tag{7.2.3}$$

We have now converted the full stochastic update into an incremental update that is nevertheless equivalent to the old update. Let us see how we can generalise this to the case where we have a mixture of temporal differences.

Temporal difference algorithm with eligibility traces.

TD(λ).
Recall the temporal difference update when the MDP is given in analytic form.

$$\boldsymbol{v}_{n+1}(i) = \boldsymbol{v}_n i + \boldsymbol{\tau}_n i, \quad \boldsymbol{\tau}_n(i) \triangleq \sum_{t=0}^{\infty} \mathbb{E}_{\pi_n, \mu} \left[(\gamma\lambda)^m d_n(s_t, s_{t+1}) \mid s_0 = i \right].$$

We can convert this to a stochastic update, which results in the well-known TD(λ) algorithm for policy evaluation.

$$\boldsymbol{v}_{n+1}(s_t) = \boldsymbol{v}_n(s_t) + \alpha \sum_{k=t}^{\infty} (\gamma\lambda)^{k-t} d_k. \tag{7.2.4}$$

Unfortunately, this algorithm is only possible to implement offline due to the fact that we are looking at future values.

This problem can be fixed by the backwards-looking Online TD(λ) algorithm. The main idea is to backpropagate changes in future states to previously encountered states. However, we wish to modify older states less than more recent states.

Algorithm 16 Online TD(λ)

input Initial parameters $\boldsymbol{v}_k$, trajectories (s_t, a_t, r_t)
$\boldsymbol{e}_0 = \mathbf{0}$.
for $t = 1, \ldots, T$ **do**
  $d_t \triangleq d(s_t, s_{t+1})$ `// temporal difference`
  $\boldsymbol{e}_t(s_t) = \boldsymbol{e}_{t-1}(s_t) + 1$ `// eligibility increase`
  for $s \in \mathcal{S}$ **do**
    $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_t \boldsymbol{e}_t(s) d_t$. `// update all eligible states`
  end for
  $\boldsymbol{e}_{t+1} = \lambda \boldsymbol{e}_t$
end for
return $\boldsymbol{v}_T$

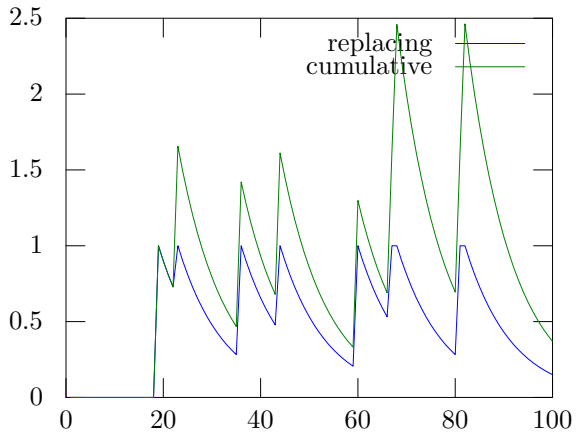


Figure 7.5: Eligibility traces, replacing and cumulative.

For replacing traces, use $\boldsymbol{e}_t(s_t) = \boldsymbol{e}_{t-1}(s_t) + 1$.

7.2.5 Stochastic value iteration methods

The main problem we had seen so far with Monte-Carlo based simulation is that we normally require a complete sequence of rewards before updating values. However, in value iteration, we can simply perform a backwards step from all the following states in order to obtain a utility estimate. This idea is explored in stochastic value iteration methods.

The standard value iteration algorithm performs a sweep over the complete state space at each iteration. However, could perform value iteration over an arbitrary sequence of states. For example, we can follow a sequence of states generated from a particular policy. This lends to the idea of *simulation-based* value iteration.

Such state sequences must satisfy various technical requirements. In particular, the policies that generate those state sequences must be *proper* for episodic problems. That is, that all policies should reach a terminating state with probability 1. For discounted non-episodic problems, this is easily achieved by using a geometric distribution for termination time. This ensures that all policies will be proper. Alternatively, of course, we could simply select starting states with an arbitrary schedule, as long as all states are visited infinitely often in the limit.

However, value iteration also requires the Markov decision process model. The question is whether it is possible to replace the MDP model with some arbitrary estimate. This estimate can itself be obtained via simulation. This leads to a whole new family of stochastic value iteration algorithms. The most important and well-known of these is Q-learning, which uses a trivial empirical MDP model.

Simulation-based value iteration

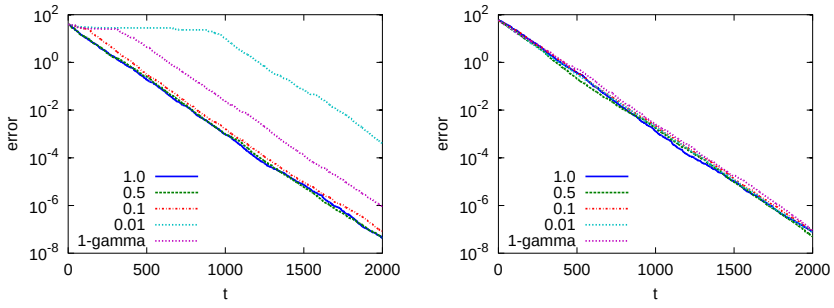
First, however, we shall discuss the extension of value iteration to the case where we obtain state data from simulation. This allows us to concentrate our estimates to the most useful states.

Algorithm 17 shows a generic simulation-based value iteration algorithm, with a uniform restart distribution $\mathcal{U}nif(\mathcal{S})$ and termination probability ϵ.

Algorithm 17 Simulation-based value iteration

Input μ, $\mathcal{S}$.
Initialise $s_t \in \mathcal{S}, \boldsymbol{v}_0 \in \mathcal{V}$.
for $t = 1, 2, \ldots$ **do**
  $s = s_t$.
  $\pi_t(s) = \arg\max_a r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}_\mu(s'|s,a) \boldsymbol{v}_{t-1}(s')$
  $\boldsymbol{v}_t(s) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}_\mu(s'|s,\pi_t(s)) \boldsymbol{v}_{t-1}(s')$
  $s_{t+1} \sim (1-\epsilon) \cdot \mathbb{P}(s_{t+1} \mid s_t = a, \pi_t, \mu) + \epsilon \cdot \mathcal{U}nif(\mathcal{S})$.
end for
Return π_n, V_n.

In the following figures, we can see the error in value function estimation in the chain task when using simulation-based value iteration. It is always a better idea to use an initial value $\boldsymbol{v}_0$ that is an upper bound on the optimal value function, if such a value is known. This is due to the fact that in that case, convergence is always guaranteed when using simulation-based value iteration, as long as the policy that we are using is proper.[2]



(a) Pessimistic (b) Optimistic

Figure 7.6: Simulation-based value iteration with pessimistic initial estimates ($\boldsymbol{v}_0 = 0$) and optimistic initial estimates ($\boldsymbol{v}_0 = 20 = 1/(1-\gamma)$), for varying ϵ. Errors indicate $\|\boldsymbol{v}_t - V^*\|_1$.

As can be seen in Figure 7.6, the value function estimation error of simulation-based value iteration is highly dependent upon the initial value function estimate $\boldsymbol{v}_0$ and the exploration parameter ϵ. It is interesting to see uniform sweeps ($\epsilon = 1$) result in the lowest estimation error in terms of the value function L_1 norm.

Q-learning

Simulation-based value iteration can be suitably modified for the actual reinforcement learning problem. Instead of relying on a model of the environment,

[2] In the case of discounted non-episodic problems, this amounts to a geometric stopping time distribution, after which the state is drawn from the initial state distribution.

we replace arbitrary random sweeps of the state-space with the actual state sequence observed in the real environment. We also use this sequence as a simple way to estimate the transition probabilities.

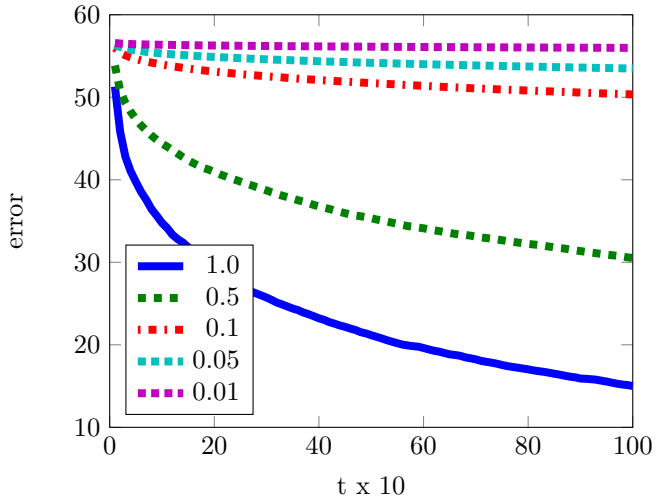
Algorithm 18 Q-learning

Input μ, $\mathcal{S}, \epsilon_t, \alpha_t$.
Initialise $s_t \in \mathcal{S}, \boldsymbol{q}_0 \in \mathcal{V}$.
for $t = 1, 2, \dots$ **do**
  $s = s_t$.
  $a_t \sim \hat{\pi}^*_{\epsilon_t}(a \mid s_t, \boldsymbol{q}_t)$
  $s_{t+1} \sim \mathbb{P}(s_{t+1} \mid s_t = s, a_t, \pi_t, \mu)$.
  $\boldsymbol{q}_{t+1}(s_t, a_t) = (1 - \alpha_t)\boldsymbol{q}_t(s_t, a_t) + \alpha_t[r(s_t) + \boldsymbol{v}_t(s_{t+1})]$, where $\boldsymbol{v}_t(s) = \max_{a \in \mathcal{A}} \boldsymbol{q}_t(s, a)$.
end for
Return π_n, V_n.

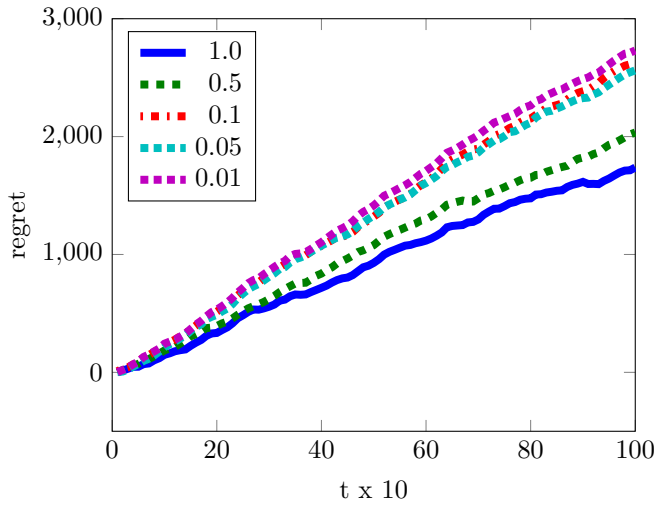
The result is Q-learning (Algorithm 18), one of the most well-known and simplest algorithms in reinforcement learning. In light of the previous theory, it can be seen as a stochastic value iteration algorithm, where at every step t, given the partial observation (s_t, a_t, s_{t+1}) you have an approximate transition model for the MDP which is as follows:

$$P(s'|s_t, a_t) = \begin{cases} 1, & \text{if } s_{t+1} = s' \\ 0, & \text{if } s_{t+1} \neq s'. \end{cases} \tag{7.2.5}$$

Even though this model is very simplistic, it still seems to work relatively well in practice, and the algorithm is simple to implement. In addition, since we cannot arbitrarily select states in the real environment, we replace the state-exploring parameter ϵ with a time-dependent exploration parameter ϵ_t for the policy we employ on the real environment.



(a) Error



(b) Regret

Figure 7.7: Q-learning with $\boldsymbol{v}_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

Figure 7.7 shows the performance of the basic Q-learning algorithm for the Chain task, in terms of value function error and regret. In this particular implementation, we used a polynomially decreasing exploration parameter ϵ_t and step size α_t. Both of these depend on the number of visits to a particular state and so perform more efficient Q-learning.

Of course, one could get any algorithm in between pure Q-learning and pure stochastic value iteration. In fact, variants ofthe Q-learning algorithm using eligibility traces (see Section 7.2.4) can be formulated in this way.

Generalised stochastic value iteration Finally, we can generalise the above ideas to the following algorithm. This is an online algorithm, which can be

applied directly to a reinforcement learning problem and it includes simulation-based value iteration and Q-learning as special cases. There are three parameters associated with this algorithm. The first is ϵ_t, the exploration amount performed by the policy we follow. The second is α_t, the step size parameter. The third one is σ_t, the state-action distribution. The final parameter is the MDP estimator $\widehat{\mu}_t$. This includes both an estimate of the transition probabilities $\mathbb{P}_{\widehat{\mu}_t}(s' \mid s, a)$ and of the expected reward $r_{\widehat{\mu}_t}(s, a)$.

Algorithm 19 Generalised stochastic value iteration

Input $\widehat{\mu_0}$, $\mathcal{S}, \epsilon_t, \alpha_t$.
Initialise $s_1 \in \mathcal{S}, \boldsymbol{q}_1 \in \mathcal{Q}, \boldsymbol{v}_0 \in \mathcal{V}$.
for $t = 1, 2, \ldots$ **do**
  $a_t \sim f(\hat{\pi}^*_{\epsilon_t}(a \mid s_t, \boldsymbol{q}_t))$
  Observe s_{t+1}, r_{t+1}.
  $\widehat{\mu}_t = \hat{\mu}_{t-1} \mid s_t, a_t, s_{t+1}, r_{t+1}$. `// update MDP estimate.`
  for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**
    With probability $\sigma_t(s, a)$ do:

$$\boldsymbol{q}_{t+1}(s, a) = (1 - \alpha_t)\boldsymbol{q}_t(s, a) + \alpha_t \left[r_{\widehat{\mu}_t}(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}_{\widehat{\mu}_t}(s' \mid s, a) \boldsymbol{v}_t(s') \right].$$

    otherwise $\boldsymbol{q}_{t+1}(s, a) = \boldsymbol{q}_t(s, a)$.
    $\boldsymbol{v}_{t+1}(s) = \max_{a \in \mathcal{A}} \boldsymbol{q}_{t+1}(s, a)$,
  end for
end for
Return π_n, V_n.

It is instructive to examine special cases for these parameters. For the case when $\sigma_t = 1$, $\alpha_t = 1$, and when $\widehat{\mu}_t = \mu$, we obtain standard value iteration.

For the case when $\sigma_t(s, a) = \mathbb{I}\{s_t = s \wedge a_t = a\}$ and

$$\mathbb{P}_{\widehat{\mu}_t}(s_{t+1} = s' \mid s_t = s, a_t = a) = \mathbb{I}\{s_{t+1} = s' \mid s_t = s, a_t = a\},$$

it is easy to see that we obtain Q-learning.

Finally, if we set $\sigma_t(s, a) = \boldsymbol{e}_t(s, a)$, then we obtain a stochastic eligibility-trace Q-learning algorithm similar to $Q(\lambda)$.

7.3 Discussion

Most of these algorithms are quite simple, and so clearly demonstrate the principle of learning by reinforcement. However, they do not aim to solve the reinforcement learning problem optimally. They have been mostly of use for finding near-optimal policies given access to samples from a simulator, as used for example to learn to play Atari games Mnih et al. [2015]. However, even in this case, a crucial issue is how much data is needed in the first place to approach optimal play. The second issue is using such methods for online reinforcement learning, i.e. in order to maximise expected utility while still learning.

Convergence. Even though it is quite simple, the convergence of Q-learning has been established in various settings. Tsitsiklis [1994] has provided an asymptotic proof based on stochastic approximation theory with less restrictive assumptions than the original paper Watkins and Dayan [1992]. Later Kearns and Singh [1999] proved finite sample convergence results under strong mixing assumptions on the MDP.

Q-learning can be seen as using a very specific type of approximate transition model. By modifying this, we can obtain more efficient algorithms, such as delayed Q-learningStrehl et al. [2006], which needs $\tilde{O}(|\mathcal{S}||\mathcal{A}|)$ samples to find an ϵ-optimal policy with high probability.

Exploration. In order to perform exploration efficiently, Q-learning does not attempt to perform optimal exploration. Another extension of Q-learning is using a population value function estimates. This was introduced in Dimitrakakis [2006b,a] through the use of random initial values and weighted bootstrapping and evaluated for bandit tasks. Recently, this idea has also been exploited in the context of deep neural networks by Osband et al. [2016] representations of value functions for the case of full reinforcement learning. We will examine this more closely in Chapter 8.

Bootstrapping and subsampling (App. B.6) use a single set of empirical data to obtain an empirical measure of uncertainty about statistics of the data. We wish to do the same thing for value functions, based on data from a one or more trajectories. Informally, this variant maintains a collection of Q-value estimates, each one of which is trained on different segments[3] of the data, with possible overlaps. In order to achieve efficient exploration, a random Q estimate is selected at every episode, or every few steps. This results in a bootstrap analogue of Thompson sampling. Figure 7.8 shows the use of weighted bootstrap estimates for the Double Chain problem introduced by Dearden et al. [1998]. Bootstrapping and subsampling (App. B.6) use a single

[3]If not dealing with bandit problems, it is important to do this with trajectories.

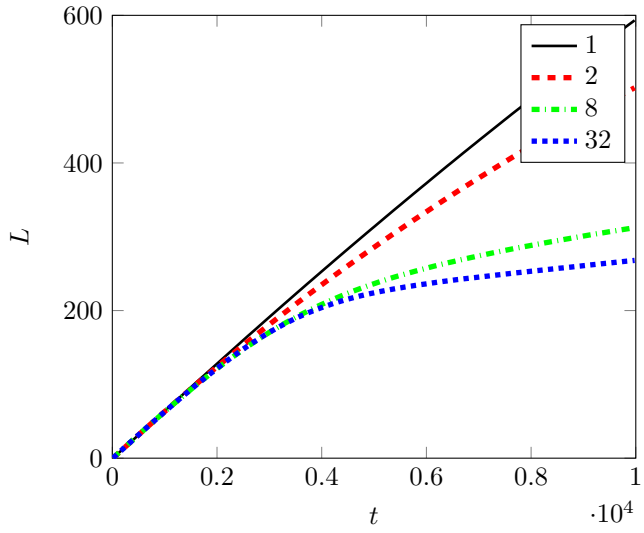


Figure 7.8: Cumulative regret of weighted Boostrap Q-learning for various amount of bootstrap replicates (1 is equivalent to plain Q-learning). Generally speaking, an increased amount of replicates leads to improved exploration performance.

7.4 Exercises

Exercise 32 (180). This is a continuation of exercise 27. Create a reinforcement learning version of the diagnostic model from exercise 27. In comparison to that exercise, here the doctor is allowed to take zero, one, or two diagnostic actions.

View the treatment of each patient as a single episode and design an appropriate state and action space to apply the standard MDP framework: note that all episodes run for at least 2 steps, and there is a different set of actions available at each state: the initial state only has diagnostic actions, while any treatment action terminates the episode and returns us the result.

1. Define the state and action space for each state.
2. Create a simulation of this problem, according to the probabilities mentioned in Exercise 27.
3. Apply a simulation-based algorithm such as Q-learning to this problem. How much times does it take to perform well? Can you improve it so as to take into account the problem structure?

Exercise 33. It is well-known that the value function of a policy π for an MDP μ with state reward function $\boldsymbol{r}$ can be written as the solution of a linear equation $V_\mu^\pi = (I - \gamma P_\mu^\pi)^{-1}\boldsymbol{r}$, where the term $\Phi_\mu^\pi \triangleq (I - \gamma P_\mu^\pi)^{-1}$ can be seen as a feature matrix. However, Sarsa and other simulation-based algorithms only approximate the value function directly rather than Φ_μ^π. This means that, if the reward function changes, they have to be restarted from scratch. Is there a way to rectify this?[4]

3h Develop and test a simulation-based algorithm (such as Sarsa) for estimating Φ_μ^π, and prove its asymptotic convergence. *Hint: focus on the fact that you'd like to estimate a value function for all possible reward functions.*

? Consider a model-based approach, where we build an empirical transition kernel P_μ^π. How good are our value function estimates in the first versus the second approach? Why would you expect either one to be better?

? Can the same idea be extended to Q-learning?

[4]This exercise stems from a discussion with Peter Auer in 2012 about this problem.

Chapter 8

Approximate representations

8.1 Introduction

In this chapter, we consider methods and representations for approximating value functions, policies, or transition kernel can only be represented by an approximation. This is the case when the state or policy space are large, so that one has to use some parameterisation that may not include the true value function, policy, or transition kernel. In general, we shall assume the existence of either some approximate value function space $\mathcal{V}_\Theta$ or some approximate policy space Π_Θ, which are the set of allowed value functions and policies, respectively. For the purposes of this chapter, we will assume that we have access to some simulator or approximate model of the transition probabilities, wherever necessary. Model-based reinforcement learning where the transition probabilities are explicitly estimated will be examined in the next two chapters.

As an introduction, let us start with the case where we have a value function space $\mathcal{V}$ and some value function $\boldsymbol{u} \in \mathcal{V}$ that is our best approximation of the optimal value function. Then we can define the greedy policy with respect to $\boldsymbol{u}$ as follows:

Definition 8.1.1 ($\boldsymbol{u}$-greedy policy and value function)**.**

$$\pi^*_{\boldsymbol{u}} \in \arg\max_{\pi \in \Pi} \mathscr{L}_\pi \boldsymbol{u}, \qquad \boldsymbol{v}^*_{\boldsymbol{u}} = \mathscr{L} \boldsymbol{u},$$

where $\pi : \mathcal{S} \to \mathbb{\Delta}(\mathcal{A})$ maps from states to action distributions.

Although the greedy policies need not be stochastic, here we are explicitly considering stochastic policies, because this sometimes facilitates finding a good approximation. If $\boldsymbol{u}$ is the optimal value function V^*, then the greedy policy is going to be optimal.

More generally, when we are trying to approximate a value function, we usually are constrained to look for it in a parameterised set of value functions $\mathcal{V}_\Theta$, where Θ is the parameter space. Hence, it might be the case that the optimal value function may not lie within $\mathcal{V}_\Theta$. Similarly, the policies that we can use lie in a space Π_Θ, which may not include the greedy policy itself. This is usually because it is not possible to represent all possible value functions and policies in complex problems.

Usually, we are not aiming at a *uniformly good* approximation to a value function or policy. Instead, we define ϕ, a distribution on $\mathcal{S}$, which specifies on which parts of the state space we want to have a good approximation, by placing higher weight on the most important states. Frequently, ϕ only has a *finite* support, meaning that we only measure the approximation error over a finite set of states. In the sequel, we shall always define the quality of an approximate value or policy with respect to ϕ.

In the remainder of this chapter, we shall examine a number of approximate dynamic programming algorithms. What all of these algorithms have in common is the requirement to calculate an approximate value function or policy. The two next sections given an overview of the basic problem of fitting an approximate value function or policy to a target.

8.1.1 Fitting a value function

Let us begin by considering the problem of finding the value function $\boldsymbol{v}_\theta \in \mathcal{V}_\Theta$ that best matches a target value function $\boldsymbol{u}$. This can be done by minimising

the difference between the target value $\boldsymbol{u}$ and the approximation v_θ:

$$\|\boldsymbol{v}_\theta - \boldsymbol{u}\|_\phi = \int_{\mathcal{S}} |\boldsymbol{v}_\theta(s) - \boldsymbol{u}(s)| \, \mathrm{d}\phi(s), \tag{8.1.1}$$

with respect to some measure ϕ on $\mathcal{S}$. If $\boldsymbol{u} = V^*$, i.e. the optimal value function, then we end up getting the best possible value function with respect to the distribution ϕ. We can formalise the idea for fitting an approximate value function to a target below:

Approximate value function fit

$$\mathcal{V}_\Theta = \{\boldsymbol{v}_\theta \mid \theta \in \Theta\}, \qquad \theta^* \in \arg\min_{\theta \in \Theta} \|\boldsymbol{v}_\theta - \boldsymbol{u}\|_\phi$$

where $\|\cdot\|_\phi \triangleq \int_{\mathcal{S}} |\cdot| \, \mathrm{d}\phi$.

Unfortunately, this minimisation problem can be difficult to solve in general. A particularly simple case is when the set of approximate functions is small enough for the minimisation to be performed via enumeration.

Example 36 (Fitting a finite number of value functions). Consider a finite space of value functions $\mathcal{V}_\Theta = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}$, which wish to fit to a target value function $\boldsymbol{u}$. In this particular scenario, $\boldsymbol{v}_1(x) = \sin(0.1x)$, $\boldsymbol{v}_2(x) = \sin(0.5x)$, $\boldsymbol{v}_3(x) = \sin(x)$, while

$$\boldsymbol{u}(x) = 0.5 \sin(0.1x) + 0.3 \sin(0.1x) + 0.1 \sin(x) + 0.1 \sin(10x).$$

Clearly, none of the given functions is a perfect fit. In addition, finding the best overall fit requires minimising an integral. So, for this problem we choose a random set of points $X = \{x_t\}$ on which to evaluate the fit, with $\phi(x_t) = 1$ for every point $x_t \in X$. This is illustrated in Figure 8.1, which shows the error of the functions at the selected points, as well as their cumulative error.

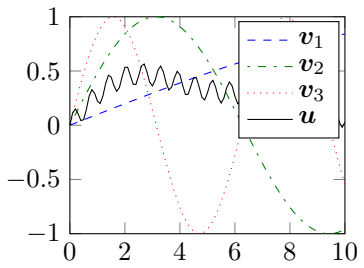
In the example above, the approximation space $\mathcal{V}_\Theta$ does not have a member that is sufficiently close to the target value function. It could be that a larger function space contains a better approximation. However, it may be difficult to find the best fit in an arbitrary set $\mathcal{V}_\Theta$.

8.1.2 Fitting a policy

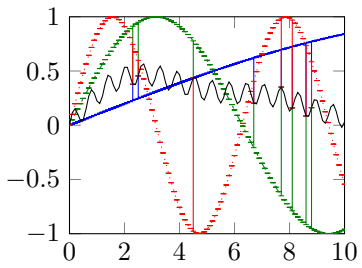
The problem of fitting a policy is not significantly different from that of fitting a value function, especially when the action space is continuous. Once more, we define an appropriate normed vector space so that it makes sense to talk about the normed difference between policies with respect to some measure ϕ on the states. In particular, we use the following error between two policies π, π':

$$\|\pi - \pi'\|_\phi = \int_{\mathcal{S}} \|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\| \, \mathrm{d}\phi(s),$$

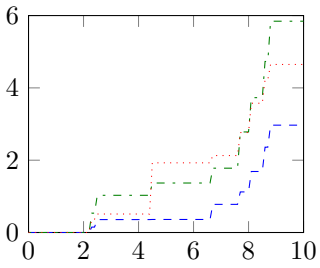
where the norm within the integral is usually the L_1 norm. For a finite action space, this corresponds to $\|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\| = \sum_{a \in \mathcal{A}} |\pi(a \mid s) - \pi'(a \mid s)|$, but



(a) The target function and the three candidates.



(b) The errors at the chosen points.



(c) The total error of each candidate.

Figure 8.1: Fitting a value function in $\mathcal{V}_\Theta = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}$ to a target value function $\boldsymbol{u}$, over a finite number of points. While none of the three candidates is a perfect fit, we clearly see that $\boldsymbol{v}_1$ has the lowest cumulative error over the measured set of points.

certainly other norms may be used and are sometimes more convenient. The optimisation problem corresponding to fitting an approximate policy from a set of policies Π_Θ to a target policy π is shown below.

The policy approximation problem

$$\Pi_\Theta = \{\pi_\theta \mid \theta \in \Theta\}, \qquad \theta^* \in \arg\min_{\theta \in \Theta} \|\pi_\theta - \pi^*_{\boldsymbol{u}}\|_\phi$$

where $\pi^*_{\boldsymbol{u}} = \arg\max_{\pi \in \Pi} \mathscr{L}_\pi \boldsymbol{u}$.

Once more, the minimisation problem may not be trivial, but there are some cases where it is particularly easy. One of these is when the policies can be efficiently enumerated, as in the example below.

Example 37 (Fitting a finite space of policies). For simplicity, consider the space of deterministic policies, with a binary action space $\mathcal{A} = \{0, 1\}$. Then each policy can be represented as a simple mapping $\pi : \mathcal{S} \to \{0, 1\}$, corresponding to a binary partition of the state space. In this example, the state space is the 2-dimensional unit cube, $\mathcal{S} = [0, 1]^2$. Figure 8.2 shows an example policy, where the light red and light green areas represent it taking action 1 and 0 respectively. The measure ϕ has support only on the crosses and circles, which indicate the action taken at that location. Consider a

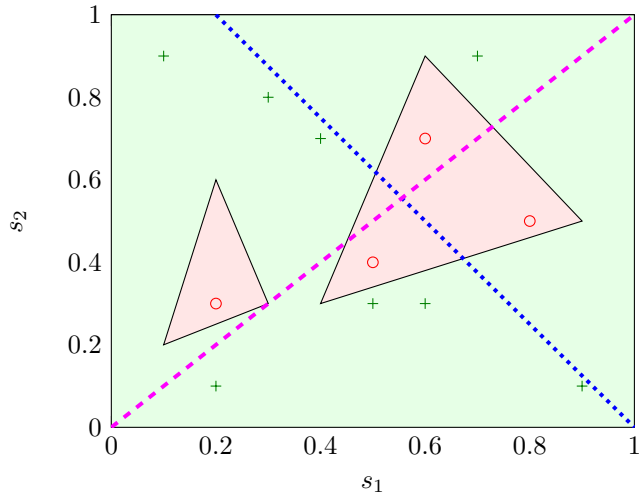


Figure 8.2: An example policy. The red areas indicate taking action 1 been taken, and the green areas action 0. The ϕ measure has finite support, indicated by the crosses and circles. The blue and magenta lines indicate two possible policies that separate the state space with a hyperplane.

policy space Π_Θ consisting of just four policies. Each set of two policies is indicated by the magenta (dashed) and blue (dotted) lines in Figure 8.2. Each line corresponds to two possible policies, one selecting action 1 in the high region, and the other selecting action 0 instead. In terms of our error metric, the best policy is the one that makes the fewest mistakes. Consequently, the best policy in this set to use the blue line and play action 1 (red) in the top-right region.

8.1.3 Features

Frequently, when dealing with large, or complicated spaces, it pays off to project the state and actions onto a feature space $\mathcal{X}$. In that way, we can make problems much more manageable. Generally speaking, a feature mapping is defined as follows.

Feature mapping $f : \mathcal{S} \times \mathcal{A} \to \mathcal{X}$**.**
For $\mathcal{X} \subset \mathbb{R}^n$, the feature mapping can be written in vector form:

$$f(s,a) = \begin{bmatrix} f_1(s,a) \\ \dots \\ f_n(s,a) \end{bmatrix}$$

We can define feature mappings $f(s)$ for states only in a similar manner.

What sort of functions should we use? A common idea is to use a set of smooth symmetric functions,such as usual radial basis functions.

EXAMPLE 38 (Radial Basis Functions). Let d be a metric on $\mathcal{S} \times \mathcal{A}$ and define the set

of centroids $\{(s_i, a_i) \mid i = 1, \dots, n\}$. Then we define each element of f as:

$$f_i(s, a) \triangleq \exp\{-d[(s, a), (s_i, a_i)]\}.$$

These functions are sometimes called *kernels*. A one-dimensional example of Gaussian radial basis functions is shown in Figure 8.3.

Another common type of functions are binary functions. These effectively discretise a continuous space through either a cover or a partition.

Definition 8.1.2. The collection of sets $\mathscr{G}$ is a *cover* of X iff $\bigcup_{S \in \mathscr{G}} S \supset X$.

Definition 8.1.3. The collection of sets $\mathscr{G}$ is a *partition* of X iff

1. If $S \neq R \in \mathscr{G}$ then $S \cap R = \emptyset$.
2. $\bigcup_{S \in \mathscr{G}} S = X$.

In reinforcement learning, these types of feature functions corresponding to partitions are usually referred to as tilings.

Example 39 (Tilings). Let $\mathscr{G} = \{X_1, \dots, X_n\}$ be a *partition* of $\mathcal{S} \times \mathcal{A}$. Then the f_i can be defined by

$$f_i(s, a) \triangleq \mathbb{I}\{(s, a) \in X_i\}. \tag{8.1.2}$$

Multiple tilings create a cover and can be used without many difficulties with most discrete reinforcement learning algorithms.Sutton and Barto [cf 1998]

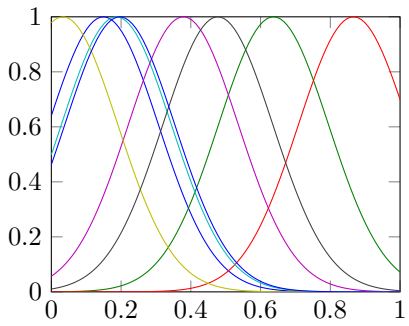


Figure 8.3: Radial Basis Functions

8.1.4 Estimation building blocks

Now that we have looked at the basic problems in approximate regimes, let us look at some methods for obtaining useful approximations. First of all, we introduce some basic concepts look-ahead and rollout policies for estimating value functions. Then we formulate value function approximation and policy estimation as an optimisation problem. These are going to be used in the remaining sections, and in particularly in Section 8.2 introduces the well known approximate policy iteration algorithm, which combining those two steps into approximate policy evaluation and approximate policy improvement.

Look-ahead policies

Given an approximate value function $\boldsymbol{u}$, the transition model P_μ of the MDP and the expected rewards r_μ we can always find the improving policy given in Def. 8.1.1 via the following single-step look-ahead.

Single-step look-ahead

Let $q(i,a) \triangleq r_\mu(i,a) + \gamma \sum_{j \in \mathcal{S}} P_\mu(j \mid i,a)\boldsymbol{u}(j)$ and

$$\pi_{\boldsymbol{q}}(a \mid i) > 0 \qquad \text{iff } a \in \arg\max_{a' \in \mathcal{A}} q(i,a').$$

We are however not necessarily limited to the first-step. By looking T steps forward into the future we can improve both our value function and policy estimates.

T-step look-ahead

Define $\boldsymbol{u}_k$ recursively as:

$$q_k(i,a) = r_\mu(i,a) + \gamma \sum_{j \in \mathcal{S}} P_\mu(j \mid i,a)\boldsymbol{u}_{k-1}(j)$$
$$\boldsymbol{u}_k(i) = \max\left\{q_k(i,a) \mid a \in \mathcal{A}\right\},$$

with $\boldsymbol{u}_0 = \boldsymbol{u}$.

$$\pi_{\boldsymbol{q}_t}(a \mid i) = \arg\max_{a \in \mathcal{A}} \boldsymbol{q}_T(i,a)$$

In fact, taking $\boldsymbol{u} = \boldsymbol{0}$, this recursion is identical to solving the k-horizon problem and in the limit we obtain solution to the original problem. In the general case, our value function estimation error is bounded by $\gamma^k \left\|\boldsymbol{u} - V^*\right\|$.

Rollout policies

As we have seen in Section 6.4.2 one way to obtain an approximate value function of an arbitrary policy π is to use Monte Carlo estimation, that is, to simulate several sequences of state-action-reward tuples by running the policy on the MDP. More specifically, we have the following rollout estimate.

Rollout estimate of the q-factor

In particular, from each state i, we take K_i rollouts to estimate:

$$q(i,a) = \frac{1}{K_i} \sum_{k=1}^{K_i} \sum_{t=0}^{T_k - 1} r(s_{t,k}, a_{t,k}), \tag{8.1.3}$$

where $s_{t,k}, a_{t,k} \sim \mathbb{P}_\mu^\pi(\cdot \mid s_0 = i, a_0 = a)$, and $T_k \sim \mathit{Geom}(1-\gamma)$.

This results in a set of samples of $\boldsymbol{q}$-factors. The next problem is to find a parametric policy π_θ that approximates the greedy policy with respect to our

samples, $\pi^*_{\boldsymbol{q}}$. For a finite number of actions, this can be seen as a classification problem [Lagoudakis and Parr, 2003a]. For continuous actions, it becomes a regression problem. Once more, we define a distribution ϕ on the states, over which we wish to perform the minimisation.

Rollout policy estimation
Given some distribution ϕ on $\hat{S}$ and a set of samples $q(i, a)$, giving us the greedy policy

$$\pi^*_{\boldsymbol{q}}(a \mid i) = \arg\max q(i \mid a)$$

and a parametrised policy space $\{\pi_\theta \mid \theta \in \Theta\}$, we estimate

$$\min_{\boldsymbol{\theta}} \left\| \pi_{\boldsymbol{\theta}} - \pi^*_{\boldsymbol{q}} \right\|_\phi$$

8.1.5 The value estimation step

We can now attempt to fit a parametric approximation to a given state or state-action value function. This is often better than simply maintaining a set of rollout estimates from individual states pairs (or state-action pairs), as it might enable us to generalise over the complete state space. The simplest way to do so is via a generalised linear model. A natural parameterisation for the value function is to use a generalised linear model on a set of features. Then the value function is a linear function of the features with parameters $\boldsymbol{\theta}$. More precisely, we can define the following model for the case where we have a feature mapping on states.

Generalised linear model using state features (or kernel)
Given a feature mapping $f : \mathcal{S} \to \mathbb{R}^n$ and parameters $\boldsymbol{\theta} \in \mathbb{R}^n$, compute the approximation

$$\boldsymbol{v}_{\boldsymbol{\theta}}(s) = \sum_{i=1}^{n} \theta_i f_i(s). \tag{8.1.4}$$

Choosing the model representation is only the first step. We now have to use it to represent a specific value function. In order to do this, we first pick a set of *representative states* $\hat{S}$ to fit our value function $\boldsymbol{v}_{\boldsymbol{\theta}}$ to $\boldsymbol{v}$. This type of estimation can be seen as a regression problem, where the observations are value function measurements at different states.

Fitting a value function to a target.
Let ϕ be a distribution over representative states $\hat{S}$. For some constants $\kappa, p > 0$, we define the weighted prediction error per state as.

$$c_s(\boldsymbol{\theta}) = \phi(s) \left\| \boldsymbol{v}_{\boldsymbol{\theta}}(s) - \boldsymbol{v}(s) \right\|_p^\kappa .$$

where the total prediction error is $c(\boldsymbol{\theta}) = \sum_{s \in \hat{S}} c_s(\boldsymbol{\theta})$.

Minimising this error can be done with gradient descent, which is a general algorithm for finding local minima of smooth cost functions. Generally, minimising a real-valued cost function $c(\boldsymbol{\theta})$ with gradient descent involves an algorithm iteratively approximating the value minimising c:

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} + \alpha_n \nabla c(\boldsymbol{\theta}^{(n)}).$$

Under certain conditions[1] on the step-size parameter α_n, $\lim_{n\to\infty} c(\boldsymbol{\theta}^{(n)}) = \min_{\boldsymbol{\theta}} c(\boldsymbol{\theta})$.

EXAMPLE 40 (Gradient descent for $p = 2$, $\kappa = 2$). In this case the square root and κ cancel out and we obtain

$$\nabla_{\boldsymbol{\theta}} c_s = \phi(s) \sum_{a\in\mathcal{A}} \nabla_{\boldsymbol{\theta}} [\boldsymbol{v}_{\boldsymbol{\theta}}(s) - \boldsymbol{v}(s)]^2 = 2[\boldsymbol{v}_{\boldsymbol{\theta}}(s) - \boldsymbol{v}(s)] \nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}},$$

where $\nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}}(s) = f(s)$. Taking partial derivatives $\partial/\partial\theta_j$, leads to the update rule:

$$\theta_j^{(n+1)} = \theta_j^{(n)} - 2\alpha\phi(s)[\boldsymbol{v}_{\boldsymbol{\theta}^{(n)}}(s) - \boldsymbol{v}(s)] f_j(s). \tag{8.1.5}$$

However, the value function is not necessarily self-consistent, we do not have the identity $\boldsymbol{v}_{\boldsymbol{\theta}}(s) = \boldsymbol{r}(s) + \int_{\mathcal{S}} \boldsymbol{v}_\theta(s') \, \mathrm{d}P(s' \mid s, a)$. For that reason, we can instead choose a parameter that tries to make the parameterised value function self-consistent by minimising the Bellman error.

Minimising the Bellman error.

$$\inf_{\boldsymbol{\theta}} \left\| r(s,a) + \gamma \int_{\mathcal{S}} \boldsymbol{v}_{\boldsymbol{\theta}}(s') \, \mathrm{d}\hat{P}(s' \mid s, a) - \boldsymbol{v}_{\boldsymbol{\theta}}(s) \right\|_{\phi}. \tag{8.1.6}$$

Here $\hat{P}$ is not necessarily the true transition kernel. It can be a model or an empirical approximation (in which case the integral would only be over the empirical support). The summation itself is performed with respect to the measure ϕ.

In this chapter, we will look at two methods for approximately minimising the Bellman error. The first, least square policy iteration is a batch algorithm for approximate policy iteration and finds the least-squares solution to the problem using the empirical transition kernel. The second is a gradient based method, which is flexible enough to use either an explicit model of the MDP or the empirical transition kernel.

8.1.6 Policy estimation

A natural parameterisation for policies is to use a generalised linear model on a set of features. Then a policy can be described as a linear function of the features with parameters $\boldsymbol{\theta}$, together with an appropriate link function. More precisely, we can define the following model.

[1] See also Sec. 7.1.1.

Generalised linear model using features (or kernel).
Given feature mapping $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^n$, parameters $\boldsymbol{\theta} \in \mathbb{R}^n$, link function $\ell : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^+$

$$\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{g_{\boldsymbol{\theta}}(s,a)}{h_{\boldsymbol{\theta}}(s)}, \qquad g_{\boldsymbol{\theta}}(s,a) = \ell\left(\sum_{i=1}^{n} \theta_i f_i(s,a)\right) \quad h_{\boldsymbol{\theta}}(s) = \sum_{b \in \mathcal{A}} g_{\boldsymbol{\theta}}(s,b) \tag{8.1.7}$$

The link function ℓ ensure that the denominator is positive, an do the policy is a distribution over actions. An alternative method would be to directly constrain the policy parameters so the result is always a distribution, but this would result in a constrained optimisation problem. A typical choice for the link function are is $\ell(x) = \exp(x)$, which results in the softmax family of policies.

In order to fit a policy, we first pick a set of representative states $\hat{S}$ and then find a $\pi_{\boldsymbol{\theta}}$ that approximates a target policy π, which is typically the greedy policy with respect to some value function. In order to do so, we can define an appropriate cost function and then estimate the optimal parameters via some arbitrary optimisation method.

Fitting a policy through a cost function.

$$c(\boldsymbol{\theta}) = \sum_{s \in \hat{S}} c_s(\boldsymbol{\theta}), \qquad c_s(\boldsymbol{\theta}) = \phi(s) \left\| \pi_{\boldsymbol{\theta}}(\cdot \mid s) - \pi(\cdot \mid s) \right\|_p^{\kappa}. \tag{8.1.8}$$

Once more, we can use gradient descent to minimise the cost function. We obtain different results for different norms, but there are three cases of main interest: $p = 1, p = 2, p \to \infty$. We present the first one here, and leave the others as an exercise.

Example 41 (The case $p = 1$, $\kappa = 1$). The derivative can be written as:

$$\nabla_{\boldsymbol{\theta}} c_s = \phi(s) \sum_{a \in \mathcal{A}} \nabla_{\boldsymbol{\theta}} |\pi_{\boldsymbol{\theta}}(a \mid s) - \pi(a \mid s)|,$$
$$\nabla_{\boldsymbol{\theta}} |\pi_{\boldsymbol{\theta}}(a \mid s) - \pi(a \mid s)| = \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a \mid s) \operatorname{sgn}[\pi_{\boldsymbol{\theta}}(a \mid s) - \pi(a \mid s)]$$

The policy derivative in turn is

$$\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{h(s) \nabla_{\boldsymbol{\theta}} g(s,a) - \nabla_{\boldsymbol{\theta}} h(s) g(s,a)}{h(s)^2},$$

with $\nabla_{\boldsymbol{\theta}} h(s) = \left(\sum_{b \in \mathcal{A}} f_i(s,b)\right)_i$ and $\nabla_{\boldsymbol{\theta}} g(s,a) = f(s,a)$. Taking partial derivatives $\partial / \partial \theta_j$, leads to the update rule:

$$\theta_j^{(n+1)} = \theta_j^{(n)} - \alpha_n \phi(s) \left(\pi_{\boldsymbol{\theta}^{(n)}}(a \mid s) \sum_{b \in \mathcal{A}} f_j(s,b) - f_j(s,a) \right).$$

Alternative cost functions. It is often a good idea to add a *penalty term* to the cost function, via a penalty term of the form $\|\boldsymbol{\theta}\|^q$, constraining the parameters to be small,. The purpose of this is to prevent overfitting of the parameters to a small number of observations.

8.2 Approximate policy iteration (API)

The main idea of approximate policy iteration is to replace the exact Bellman operator $\mathscr{L}$ with an approximate version $\hat{\mathscr{L}}$ and the exact value of the policy with an approximate version.

Just as in standard policy iteration, there is a policy improvement step and a policy evaluation step. In the policy improvement step, we simply try to get as close as possible to the best possible improvement, in a restricted set of policies, using an approximate operator. Similarly, in the policy evaluation step, we try to get as close as possible to the actual value of the improved policy, as shown in Algorithm 46.

Algorithm 20 Generic approximate policy iteration algorithm

input Initial value function approximation $\boldsymbol{v}_0$, approximate Bellman operator $\hat{\mathscr{L}}$, approximate value estimator $\hat{V}$, policy space $\hat{\Pi}$, value function space $\hat{\mathcal{V}}$, norms $\|\cdot\|_\phi$, and $\|\cdot\|_\psi$,
for $k = 1, \dots$ **do**
  $\pi_k = \arg\min_{\pi \in \hat{\Pi}} \left\| \hat{\mathscr{L}}_\pi \boldsymbol{v}_{k-1} - \mathscr{L} \boldsymbol{v}_{k-1} \right\|_\phi$ `// policy improvement`
  $\boldsymbol{v}_k = \arg\min_{\boldsymbol{v} \in \hat{\mathcal{V}}} \|\boldsymbol{v} - \hat{V}_\mu^{\pi_k}\|_\psi$ `// policy evaluation`
end for

More precisely, at the k-th iteration, we use the approximate value $\boldsymbol{v}_{k-1}$ of the previous policy, π_{k-1}, to obtain an improved policy π_k. However, we may not be able to implement the policy $\arg\max_\pi \mathscr{L}_\pi \boldsymbol{v}_{k-1}$ for two reasons. Firstly, our policy space may not include all possible policies, due to the policy parameterisation. Secondly, the Bellman operator we have available may only be approximate. We'd like to find the value function $\boldsymbol{v}$ that is the closest to the true value function of policy π_k for the MDP μ. However, even if our value function space is rich enough to do that, the minimisation is done over a norm that integrates over a finite subset of the state space. The following section discusses the effect of those errors on the convergence of approximate policy iteration.

8.2.1 Error bounds for approximate value functions

If the approximate value function $\boldsymbol{u}$ is close to V^* then the greedy policy with respect to $\boldsymbol{u}$ is close to optimal. For a finite state and action space, the following holds.

Theorem 8.2.1. *Consider a finite MDP μ with discount factor $\gamma < 1$ and a vector $\boldsymbol{u} \in \mathcal{V}$ such that $\left\|\boldsymbol{u} - V_\mu^*\right\|_\infty = \epsilon$. If π is the $\boldsymbol{u}$-greedy policy then*

$$\left\|V_\mu^\pi - V_\mu^*\right\|_\infty \leq \frac{2\gamma\epsilon}{1-\gamma}.$$

In addition, $\exists \epsilon_0 > 0$ *s.t. if* $\epsilon < \epsilon_0$*, then* π *is optimal.*

Proof. Recall that $\mathscr{L}$ is the one-step Bellman operator and $\mathscr{L}_\pi$ is the one-step policy operator on the value function. Then (skipping the index for μ)

$$\begin{aligned}\|V^\pi - V^*\|_\infty &= \|\mathscr{L}_\pi V^\pi - V^*\|_\infty \\ &\leq \|\mathscr{L}_\pi V^\pi - \mathscr{L}_\pi \boldsymbol{u}\|_\infty + \|\mathscr{L}_\pi \boldsymbol{u} - V^*\|_\infty \\ &\leq \gamma \|V^\pi - \boldsymbol{u}\|_\infty + \|\mathscr{L}\boldsymbol{u} - V^*\|_\infty\end{aligned}$$

by contraction, and by the fact that π is $\boldsymbol{u}$-greedy

$$\begin{aligned}&\leq \gamma \|V^\pi - V^*\|_\infty + \gamma \|V^* - \boldsymbol{u}\|_\infty + \gamma \|\boldsymbol{u} - V^*\|_\infty \\ &\leq \gamma \|V^\pi - V^*\|_\infty + 2\gamma\epsilon.\end{aligned}$$

This proves the first part.

For the second part, note that the state and action sets are finite. Consequently, the set of policies is finite. Thus, there is some $\epsilon_0 > 0$ such that the best sub-optimal policy is ϵ_0-close to the optimal policy in value. So, if $\epsilon < \epsilon_0$, the obtained policy must be optimal. □

Building on this result, we can prove some simple bounds for approximate policy iteration. These are based on the following assumption.

Assumption 8.2.1. *There are* ϵ, δ *such that, for all* k*, the iterates* $\boldsymbol{v}_k, \pi_k$ *satisfy:*

$$\|\boldsymbol{v}_k - V^{\pi_k}\|_\infty \leq \epsilon, \qquad \forall k \tag{8.2.1}$$

$$\left\|\mathscr{L}_{\pi_{k+1}} \boldsymbol{v}_k - \mathscr{L}\boldsymbol{v}_k\right\|_\infty \leq \delta, \qquad \forall k \tag{8.2.2}$$

This assumption uniformly bounds the error in approximating the value of a policy by ϵ. It also demands that our approximate Bellman operator is δ-close to $\mathscr{L}$. Even though these assumptions are quite strong, we still only obtain the following rather weak asymptotic convergence result.[2]

Theorem 8.2.2 (Bertsekas and Tsitsiklis [1996], Proposition 6.2)**.** *Under Assumption 8.2.1*

$$\limsup_{k\to\infty} \|V^{\pi_k} - V^*\|_\infty \leq \frac{\delta + 2\gamma\epsilon}{(1-\gamma)^2}. \tag{8.2.3}$$

8.2.2 Rollout-based policy iteration methods

One idea for estimating the value function is to simply perform rollouts, while the policy itself is estimated in parametric form, as suggested by Bertsekas and Tsitsiklis [1996]. The first practical algorithm in this direction was Rollout Sampling Approximate Policy Iteration Dimitrakakis and Lagoudakis [2008b]. The main idea is to concentrate rollouts in interesting parts of the state space, so as to maximise the expected amount of improvement we can obtain with a given rollout budget.

[2] For $\delta = 0$, this is identical to the result for ϵ-equivalent MDPs by Even-Dar and Mansour [2003]

Algorithm 21 Rollout Sampling Approximate Policy Iteration

for $k = 1, \ldots$ **do**
  Select a set of representative states $\hat{S}_k$
  for $n = 1, \ldots$ **do**
    Calculate U_n from (8.2.6)
    Select a state $s_n \in \hat{S}_k$ maximising $U_n(s)$ and perform a rollout obtaining $\{s_{t,k}, a_{t,k}\}$.
    If $\hat{a}^*(s_n)$ is optimal w.p. $1-\delta$, put s_n in $\hat{S}_k(\delta)$ and remove it from $\hat{S}_k$.
  end for
  Calculate $\boldsymbol{q}_k \approx Q^{\pi_k}$ from the rollouts (8.1.3)
  Train a classifier $\pi_{\boldsymbol{\theta}_{k+1}}$ on the set of states $\hat{S}_k(\delta)$ with actions $\hat{a}^*(s)$.
end for

The first question is which states to look at in the first place. If we have data collects We can use the empirical state distribution to select starting states. The main idea is to concentrate rollouts on promising states. These are states where it will be the easiest to determine the optimal action. To do this, we always choose the state s with the highest upper bound $U_n(s)$, where this is typically defined as

$$\hat{a}_s^* \triangleq \arg\max_a \boldsymbol{q}_k(s,a) \tag{8.2.4}$$

$$\hat{\Delta}_k(s) \triangleq \boldsymbol{q}_k(s, \hat{a}_s^*) - \max_{a \neq \hat{a}_s^*} \boldsymbol{q}_k(s,a) \tag{8.2.5}$$

$$U_n(s) \triangleq \hat{\Delta}_k - \max_{a \neq \hat{a}_s^*} \boldsymbol{q}_k(s,a) + \sqrt{\frac{1}{1+c(s)}}, \tag{8.2.6}$$

where $c(s)$ is the number of rollouts from state s. If the sampling of a state s stops whenever

$$\hat{\Delta}_k(s) \geq \sqrt{\frac{2}{c(s)(1-\gamma)^2} \ln\left(\frac{|\mathcal{A}|-1}{\delta}\right)}, \tag{8.2.7}$$

then we are certain that the optimal action has been identified with probability $1-\delta$ for that state, due to Hoeffding's inequality. Unfortunately, guaranteeing a policy improvement for the complete state space is impossible, even with strong assumptions.[3]

8.2.3 Least Squares Methods

When c is a quadratic error, it is tempting to use linear methods, such as least squares, which are very efficient. This requires formulate the problem in linear form, using a feature mapping that projects individual states (or state action pairs) onto a high-dimensional space. Then the value function can be represented as linear function of the parameters and this mapping, which minimises a squared error over the observed trajectories.

[3]First, note that if we need to identify the optimal action for k states, then the above stopping rule has an overall error probability of $k\delta$. In addition, even if we assume that value functions are smooth, it will be impossible to identify the boundary in the state space where the optimal policy should switch actions [Dimitrakakis and Lagoudakis, 2008a].

To get an intuition for these methods, recall from Theorem 6.5.1 that the solution of

$$\boldsymbol{v} = \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi} \boldsymbol{v}$$

is the value function of π and can be obtained via

$$\boldsymbol{v} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})^{-1} \boldsymbol{r}.$$

Here we consider the setting we do not have access to the transition matrix, but instead have some observations of transition (s_t, a_t, s_{t+1}). In addition, our state space can be continuous (e.g. $\mathcal{S} \subset \mathbb{R}^n$), so that the transition matrix becomes a general transition kernel. Consequently, the set of value functions $\mathcal{V}$ becomes a Hilbert space, while it previously was a Euclidean subset.

In general, we deal with this case via projections. We project from the infinite-dimensional Hilbert space to one with finite dimension on a subset of states: namely, the ones that we have observed. We also replace the transition kernel with the empirical transition matrix on the observed states.

Parametrisation. Let us first deal with parametrising a linear value function. Setting $\boldsymbol{v} = \boldsymbol{\Phi}\boldsymbol{\theta}$ where $\boldsymbol{\Phi}$ is a feature matrix and $\boldsymbol{\theta}$ is a parameter vector we have

$$\begin{aligned} \boldsymbol{\Phi}\boldsymbol{\theta} &= \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi} \boldsymbol{\Phi}\boldsymbol{\theta} \\ \boldsymbol{\theta} &= \left[(\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi}\right]^{-1} \boldsymbol{r} \end{aligned}$$

This simple linear parametrisation of a value function is perfectly usable in a discrete MDP setting where the transition matrix is given. However, when neither of these things is true, using this parametrisation is not so straightforward. The first problem is how to define the transition matrix itself, since there is an infinite number of states. The simple solution to this problem is to only define the matrix on the observed states, and furthermore, so that the probability of transiting to a particular state is 1 if that transition has been observed. This makes the matrix off-diagonal. More precisely, the construction is as follows.

Empirical construction. Given a set of data points $\{(s_i, a_i, r_i, s'_i) \mid i = 1, \dots, n\}$, we define:

1. The empirical reward vector $\boldsymbol{r} = (r_i)_i$.
2. The feature matrix $\boldsymbol{\Phi} = (\boldsymbol{\Phi}_i)_i$, with $\boldsymbol{\Phi}_i = f(s_i, a_i)$,
3. The empirical transition matrix $\boldsymbol{P}_{\mu,\pi}(i, j) = \mathbb{I}\{j = i + 1\}$

However, generally the value function space generated by the features and the linear parameterisation does not allow us to obtain exact value functions. For this reason we replace the inverse with the *pseudo-inverse*, defined as:

$$\widetilde{\boldsymbol{A}}^{-1} \triangleq \boldsymbol{A}^\top \left(\boldsymbol{A}\boldsymbol{A}^\top\right)^{-1}, \qquad \boldsymbol{A} = (\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi}$$

If the inverse exists, then it is equal to the pseudo-inverse. However, in our setting, the matrix can be low rank, in which case we instead obtain the matrix minimising the squared error, which in turn can be used to obtain a good

estimate for the parameters. This immediately leads to the Least Squares Temporal Difference algorithm [Bradtke and Barto, 1996, LSTD], an algorithm that estimates an approximate value function for some policy π given some data D and a feature mapping f.

State-action value functions. However, estimating a state-value function is not directly useful for obtaining an improved policy without a model. For that reason, we can instead consider estimating a state-action value function. This leads to the following estimator.

$$\begin{aligned}\boldsymbol{q} &= \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{q} \\ \boldsymbol{\Phi\theta} &= \boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{\Phi\theta} \\ \boldsymbol{\theta} &= \widetilde{((\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi})}^{-1}\boldsymbol{r}.\end{aligned}$$

This approach has two drawbacks. The first is that it is difficult to get an unbiased estimate of θ. The second is that when we apply the Bellman operator to $\boldsymbol{q}$, the result may lie outside the space spanned by the features. For this reason, we can instead consider the least-square projection $\boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\top$:

$$\boldsymbol{q} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\top \left(\boldsymbol{r} + \gamma \boldsymbol{P}_{\mu,\pi}\boldsymbol{q}\right).$$

Replacing $\boldsymbol{q} = \boldsymbol{\Phi\theta}$ leads to the estimate:

$$\boldsymbol{\theta} = \left(\boldsymbol{\Phi}^\top(\boldsymbol{I} - \gamma \boldsymbol{P}_{\mu,\pi})\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^\top\boldsymbol{r}.$$

In practice, of course, we do not have the transitions $\boldsymbol{P}$ but estimate them from data. Note that for any deterministic policy π, and a set of n data points, we have:

$$\begin{aligned}\boldsymbol{P}_{\mu,\pi}\boldsymbol{\Phi} &= \sum_{s'} P(s' \mid s,a)\boldsymbol{\Phi}(s',\pi(s')) \\ &\approx \frac{1}{T}\sum_{t=1}^{T} \hat{P}(s_{t+1} \mid s_t,a_t)\boldsymbol{\Phi}(s_{t+1},\pi(s_{t+1})).\end{aligned}$$

As shown in Algorithm 22 we can now use this to maintain $\boldsymbol{q}$-factors, so that $\boldsymbol{q}(s,a) = f(s,a)\boldsymbol{\theta}$, and use the empirical estimate of the Bellman operator.

Algorithm 22 LSTDQ - Least Squares Temporal Differences

input data $D = \{(s_t, a_t, r_t, s'_t) \mid t = 1,\dots,T\}$, feature mapping f, policy π
$\boldsymbol{A} = \sum_{t=1}^{n} \boldsymbol{\Phi}(s_t,a_t)\boldsymbol{P}[\boldsymbol{\Phi}(s_t,a_t) - \gamma\boldsymbol{\Phi}(s_{t+1},\pi(s_{t+1}))]$.
$\boldsymbol{b} = \sum_{t=1}^{n} \boldsymbol{\Phi}(s_t,a_t)r_t$
$\boldsymbol{\theta} = \boldsymbol{A}^{-1}\boldsymbol{b}$

The algorithm can be easily extended to approximate policy iteration, giving us the well-known Least Squares Policy Iteration [Lagoudakis and Parr, 2003b, LSPI] algorithm shown in Alg. 23. The idea is to repeatedly estimate the value function for improved policies using a least squares estimate, and then obtain the greedy policy for each estimate.

Algorithm 23 LSPI - Least Squares Policy Iteration

input data $D = \{(s_i, a_i, r_i, s_i') \mid i = 1, \dots, n\}$, feature mapping f
Set π_0 arbitrarily.
for $k = 1, \dots$ **do**
  $\boldsymbol{\theta}^{(k)} = \text{LSTDQ}(D, f, \pi_{k-1})$.
  $\pi^{(k)} = \pi^*_{\boldsymbol{\Phi}\boldsymbol{\theta}^{(k)}}$.
end for

8.3 Approximate Value Iteration

Approximate algorithms can also be defined for backwards induction. The general algorithmic structure remains the same as exact backwards induction. We only need to replace the exact steps with approximations. There are two reasons for this. Firstly, the value function cannot be updated for all states. Secondly, our value function representations are not complex enough to capture the true value function.

8.3.1 Approximate backwards induction

The first algorithm is approximate backwards induction. Let us start with the basic backwards induction algorithm:

$$V_t^*(s) = \max_{a \in \mathcal{A}} \left\{ r(s,a) + \gamma \, \mathbb{E}_\mu \left(V_{t+1}^* \mid s_t = s, a_t = a \right) \right\} \tag{8.3.1}$$

This is essentially the same both for finite and infinite-horizon problems.If we have to pick the value function from a set of functions $\mathcal{V}$, we can use the following value function approximation.

Let our estimate at time t be $\boldsymbol{v}_t \in \mathcal{V}$, with $\mathcal{V}$ being a set of functions. Let $\hat{V}_t$ be our one-step update given the value function approximation at the next step, $\boldsymbol{v}_{t+1}$. Then $\boldsymbol{v}_t$ will be the closest approximation in that set.

Iterative approximation

$$\hat{V}_t(s) = \max_{a \in \mathcal{A}} \left\{ r(s,a) + \gamma \sum_{s'} \boldsymbol{P}_\mu(s' \mid s, a) \boldsymbol{v}_{t+1}(s') \right\}$$
$$\boldsymbol{v}_t = \arg\min_{\boldsymbol{v} \in \mathcal{V}} \left\| \boldsymbol{v} - \hat{V}_t \right\|$$

The above minimisation can for example be perfoemed by gradient descent: Consider the case where $\boldsymbol{v}$ is a parameterised function from a set of parametrised value functions $\mathcal{V}_\Theta$ with parameters $\boldsymbol{\theta}$. Then it is sufficient for us to maintain the parameter $\boldsymbol{\theta}^{(t)}$ at time t. These can be updated with a gradient scheme at every step. In the online case, our next-step estimates can be given by gradient descent, with a step size sequence α_t:

Online gradient estimation

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \nabla_{\boldsymbol{\theta}} \left\| \boldsymbol{v}_t - \hat{V}_t \right\| \tag{8.3.2}$$

This gradient descent algorithm can also be made stochastic, if we sample s' from the probability distribution $\boldsymbol{P}_\mu(s' \mid s, a)$ used in the iterative approximation. The next sections give some examples.

8.3.2 State aggregation

In state aggregation, multiple different states are identified in order to obtain a new aggregated state in an aggregated MDP with a smaller state space. Unfortunately, it is very rarely the case that aggregated states really are indeed indistinguishable with respect to rewards and transition probabilities. Nevertheless, as we can see in the example below, aggregation can significantly simplify computation through the reduction of the size of the state space.

EXAMPLE 42 (Aggregated value function.). A simple method for aggregation is to set the value of every state in an aggregate set to be the same. More precisely, Let $\mathscr{G} = \{S_1, \dots, S_n\}$ be a partition of $\mathcal{S}$, with $\boldsymbol{\theta} \in \mathbb{R}^n$ and let $f_k(s_t) = \mathbb{I}\{s_t \in S_k\}$. Then the approximate value function is

$$\boldsymbol{v}(s) = \boldsymbol{\theta}(k), \qquad \text{if } s \in S_k,\ k \neq 0.$$

In the above example, the value of every state corresponds to the value of the k-th set in the partition. Of course, this is only a very rough approximation if the sets S_k are very large. However, this is a convenient approach to use for gradient descent updates, as only one parameter needs to be updated at every step.

Online gradient estimate for aggregated value functions
Consider the case $\|\cdot\| = \|\cdot\|_2^2$. For $s_t \in S_k$ and some step size sequence α_t:

$$\boldsymbol{\theta}_{t+1}(k) = (1 - \alpha_t)\boldsymbol{\theta}_t(k) + \alpha \max_{a \in \mathcal{A}} r(s_t, a) + \gamma \sum_j P(j \mid s_t, a)\boldsymbol{\theta}_t(f_k(j)),$$

while $\boldsymbol{\theta}_{t+1}(k) = \boldsymbol{\theta}(k)$ for $s_t \notin S_k$.

Of course, whenever we perform the estimation online, we are limited to estimation on the sequence of states s_t that we visit. Consequently, estimation on other states may not be very good. It is indeed possible that we suffer from convergence problems as we alternate between estimating the values of different states in the aggregate.

8.3.3 Representative state approximation

A more refined approach is to choose some representative states and try to approximate the value function of all other states as a convex combination of

the value of the representative states.

Representative state approximation.
Let $\hat{\mathcal{S}}$ be a set of n representative states and $\boldsymbol{\theta} \in \mathbb{R}^n$ and a feature mapping f with

$$\sum_{i=1}^{n} f_i(s) = 1, \qquad \forall s \in \mathcal{S}.$$

The feature mapping is used to perform the convex combination. Usually, $f_i(s)$ is larger for representative states i which are "closer" to s. In general, the feature mapping is fixed, and we want to find a set of parameters for the values of the representative states. At time t, for each representative state i, we obtain a new estimate of its value function and plug it back in.

Representative state update.
For $i \in \hat{\mathcal{S}}$:

$$\boldsymbol{\theta}_{t+1}(i) = \max_{a \in \mathcal{A}} \left\{ r(i,a) + \gamma \int \boldsymbol{v}_t(s) \, \mathrm{d}P(s \mid i, a) \right\} \tag{8.3.3}$$

with

$$\boldsymbol{v}_t(s) = \sum_{i=1}^{n} f_i(s) \boldsymbol{\theta}_t(i). \tag{8.3.4}$$

When the integration in (8.3.3) is not possible, we may instead approximate the expectation with a Monte-Carlo method. One particular problem with this method arises when the transition kernel is very sparse. Then we are basing our estimates on approximate values of other states, which may be very far from any other representative state. This is illustrated in Figure 8.4, which presents the value function error for the chain environment and random MDPs. Due to the linear structure of the chain environment, the states are far from each other. In contrast, the random MDPs are generally both quite dense and the state distribution for any particular policy mixes rather fast. Thus, states in the former tend to have very different values and in the latter very similar ones.

8.3.4 Bellman error methods

The problems with the representative state update can be alleviated through Bellman error minimisation. The idea here is to obtain as a *consistent* value function as possible. The basic Bellman error minimisation is as follows:

$$\min_{\boldsymbol{\theta}} \|\boldsymbol{v}_{\boldsymbol{\theta}} - \mathscr{L} \boldsymbol{v}_{\boldsymbol{\theta}}\| \tag{8.3.5}$$

This is different from the approximate backwards induction algorithm we saw previously, since the same parameter $\boldsymbol{\theta}$ appears in both sides of the equality.

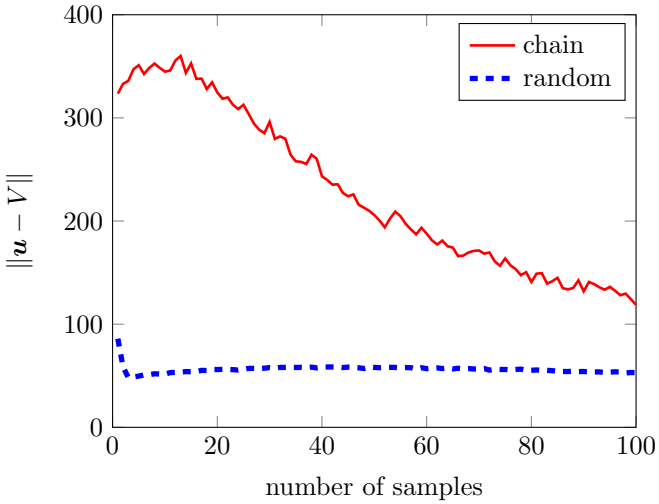


Figure 8.4: Error in the representative state approximation for two different MDPs structures as we increase the number of sampled states. The first is the chain environment from Example 34, extended to 100 states. The second involves randomly generated MDPs with 2 actions and 100 states.

Furthermore, if the norm has support in all of the state space and the approximate value function space contains the actual set of value functions then the minimum is 0 and we obtain the optimal value function.

Gradient update.
For the L2-norm, we have:

$$\|\boldsymbol{v}_{\boldsymbol{\theta}} - \mathscr{L}\boldsymbol{v}_{\boldsymbol{\theta}}\| = \sum_{s \in \hat{S}} D_{\boldsymbol{\theta}}(s)^2, \quad D_{\boldsymbol{\theta}}(s) = \boldsymbol{v}_{\boldsymbol{\theta}}(s) - \max_{a \in \mathcal{A}} \int_{\mathcal{S}} \boldsymbol{v}_{\boldsymbol{\theta}}(j) \, \mathrm{d}P(j \mid s, a). \tag{8.3.6}$$

Then the gradient update becomes $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha D_{\boldsymbol{\theta}_t}(s_t) \nabla_{\boldsymbol{\theta}} D_{\boldsymbol{\theta}_t}(s_t)$, where

$$\nabla_{\boldsymbol{\theta}} D_{\boldsymbol{\theta}_t}(s_t) = \nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}_t}(s_t) - \int_{\mathcal{S}} \nabla_{\boldsymbol{\theta}} \boldsymbol{v}_{\boldsymbol{\theta}_t}(j) \, \mathrm{d}P(j \mid s_t, a_t^*),$$

with $a_t^* \triangleq \arg\max_{a \in \mathcal{A}} \left\{ r(s_t, a) + \gamma \int_{\mathcal{S}} \boldsymbol{v}_{\boldsymbol{\theta}}(j) \, \mathrm{d}P(j \mid s_t, a) \right\}$.

We can also construct a Q-factor approximation for the case where no model is available. This is going to be simply done by replacing P with the empirical transition observed at time t.

8.4 Policy gradient

In the previous section, we saw how we could use gradient methods for value function approximation. However, it is also possible to use these methods to

estimate policies – the only necessary ingredients are a policy representation and a way to evaluate a policy. The representation is usually parametric, but non-parametric representations are also possible. A common choice for parameterised policies is to use a feature function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^k$ and a linear parameterisation with parameters $\theta \in \mathbb{R}^k$ leading to the following Softmax distribution:

$$\pi(a \mid s) = \frac{e^{F(s,a)}}{\sum_{a' \in \mathcal{A}} e^{F(s,a')}}, \qquad F(s,a) \triangleq \theta^\top f(s,a). \tag{8.4.1}$$

As usual, we would like to find a policy maximising expected utility. Policy gradient algorithms employ gradient ascent on the expected utility to find a locally maximising policy. Here we focus on the discounted reward criterion, with discount factor γ, where a policy's expected utility is defined with respect to a starting state distribution $\boldsymbol{y}$ so that

$$\mathbb{E}^\pi_{\boldsymbol{y}}(U) = \sum_s \boldsymbol{y}(s) V^\pi(s) = \sum_s \boldsymbol{y}(s) \sum_h \mathbb{P}^\pi(h \mid s_1 = s) U(h),$$

where $U(h)$ is the utility of a trajectory h. This definition leads to a number of simple expressions for the gradient of the expected utility.

Theorem 8.4.1. *Assuming that the reward only depends on the state, for any $\boldsymbol{\theta}$-parameterised policy space Π, the gradient of the utility from starting state distribution $\boldsymbol{y}$ can be equivalently written in the three following forms:*

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}^\pi_{\boldsymbol{y}} U = \boldsymbol{y}^\top \gamma (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \nabla_{\boldsymbol{\theta}} \boldsymbol{P}^\pi_\mu (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \boldsymbol{r} \tag{8.4.2}$$

$$= \sum_s x^{\pi,\gamma}_{\mu,\boldsymbol{y}}(s) \sum_a \nabla_{\boldsymbol{\theta}} \pi(a \mid s) Q^\pi_\mu(s,a) \tag{8.4.3}$$

$$= \sum_h U(h)\, \mathbb{P}^\pi_\mu(h) \nabla \ln \mathbb{P}^\pi_\mu(h), \tag{8.4.4}$$

where (as in Sec. 6.5.4) we use

$$x^{\pi,\gamma}_{\mu,\boldsymbol{y}}(s) = \sum_{s'} \sum_{t=0}^{\infty} \gamma^t\, \mathbb{P}^\pi_\mu(s_t = s \mid s_0 = s') y(s').$$

to denote the γ-discounted sum of state visits. Finally $h \in (\mathcal{S} \times \mathcal{A})^*$ is a state-action history and $\mathbb{P}^\pi_\mu(h)$ its probability under the policy π and MDP μ and $U(h)$ is the utility of history h.

Proof. We begin by proving the claim (8.4.2). Note that the

$$\mathbb{E}^\pi_\mu U = \boldsymbol{y}^\top (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \boldsymbol{r}$$

where $\boldsymbol{y}$ is a starting state distribution vector and $\boldsymbol{P}^\pi_\mu$ is the transition matrix resulting from applying policy π to μ. Then we can calculate the derivative of the above expression using matrix calculus, that is:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}\, U = \boldsymbol{y}^\top \nabla_{\boldsymbol{\theta}} (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \boldsymbol{r},$$

as the only term involving $\boldsymbol{\theta}$ is π. The derivative of a matrix inverse can be written as:

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} &= -(\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \nabla_{\boldsymbol{\theta}} (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu) (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \\ &= \gamma (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1} \nabla_{\boldsymbol{\theta}} \boldsymbol{P}^\pi_\mu (\boldsymbol{I} - \gamma \boldsymbol{P}^\pi_\mu)^{-1}, \end{aligned}$$

which concludes the proof of the first claim. We can also expand the $\boldsymbol{P}_\mu^\pi$ term, thus obtaining a formula that only has a derivative for π:

$$\frac{\partial}{\partial \theta_i} \mathbb{P}_\mu^\pi(s' \mid s) = \sum_a \mathbb{P}_\mu(s' \mid s, a) \frac{\partial}{\partial \theta_i} \pi(a \mid s).$$

Defining the state visitation matrix $\boldsymbol{X} \triangleq (\boldsymbol{I} - \gamma \boldsymbol{P})^{-1}$ we have, rewriting (8.4.2):

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_\mu^\pi U = \gamma \boldsymbol{y}^\top \boldsymbol{X} \nabla_{\boldsymbol{\theta}} \boldsymbol{P}_\mu^\pi \boldsymbol{X} \boldsymbol{r}. \tag{8.4.5}$$

We are now ready to prove claim (8.4.3). Now define the expected state visitation from the starting distribution to be $\boldsymbol{x} \triangleq \boldsymbol{y}^\top \boldsymbol{X}$, so that the above becomes:

$$\begin{aligned}
\nabla \mathbb{E}_\mu^\pi U &= \gamma \boldsymbol{x}^\top \nabla_{\boldsymbol{\theta}} \boldsymbol{P}_\mu^\pi \boldsymbol{X} \boldsymbol{r} && (8.4.6)\\
&= \gamma \sum_s \boldsymbol{x}(s) \sum_{a,s'} \boldsymbol{P}_\mu(s' \mid s, a) \nabla_{\boldsymbol{\theta}} \pi(a \mid s) V(s') && (8.4.7)\\
&= \gamma \sum_s \boldsymbol{x}(s) \sum_a \nabla_{\boldsymbol{\theta}} \pi(a \mid s) \sum_{s'} \boldsymbol{P}_\mu(s' \mid s, a) V_\mu^\pi(s') && (8.4.8)\\
&= \gamma \sum_s x(s) \sum_a \nabla_{\boldsymbol{\theta}} \pi(a \mid s) Q_\mu^\pi(s, a). && (8.4.9)
\end{aligned}$$

We thus obtain the policy gradient theorem of [Sutton et al., 1999]. The last claim (8.4.4) is straightforward. Indeed,

$$\nabla \mathbb{E} U = \sum_h U(h) \nabla \mathbb{P}(h) = \sum_h U(h) \mathbb{P}(h) \nabla \ln \mathbb{P}(h), \tag{8.4.10}$$

as $\nabla \ln \mathbb{P}(h) = \frac{1}{\mathbb{P}(h)} \nabla \mathbb{P}(h)$. □

8.4.1 Stochastic policy gradient;

For finite MDPs, we can obtain $\boldsymbol{x}_\pi$ through state occupancy matrix (6.5.5) by left multiplying the latter with the initial state distribution $\boldsymbol{y}$. However, in the context of gradient methods, it makes more sense to use a stochastic estimate of $\boldsymbol{x}_\pi$ to calculate the gradient, since:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}^\pi U = \mathbb{E}_y^\pi \sum_a \nabla_{\boldsymbol{\theta}} \pi(a \mid s) Q^\pi(s, a). \tag{8.4.11}$$

For the discounted reward criterion, we can easily obtain unbiased samples through geometric stopping (see exercise 28).

Importance sampling.

The last formulation is especially useful as it allows us to use importance sampling rather easily. This allows us to compute the gradient even on data obtained for different policies, and might be more data efficient. First note that for any history $h \in (\mathcal{S} \times \mathcal{A})^*$, we have

$$\mathbb{P}_\mu^\pi(h) = \prod_{t=1}^T \mathbb{P}_\mu(s_t \mid s^{t-1}, a^{t-1}) \, \mathbb{P}^\pi(a_t \mid s^t, a^{t-1}) \tag{8.4.12}$$

without any Markovian assumptions on the model or policy. We can now rewrite (8.4.10) in terms of the expectation with respect to an alternative policy π':

$$\begin{aligned}\nabla \mathbb{E}^{\pi}_{\mu} U &= \mathbb{E}^{\pi'}_{\mu}\left(U(h)\nabla \ln \mathbb{P}^{\pi}_{\mu}(h)\frac{\mathbb{P}^{\pi}_{\mu}(U)}{\mathbb{P}^{\pi'}_{\mu}(U)}\right)\\ &= \mathbb{E}^{\pi'}_{\mu}\left(U(h)\nabla \ln \mathbb{P}^{\pi}_{\mu}(h)\prod_{t=1}^{T}\frac{\pi(a_t \mid s^t, a^{t-1})}{\pi'(a_t \mid s^t, a^{t-1})}\right),\end{aligned}$$

as the μ-dependent terms in (8.4.12) cancel out. In practice the expectation would be approximated through sampling trajectories h.

$$\nabla \ln \mathbb{P}^{\pi}_{\mu}(h) = \sum_t \nabla \ln \pi(a_t \mid s^t, a^{t-1}) = \sum_t \frac{\nabla \pi(a_t \mid s^t, a^{t-1})}{\pi(a_t \mid s^t, a^{t-1})}.$$

This allows us to perform stochastic gradient descent on data collected from any arbitrary previous policy π', and perform gradient descent on a parametrised policy π.

8.4.2 Practical considerations.

The first design choice in any gradient algorithm is how to parameterise the policy. For the discrete case, a common parameterisation is to have a separate and independent parameter for each state-action pair, i.e. $\theta_{s,a} = \pi(a|s)$. This leads to a particularly simple expression for the second form (8.4.3), which is $\partial/\partial_{\theta_{s,a}} \mathbb{E}^{\pi}_{\mu} U = y(s)Q(s,a)$. However, it is easy to see that in this case the parameterisation will lead to all parameters increasing if rewards are positive. This can be avoided by either a Softmax parameterisation or by subtracting a bias term[4] from the derivative. Nevertheless, this parameterisation implies stochastic discrete policies.

We could also suitably parameterise continuous policies. For example, if $\mathcal{A} \subset \mathbb{R}^n$, we can consider a linear policy. Most of the derivation carries over to Euclidean state-action spaces. In particular, the second form (8.4.3) is also suitable for deterministic policies.

Finally, in practice, we may not need to accurately calculate the expectations involved in the gradient. Sample trajectories are sufficient to update the gradient in a meaningful way, especially for the third form (8.4.4), as we can naturally sample from the distribution of trajectories. However, the fact that this form doesn't need a Markovian assumption also means that it cannot take advantage of Markovian environments.

Policy gradient methods are useful, especially in cases where the environment model or value function is extremely complicated, while the optimal policy itself might be quite simple. The main difficulty lies in obtaining an appropriate estimate of the gradient itself, but convergence to a local maximum is generally good as long as we are adjusting the parameters in a gradient-related direction.[5]

[4] e.g. $Q^{\pi}_{\mu}(s, a_1)$

[5] In the sense of Ass 7.1.1(iii).

8.5 An extended example

Let us now consider two well-known problems with a 2-dimensional continuous state space and a discrete set of actions. The first is the *inverted pendulum* problemLagoudakis and Parr [2003b, the version in], where a controller must balance a rod upside-down. The state information is the rotational velocity and position of the pendulum. The second is the *mountain car* problem, where we must drive an underpowered vehicle to the top of a hill[Sutton and Barto, 1998]. The state information is the velocity and location of the car. In both problems, there are three actions: "push left", "push right" and "do nothing".

Let us first consider the effect of model and features in representing the value function of the inverted pendulum problem. Figure 8.5 shows value function approximations for policy evaluation under a uniformly random policy, for under different choices of model and features. Here we need to fit an approximate value function to samples of the utility obtained from different states. The quality of the approximation depends on both the model and the features used. The first choice of features is simply the raw state representation, while the second a 16×16 uniform RBF tiling. The two models are very simple: the first is a linear model-Gaussian model[6] assumption on observation noise (LG), and the second is a k-nearest neighbour (kNN) model.

As can be seen from the figure the linear model results in a smooth approximation, but is inadequate for modelling the value function in the original 2-dimensional state space. However, a high-dimensional non-linear projection using RBF kernels results in a smooth and accurate value function representation. Non-parametric models such as k-nearest neighbours behave rather well under either state representation.

Now consider instead the problem of finding the optimal value function. Now we must additionally consider the question of which algorithm to use. In Figure 8.6 we see the effect of choosing either approximate value iteration (AVI) or representative state representations and value iteration (RSVI) for this model.

8.6 Further reading

Among value function approximation methods, the two most well known are fitted Q-iteration [Antos et al., 2008b], and fitted value iteration, which has been analysed in [Munos and Szepesvári, 2008]. Minimising the Bellman error [Antos et al., 2008a, Dimitrakakis, 2013, Ghavamzadeh and Engel, 2006] is generally a good way to ensure that approximate value iteration is stable.

In approximate policy iteration methods, one needs to approximate both the value function and policy. In rollout sampling policy iteration [Dimitrakakis and Lagoudakis, 2008b,a], an empirical approximation of the value function is maintained. However, one can employ least-squares methods [Bradtke and Barto, 1996, Boyan, 2002, Lagoudakis and Parr, 2003b] for example.

The general technique of state aggregation [Singh et al., 1995, Bernstein, 2007] is applicable to a variety of reinforcement learning algorithms. While the more general question of selecting features appropriately is open, there has been

[6] Essentially, this is the a linear model of the form $s_{t+1} \mid s_t = s, a_t = a \sim \mathcal{N}\mu_a^\top s, \Sigma_a$, where μ_a has a normal prior and Σ a Wishart priorl

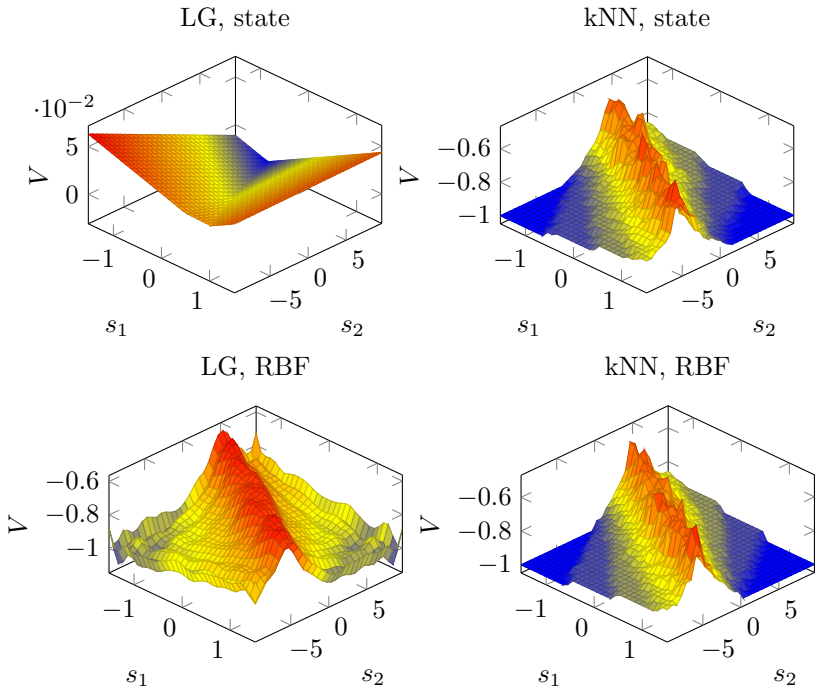


Figure 8.5: Estimated value function of a uniformly random policy on the two-dimensional state-space of the pendulum problem. Results are shown for a k-nearest neighbour model (kNN) with $k = 3$ and a Bayesian linear-Gaussian model (LG), for either the case when the model uses the plain state information (state) or an 256-dimensional RBF embedding (RBF).

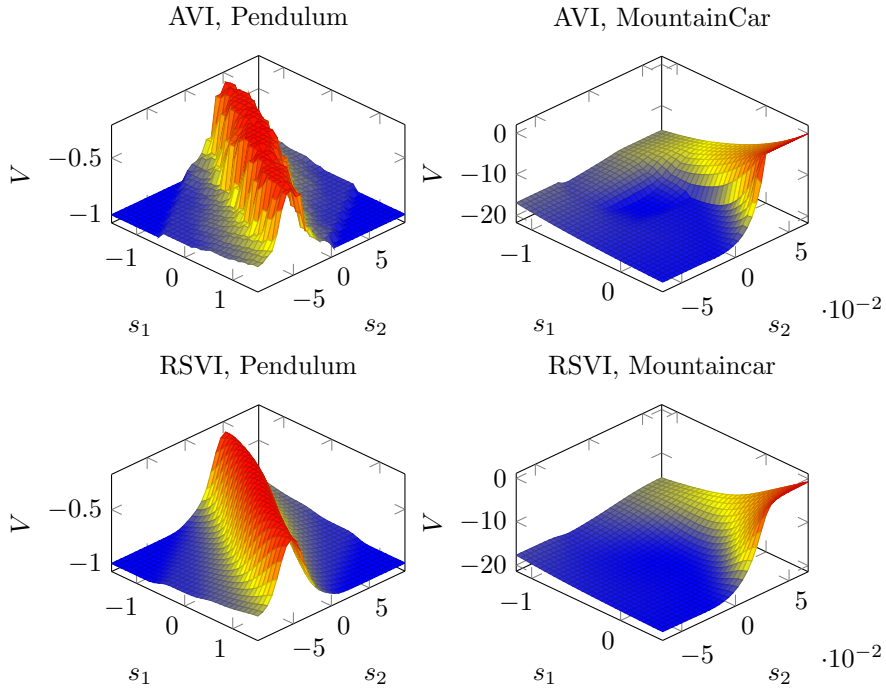


Figure 8.6: Estimated optimal value function for the pendulum problem. Results are shown for approximate value iteration (AVI) with a Bayesian linear-Gaussian model, and a representative state representation (RSVI) with an RBF embedding. Both the embedding and the states where the value function is approximated are a 16×16 uniform grid over the state space.

some progress in the domain of feature reinforcement learning [Hutter, 2009]. In general, learning internal representations (i.e. features) has been a prominent aspect of neural network research [Rumelhart et al., 1987]. Even if it is unclear to what extent recently proposed approximation architectures that employ deep learning actually learn useful representations, they have been successfully used in combination with simple reinforcement learning algorithms [Mnih et al., 2015]. Another interesting direction is to establish links between features and approximately sufficient statistics [Dimitrakakis and Tziortziotis, 2013, 2014].

Finally, the policy gradient theorem in the state visitation form was first proposed by Sutton et al. [1999], while Williams [1992] was the first to use the log-ratio trick (8.4.4) in reinforcement learning. To our knowledge, the analytical gradient has not actually been applied (or indeed, described) in prior literature. Extensions of the policy gradient idea are also natural. They have also been used in a Bayesian setting by Ghavamzadeh and Engel [2006], while the natural gradient has been proposed by Kakade [2002]. A survey of policy gradient methods can be found in [Peters and Schaal, 2006].

8.7 Exercises

Exercise 34 (Enlarging the function space.). Consider the problem in example 36. What would be a simple way to extend the space of value functions from the three given candidates to an infinite number of value functions? How could we get a good fit?

Exercise 35 (Enlarging the policy.). Consider example 37. This represents and example of a linear deterministic policies. In which two ways can this policy space be extended and how?

Exercise 36. Find the derivative for the two other cases, specifically:

1. $p = 2$, $\kappa = 2$.
2. $p \to \infty$, $\kappa = 1$.

Solution. For $p = 2, \kappa = 2$, the derivative can be written as:

□

Chapter 9

Bayesian reinforcement learning

9.1 Introduction

In this chapter, we will return to the setting of subjective probability and utility, by formalising the reinforcement learning problem as a Bayesian decision problem and solving it directly. In the Bayesian setting, we are acting in an MDP which is not known, but we have a subjective belief about what the environment is. We shall first consider the case of acting in unknown MDPs, which is the focus of the reinforcement learning problem. We will examine a few different heuristics for maximising expected utility in the Bayesian setting, and contrast them with tractable approximations to the Bayes-optimal solution. In Section 9.4, we will connect this problem to partially observable MDPs. Finally, we shall present extensions of these ideas to continuous domains.

9.2 Acting in unknown MDPs

The reinforcement learning problem can be formulated as the problem of learning to act in an unknown environment, only by interaction and reinforcement. All of these elements of the definition are important. Firstly and foremostly it is a *learning* problem: We have only partial prior knowledge about the environment we are acting in. This knowledge is arrived at via *interaction* with the environment. We do not have a fixed set of data to work with, but we must actively explore the environment to understand how it works. Finally, there is a *reinforcement* signal that punishes some behaviours and rewards others. We can formulate some of these problems as Markov decision processes.

Let us consider the case where the environment can be represented as an MDP μ. That is, at each time step t, we observe the environment's state $s_t \in \mathcal{S}$, take an action $a_t \in \mathcal{A}$ and receive *reward* $r_t \in \mathbb{R}$. In the MDP setting, the state and our action fully determine the distribution of the immediate reward, as well as that of the next state, as described in Definition 6.3.1. For a specific MDP μ the probability of the immediate reward is given by $\mathbb{P}_\mu(r_t \mid s_t, a_t)$, with expectation $\bar{r}_\mu(s,a) \triangleq \mathbb{E}_\mu(r_t \mid s_t = s, a_t = a)$, while the next state distribution is given by $\mathbb{P}_\mu(s_{t+1} \mid s_t, a_t)$. If these quantities are known, or if we can at least draw samples from these distributions, it is possible to employ stochastic approximation and approximate dynamic programming to estimate the optimal policy and value function for the MDP.

More precisely, when μ is known, we wish to find a *policy* $\pi : \mathcal{S} \to \mathcal{A}$ maximising the *utility* in expectation. This requires us to solve the maximisation problem $\max_\pi \mathbb{E}^\pi_\mu U$, where the utility is an additive function of rewards, $U = \sum_{t=1}^T r_t$. this can be accomplished using standard algorithms, such as value or policy iteration. However, knowing μ is contrary to the problem definition.

In Chapter 7 we have seen a number of stochastic approximation algorithms which allow us to learn the optimal policy for a given MDP eventually. However, these generally give few guarantees on the performance of the policy while learning. How can we create an algorithm for optimally learning MDPs? This should trade off exploring the environment to obtain further knowledge, and simultaneously exploiting this knowledge.

Within the subjective probabilistic framework, there is a natural formalisation we only need to define a prior belief ξ on the set of MDPs $\mathcal{M}$, and then find the policy that maximises the expected utility with respect to the prior $\mathbb{E}^\pi_\xi(U)$.

The structure of the unknown MDP process is shown in Figure 9.1 below. We

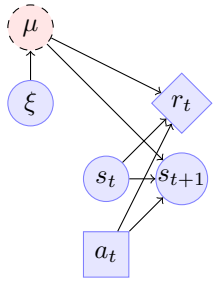


Figure 9.1: The unknown Markov decision process. ξ is our prior over the unknown μ, which is not directly observed. However, we always observe the result of our actions a_t in terms of rewards r_t and next states s_{t+1}.

have previously seen two simpler sequential decision problems in the Bayesian setting. The first was the simple optimal stopping procedure in Section 5.2.2, which introduced the backwards induction algorithm. The second was the optimal experiment design problem, which resulted in the bandit Markov decision process of Section 6.2. Let us now formulate the reinforcement learning problem as a Bayesian maximisation problem.

Let ξ be a prior over $\mathcal{M}$ and Π be a set of policies. Then the expected utility of the optimal policy is:

$$U^*_\xi \triangleq \max_{\pi \in \Pi} \mathbb{E}(U \mid \pi, \xi) = \max_{\pi \in \Pi} \int_{\mathcal{M}} \mathbb{E}(U \mid \pi, \mu) \, \mathrm{d}\xi(\mu) \tag{9.2.1}$$

Solving this optimisation problem, and hence finding the optimal policy is not easy, as in general the optimal policy π must incorporate the information it obtained while interacting with the MDP. Formally, this means that it must map from *histories* to actions. For any such history-dependent policy, the action we take at step t must depend on what we observed in steps $1, \dots, t$. Consequently, when we wish find the optimal policy, we must specify it for all future time steps. In other words, the policy must take into account the learning that will take place in this interval. Thus, in some sense, the value of information is automatically taken into account in this model. This is illustrated in the following example.

EXAMPLE 43. Consider two MDPs, μ_1, μ_2 with a single state (i.e. $\mathcal{S} = \{1\}$) and actions $\mathcal{A} = \{1, 2\}$. In the MDP μ_i, whenever you take action $a_t = i$, you obtain reward $r_t = 1$, otherwise you obtain reward 0. The expected utility of a memoryless policy taking action i with probability $\pi(i)$ is

$$\mathbb{E}^\pi_\xi U = T \sum_i \xi(\mu_i)\pi(i),$$

for horizon T. Consequently, if the prior ξ is not uniform, the optimal memoryless policy selects the action corresponding to the MDP with the highest prior probability. Then, the maximal expected utility is:

$$\max_{\pi \in \Pi_1} \mathbb{E}^\pi_\xi U = T \max_i \xi(\mu_i).$$

However, in this example, we become certain of what is the right MDP as soon as we take one action. Consequently, an improved policy is the following: First select the apparently best action, and then switche to the best action for the MDP we have seen. Then, our utility is simply $\max_i \xi(\mu_i) + (T-1)$.

As we have to consider quite general policies in this setting, it is useful to differentiate between different policy types. We use Π to denote the set of all policies. We use Π_k to denote the set of k-order Markov policies. so that

$$\pi(a_t \mid s^t, a^{t-1}, r^{t-1}) = \pi(a_t \mid s_t, a_{t-1}, r_{t-1}, \ldots, s_{t-k+1}, a_{t-k}, r_{t-k}) \qquad \text{if } \pi \in \Pi_k$$

Important special cases are the set of *blind* policies Π_0 and the set of *memoryless* policies Π_1. A policy in $\pi \in \bar{\Pi}_k \subset \Pi_k$ is *stationary*, when $\pi(A \mid s_{t-k+1}^t, a_{t-k+1}^{t-1}) = \pi(A \mid s^k, a^{k-1})$ for all t. Finally, policies may be indexed by some parameter set Θ, in which case the set of parameterised policies is given by Π_Θ.

Let us now discuss turn to the problem of how to construct an optimal policy. As the optimal policy must include learning, we must first examine how to update the belief. Given that, we shall examine methods for exact and approximate methods of policy optimisation.

9.2.1 Updating the belief

Strictly speaking, in order to update our belief, we must condition the prior distribution on all the information. This includes the sequence of observations up to this point in time, including the states s^t, actions a^{t-1}, and rewards r^{t-1}, as well the policy π that we followed. Let $D_t = \left\langle s^t, a^{t-1}, r^{t-1} \right\rangle$ be the observed data to time t. Then the posterior measure for any measurable subset $B \subset \mathcal{M}$ is:

$$\xi(B \mid D_t, \pi) = \frac{\int_B \mathbb{P}_\mu^\pi(D_t) \, \mathrm{d}\xi(\mu)}{\int_{\mathcal{M}} \mathbb{P}_\mu^\pi(D_t) \, \mathrm{d}\xi(\mu)}. \tag{9.2.2}$$

However, as we shall see in the following remark, we can usually[1] ignore the policy itself when calculating the posterior.

Remark 9.2.1. The dependence on the policy can be removed, since the posterior is the same for all policies that put non-zero mass on the observed data: Let $D_t \sim \mathbb{P}_\mu^\pi$. Then it is easy to see that $\forall \pi' \neq \pi$ such that $\mathbb{P}_\mu^{\pi'}(D_t) > 0$,

$$\xi(B \mid D_t, \pi) = \xi(B \mid D_t, \pi').$$

The proof is left as an exercise for the reader. In the specific case of MDPs, the posterior calculation is easy to perform incrementally. This also more clearly demonstrates why there is no dependence on the policy. Let ξ_t be the (random)

[1]The exception involves any type of inference where $\mathbb{P}_\mu^\pi(D_t)$ is not directly available. This includes methods of approximate Bayesian computation Csilléry et al. [2010], so that trajectories from past policies are used to approximate it. See Dimitrakakis and Tziortziotis [2013] for an example of this in reinforcement learning.

posterior at time t. Then, the next-step belief is going to be:

$$\begin{aligned}\xi_{t+1}(B) \triangleq \xi(B \mid D_{t+1}) &= \frac{\int_B \mathbb{P}_\mu^\pi(D_t)\,\mathrm{d}\xi(\mu)}{\int_{\mathcal{M}} \mathbb{P}_\mu^\pi(D_t)\,\mathrm{d}\xi(\mu)} \\ &= \frac{\int_B \mathbb{P}_\mu(s_{t+1}, r_t \mid s_t, a_t)\pi(a_t \mid s^t, a^{t-1}, r^{t-1})\,\mathrm{d}\xi(\mu \mid D_t)}{\int_{\mathcal{M}} \mathbb{P}_\mu(s_{t+1}, r_t \mid s_t, a_t)\pi(a_t \mid s^t, a^{t-1}, r^{t-1})\,\mathrm{d}\xi(\mu \mid D_t)} \\ &= \frac{\int_B \mathbb{P}_\mu(s_{t+1}, r_t \mid s_t, a_t)\,\mathrm{d}\xi_t(\mu)}{\int_{\mathcal{M}} \mathbb{P}_\mu(s_{t+1}, r_t \mid s_t, a_t)\,\mathrm{d}\xi_t(\mu)}.\end{aligned}$$

The above calculation is easy to perform for arbitrarily complex MDPs when the set $\mathcal{M}$ is finite. The posterior calculation is also simple under certain conjugate priors, such as the Dirichlet-multinomial prior for transition distributions.

9.2.2 Finding Bayes-optimal policies

The problem of policy optimisation in the Bayesian case is much harder than when the MDP is known. This is simply because of the history dependence of the policies we have to consider. This makes the policy space much larger, as we need to consider history dependent policies.

In this section, we first consider two simple heuristics for finding sub-optimal policies. Then we examine policies which construct upper and lower bounds on the expected utility. Finally, we consider finite look ahead backwards induction, that uses the same upper and lower bounds to perform efficient tree search.

The expected MDP heuristic

One simple heuristic is to simply calculate the expected MDP $\widehat{\mu}(\xi) \triangleq \mathbb{E}_\xi\, \mu$ for the belief ξ. In particular, the transition kernel of the expected MDP is simply the expected transition kernel:

$$\mathbb{P}_{\widehat{\mu}(\xi)}(s'|s,a) = \int_{\mathcal{M}} \mathbb{P}_\mu(s'|s,a)\,\mathrm{d}\xi(\mu).$$

Then, we simply calculate the optimal memoryless policy for $\widehat{\mu}(\xi)$:

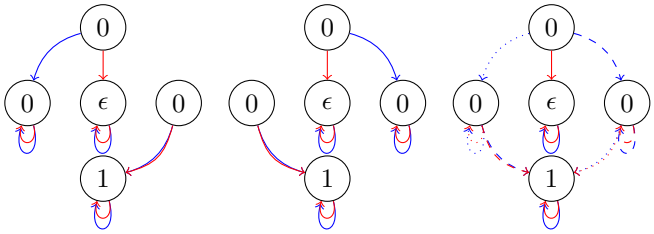
$$\pi^*(\widehat{\mu}(\xi)) \in \arg\max_{\pi \in \Pi_1} V^\pi_{\widehat{\mu}(\xi)},$$

where $\Pi_1 = \left\{\pi \in \Pi \;\middle|\; \mathbb{P}^\pi(a_t \mid s^t, a^{t-1}) = \mathbb{P}^\pi(a_t \mid s_t)\right\}$ is the set of Markov policies. Finally, we execute $\pi^*(\widehat{\mu}(\xi))$ on the real MDP. The algorithm can be written as follows. Algorithm 24 gives the pseudocode for this heuristic. One important detail is that we are only generating the k-th policy at step T_k. This is sometimes useful to ensure policies remain consistent, as small changes in the mean MDP may create a large change in the resulting policy. It is natural to have $T_k - T_{k-1}$ in the order of $1/1-\gamma$ for discounted problems, or simply the length of the episode for episodic problems. In the undiscounted case, switching policies whenever sufficient information has been obtained to significantly change the belief gives good regret guarantees, as we shall see in Chapter 10.

Unfortunately, the policy returned by this heuristic may be far from the Bayes-optimal policy in Π_1, as shown by the following example.

Algorithm 24 The expected MDP heuristic

for $k = 1, \ldots$ **do**
  $\pi_k \approx \arg\max_\pi \mathbb{E}^\pi_{\widehat{\mu}(\xi_{T_k})} U$.
  for $t = T_{k-1} + 1, \ldots, T_k$ **do**
    Observe s_t.
    Update belief $\xi_t(\cdot) = \xi_{t-1}(\cdot \mid s_t, a_{t-1}, r_{t-1}, s_{t-1})$.
    Take action $a_t \sim \pi_k(a_t \mid s_t)$.
    Observe reward r_t.
  end for
end for



(a) μ_1 (b) μ_2 (c) $\widehat{\mu}(\xi)$

Figure 9.2: The two MDPs and the expected MDP from Example 44

.

EXAMPLE 44 (Counterexample[2]). In this example, illustrated in Figure 9.2, $\mathcal{M} = \{\mu_1, \mu_2\}$ is the set of MDPs, and the belief is $\xi(\mu_1) = \theta$, $\xi(\mu_2) = 1 - \theta$. All transitions are deterministic, and there are two actions, the blue and the red action. When we calculate the expected MDP, we see that now the state with reward 1 is reachable, and that $\widehat{\mu}(\xi) \notin \mathcal{M}$. One can compute that even when when $T \to \infty$, the $\widehat{\mu}(\xi)$-optimal policy is not optimal in Π_1 if:

$$\epsilon < \frac{\gamma\theta(1-\theta)}{1-\gamma}\left(\frac{1}{1-\gamma\theta} + \frac{1}{1-\gamma(1-\theta)}\right)$$

9.2.3 The maximum MDP heuristic

An alternative idea is to simply pick the maximum-probability MDP, as shown in Algorithm 25. This at least guarantees that the MDP for which one chooses the optimal policy is actually within the set of MDPs. However, it may still be the case that the resulting policy is sub-optimal, as shown by the following example.

EXAMPLE 45 (Counterexample for $\widehat{\mu}^*(\xi) \triangleq \arg\max_\mu \xi(\mu)$). Let the MDP set (illustrated in Figure 9.3 be $\mathcal{M} = \{\mu_i \mid i = 1, \ldots, n\}$ with $\mathcal{A} = \{0, \ldots, n\}$. In all MDPs, action 0 gives a reward of ϵ. In the i-th MDP, the i-th gives you a reward of 1, and all remaining actions give a reward of 0. For any action a, the MDP terminates after an action is chosen and the reward received.

If $\xi(\mu_i) < \epsilon$ for all i, then it is optimal to choose action 0, while the max heuristic would pick the sub-optimal $\max_i \xi(\mu_i)$.

[2]Based on an example of Remi Munos.

Algorithm 25 The maximum MDP heuristic

for $k = 1, \dots$ **do**
  $\pi_k \approx \arg\max_\pi \mathbb{E}^\pi_{\hat{\mu}^*(\xi(T_k))} U$.
  for $t = 1 + T_{k-1}, \dots, T_k$ **do**
    Observe s_t.
    Update belief $\xi_t(\cdot) = \xi_{t-1}(\cdot \mid s_t, a_{t-1}, r_{t-1}, s_{t-1})$.
    Take action $a_t \sim \pi_k(a_t \mid s_t)$.
    Observe reward r_t.
  end for
end for

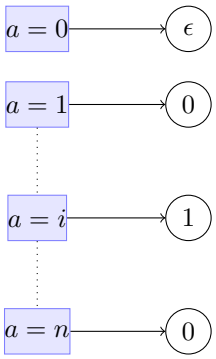


Figure 9.3: The MDP μ_i from Example 45

9.2.4 Bounds on the expected utility

Another approach is to try finding a policy that directly maximises a bound on the Bayes-expected utility. This is certainly a more principle approach, and so also affords us some performance guarantees. Before we discuss these bounds, we shall first consider how to calculate the expected utility of an arbitrary policy, which is a good step towards obtaining lower bounds. As it turns out, this operation is relatively simple in the Bayesian case, even when the set of MDPs is infinite.

Policy evaluation is particularly simple in Bayesian MDP problems for any fixed policy. We simply apply the basic utility theory definitions in order to calculate the expected utility of the policy under our belief. In the following, we will find it useful to also define the the Bayes-value function of a policy π as the conditional expected utility under that policy and our belief ξ. Analogously to an MDP value function, this is defined as follows.

Definition 9.2.1 (Bayesian value function π for a belief ξ)**.**

$$V^\pi_\xi(s) \triangleq \mathbb{E}^\pi_\xi(U \mid s).$$

It is easy to see that the Bayes value function of a policy is simply the expected value function under ξ:

$$V^\pi_\xi(s) = \int_{\mathcal{M}} \mathbb{E}^\pi_\mu(U \mid s) \, \mathrm{d}\xi(\mu) = \int_{\mathcal{M}} V^\pi_\mu(s) \, \mathrm{d}\xi(\mu).$$

However, the Bayes-optimal value function is not equal to the expected value

Algorithm 26 Bayesian Monte-Carlo policy evaluation

input policy π, belief ξ
for $k = 1, \dots, K$ **do**
  $\mu_k \sim \xi$
  $\boldsymbol{v}_k = V^\pi_{\mu_k}$
end for
$\boldsymbol{u} = \frac{1}{K} \sum_{k=1}^K \boldsymbol{v}_k$.
return $\boldsymbol{u}$.

function of the optimal policy for each MDP. In fact, the Bayes-value of any policy is a natural lower bound on the Bayes-optimal value function, as the Bayes-optimal policy is the maximum by definition. We can however use the expected optimal value function as an upper bound on the Bayes-optimal value:

$$V^*_\xi \triangleq \sup_\pi \mathbb{E}^\pi_\xi(U) = \sup_\pi \int_{\mathcal{M}} \mathbb{E}^\pi_\mu(U) \, \mathrm{d}\xi(\mu) \tag{9.2.3}$$

$$\leq \int_{\mathcal{M}} \sup_\pi V^\pi_\mu \, \mathrm{d}\xi(\mu) = \int_{\mathcal{M}} V^*_\mu \, \mathrm{d}\xi(\mu) \triangleq V^+_\xi \tag{9.2.4}$$

Algorithm 27 Bayesian Monte-Carlo upper bound

input belief ξ
for $k = 1, \dots, K$ **do**
  $\mu_k \sim \xi$
  $\boldsymbol{v}_k = V^*_{\mu_k}$
end for
$\boldsymbol{u}^* = \frac{1}{K} \sum_{k=1}^K \boldsymbol{v}_k$
return $\boldsymbol{u}^*$

Given the previous development, it is easy to see that the following inequalities always hold, giving us upper and lower bounds on the value function:

$$V^\pi_\xi \leq V^*_\xi \leq V^+_\xi, \qquad \forall \pi. \tag{9.2.5}$$

These bounds are geometrically demonstrated in Fig. 9.4. They are entirely analogous to the Bayes bounds of Sec. 3.3.1, with the only difference being that we are now considering complete policies rather than simple decisions.

9.2.5 Tighter lower bounds

A lower bound on the value function is useful to tell us how tight our upper bounds are. It is possible to obtain one by evaluating any arbitrary policy. So, tighter lower bounds can be obtained by finding better policies, something that was explored in Dimitrakakis [2011].

In particular, we can consider the problem of finding the best memoryless policy. This involves two approximations. Firstly, approximating our belief over MDPs with a sample over a finite set of n MDPs. Secondly, assuming that the belief is nearly constant over time, and performing backwards induction those n MDPs simultaneously. While this greedy procedure might not find the optimal memoryless policy, it still improves the lower bounds considerably.

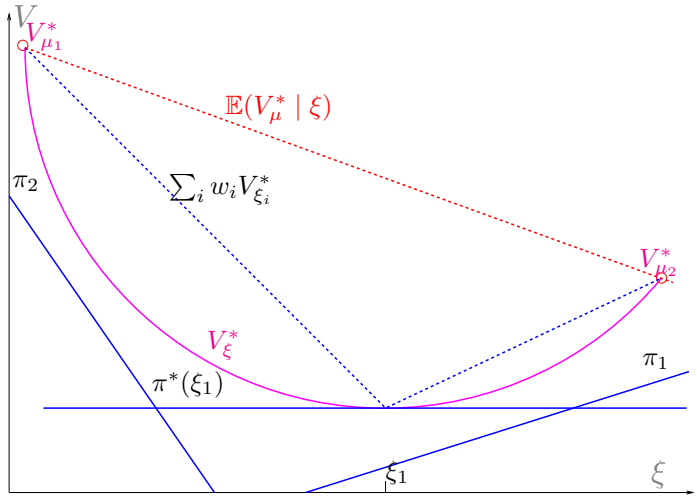


Figure 9.4: A geometric view of the bounds

The central step backwards induction over multiple MDPs is summarised by the following equation, which simply involves calculating the expected utility of a particular policy over all MDPs.

$$Q^\pi_{\xi,t}(s,a) \triangleq \int_{\mathcal{M}} \left\{ \bar{r}_\mu(s,a) + \gamma \int_{\mathcal{S}} V^\pi_{\mu,t+1}(s') \, \mathrm{d}P_\mu(s' \mid s,a) \right\} \mathrm{d}\xi(\mu) \tag{9.2.6}$$

The algorithm greedily performs backwards induction as shown in Algorithm 28. However, this is not an optimal procedure, since the belief at any time-step t is not constant. Indeed, even though the policy is memoryless, $\xi(\mu \mid s_t, \pi) \neq \xi(\mu \mid s_t, \pi')$. This is because the probability of being at a particular state is different under different policies and at different time-steps (e.g. if you consider periodic MDPs). For the same reason, this type of backwards induction may not converge in the manner of value iteration, but can exhibit cyclic convergence similar to the cyclic equilibria in Markov games [Zinkevich et al., 2006].

In practice, we maintain a belief over an infinite set of MDPs, such as the class of all discrete MDPs with a certain number of state and actions. In order to apply this idea in practice, we can sample a finite number of MDPs from the current belief and then find the optimal policy for this sample, as shown in Algorithm 29. For $n = 1$, this method is equivalent to Thompson sampling [Thompson, 1933], which was first used in the context of Bayesian reinforcement learning by Strens [2000]. Even though Thompson sampling is good exploration heuristic with formal performance guarantees Kaufmanna et al. [2012], Osband et al. [2013], it is not optimal. In fact, as we can see in Figure 9.6, Algorithm 29 performs better when the number of samples is increased.

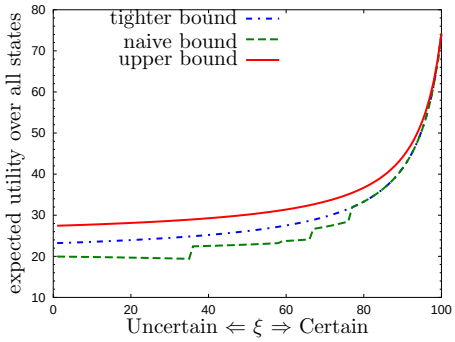
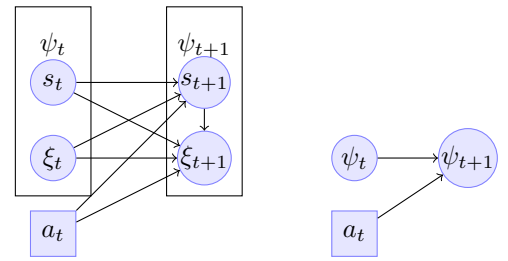


Figure 9.5: Illustration of the improved bounds. The naive and tighter bound refer to the lower bound obtained by calculating the value of the policy that is optimal for the expected MDP and that obtained by calculating the value of the MMBI policy respectively. The upper bound is V_ξ^+. The horizontal axis refers to our belief: At the left edge, our belief is uniform over all MDPs, while on the right edge, we are certain about the true MDP.



(a) The complete MDP model

(b) Compact form of the model

Figure 9.7: Belief-augmented MDP

9.2.6 The Belief-augmented MDP

The most direct way to actually solve the Bayesian reinforcement learning problem of (9.2.1) is to cast it as a yet another MDP. We already saw how this can be done with bandit problems in Section 6.2.2, but we shall now see that the general methodology is also applicable to MDPs.

We are given an initial belief ξ_0 on a set of MDPs $\mathcal{M}$. Each $\mu \in \mathcal{M}$ is a tuple $(\mathcal{S}, \mathcal{A}, P_\mu, \rho)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel P_μ and reward function $\rho : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Let s_t, a_t, r_t be the state and action observed in the original MDP and let ξ_t to denote our belief over MDPs $\mu \in \mathcal{M}$ at time t. Note that the marginal next-state distribution is

$$P(s_{t+1} \in S \mid \xi_t, s_t, a_t) \triangleq \int_{\mathcal{M}} P_\mu(s_{t+1} \in S \mid s_t, a_t) \, \mathrm{d}\xi_t(\mu), \tag{9.2.7}$$

while the next belief deterministically depends on the next state:

$$\xi_{t+1}(\cdot) \triangleq \xi_t(\cdot \mid s_{t+1}, s_t, a_t). \tag{9.2.8}$$

Algorithm 28 Multi-MDP backwards induction

input $\mathcal{M}, \xi, \gamma, T$
Set $V^\pi_{\mu,T+1}(s) = 0$ for all $s \in \mathcal{S}$.
for $t = T, T-1, \ldots, 0$ **do**
  for $s \in \mathcal{S}, a \in \mathcal{A}$ **do**
    Calculate $Q^\pi_{\xi,t}(s,a)$ from (9.2.6) using $\{V^\pi_{\mu,t+1}\}$.
  end for
  for $s \in \mathcal{S}$ **do**
    $\pi_t(s) \in \arg\max_{a\in\mathcal{A}} Q_{\xi,t}(s,a)$.
    for $\mu \in \mathcal{M}$ **do**
      $V^\pi_{\mu,t}(s) = Q^\pi_{\mu,t}(s, \pi_t(s))$.
    end for
  end for
end for
return π, Q_ξ.

Algorithm 29 Monte Carlo Bayesian Reinforcement Learning

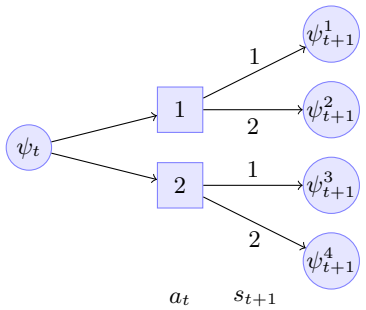
for Epochs $i = 1, \ldots$ **do**
  At the start-time t_i of the epoch, sample n MDPs $\mu_1, \ldots, \mu_n$ from ξ_{t_i}.
  Calculate the best memoryless policy $\pi_i \approx \arg\max_{\pi\in\Pi_1} \sum_{k=1}^n V^\pi_\mu$ wrt the sample.
  Execute π_i until $t = t_{i+1}$.
end for

We now construct the following augmented Markov decision process: $(\Psi, \mathcal{A}, P, \rho')$, with state space $\Psi = \mathcal{S} \times \Xi$ being the product of the original MDP states $\mathcal{S}$ and possible beliefs Ξ. It has a corresponding transition distribution:

$$P(\psi_{t+1} \mid \psi_t, a_t) = P(\xi_{t+1}, s_{t+1} \mid \xi_t, s_t, a_t) \tag{9.2.9}$$
$$= P(\xi_{t+1} \mid \xi_t, s_{t+1}, s_t, a_t) P(s_{t+1} \mid \xi_t, s_t, a_t), \tag{9.2.10}$$

where $P(\xi_{t+1} \mid \xi_t, s_{t+1}, s_t, a_t)$ is the singular distribution centred on the posterior distribution $\xi_t(\cdot \mid s_{t+1}, s_t, a_t)$. This construction is illustrated in Figure 9.7. The optimal policy for the augmented MDP is the ξ-optimal for the original problem. The augmented MDP has a pseudo-tree structure (since belief states might repeat), as shown in the example below.

Example 46. Consider an MDP family $\mathcal{M}$ with $\mathcal{A} = \{1, 2\}$, $\mathcal{S} = \{1, 2\}$. Then, for any hyper-state $\psi_t = (s_t, \xi_t)$, we can graphically represent the expansion as seen below:



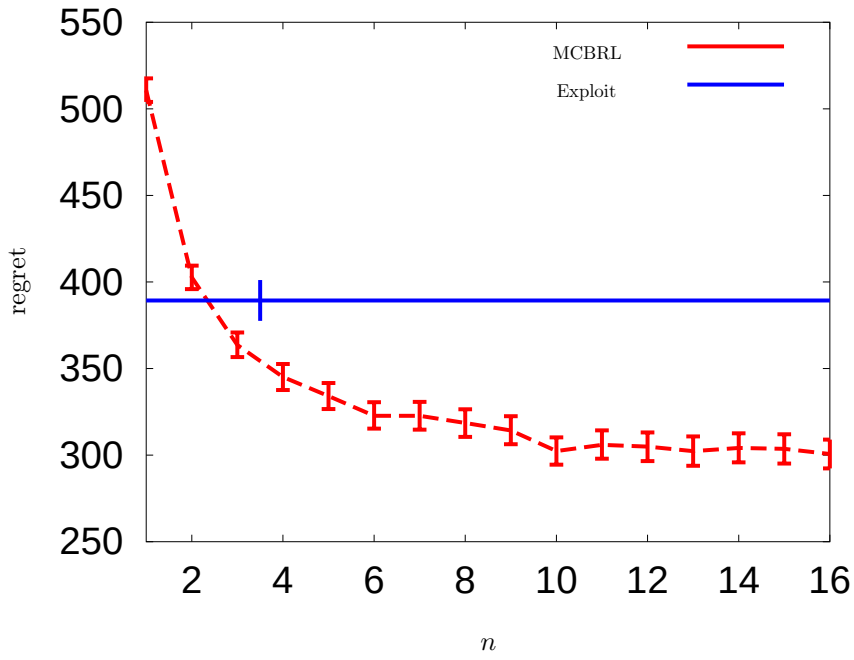


Figure 9.6: Comparison of the regret between the expected MDP heuristic and sampling with Multi-MDP backwards induction for the Chain environment. The error bars show the standard error of the average regret.

In this example, each possible action-state transition results in one specific hyper-state. As the diagram above shows, each possible action a_t, and each possible transition to a new state s_{t+1} leads to a new possible hyperstate ψ_{t+1}. When the branching factor is very large, or when we need to deal with very large tree depths, it becomes necessary to approximate the MDP structure.

9.2.7 Branch and bound

Branch and bound is a general technique for solving large problems. It can be applied in all cases where upper and lower bounds on the value of solution sets can be found. For Bayesian reinforcement learning, we can consider upper and lower bounds $\boldsymbol{q}^+$ and $\boldsymbol{q}^-$ such that:

$$\boldsymbol{q}^+(\psi, a) \geq Q^*(\psi, a) \geq \boldsymbol{q}^-(\psi, a) \tag{9.2.11}$$

$$\boldsymbol{v}^+(\psi) = \max_{a \in \mathcal{A}} \boldsymbol{q}^+(\psi, a), \qquad \boldsymbol{v}^-(\psi) = \max_{a \in \mathcal{A}} \boldsymbol{q}^-(\psi, a). \tag{9.2.12}$$

Let us now consider an incremental expansion of the belief-augmented MDP (BAMDP) so that, starting from some hyperstate ψ_t, we create the BAMDP tree for all subsequent states $\psi_{t+1}, \psi_{t+2}, \ldots$. For any leaf node $\psi_{t'} = (\xi_{t'}, s_{t'})$ in the tree, we can define upper and lower value function bounds according to

(9.2.5):

$$\boldsymbol{v}^-(\psi_{t'}) = V^{\pi(\xi_{t'})}_{\xi_{t'}}(s_{t'}), \qquad \boldsymbol{v}^+(\psi_{t'}) = V^{+}_{\xi_{t'}}(s_{t'}),$$

where $\pi(\xi_{t'})$ can be any approximately optimal policy for $\xi_{t'}$ Using backwards induction, we can calculate tighter upper $\boldsymbol{q}^+$ and lower bounds $\boldsymbol{q}^-$ for all all non-leaf hyperstates, as shown below.

$$\boldsymbol{q}^+(\psi_t, a_t) = \sum_{\psi_{t+1}} P(\psi_{t+1} \mid \psi_t, a_t) \left[\rho(\psi_t, a_t) + \gamma \boldsymbol{v}^+(\psi_{t+1})\right] \tag{9.2.13}$$

$$\boldsymbol{q}^-(\psi, a_t) = \sum_{\psi'} P(\psi_{t+1} \mid \psi_t, a_t) \left[r(\psi, a_t) + \gamma \boldsymbol{v}^-(\psi')\right] \tag{9.2.14}$$

We can then use the upper bounds expand the tree (e.g. selecting actions in the tree that maximise $\boldsymbol{v}^+$) while the lower bounds can be used to select the final policy. Sub-optimal branches can be discarded once their upper bounds become lower than the lower bound of some other branch.

Remark 9.2.2. If $\boldsymbol{q}^-(\psi, a) \geq \boldsymbol{q}^+(\psi, b)$ then b is sub-optimal at ψ.

However, such an algorithm is only possible to implement when the number of possible MDPs and states are finite. We can generalise this to the infinite case, by applying *stochastic* branch and bound methods Dimitrakakis [2010b, 2008]. This involves estimating upper and lower bounds on the values of leaf nodes through Monte-Carlo sampling.

9.2.8 Further reading.

One of the first treatments of this idea was due to Bellman [1957]. Although the idea was well-known in the statistical community [DeGroot, 1970], the popularisation of the idea in reinforcement learning was achieved with Duff's thesis [Duff, 2002]. Most recent advances in this area involve the use of intelligent methods for exploring the tree, such as sparse sampling [Wang et al., 2005] and Mote-Carlo tree search [Veness et al., 2009].

Instead of sampling MDPs, one could sample beliefs. This would lead to a finite hyper-state approximate of the complete belief MDP. One such approach is BEETLE [Poupart et al., 2006, Poupart and Vlassis, 2008] is a belief-sampling approach. It examines a set of possible future beliefs and approximates the value of each belief with a lower bound. In essence, it then creates the set of policies which are optimal with respect to these bounds.

Another idea is to take advantage of the expectation-maximisation view of reinforcement learning [Toussaint et al., 2006]. This allows us to apply a host of different probabilistic inference algorithms. This approach was investigated by Furmston and Barber [2010].

9.3 Bayesian methods in continuous spaces

Formally, Bayesian reinforcement learning in continuous state spaces is not significantly different from the discrete case. Typically, we assume that the agent acts within a fully observable discrete-time Markov decision process (MDP), with a metric state space $\mathcal{S}$, for example $\mathcal{S} \subset \mathbb{R}^d$. The action space $\mathcal{A}$ itself can

be either discrete or continuous. We can know define the transition kernel as a collection of probability measureson the continuous state space, indexed by $(\boldsymbol{s}, a)$

$$P_\mu(S \mid \boldsymbol{s}, a) \triangleq \mathbb{P}_\mu(\boldsymbol{s}_{t+1} \in S \mid \boldsymbol{s}_t = \boldsymbol{s}, a_t = a). \qquad S \subset \mathcal{S}.$$

There are a number of transition models we could consider for the continuous case. For the purposes of this textbook, we shall limit ourselves to the relatively simple case of linear-Gaussian models.

9.3.1 Linear-Gaussian transition models.

The simplest type of transition model for an MDP defined on a continuous state space is a linear-Gaussian model, which also results in a closed form posterior calculation due to the conjugate prior. While typically the real system dynamics may not be linear, one can usually find some mapping $f : \mathcal{S} \to \mathcal{X}$ to a k-dimensional vector space $\mathcal{X}$ such that the transformed state at time t is $\boldsymbol{x}_t \triangleq f(\boldsymbol{s}_t)$, whose dynamics may be well-approximated by a linear system. In this case, the next state $\boldsymbol{s}_{t+1}$ is given by the output of a function $g : \mathcal{X} \times \mathcal{A} \to \mathcal{S}$ of the transformed state, the action and some additive noise $\boldsymbol{\varepsilon}_t$:

$$\boldsymbol{s}_{t+1} = g(\boldsymbol{x}_t, a_t) + \boldsymbol{\varepsilon}_t.$$

design matrices
covariance

When g is linear and $\boldsymbol{\varepsilon}_t$ is normal, the this corresponds to a multivariate linear-Gaussian model. In particular, we can parametrise g with a set of $k \times k$ *design matrices* $\{\boldsymbol{A}_i \mid i \in \mathcal{A}\}$, such that $g(\boldsymbol{x}_t, a_t) = \boldsymbol{A}_{a_t}\boldsymbol{x}_t$. We can also define a set of *covariance* matrices $\{\boldsymbol{V}_i \mid i \in \mathcal{A}\}$ for the noise distribution. Then, the next state distribution is:

$$\boldsymbol{s}_{t+1} \mid \boldsymbol{x}_t = \boldsymbol{x}, a_t = i \sim \mathcal{N}(\boldsymbol{A}_i\boldsymbol{x}, \boldsymbol{V}_i),$$

i.e. the next state is drawn from a normal distribution with mean $\boldsymbol{A}_i\boldsymbol{x}$ and covariance matrix $\boldsymbol{V}_i$.

In order to model our uncertainty with a (subjective) prior distribution ξ, we have to specify the model structure. Fortunately, in this particular case, a conjugate prior exists in the form of the *matrix-normal distribution* for $\boldsymbol{A}$ and the *inverse-Wishart* distribution for $\boldsymbol{V}$. Given $\boldsymbol{V}_i$, the distribution for $\boldsymbol{A}_i$ is matrix-normal, while the marginal distribution of $\boldsymbol{V}_i$ is inverse-Wishart. More specifically,

$$\boldsymbol{A}_i \mid \boldsymbol{V}_i = \widehat{\boldsymbol{V}} \sim \phi(\boldsymbol{A}_i \mid \boldsymbol{M}, \boldsymbol{C}, \widehat{\boldsymbol{V}}) \tag{9.3.1}$$

$$\boldsymbol{V}_i \sim \psi(\boldsymbol{V}_i \mid \boldsymbol{W}, n), \tag{9.3.2}$$

where ϕ_i is the prior distribution on dynamics matrices conditional on the covariance and two prior parameters: $\boldsymbol{M}$, which is the prior mean and $\boldsymbol{C}$ which is the prior output (dependent variable) covariance. Finally, ψ is the marginal prior on covariance matrices, which has an inverse-Wishart distribution with $\boldsymbol{W}$ and n. More precisely, the distributions are:

$$\phi(\boldsymbol{A}_i \mid \boldsymbol{M}, \boldsymbol{C}, \widehat{\boldsymbol{V}}) \propto e^{-\frac{1}{2}\operatorname{trace}\left[\boldsymbol{P}(\boldsymbol{A}_i - \boldsymbol{M})\boldsymbol{V}_i^{-1}(\boldsymbol{A}_i - \boldsymbol{M})\boldsymbol{C}\right]}, \tag{9.3.3}$$

$$\psi(\boldsymbol{V}_i \mid \boldsymbol{W}, n) \propto |\boldsymbol{V}^{-1}\boldsymbol{W}/2|^{n/2} e^{-\frac{1}{2}\operatorname{trace}(\boldsymbol{V}^{-1}\boldsymbol{W})}. \tag{9.3.4}$$

Essentially, the model is an extension of the univariate Bayesian linear regression model (see for example DeGroot [1970]) to the multivariate case via vectorisation of the mean matrix. Since the prior is conjugate, it is relatively simple to calculate posterior values of the parameters after each observation. While we omit the details, a full description of inference using this model is given in Minka [2001b].

Further reading. More complex transition models include the non-parametric extension of the above model, *Gaussian processes* (GP) [Rasmussen and Williams, 2006]. For an n-dimensional state space, GPs are typically applied by using independent GP for predicting each state dimension of the state space, i.e. $f_i : \mathbb{R}^n \to \mathbb{R}$. As this completely decouples the state dimensions, it's best to consider a joint model, but this requires various approximations (e.g. [Álvarez et al., 2010]). A well-known method for model-based Gaussian process reinforcement learning is GP-Rmax[Jung and Stone, 2010], which has been recently shown by Grande et al. [2014] to be KWIK-learnable.[3]

Gaussian processes

Another straightforward extension of linear models are piecewise linear models, which can be described in a Bayesian non-parametric frameworkTziortziotis et al. [2014]. This avoids the computational complexity of GPs.

9.3.2 Approximate dynamic programming

Bayesian methods are also frequently used as part of a dynamic programming approach. Typically, this requires maintaining a distribution over value functions in some sense. This generally involves defining some generative model over which inference can be performed. For continuous state spaces particularly, one can assume that the value function $\boldsymbol{v}$ is drawn from a Gaussian process. However, to perform inference we also need to specify some generative model for the observations.

Temporal differences. Engel et al. [2003] consider temporal differences from a Bayesian perspective in conjunction with a GP model, so that the rewards are distributed as follows:

$$r_t \mid \boldsymbol{v}, s_t, s_{t+1} \sim \mathcal{N}(\boldsymbol{v}(s_{t+1}) - \gamma \boldsymbol{v}(s_t), \sigma),$$

which essentially gives a simple model for $P(r^T|\boldsymbol{v}, s^T)$. We can now write the posterior as $\xi(\boldsymbol{v} \mid r^T, s^T) \propto P(r^T \mid \boldsymbol{v}, s^T)\xi(\boldsymbol{v})$, where the dependence $\xi(\boldsymbol{v}|s^T)$ is suppressed. This model was later updated in Engel et al. [2005], with the following reward distribution

$$r_t \mid \boldsymbol{v}, s_t, s_{t+1} \sim \mathcal{N}(\boldsymbol{v}(s_t) - \gamma \boldsymbol{v}(s_{t+1}), N(s_t, s_{t+1})).$$

The main part of the model is $N(s, s') \triangleq \Delta_U(s) - \gamma \Delta_U(s'))$, where the $\Delta_U(s) \triangleq U(s) - \boldsymbol{v}(s)$ denotes the distribution of the residual, i.e. the utility when starting from s minus its expectation. The correlation between $U(s)$ and $U(s')$ is captured via N, and the residuals are modelled as a Gaussian process. While the model is still an approximation, it is equivalent to performing GP regression using Monte-Carlo samples of the discounted return.

[3] Informally, a class is KWIK learnable if the number of mistakes made by the algorithm is polynomially bounded in the problem parameters. In the context of reinforcement learning this would be the number of steps for which no guarantee of utility can be provided.

Bayesian finite-horizon dynamic programming for deterministic systems. Instead of using an approximate model, Deisenroth et al. [2009] employ a series of GPs, each for one dynamic programming stage, under the assumption that the dynamics are deterministic and the rewards are Gaussian-distributed. It is possible to extend this approach to the case of non-deterministic transitions, at the cost of requiring additional approximations. However, since a lot of real-world problems do in fact have deterministic dynamics, the approach is consistent.

Bayesian least-squares temporal differences. Tziortziotis and Dimitrakakis [2017] instead consider a model for the value function itself, where the random quantity is the empirical transition matrix $\hat{P}$ rather than the reward (which can be assumed to be known):

$$\hat{P}\boldsymbol{v} \mid \boldsymbol{v}, P \sim \mathcal{N}(P\boldsymbol{v}, \beta I). \tag{9.3.5}$$

This model makes a different trade-off in its distributional assumptions. It allows us to model the uncertainty about P in a Bayesian manner, but instead of explicitly modelling this as a distribution on P itself, we are modelling a distribution on the resulting Bellman operator.

Gradient methods. Generally speaking, if we are able to sample from the posterior distribution, we can leverage stochastic gradient descent methods to extend any gradient algorithm for reinforcement learning with a given model to the Bayesian setting. More precisely, if we have a utility gradient $\nabla_\pi U(\mu, \pi)$ for model μ, then by linearity of expectations we obtain that

$$\nabla_\pi \mathbb{E}_\xi^\pi U = \int_{\Pi} \nabla_{\mathcal{M}} U(\mu, \pi) \, \mathrm{d}\xi(\mu)$$

and stochastic gradient descent can be implemented simply by sampling $\mu \sim \xi$ and updating the parameters using the gradient of the sampled MDP. This approach was originally suggested by Ghavamzadeh and Engel [2006].

9.4 Partially observable Markov decision processes

In most real world applications s_t, the state of the system at time t is not observed. Instead, we obtain some observation x_t, which depends on the state of the system. While this does give us some information about the system state, it nevertheless is not sufficient to pin-point it exactly. This idea can be formalised as a partially observable Markov decision process.

Definition 9.4.1 (Partially observable Markov decision process (POMDP))**.** A POMDP $\mu \in \mathcal{M}_P$ is a tuple $(\mathcal{X}, \mathcal{S}, \mathcal{A}, P, y)$ where $\mathcal{X}$ is an observation space, $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, and P is a conditional distribution on observations, states and rewards. The reward, observation and next state are Markov with respect to the current state and action. In this book, we shall assume the following dependencies:

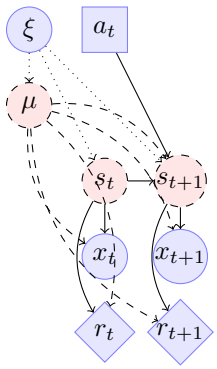
$$\mathbb{P}_\mu(s_{t+1}, r_t, x_t \mid s_t, a_t, \ldots) = P(s_{t+1} \mid s_t, a_t) P(x_t \mid s_t) P(r_t \mid s_t). \tag{9.4.1}$$

ition distribution

$P(s_{t+1} \mid s_t, a_t)$ is the *transition distribution*, giving the probabilities of next states given the current state and action. $P(x_t \mid s_t)$ is the *observation distribution*, giving the probabilities of different observations given the current state. Finally, $P(r_t \mid s_t)$ is the *reward distribution*, which we make dependent only on the current state for simplicity. Different dependencies are possible, but they are all equivalent to the one given here.

observation distribution
reward distribution

Partially observable Markov decision processes
The following graphical model illustrates the dependencies in a POMDP.



- The system *state* $s_t \in \mathcal{S}$ is not observed.
- We receive an *observation* $x_t \in \mathcal{X}$ and a *reward* $r_t \in \mathcal{R}$.
- We take *action* $a_t \in \mathcal{A}$.
- The system transits to state s_{t+1}.

9.4.1 Solving known POMDPs

When we know a POMDP's parameters, that is to say, when we know the transition, observation and reward distributions, the problem is formally the same as solving an unknown MDP. In particular, we can similarly define a *belief state* summarising our knowledge. This takes the form of a probability distribution on the hidden state variable s_t, rather than on the model μ. If μ defines starting state probabilities, then the belief is not subjective, as it only relies on the actual POMDP parameters. The transition distribution on states given our belief is as follows.

Belief ξ
For any distribution ξ on $\mathcal{S}$, we define:

$$\xi(s_{t+1} \mid a_t, \mu) \triangleq \int_{\mathcal{S}} P_\mu(s_{t+1} \mid s_t a_t) \, \mathrm{d}\xi(s_t) \tag{9.4.2}$$

When there is no ambiguity, we shall use ξ to denote arbitrary marginal distributions on states and state sequence given the belief ξ.

When the model μ is given, calculating a belief update is not particularly difficult, but we must take care to properly use the time index t. Starting from Bayes' theorem, it is easy to derive the belief update from ξ_t to ξ_{t+1} as follows,

Belief update

$$\xi_{t+1}(s_{t+1} \mid \mu) \triangleq \xi_t(s_{t+1} \mid x_{t+1}, r_{t+1}, a_t, \mu) \tag{9.4.3}$$

$$= \frac{P_\mu(x_{t+1}, r_{t+1} \mid s_{t+1}) \xi_t(s_{t+1} \mid a_t, \mu)}{\xi_t(x_{t+1} \mid a_t, \mu)} \tag{9.4.4}$$

$$\xi_t(s_{t+1} \mid a_t, \mu) = \int_{\mathcal{S}} P_\mu(s_{t+1} \mid s_t, a_t, \mu) \, \mathrm{d}\xi_t(s_t) \tag{9.4.5}$$

$$\xi_t(x_{t+1} \mid a_t, \mu) = \int_{\mathcal{S}} P_\mu(x_{t+1} \mid s_{t+1}) \, \mathrm{d}\xi_t(s_{t+1} \mid a_t, \mu) \tag{9.4.6}$$

A particularly attractive example is when the model is finite. Then the sufficient statistic also has finite dimension and all updates are in closed form.

Remark 9.4.1. If $\mathcal{S}, \mathcal{A}, \mathcal{X}$ are finite, and then we can define the sequence of vectors $\boldsymbol{p}_t \in \mathbb{\Delta}^{|\mathcal{S}|}$, matrices $\boldsymbol{A}_t$

- $\boldsymbol{p}_t(j) = P(x_t \mid s_t = j)$
- $\boldsymbol{A}_t(i, j) = P(s_{t+1} = j \mid s_t = i, a_t)$.
- $\boldsymbol{b}_t(i) = \xi_t(s_t = i)$

We can then use Bayes theorem:

$$\boldsymbol{b}_{t+1} = \frac{\operatorname{diag}(\boldsymbol{p}_{t+1}) \boldsymbol{A}_t \boldsymbol{b}_t}{\boldsymbol{p}_{t+1}^\top \boldsymbol{A}_t \boldsymbol{b}_t}, \tag{9.4.7}$$

Even though inference is tractable in finite models, there is a small number of cases

9.4.2 Solving unknown POMDPs

This is a much harder problem, unfortunately. Let us take a look at the basic update equation, where we need to define a joint belief on both possible states and possible models.

$$\xi(\mu, s^t \mid x^t, a^t) \propto P_\mu(x^t \mid s^t) P_\mu(s^t \mid a^t) \xi(\mu) \tag{9.4.8}$$

Unfortunately, even for the simplest possible case of two possible models μ_1, μ_2 and binary observations, there is no finite-dimensional representation of the belief at time t.

Strategies for solving unknown POMDPs include solving the full Bayesian decision problem, but this requires exponential inference and planning for exact solutions Ross et al. [2008]. For this reason, we must use approximations.

ble order Markov de-
n process

One very simple approximation involves replacing a partially observable Markov process with a *variable order Markov decision process*. Fortunately, inference in variable order Markov processes has only logarithmic computational complexityDimitrakakis [2010a]. Of course, the memory complexity is still linear.

In general, finding optimal controllers for POMDPs is hard even for restricted classes of policiesVlassis et al. [2012]. However, approximations Spaan and Vlassis [2005] and stochastic methods and policy search methodsBaxter and Bartlett [2000], Toussaint et al. [2006] work quite well in practice.

9.5 Exercises

Exercise 37. Consider the algorithms we have seen in Chapter 8. Are any of those applicable to belief-augmented MDPs? Outline a strategy for applying one of those algorithms to the problem. What would be the biggest obstacle we would have to overcome in your specific example?

Exercise 38. Prove Remark 9.2.1

Exercise 39. A practical case of Bayesian reinforcement learning in discrete spaces is when we have an independent belief over the transition probabilities of each state-action pair. Consider the case where we have n states and k actions. Similar to the product-prior in the bandit case in Section 6.2, we assign a probability (density) $\xi_{s,a}$ to the probability vector $\boldsymbol{\theta}_{(s,a)} \in \mathbb{\Delta}^n$. We can then define our joint belief on the $(nk) \times n$ matrix $\boldsymbol{\Theta}$ to be

$$\xi(\boldsymbol{\Theta}) = \prod_{s \in \mathcal{S}, a \in \mathcal{A}} \xi_{s,a}(\boldsymbol{\theta}_{(s,a)}).$$

(i) Derive the updates for a product-Dirichlet prior on transitions.

(ii) Derive the updates for and a product-Normal-Gamma prior on rewards.

(iii) What would be the meaning of using a Normal-Wishart prior on rewards?

Exercise 40. Consider the Gaussian process model of (9.3.2). What is the implicit assumption made about the transition model? If this assumption is satisfied, what does the corresponding posterior distribution represent?

Chapter 10

Regret bounds for reinforcement learning

10.1 Introduction

The Bayesian framework requires specifying a prior distribution. For many reasons, we may frequently be unable to do that. In addition, as we have seen, the Bayes-optimal solution is often intractable. In this chapter we shall take a look at algorithms that do not require specifying a prior distribution. Instead, they employ the heuristic of "optimism under uncertainty" to select policies. This idea is very similar to heuristic search algorithms, such as A^* [Hart et al., 1968]. All these algorithms assume the best possible model that is consistent with the observations so far and choose the optimal policy in this "optimistic" model. Intuitively, this means that for each possible policy we maintain an upper bound on the value/utility we can reasonably expect from it. In general we want this upper bound to

1. be as tight as possible (i.e., to be close to the true value),
2. still hold with high probability.

We begin with an introduction to these ideas in bandit problems, when the objective is to maximise total reward. We then expand this discussion to structured bandit problems, which have many applications in optimisation. Finally, we look at the case of maximising total reward in unknown MDPs.

10.2 Finite Stochastic Bandit problems

First of all, let us remind the reader of the stochastic bandit problem. The learner in each time step t chooses an *arm* a_t from a given set $\mathcal{A} = \{1, \dots, K\}$ of K arms. The expected reward for choosing each arm i is $\mu_i = \mathbb{E}(r_t | a_t = i)$ independent of the step t and unknown to the learner. Further, we can assume that the rewards are bounded, e.g. $r_t \in [0, 1]$. The goal is to maximise the total reward $\sum_{t=1}^{T} r_t$ after T time steps.

Let $\mu^* \triangleq \max_i \mu_i$ be the highest expected reward that can be achieved. Obviously, the optimal policy π^* in each time step chooses the arm giving the highest expected reward μ^*. The learner who does not know which arm is optimal will choose at each time step t an arm a_t from $\mathcal{A}$, or more generally, a probability distribution π_t over the arms from which a_t then is drawn. It is important to notice that maximising the total reward is equivalent to minimising total regret with respect to that policy.

Definition 10.2.1 (Total regret)**.** The *(total) regret* of a policy π relative to the optimal fixed policy π^* after T steps is

$$L_T(\pi) \triangleq \sum_{t=1}^{T} \left(r_t^* - r_t^\pi \right),$$

where r_t^π is the reward obtained by the policy π at step t and $r_t^* \triangleq r_t^{\pi^*}$. Accordingly, the *expected (total) regret* is

$$\mathbb{E}\, L_T(\pi) \triangleq T\mu^* - \mathbb{E}_\pi \sum_{t=1}^{T} r_t.$$

The regret compares the collected rewards to those of the best fixed policy. Comparing instead to the best rewards obtained by the arms at each time would be too hard, as these rewards are by definition unpredictable, which would make learning impossible.

[**ro:** I've skipped the proto-UCB algorithm you gave here, as I think it's not very interesting. In particular, the analysis for the deterministic case is not very meaningful in my view, as in this case the silly 'choose arm with best estimate'-algorithm would work even better.]

10.2.1 The UCB1 algorithm

It makes sense for a learning algorithm to use the empirical average rewards obtained for each arm so far.

Empirical average

$$\hat{\mu}_{t,i} \triangleq \frac{1}{N_{t,i}} \sum_{k=1}^{t} r_{k,i} \mathbb{I}\{a_k = i\}, \qquad \text{where } N_{t,i} \triangleq \sum_{k=1}^{t} \mathbb{I}\{a_k = i\}$$

and $r_{k,i}$ denotes the (random) reward the learner receives upon choosing arm i at step k.

Simply always choosing the arm with best the empirical average reward so far is not a very good idea, because you might get stuck with a sub-optimal arm: If the optimal arm underperforms at the beginning, so that its empirical average is far below the true mean of a suboptimal arm, it will never be chosen again. A better idea is to choose arms optimistically. Intuitively, as long as an arm has a significant chance of being the best, you play it every now and then. One simple way to implement this is shown in the following UCB1 algorithm [Auer et al., 2002a].

Algorithm 30 UCB1

Input $\mathcal{A}$
For all i, initialise $\hat{\mu}_{0,i} = 1$.
for $t = 1, \ldots$ **do**
  $a_t = \arg\max_{i \in \mathcal{A}} \left\{ \hat{\mu}_{t-1,i} + \sqrt{\frac{2 \ln t}{N_{t-1,i}}} \right\}$
end for

Thus, the algorithm adds a bonus value of order $O(\sqrt{\ln t / N_{t,i}})$ to the empirical value of each arnm thus forming *upper confidence bound*. This upper bound is such that by the Hoeffding bound (4.5.5), with high probability the true mean reward of each arm will be below the upper confidence bound value. *upper confidence bound*

Theorem 10.2.1 (Auer et al. [2002a])**.** *The expected regret of UCB1 after T rounds is at most*

$$\mathbb{E}\, L_T(UCB1) \leq \sum_{i:\mu_i < \mu^*} \frac{8 \ln T}{\mu^* - \mu_i} + 5 \sum_i (\mu^* - \mu_i).$$

Proof. By Wald's identity (5.2.13) the expected regret can be written as

$$\mathbb{E}\, L_T \;=\; \mathbb{E}\sum_{t=1}^{T}(\mu^* - r_t) \;=\; \sum_i \mathbb{E}\, N_{T,i}(\mu^* - \mu_i), \tag{10.2.1}$$

so that we focus on bounding $\mathbb{E}\, N_{t,i}$. Thus, let i be an arbitrary suboptimal arm and consider when it will be chosen by the algorithm. Write $B_{t,s} = \sqrt{(2\ln t)/s}$ for the bonus value at step t after s observations. Note that by the Hoeffding bound (4.5.5) for fixed values of $t, s, s_i \in \mathbb{N}$ under the assumption that $N_{t,i} = s_i$ and (the count of the optimal action) $N_{t,*} = s$, we have that

$$\mathbb{P}(\hat{\mu}_{t,i} \geq \mu_i + B_{t,s_i}) \leq e^{-4\ln t} = t^{-4}, \tag{10.2.2}$$

$$\mathbb{P}(\hat{\mu}^*_t \leq \mu^* - B_{t,s}) \leq e^{-4\ln t} = t^{-4}, \tag{10.2.3}$$

so that we may assume (we take care of the contribution of the error probabilities to $\mathbb{E}\, N_{t,i}$ below)

$$\hat{\mu}_{t,i} < \mu_i + B_{t,N_{t,i}}, \tag{10.2.4}$$

$$\mu^* < \hat{\mu}^*_t + B_{t,N_{t,*}}. \tag{10.2.5}$$

Now note that for $s \geq \lceil (8\ln T)/(\mu^* - \mu_i)^2 \rceil$ it holds that

$$2B_{t,s} \leq (\mu^* - \mu_i), \tag{10.2.6}$$

so that after arm i has been chosen $\lceil (8\ln T)/(\mu^* - \mu_i)^2 \rceil$ times we get from (10.2.4), (10.2.6), and (10.2.5) that

$$\begin{aligned} \hat{\mu}_{t,i} + B_{t,N_{t,i}} \;&<\; \mu_i + 2B_{t,N_{t,i}} \;\leq\; \mu^* \\ &<\; \hat{\mu}^*_t + B_{t,N_{t,*}}, \end{aligned}$$

showing that the algorithm won't choose arm i. Taking into account the error probabilities for (10.2.4) and (10.2.5) we might play arm i once whenever either equation does not hold. Thus, summing over all possible values for t, $N_{t,i}$ and $N_{t,*}$ this shows that

$$\mathbb{E}\, N_{t,i} \;\leq\; \left\lceil \frac{8\ln T}{(\mu^* - \mu_i)^2} \right\rceil + \sum_{\tau \geq 1} \sum_{s \leq \tau} \sum_{s_i \leq \tau} 2\tau^{-4}.$$

Combining this with (10.2.1) and noting that the sum converges to a value < 4, proves the regret bound. □

The UCB1 algorithm is actually not the first algorithm employing *optimism in the face of uncertainty* to deal with the exploration-exploitation dilemma, nor the first that uses confidence intervals for that purpose. This idea goes back to the seminal work of [Lai and Robbins, 1985] that used the same approach, however in a more complicated form. In particular, the whole history is used for computing the arm to choose. The derived bounds of Lai and Robbins [1985] show that after T steps each suboptimal arm is played at most $\left(\frac{1}{D_{\mathrm{KL}}} + o(1)\right)\log T$ times in expectation, where D_{KL} measures the distance between the reward distributions of the optimal and the suboptimal arm by the Kullback-Leibler divergence, and $o(1) \to 0$ as $T \to \infty$. This bound was also shown to be asymptotically

optimal by Lai and Robbins [1985]. A lower bound logarithmic in T for any finite T that is close to matching the bound of Theorem 10.2.1 can be found in [Mannor and Tsitsiklis, 2004]. Improvements that get closer to the lower bound (and are still based on the UCB1 idea) can be found in [Auer and Ortner, 2010].

For so-called distribution-independent bounds that do not depend on problem parameters like the 'gaps' $\mu^* - \mu_i$, see e.g. Audibert and Bubeck [2009]. In general, these bounds cannot be logarithmic in T anymore—as the gaps may be of order $1/\sqrt{T}$—and are $O(\sqrt{T})$.

10.2.2 Non i.i.d. Rewards

The stochastic setting just considered is only one among several variants of the multi-armed bandit problem. While it is impossible to cover them all, we give a brief of the most common scenarios and refer to Bubeck and Cesa-Bianchi [2012] for a more complete overview.

What is common to most variants of the classic stochastic setting is the assumption of receiving i.i.d. rewards when sampling a fixed arm is loosened. The most extreme case is the so-called *nonstochastic*, sometimes also termed *adversarial bandit* setting, where the reward sequence for each arm is assumed to be fixed in advance (and thus not random at all). In this case, the reward is maximised when choosing in each time step the arm that maximises the reward at this step. Obviously, since the reward sequences can be completely arbitrary, no learner can stand a chance to perform well with respect to this optimal policy. Thus, one confines oneself to consider the regret with respect to the best *fixed* arm in hindsight, that is, $\arg\max_i \sum_{t=1}^{T} r_{t,i}$ where $r_{t,i}$ is the reward of arm i at step t. It is still not clear that this is not too much to ask for, but it turns out that one can achieve regret bounds of order $O(\sqrt{KT})$ in this setting. Clearly, algorithms that choose arms deterministically can always be tricked by an adversarial reward sequence. However, algorithms that at each time step choose an arm from a suitable distribution over the arms (that is updated according to the collected rewards), such the Exp3 algorithm of [Auer et al., 2002b], or similar algorithms that use an exponential weighting scheme meet the mentioned upper bound on the regret, which can be shown to be optimal.

adversarial bandit

In the *contextual bandit* setting *bandit!contextual* the learner receives some additional side information called the *context*. The reward for choosing an arm is assumed to depend on the context as well as on the chosen arm and can be either stochastic or adversarial. The learner usually competes against the best policy that maps contexts to arms. There is a notable amount of literature dealing with various settings that are usually also interesting for applications like web advertisement where user data takes the role of provided side information. For an overview see e.g. Chapter 4 of Bubeck and Cesa-Bianchi [2012].

bandit!contextual

In other settings the i.i.d. assumption about the rewards of a fixed arm is replaced by more general assumptions, such as that underlying each arm there is a Markov chain and rewards depend on the state of the Markov chain when sampling the arm. This is called the *restless bandits* problem, that is already quite close to the general reinforcement learning setting with an underlying Markov decision process (see Section 10.3.2 below). Regret bounds in this setting can be shown to be $\tilde{O}(\sqrt{T})$ even if at each time step the learner can observe only the state of the arm he chooses, see [Ortner et al., 2014].

10.3 Reinforcement learning problems

10.3.1 Introduction

Now we want to take a step further from the bandit problems of the previous sections to the general reinforcement learning setting with an underlying MDP unknown to the learner. Note that the stochastic bandit problem corresponds to a single state MDP.

Thus, consider an MDP μ^* with state space $\mathcal{S}$, action space $\mathcal{A}$, and let $r(s,a) \in [0,1]$ and $P(\cdot|s,a)$ be the mean reward and the transition probability distribution on $\mathcal{S}$ for each state $s \in \mathcal{S}$ and each action $a \in \mathcal{A}$, respectively. For the moment we assume that $\mathcal{S}$ and $\mathcal{A}$ are finite. As we have seen in Section 6.6 there are various optimality criteria for MDPs. In the spirit of the bandit problems considered so far we consider undiscounted rewards and examine the regret after any T steps with respect to an optimal policy.

Since the optimal T-step policy in general will be non-stationary and different for different horizons T and different initial states, we will compare to a *gain optimal* policy π^*.[1] Further, we assume that the MDP is *communicating*, that is, for any two states s, s' there is a policy $\pi_{s,s'}$ that with positive probability reaches s' when starting in s and playing actions according to $\pi_{s,s'}$. This also means that when learning in the MDP we can always recover when making a mistake. Note that in MDPs that are not communicating one wrong step may lead to a suboptimal region of the state space that cannot be left anymore, which makes competing to an optimal policy in a learning setting impossible. For communicating MDPs we can define the diameter to be the maximal expected time it takes to connect any two states.

Definition 10.3.1. Let $T(\pi, s, s')$ the expected number of steps it takes to reach state s' when starting in s and playing policy π. Then the *diameter* is defined as

$$D \triangleq \max_{s,s'} \min_{\pi} T(\pi, s, s').$$

Given that our rewards are assumed to be bounded in $[0,1]$, intuitively, when we make one wrong step in some state s, in the long run we won't lose more than D. After all, in D steps we can go back to s and continue optimally.

Under the assumption that the MDP is communicating, the gain g^* can be shown to be independent of the initial state, that is, $g^*(s) = g^*$ for all states s. Then we define the T-step regret of a learning algorithm as

$$L_T \triangleq \sum_{t=1}^{T} \big(g^* - r_t\big),$$

where r_t is the reward collected by the algorithm at step t. Note that in general (and depending on the initial state) the value Tg^* we compare to will differ from the optimal T-step reward. However, this difference can be shown to be upper bounded by the diameter and is therefore negligible when considering the regret.

[1] See Def. 6.6.4.

10.3.2 An upper-confidence bound algorithm

Now we would like to extend the idea underlying the UCB1 algorithm to the general reinforcement learning setting. Again, we would like to have for each (stationary) policy π an upper bound on the gain that is reasonable to expect. Note that simply taking each policy to be the arm of a bandit problem does not work well. First, to approach the true gain of a chosen policy, it will not be sufficient to choose it just once. It would be necessary to follow each policy for a sufficiently high number of consecutive steps. Without knowledge of some characteristics of the underlying MDP like mixing times, it might be however difficult to determine how long a policy shall be played. Further, due to the large number of stationary policies, which is $|\mathcal{A}|^{|\mathcal{S}|}$, the regret bounds that would result from such an approach would be exponential in the number of states. Thus, we rather maintain confidence regions for the rewards and transition probabilities of each state-action pair s, a. Then, at each step t, these confidence regions implicitly define a confidence region for the true underlying MDP μ^*, that is, a set M_t of *plausible* MDPs. For suitably chosen confidence intervals for the rewards and transition probabilities one can obtain that

$$\mathbb{P}(\mu^* \notin M_t) < \delta. \tag{10.3.1}$$

Given this confidence region M_t, one can define the optimistic value for any policy π to be

$$g^\pi_+(M_t) \triangleq \max \left\{ g^\pi_\mu \;\middle|\; \mu \in M_t \right\}. \tag{10.3.2}$$

Note that similar to the bandit setting this estimate is optimistic for each policy, as due to (10.3.1) it holds that $g^\pi_+(M_t) \geq g^\pi_{\mu^*}$ with high probability. Analogously to UCB1 we would like to make an optimistic choice among the possible policies, that is, we choose a policy π that maximises $g^\pi_+(M_t)$.

However, unlike in the bandit setting where we immediately receive a sample from the reward of the chosen arm, in the MDP setting we only obtain information about the reward in the current state. Thus, we should not play the chosen optimistic policy just for one but a sufficiently large number of steps. An easy way is to play policies in episodes of increasing length, such that sooner or later each action is played for a sufficient number of steps in each state. Summarized, we obtain an algorithm as shown below.

UCRL2 Jaksch et al. [2010] outline

In episodes $k = 1, 2, \ldots$

- At the first step t_k of episode k, update the confidence region M_{t_k}.
- Compute an optimistic policy $\tilde{\pi}_k \in \arg\max_\pi g^\pi_+(M_{t_k})$.
- Execute $\tilde{\pi}_k$, observe rewards and transitions until t_{k+1}.

Technical details for UCRL2

To make the algorithm complete, we have to fill in some technical details. In the following, let S be the number of states and A the number of actions of the underlying MDP μ^*. Further, $\delta > 0$ is a confidence parameter of the algorithm.

The confidence region First, concerning the confidence regions, for the rewards it is sufficient to use confidence intervals similar to those for UCB1. For the transition probabilities we consider all those transition probability distributions to be plausible if their $\|\cdot\|_1$-norm is close to the empirical distribution $\hat{\boldsymbol{P}}_t(\cdot \mid s,a)$. That is, the confidence region M_t at step t used to compute the optimistic policy in each episode can be defined as the set of MDPs with mean rewards $r(s,a)$ and transition probabilities $\boldsymbol{P}(\cdot \mid s,a)$ such that

$$\left|r(s,a) - \hat{r}(s,a)\right| \leq \sqrt{\tfrac{7\log(2SAt/\delta)}{2N_t(s,a)}}, \tag{10.3.3}$$

$$\left\|\boldsymbol{P}(\cdot \mid s,a) - \hat{\boldsymbol{P}}_t(\cdot \mid s,a)\right\|_1 \leq \sqrt{\tfrac{14S\log(2At/\delta)}{N_t(s,a)}}, \tag{10.3.4}$$

where $\hat{r}(s,a)$ and $\hat{\boldsymbol{P}}_t(\cdot \mid s,a)$ are the estimates for the rewards and the transition probabilities, and $N_t(s,a)$ denotes the number of samples of action a in state s (at time step t).

One can show via a bound due to Weissman et al. [2003] that given n samples of the transition probability distribution $\boldsymbol{P}(\cdot \mid s,a)$, one has

$$\mathbb{P}\left(\left\|\boldsymbol{P}(\cdot \mid s,a) - \hat{\boldsymbol{P}}_t(\cdot \mid s,a)\right\|_1 \geq \varepsilon\right) \leq 2^S \exp\left(-\frac{n\varepsilon}{2}\right). \tag{10.3.5}$$

Using this together with standard Hoeffding bounds for the reward estimates, it can be shown that the confidence region contains the true underlying MDP with high probability.

Lemma 10.3.1. $\mathbb{P}(\mu^* \in M_t) > 1 - \frac{\delta}{15t^6}$.

Episode lengths Concerning the termination of episodes, as already mentioned, we would like to have episodes that are long enough so that we do not suffer suffer large regret when playing a suboptimal policy. Intuitively, it only pays off to recompute the optimistic policy when the estimates/confidence intervals have changed sufficiently. One option is e.g. to terminate an episode when the confidence interval for one state-action pair has shrinked by some factor. Even simpler, one can terminate an episode when a state-action pair has been sampled often (compared to the samples one had before the episode has started), e.g. when one has doubled the number of visits in some state-action pair. This also allows to bound the total number of episodes up to step T.

Lemma 10.3.2. *If an episode of UCRL2 is terminated when the number of visits in some state-action pair has been doubled, the total number of episodes up to step T is upper bounded by $SA\log_2 \frac{8T}{SA}$.*

The episode termination criterion also allows to bound the sum over all fractions of the form $\frac{v_k(s,a)}{\sqrt{N_k(s,a)}}$, where $v_k(s,a)$ is the number of times action a has been chosen *in* state s during episode k, while $N_k(s,a)$ is the respective count of visits *before* episode k. The evaluation of this sum will turn out to be important to bound the sum over all confidence intervals over the visited state-action pairs in the regret analysis below.

Lemma 10.3.3. $\sum_k \sum_{s,a} \frac{v_k(s,a)}{\sqrt{N_k(s,a)}} \leq (\sqrt{2}+1)\sqrt{SAT}$.

Calculating the optimistic policy It is important to note that the computation of the optimistic policy can be performed efficiently by using a modification of value iteration. Intuitively, for each policy π the optimistic value $g_+^\pi(M_t)$ maximises the gain over all possible values in the confidence intervals for the rewards and the transition probabilities for π. This is an optimisation problem over a compact space that can be easily solved. In order to find $\arg\max_\pi g_+^\pi(M_t)$, for each considered policy one additionally has to determine the precise values for rewards and transition probabilities within the confidence region. This corresponds to finding the optimal policy in an MDP with compact action space, which can be solved by an extension of value iteration that in each iteration now not only maximises over the original action space but also within the confidence region of the respective action. Noting that $g_+^\pi(M_t)$ is maximised when the rewards are set to their upper confidence values, this results in the following value iteration scheme:

1. Set the optimistic rewards $\tilde{r}(s,a)$ to the upper confidence values for all states s and all actions a.

2. Set $u_0(s) := 0$ for all s.

3. For $i = 0, 1, 2, \ldots$ set

$$u_{i+1}(s) \quad := \quad \max_a \left\{ \tilde{r}(s,a) \; + \max_{P \in \mathcal{P}(s,a)} \left\{ \sum_{s'} P(s') u_i(s') \right\} \right\},$$

   where $\mathcal{P}(s,a)$ is the set of all plausible transition probabilities for choosing a in s.

Similarly to the value iteration algorithm in Section 6.5.4, this scheme can be shown to converge. More precisely one can show that $\max_s\{u_{i+1}(s) - u_i(s)\} - \min_s\{u_{i+1}(s) - u_i(s)\} \to 0$ and also

$$u_{i+1}(s) \to u_i(s) + g_+^{\tilde{\pi}} \text{ for all } s. \tag{10.3.6}$$

After convergence the maximizing actions constitute the optimistic policy $\tilde{\pi}$, and the maximizing transition probabilities are the respective optimistic transition values $\tilde{P}$.

One can also show that the *span* $\max_s u_i(s) - \min_s u_i(s)$ of the converged value vector u_i is upper bounded by the diameter. This follows by optimality of the vector u_i. Intuitively, if the span would be larger than D one could increase the collected reward in the lower value state s^- by going (as fast as possible) to the higher value state s^+. (Here we use the fact that the true MDP w.h.p. is plausible, so that we may take the true transitions to go from s^- to s^+.)

Lemma 10.3.4. *Let $u_i(s)$ the converged value vector. Then*

$$\max_s u_i(s) - \min_s u_i(s) \leq D.$$

Analysis of UCRL2

In this section we derive the following regret bound for UCRL2.

Theorem 10.3.1 (Jaksch et al. [2010])**.** *In an MDP with S states, A actions, and diameter D with probability of at least $1-\delta$ the regret of UCRL2 after any T steps is bounded by*

$$\text{const} \cdot DS\sqrt{AT\log\left(\tfrac{T}{\delta}\right)}.$$

Proof. The main idea of the proof is that by Lemma 10.3.1 we have that

$$\tilde{g}_k^* \triangleq g_+^{\tilde{\pi}_k}(M_{t_k}) \;\geq\; g^* \;\geq\; g^{\tilde{\pi}_k}, \tag{10.3.7}$$

so that the regret in each step is upper bounded by the width of the confidence interval for $g^{\tilde{\pi}_k}$, that is, by $\tilde{g}_k^* - g^{\tilde{\pi}_k}$. In what follows we need to break down this confidence interval to the confidence intervals we have for rewards and transition probabilities.

In the following, we consider that the true MDP μ^* is always contained in the confidence regions M_t considered by the algorithm. Using Lemma 10.3.1 it is not difficult to show that with probability at least $1 - \frac{\delta}{12T^{5/4}}$ the regret accumulated due to $\mu^* \notin M_t$ at some step t is bounded by $\sqrt{T}$.

Further, note that the random fluctuation of the rewards can be easily bounded by Hoeffding's inequality (4.5.5), that is, if s_t and a_t denote the state and action at step t, we have

$$\sum_{t=1}^{T} r_t \geq \sum_t r(s_t,a_t) - \sqrt{\tfrac{5}{8}T\log\tfrac{8T}{\delta}}$$

with probability at least $1 - \frac{\delta}{12T^{5/4}}$.

Therefore, writing $v_k(s,a)$ for the number of times action a has been chosen in state s in episode k we have $\sum_t r(s_t,a_t) = \sum_k \sum_{s,a} v_k(s,a)\, r(s,a)$ so that by (10.3.7) we can bound the regret by

$$\sum_{t=1}^{T}(g^* - r_t) \;\leq\; \sum_k \sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - r(s,a)\big) + \sqrt{T} + \sqrt{\tfrac{5}{8}T\log\tfrac{8T}{\delta}} \tag{10.3.8}$$

with probability at least $1 - \frac{2\delta}{12T^{5/4}}$.

Thus, let us consider an arbitrary but fixed episode k, and consider the regret

$$\sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - r(s,a)\big)$$

the algorithm accumulates in this episode. Let $\text{conf}_k^r(s,a)$ and $\text{conf}_k^p(s,a)$ be the width of the confidence intervals for rewards and transition probabilities in episode k. First, we simply have

$$\begin{aligned}\sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - r(s,a)\big) \;&\leq\; \sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - \tilde{r}_k(s,a)\big) \\ &+\; \sum_{s,a} v_k(s,a)\big(\tilde{r}_k(s,a) - r(s,a)\big),\end{aligned} \tag{10.3.9}$$

where the second term is bounded by $|\tilde{r}_k(s,a) - \hat{r}_k(s,a)| + |\hat{r}_k(s,a) - r(s,a)| \leq 2\text{conf}_k^r(s,a)$ w.h.p. by Lemma 10.3.1, so that

$$\sum_{s,a} v_k(s,a)\big(\tilde{r}_k(s,a) - r(s,a)\big) \leq 2\sum_{s,a} v_k(s,a)\cdot \text{conf}_k^r(s,a). \tag{10.3.10}$$

For the first term in (10.3.9) we use that after convergence of the value vector u_i we have by (10.3.6) and (10.3.6)

$$\tilde{g}_k^* - \tilde{r}_k(s,\tilde{\pi}_k(s)) = \sum_{s'} \tilde{P}_k(s'|s,\tilde{\pi}_k(s)) \cdot u_i(s') - u_i(s),$$

so that noting that $v_k(s,a) = 0$ for $a \neq \tilde{\pi}_k(s)$ and using vector/matrix notation we have

$$\begin{aligned}
\sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - \tilde{r}_k(s,\tilde{\pi}_k(s))\big) &= \sum_{s,a} v_k(s,a)\Big(\sum_{s'} \tilde{P}_k(s'|s,\tilde{\pi}_k(s)) \cdot u_i(s') - u_i(s)\Big) \\
&= \mathbf{v}_k\big(\tilde{\mathbf{P}}_k - \mathbf{I}\big)\mathbf{u} \\
&= \mathbf{v}_k\big(\tilde{\mathbf{P}}_k - \mathbf{P}_k + \mathbf{P}_k - \mathbf{I}\big)\mathbf{w}_k \\
&= \mathbf{v}_k\big(\tilde{\mathbf{P}}_k - \mathbf{P}_k\big)\mathbf{w}_k + \mathbf{v}_k\big(\mathbf{P}_k - \mathbf{I}\big)\mathbf{w}_k, \qquad (10.3.11)
\end{aligned}$$

where $\mathbf{P}_k$ is the true transition matrix (in μ^*) of the optimistic policy $\tilde{\pi}_k$ in episode k, and $\mathbf{w}_k$ is a renormalisation of the vector $\mathbf{u}$ (with entries $u_i(s)$) where $w_k(s) := u_i(s) - \frac{1}{2}(\min_s u_i(s) + \max_s u_i(s))$, so that $\|\mathbf{w}_k\|_\infty \leq \frac{D}{2}$ by Lemma 10.3.4.

Since $\|\tilde{\mathbf{P}}_k - \mathbf{P}_k\|_1 \leq \|\tilde{\mathbf{P}}_k - \hat{\mathbf{P}}_k\|_1 + \|\hat{\mathbf{P}}_k - \mathbf{P}_k\|_1$, the first term of (10.3.11) is bounded as

$$\begin{aligned}
\mathbf{v}_k\big(\tilde{\mathbf{P}}_k - \mathbf{P}_k\big)\mathbf{w}_k &\leq \big\|\mathbf{v}_k\big(\tilde{\mathbf{P}}_k - \mathbf{P}_k\big)\big\|_1 \cdot \big\|\mathbf{w}_k\big\|_\infty \\
&\leq 2\sum_{s,a} v_k(s,a)\,\mathrm{conf}_k^p(s,a)\,D. \qquad (10.3.12)
\end{aligned}$$

The second term can be rewritten as martingale difference sequence

$$\begin{aligned}
\mathbf{v}_k\big(\mathbf{P}_k - \mathbf{I}\big)\mathbf{w}_k &= \sum_{t=t_k}^{t_{k+1}-1} \Big(P(\cdot|s_t,a)\mathbf{w}_k - w_k(s_t)\Big) \\
&= \sum_{t=t_k}^{t_{k+1}-1} \Big(P(\cdot|s_t,a)\mathbf{w}_k - w_k(s_{t+1})\Big) + w_k(s_{t_{k+1}}) - w_k(s_{t_k}),
\end{aligned}$$

so that its sum over all episodes can be bounded by Azuma-Hoeffding inequality (5.3.4) and Lemma 10.3.2, that is,

$$\sum_k \mathbf{v}_k\big(\mathbf{P}_k - \mathbf{I}\big)\mathbf{w}_k \leq D\sqrt{\tfrac{5}{2}T\log\left(\tfrac{8T}{\delta}\right)} + DSA\log_2\left(\tfrac{8T}{SA}\right) \qquad (10.3.13)$$

with probability at least $1 - \frac{\delta}{12T^{5/4}}$.

Summing (10.3.10) and (10.3.12) over all episodes, by definition of the confidence intervals and Lemma 10.3.3 we have

$$\begin{aligned}
&\sum_k \sum_{s,a} v_k(s,a)\,\mathrm{conf}_k^r(s,a) + 2D\sum_k \sum_{s,a} v_k(s,a)\,\mathrm{conf}_k^p(s,a) \\
&\leq \text{const} \cdot D\sqrt{S\log(AT/\delta)} \sum_k \sum_{s,a} \tfrac{v_k(s,a)}{\sqrt{N_k(s,a)}} \\
&\leq \text{const} \cdot D\sqrt{S\log(AT/\delta)}\,\sqrt{SAT}. \qquad (10.3.14)
\end{aligned}$$

Thus, combining (10.3.9)–(10.3.14) we obtain that

$$\sum_{s,a} v_k(s,a)\big(\tilde{g}_k^* - r(s,a)\big) \quad \leq \quad \text{const} \cdot D\sqrt{S\log(AT/\delta)}\,\sqrt{SAT} \tag{10.3.15}$$

with probability at least $1 - \frac{\delta}{12T^{5/4}}$.

Finally by (10.3.8) and (10.3.15) the regret of UCRL2 is upper bounded by $\text{const} \cdot D\sqrt{S\log(AT/\delta)}\,\sqrt{SAT}$ with probability at least $1 - 3\sum_{T\geq 2} \frac{\delta}{12T^{5/4}} \geq 1 - \delta$. □

The following is a corresponding lower bound on the regret that shows that the upper bound of Theorem 10.3.1 is optimal in T and A. In the meantime, upper bounds of $O(D\sqrt{SAT})$ have been derived for Tompson sampling.

Theorem 10.3.2. *[Jaksch et al., 2010] For any algorithm and any natural numbers T, S, $A > 1$, and $D \geq \log_A S$ there is an MDP with S states, A actions, and diameter D, the expected regret after T steps is*

$$\Omega\big(\sqrt{DSAT}\big).$$

Similar to the distribution dependent regret bound of Theorem 10.2.1 for UCB1, one can derive a logarithmic bound on the expected regret of UCRL2.

Theorem 10.3.3. *[Jaksch et al., 2010] Let $\Delta \triangleq \rho^*(\mathcal{M}) - \max_\pi \big\{\rho(\mathcal{M},\pi) : \rho(\mathcal{M},\pi) < \rho^*(\mathcal{M})\big\}$ be the gap between the optimal gain and the second largest gain achievable in $\mathcal{M}$. Then the expected regret of UCRL2 is*

$$O\left(\frac{D^2S^2A\log(T)}{\Delta}\right).$$

10.3.3 Bibliographical remarks

Similar to UCB1, UCRL2 is not the first optimistic algorithm with theoretical guarantees. Thus, the *index policies* of Burnetas and Katehakis [1997] and Tewari and Bartlett [2008] choose actions optimistically by using confidence bounds for the estimates in the current state. However, the logarithmic regret bounds are derived only for *ergodic* MDPs in which each policy visits each state with probability 1.

The most well-known optimistic RL algorithm is R-Max [Brafman and Tennenholtz, 2003], that assumes in each not sufficiently visited state to receive the maximal possible reward. UCRL2 offers a refinement of this idea to motivate exploration. Sample complexity bounds as derived for R-Max can also be obtained for UCRL2, cf. [Jaksch et al., 2010].

The gap between the lower bound of Theorem 10.3.2 and the bound for UCRL2 has not been closed so far. There have been claims of an improved upper bound for a Thompson sampling-like algorithm by Agrawal and Jia [2017]. However, there is a problem with the published proof that has not been resolved up-to-date.

The situation is settled in the simpler episodic setting, where after any H steps there is a restart. Here there are matching upper and lower bounds of order $\sqrt{HSAT}$ on the regret, see [Azar et al., 2017].

In the discounted setting, the MBIE algorithm of Strehl and Littman [2005, 2008] is a precursor of UCRL2 that is based on the same ideas. The derived regret bounds are not easily comparable to Theorem 10.2.1, however they are weaker, as the regret is measured along the trajectory of the algorithm, while the regret considered for UCRL2 is with respect to the trajectory an optimal policy would have taken. In general, regret in the discounted setting seems to be a less satisfactory concept. However, sample complexity bounds in the discounted setting for a UCRL2 variant have been given in [Lattimore and Hutter, 2014].

Chapter 11

Conclusion

This book touched upon the basic principles of decision making under uncertainty in the context of reinforcement learning. While one of the main streams of thought is Bayesian decision theory, we also discussed the basics of approximate dynamic programming and stochastic approximation as applied to reinforcement learning problems.

Consciously, however, we have avoided going into a number of topics related to reinforcement learning and decision theory, some of which would need a book of their own to be properly addressed. Even though it was fun writing the book, we at some point had to decide to stop and consolidate the material we had, sometimes culling partially developed material in favour of a more concise volume.

Firstly, we haven't explicitly considered many models that can be used for representing transition distributions, value functions or policies, beyond the simplest ones, as we felt that this would detract from the main body of the text. Textbooks for the latest fashion are always going to be abundant, and we hope that this book provides a sufficient basis to enable the use of any current methods. There are also a large number of areas which have not been covered at all. In particular, while we touched upon the setting of two-player games and its connection to robust statistical decisions, we have not examined problems which are also relevant to sequential decision making, such as Markov games and Bayesian games. In relation to this, while early in the book we discuss risk aversion and risk seeking, we have not discussed specific sequential decision making algorithms for such problems. Furthermore, even though we discuss the problem of preference elicitation, we do not discuss specific algorithms for it or the related problem of inverse reinforcement learning. Another topic which went unmentioned, but which may become more important in the future, is hierarchical reinforcement learning as well as options, which allow constructing long-term actions (such as "go to the supermarket") from primitive actions (such as "open the door"). Finally, even though we have mentioned the basic framework of regret minimisation, we focused on the standard reinforcement learning problem, and ignored adversarial settings and problems with varying amounts of side information.

It is important to note that the book almost entirely elides social aspects of decision making. In practice, any algorithm that is going to be used to make autonomous decision is going to have a societal impact. In such cases, the algorithm designer must guard against negative externalities, such as hurting disadvantaged groups, violating privacy, or environmental damage. However, as a lot of these issues are context dependent, we urge the reader to consult recent work in economics, algorithmic fairness and differential privacy.

Appendix A

Symbols

$\triangleq$	definition
$\wedge$	logical and
$\vee$	logical or
$\Rightarrow$	implies
$\Leftrightarrow$	if and only if
$\exists$	there exists
$\forall$	for every
s.t.	such that

Table A.1: Logic symbols

$\{x_k\}$	a set indexed by k
$\{x \mid xRy\}$	the set of x satisfying relation xRy
$\mathbb{N}$	set of natural numbers
$\mathbb{Z}$	set of integers
$\mathbb{R}$	set of real numbers
Ω	the universe set (or sample space)
$\emptyset$	the empty set
$\mathbb{\Delta}^n$	the n-dimensional simplex
$\mathbb{\Delta}(A)$	the collection of distributions over a set A
$\mathfrak{B}(A)$	the Borel σ-algebra induced by a set A
A^n	the product set $\prod_{i=1}^{n} A$
A^*	$\bigcup_{n=0}^{\infty} A^n$ the set of all sequences from set A
$x \in A$	x belongs to A
$A \subset B$	A is a (strict) subset of B
$A \subseteq B$	A is a (non-strict) subset of B
$B \setminus A$	set difference
$B \triangle A$	symmetric set difference
$A^{\complement}$	set complement
$A \cup B$	set union
$A \cap B$	set intersection

Table A.2: List of set theory symbols

$\boldsymbol{x}^\top$	the transpose of a vector $\boldsymbol{x}$
$\lvert\boldsymbol{A}\rvert$	the determinant of a matrix A
$\lVert x\rVert_p$	The p-norm of a vector $(\sum_i \lvert x_i\rvert^p)^{1/p}$
$\lVert f\rVert_p$	The p-norm of a function $(\int \lvert f(x)i\rvert^p \,\mathrm{d}x)^{1/p}$
$\lVert A\rVert_p$	The operator norm of a matrix $\max\{Ax \mid \lVert x\rVert_p = 1\}$
$\partial f(x)/\partial x_i$	Partial derivative with respect to x_i
∇f	Gradient vector of partial derivatives with respect to vector x

Table A.3: Analysis and linear algebra symbols

$\mathcal{Beta}(\alpha, \beta)$ Beta distribution with parameters (α, β). $\mathcal{Geom}(\omega)$ Geometric distribution with parameter ω $\mathcal{Wish}(n -$

Table A.4: Miscellaneous statistics symbols

Appendix B

Probability concepts

This chapter is intended as a refresher of basic concepts in probability. This includes the definition of probability functions, expectations and moments. Perhaps unusually for an introductory text, we use the modern definition of probability as a *measure*, i.e. an additive function on sets.

Probability measures the likelihood of different events; where each event corresponds to a set in some universe of set. For that reason, we first remind the reader of elementary set theory and then proceed to describe how this relates to events.

B.1 Fundamental definitions

We start with ground set Ω that contains all objects we want to talk about. These objects are called the *elements* of Ω. Given a property Y of elements in Ω, one can define the set of all objects that satisfy this property. That is,

$$A \triangleq \{x \mid x \text{ have property } Y\}.$$

Example 47.

$$B(c,r) \triangleq \{x \in \mathbb{R}^n \mid \|x - c\| \leq r\}$$

describes the set of points enclosed in an n-dimensional sphere of radius r with center $c \in \mathbb{R}^n$.

We use the following notations and definitions for sets. If an element x *belongs* to a set A, we write $x \in A$. Let the *sample space* Ω be a set such that $\omega \in \Omega$ always.We say that A is a *subset* of B or that B *contains* A, and write $A \subset B$, iff, $x \in B$ for any $x \in A$. Let $B \setminus A \triangleq \{x \mid x \in B \text{ and } x \notin A\}$ be the set difference. Let $A \triangle B \triangleq (B \setminus A) \cup (A \setminus B)$ be the symmetric set difference. The *complement* of any $A \subseteq \Omega$ is $A^{\complement} \triangleq \Omega \setminus A$. The *empty set* is $\emptyset = \Omega^{\complement}$. The *union* of n sets: $A_1, \ldots, A_n$ is $\bigcup_{i=1}^{n} A_i = A_1 \cup \cdots \cup A_n$. The *intersection* of n sets $A_1, \ldots, A_n$ is $\bigcap_{i=1}^{n} A_i = A_1 \cap \cdots \cap A_n$. A and B are *disjoint* if $A \cap B = \emptyset$. The *Cartesian product* or *product space* is defined as

$$\Omega_1 \times \cdots \times \Omega_n = \{(s_1, \ldots, s_n) \mid s_i \in \Omega_i, i = 1, \ldots, n\} \tag{B.1.1}$$

the set of all ordered n-tuples $(s_1, \ldots, s_n)$.

B.1.1 Experiments and sample spaces

Conceptually, it might be easier to discuss concepts of probability if we think about this in terms of an experiment performed by a statistician. For example, such an experiment could be tossing a coin. The coin could come up heads, tails, balance exactly on the edge, get lost under the furniture, or simple disintegrate when it is tossed. The *sample space* of the experiment must contain *all possible* outcomes.

However, it is the statistician which determines what this set is. For example one statistician may only care whether the coin lands heads, or not (two outcomes). Another may care about how many times it bounces on the ground. Yet another may be interested in both the maximum height reached by the coin and how it lands. Thus, the sample space represents different aspects of the

experiment we are interested in. At the extreme, the sample space and corresponding outcomes may completely describe everything there is to know about the experiment.

Experiments
The set of possible experimental outcomes of an experiment is called the *sample space* Ω.

- Ω must contain all possible outcomes.
- After the experiment is performed, exactly one outcome ω in Ω is true.
- Each statistician i may consider a different Ω_i for the same experiment.

The following example considers the case where three different statisticians care about three different types of outcomes of an experiment where a drug is given to a patient. The first is interested in whether the patient recovers, the second in whether the drug has side-effects, while the third is interested in both.

EXAMPLE 48. Experiment: give medication to a patient.

- $\Omega_1 = \{\text{Recovery within a day}, \text{No recovery after a day}\}$.
- $\Omega_2 = \{\text{The medication has side-effects}, \text{No side-effect}\}$.
- $\Omega_3 =$ all combinations of the above.

Clearly, the drug's effects are much more complex than the above simplified view. One could for example consider a very detailed patient state $\omega \in \Omega$ (which would e.g. describe every molecule in the patient's body)

Product spaces and repeated experiments

Sometimes we perform repeated experiments. Each experiment could be defined in a different outcome space, but many times we are specifically interested in repeated identical experiments. This occurs for example in situations where we give a treatment to patients suffering from a particular disease, and we measure the same outcomes (recovery, side-effects) in each one of them.

More formally, the set-up is as follows: We perform n experiments. The i-th experiment has sample space Ω_i. The sample space $\prod_{i=1}^{n} \Omega_i := \Omega_1 \times \dots \Omega_n$ can be thought of as a sample space of a *composite* experiment in which all n experiments are performed.

Identical experiment sample spaces In many cases, $\Omega_i = \Omega$ for all i, i.e. the sample space is identical for all individual experiments (e.g. n coin tosses). In this case we write $\Omega^n = \prod_{i=1}^{n} \Omega$.

B.2 Events, measure and probability

Probability is a type of function that is called a *measure*. In that sense it is similar to a function that weights, or measures things. Just like when weighing

two apples and adding the total gives you the same answer as weighing both apples together, so does the total probability of either of two mutually exclusive events equals the sum of their individual probabilities. However, sets are complex beasts and formally we wish to define exactly when we can measure them.

Many times the natural outcome space Ω that we wish to consider is extremely complex, but we only care about whether a specific *event* occurs or not. For example, when we toss a coin in the air, the natural outcome is the complete trajectory that the coin follows and its final resting position. However, we might only care about whether the coin lands heads or not. Then, the event of the coin landing "heads" is defined as all the trajectories that the coin follows which result in it landing heads. These trajectories form a subset $A \subset \Omega$.

Probabilities will always be defined on subsets of the outcome space. These subsets are termed events. The probability of events will simply be a function on sets, and more specifically a *measure*. The following gives some intuition and formal definitions about what this means.

B.2.1 Events and probability

Probability of a set
If A is a subset of Ω, the probability of A is a measure of the chances that the outcome of the experiment will be an element of A.

Which sets?

Ideally, we would like to be able to assign a probability to *every subset of* Ω. However, for technical reasons, this is not always possible.

EXAMPLE 49. Let X be uniformly distributed on $[0, 1]$. By definition, this means that the probability that X is in $[0, p]$ is equal to p for all $p \in [0, 1]$. However, even for this simple distribution, it might be difficult to define the probability of all events.

- What is the probability that X will be in $[0, 1/4)$?
- What is the probability that X will be in $[1/4, 1]$?
- What is the probability that X will be a rational number?

B.2.2 Measure theory primer

Imagine that you have an apartment Ω composed of three rooms, A, B, C. There are some coins on the floor and a 5-meter-long red carpet. We can measure various things in this apartment.

Area

- A: $4 \times 5 = 20m^2$.

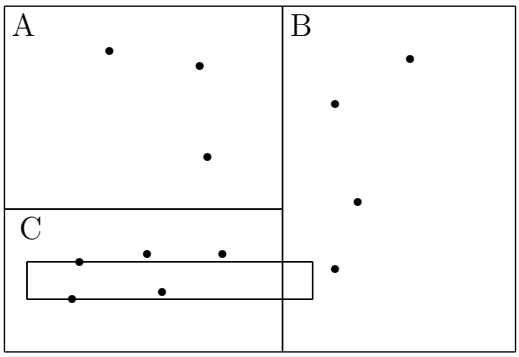


Figure B.1: A fashionable apartment

- B: $6 \times 4 = 24m^2$.
- C: $2 \times 5 = 10m^2$.

Coins on the floor

- A: 3.
- B: 4
- C: 5.

Length of red carpet

- A: $0m$
- B: $0.5m$
- C: $4.5m$.

Measure the sets: $\mathcal{F} = \{\emptyset, A, B, C, A \cup B, A \cup C, B \cup C, A \cup B \cup C\}$. It is easy to see that the union of any sets in $\mathcal{F}$ is also in $\mathcal{F}$. In other words, $\mathcal{F}$ is closed under union. Furthermore, $\mathcal{F}$ contains the whole space Ω.

Note that all those measures have an *additive property*.

B.2.3 Measure and probability

As previously mentioned, the probability of $A \subseteq \Omega$ is a measure of the chances that the outcome of the experiment will be an element of A. Here we give a precise definition of what we mean by measure and probability.

If we want to be able to perform probabilistic logic, we need to define some appropriate algebraic construction that relates events to each other. In particular, if we have a family of events $\mathcal{F}$, i.e. a collection of subsets of Ω, we want this to be closed under union and complement.

Definition B.2.1 (A field on Ω)**.** A family $\mathcal{F}$ of sets, such that for each $A \in \mathcal{F}$, one also has $A \subseteq \Omega$, is called a *field on* Ω if and only if

1. $\Omega \in \mathcal{F}$
2. if $A \in \mathcal{F}$, then $A^{\complement} \in \mathcal{F}$.
3. For any $A_1, A_2, \ldots, A_n$ such that $A_i \in \mathcal{F}$, it holds that: $\bigcup_{i=1}^{n} A_i \in \mathcal{F}$.

From the above definition, it is easy to see that $A_i \cap A_j$ is also in the field.

Since many times our family may contain an infinite number of sets, we also want to extend the above to countably infinite unions.

Definition B.2.2 (σ-field on Ω)**.** A family $\mathcal{F}$ of sets, such that $\forall A \in \mathcal{F}$, $A \subseteq \Omega$, is called a *σ-field on* Ω if and only if

1. $\Omega \in \mathcal{F}$
2. if $A \in \mathcal{F}$, then $A^{\complement} \in \mathcal{F}$.
3. For any sequence $A_1, A_2, \ldots$ such that $A_i \in \mathcal{F}$, it holds that: $\bigcup_{i=1}^{\infty} A_i \in \mathcal{F}$.

It is easy to verify that the $\mathcal{F}$ given in the apartment example satisfies these properties. In general, for any finite Ω, it is easy to find a family $\mathcal{F}$ containing all possible events in Ω. Things become trickier when Ω is infinite. Can we define an algebra $\mathcal{F}$ that contains all events? In general no, but we can define an algebra on the so-called Borel sets of Ω, defined in B.2.3.

Definition B.2.3 (Measure)**.** A measure λ on $(\Omega, \mathcal{F})$ is a function $\lambda : \mathcal{F} \to \mathbb{R}^+$ such that

1. $\lambda(\emptyset) = 0$.
2. $\lambda(A) \geq 0$ for any $A \in \mathcal{F}$.
3. For any collection of subsets $A_1, \ldots, A_n$ with $A_i \in \mathcal{F}$ and $A_i \cap A_j = \emptyset$.

$$\lambda\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \lambda(A_i) \tag{B.2.1}$$

It is easy to verify that the floor area, the number of coins, and the length of the red carpet are all measures. In fact, the area and length correspond to what is called a *Lebesgue measure*[1] and the number of coins to a *counting measure*.

Definition B.2.4 (Probability measure)**.** A probability measure P on $(\Omega, \mathcal{F})$ is a function $P : \mathcal{F} \to [0, 1]$ such that:

1. $P(\Omega) = 1$

[1] See Section B.2.3 for a precise definition.

2. $P(\emptyset) = 0$
3. $P(A) \geq 0$ for any $A \in \mathcal{F}$.
4. If $A_1, A_2, \ldots$ are (pairwise) disjoint then

$$P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i) \qquad \text{(union)}$$

$(\Omega, \mathcal{F}, P)$ is called a *probability space*.

So, probability is just a special type of measure.

The Lebesgue measure*

Definition B.2.5 (Outer measure)**.** Let $(\Omega, \mathcal{F}, \lambda)$ be a measure space. The outer measure of a set $A \subseteq \Omega$ is:

$$\lambda^*(A) \triangleq \inf_{A \subseteq \bigcup_k B_k} \sum_k \lambda(B_k). \tag{B.2.2}$$

In other words, it is the measure λ-measure of the smallest cover $\{B_k\}$ of A.

Definition B.2.6 (Inner measure)**.** Let $(\Omega, \mathcal{F}, \lambda)$ be a measure space. The inner measure of a set $A \subseteq \Omega$ is:

$$\lambda_*(A) \triangleq \lambda(\Omega) - \lambda(\Omega \setminus A). \tag{B.2.3}$$

Definition B.2.7 (Lebesgue measurable sets)**.** A set A is (Lebesgue) measurable if the outer and inner measures are equal.

$$\lambda^*(A) = \lambda_*(B). \tag{B.2.4}$$

The common value of the inner and outer measure is called the Lebesgue measure[2] $\bar{\lambda}(A) = \lambda^*(A)$.

The Borel σ-algebra*

When Ω is a finite collection $\{\omega_1, \ldots, \omega_n\}$, there is a σ-algebra containing all possible events in Ω, denoted 2^Ω. This is called the *powerset*. However, in general this is not possible. For infinite sets equipped with a metric, we can instead define the *Borel σ-algebra* $\mathfrak{B}(\Omega)$, which is the smallest σ-algebra containing all open sets of Ω.

powerset

Borel σ-algebra

B.3 Conditioning and independence

A probability measure can give us the probability of any set in the algebra. Each one of these sets can be seen as an *event*. For example, the set of all states where a patient has a fever constitutes the event that the patient has a fever. Thus, generally we identify events with subsets of Ω.

[2] It is easy to see that $\bar{\lambda}$ is a measure.

However, the basic probability on Ω does not tell us anything about what the probability of some event A, given the fact that some event B has occurred. Sometimes, these events are *mutually exclusive*, meaning that when B happens, A cannot be true; other times B implies A, and sometimes they are *independent*. To quantify exactly how knowledge of whether B has occurs can affect what we know about A, we need the notion of *conditional probability*.

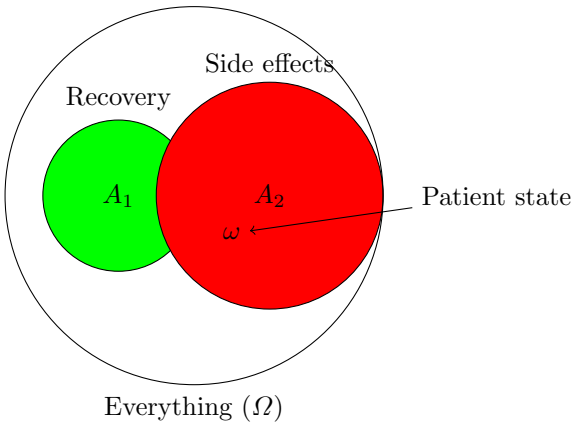


Figure B.2: Events as sets. The patient state $\omega \in \Omega$ after submitting to a treatment may belong to either of the two possible sets A_1, A_2.

B.3.1 Mutually exclusive events

By events, we mean subsets of Ω. Thus, the probability of the event that a draw from Ω is in A is equal to the probability measure of A, $P(A)$. Some events are mutually exclusive, meaning that they can never happen at the same time. This is the same as saying that the corresponding sets have an empty intersection.

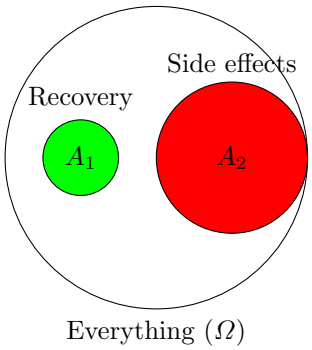


Figure B.3: Mutually exclusive events A_1, A_2.

Definition B.3.1 (Mutually exclusive events)**.** Two events A, B are mutually exclusive if and only if $A \cap B = \emptyset$.

By definition of the measure, $P(A \cup B) = P(A) + P(B)$ for any mutually exclusive events.

Lemma B.3.1 (Union bound)**.** *For any events A, B, it holds that*

$$P(A \cup B) \leq P(A) + P(B). \tag{B.3.1}$$

Proof. Let $C = A \cap B$. Then

$$\begin{aligned} P(A) + P(B) &= P(A \setminus C) + P(C) + P(B \setminus C) + P(C) \\ &\geq P(A \setminus C) + P(C) + P(B \setminus C) = P(A \cup B) \end{aligned}$$

□

The union bound is extremely important, and one of the basic proof methods in many applications of probability.

Finally, let us consider the general case of multiple disjoint events, shown in Figure B.4. When B is decomposed in a set of disjoint events $\{B_i\}$, we can

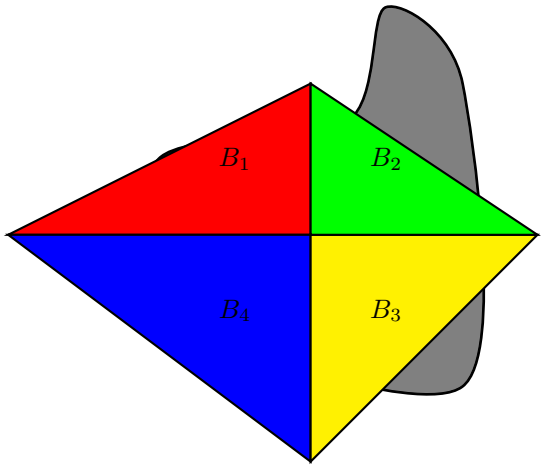


Figure B.4: Disjoint events and marginalisation

write:

$$P(B) = P\left(\bigcup_i B_i\right) = \sum_i P(B_i) \tag{B.3.2}$$

$$P(A \cap B) = P\left(\bigcup_i (A \cap B_i)\right) = \sum_i P(A \cap B_i), \tag{B.3.3}$$

for any other set A. An interesting special case occurs when $B = \Omega$, in which case $P(A) = P(A \cap \Omega)$, since $A \subset \Omega$ for any A in the algebra. This results in the *marginalisation* or *sum rule* of probability.

marginalisation
sum rule

$$P(A) = P\left(\bigcup_i (A \cap B_i)\right) = \sum_i P(A \cap B_i), \qquad \bigcup_i B_i = \Omega. \tag{B.3.4}$$

B.3.2 Independent events

Sometimes different events are independent, in the sense there is no interaction between their probabilities. This can be formalised as follows.

Definition B.3.2 (Independent events)**.** Two events A, B are independent if $P(A \cap B) = P(A)P(B)$. The events in a family $\mathcal{F}$ of events are independent if for any sequence $A_1, A_2, \ldots$ of events in $\mathcal{F}$,

$$P\left(\bigcap_{i=1}^{n} A_i\right) = \prod_{i=1}^{n} P(A_i) \qquad \text{(independence)}$$

As a simple example, consider Figure B.5, where the universe is a rectangle of dimensions $(1, 1)$, two events A_1, A_2 are rectangles with A_1 having dimensions $(1, h)$ and A_2 having dimensions (w, h). Let's take the probability distribution P which assigns probability $P(A)$ equal to the *area* of the set A. Then

$$P(A_1) = 1 \times h = h, \qquad P(A_2) = w \times 1 = w.$$

Similarly, the intersecting rectangle has dimensions (w, h). Consequently

$$P(A_1 \cup A_2) = w \times h = P(A_1) \times P(A_2)$$

and the two events are independent. Independent events are particularly im-

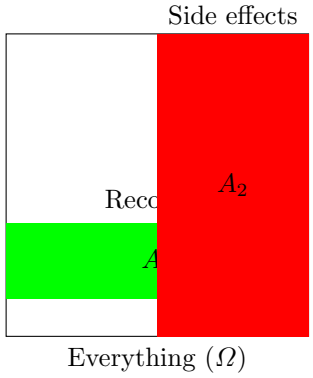


Figure B.5: Independent events A_1, A_2.

portant in repeated experiments, where the outcomes of one experiment are independent of the outcome of another.

B.3.3 Conditional probability

Now that we have defined a distribution for all possible events, and we have also defined basic relationships between events, we'd also like to have a way of determining the probability of one event given that another has occurred. This is given by the notion of conditional probability.

Definition B.3.3 (Conditional probability)**.** The conditional probability of A when B, s.t. $P(B) > 0$, is given is:

$$P(A \mid B) \triangleq \frac{P(A \cap B)}{P(B)}. \tag{B.3.5}$$

Note that we can always write $P(A \cap B) = \mathbb{P}(A \mid B)\,\mathbb{P}(B)$ even if A, B are not independent.

Finally, we say that two events A, B are *conditionally independent* given C if *conditionally independent*

$$P(A \cap B \mid C) = P(A \mid C)P(B \mid C). \tag{B.3.6}$$

This is an important notion when dealing with probabilistic graphical models.

B.3.4 Bayes' theorem

The following theorem trivially follows from the above discussion. However, versions of it shall be used repeatedly throughout the book. For this reason we present it here together with a detailed proof.

Theorem B.3.1 (Bayes' theorem)**.** *Let $A_1, A_2, \ldots$ be a (possibly infinite) sequence of disjoint events such that $\bigcup_{i=1}^{\infty} A_i = \Omega$ and $P(A_i) > 0$ for all i. Let B be another event with $P(B) > 0$. Then*

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{\sum_{j=1}^{\infty} P(B \mid A_j)P(A_j)} \tag{B.3.7}$$

Proof. From (B.3.5), $P(A_i \mid B) = P(A_i \cap B)/P(B)$ and also $P(A_i \cap B) = P(B \mid A_i)P(A_i)$. Thus

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{P(B)},$$

and we continue analyzing the denominator $P(B)$. First, due to $\bigcup_{i=1}^{\infty} A_i = \Omega$ we have $B = \bigcup_{j=1}^{\infty}(B \cap A_j)$. Since A_i are disjoint, so are $B \cap A_i$. Then from the union property of probability distributions we have

$$P(B) = P\left(\bigcup_{j=1}^{\infty}(B \cap A_j)\right) = \sum_{j=1}^{\infty} P(B \cap A_j) = \sum_{j=1}^{\infty} P(B \mid A_j)P(A_j),$$

which finishes the proof. □

B.4 Random variables

A random variable X is a special kind of random quantity, defined as a function of outcomes in Ω to some vector space. Unless otherwise stated, the mapping is on the real numbers $\mathbb{R}$. Thus, it also defines a mapping from a probability measure P on $(\Omega, \mathcal{F})$ to a probability measure P_X on $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$. More precisely, we define the following.

Definition B.4.1 (Measurable function)**.** Let $\mathcal{F}$ on Ω be a σ-field. A function $g : \Omega \to \mathbb{R}$ is said to be *measurable with respect to* $\mathcal{F}$, or $\mathcal{F}$-measurable, if, for any $x \in \mathbb{R}$,

$$\{s \in \Omega \mid g(s) \leq x\} \in \mathcal{F}.$$

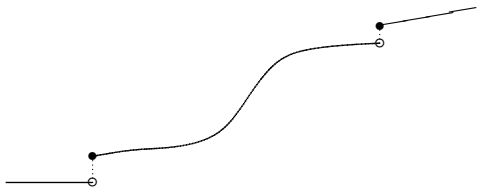


Figure B.6: A distribution function F

Definition B.4.2 (Random variable)**.** Let $(\Omega, \mathcal{F}, P)$ be a probability space. A *random variable* $X : \Omega \to \mathbb{R}$ is a real-valued, $\mathcal{F}$-measurable function.

The distribution of X

Every random variable X induces a probability measure P_X on $\mathbb{R}$. For any $B \subseteq \mathbb{R}$ we define

$$P_X(B) \triangleq \mathbb{P}(X \in B) = P(\{s \mid X(s) \in B\}). \tag{B.4.1}$$

Thus, the probability that X is in B is equal to the P-measure of the points $s \in \Omega$ such that $X(s) \in B$ and also equal to the P_X-measure of B.

Here $\mathbb{P}$ is used as a *short-hand* notation.

Exercise 41. Ω is the set of 52 playing cards. $X(s)$ is the value of each card $(1, 10$ for the ace and figures respectively). What is the probability of drawing a card s with $X(s) > 7$?

B.4.1 (Cumulative) Distribution functions

Definition B.4.3 ((Cumulative) Distribution function)**.** The distribution function of a random variable X is the function $F : \mathbb{R} \to \mathbb{R}$:

$$F(t) = \mathbb{P}(X \leq t). \tag{B.4.2}$$

Properties

- If $x \leq y$, then $F(x) \leq F(y)$.
- F is right-continuous.
- At the limit,
$$\lim_{t \to -\infty} F(t) = 0, \qquad \lim_{t \to \infty} F(t) = 1.$$

B.4.2 Discrete and continuous random variables

On the real line, there are two types of distributions for a random variable. Here, once more, we employ the $\mathbb{P}$ notation as a shorthand for the probability of general events involving random variables, so that we don't have to deal with the measure notation. The two following examples should give some intuition.

Discrete distributions
$X : \Omega \to \{x_1, \ldots, x_n\}$ takes n discrete values (n can be infinite). The probability function of X is

$$f(x) \triangleq \mathbb{P}(X = x),$$

defined for $x \in \{x_1, \ldots, x_n\}$. For any $B \subseteq \mathbb{R}$:

$$P_X(B) = \sum_{x_i \in B} f(x_i).$$

In addition, we write $\mathbb{P}(X \in B)$ to mean $P_X(B)$.

Continuous distributions
X has a continuous distribution if there exists a *probability density function* f s.t. $\forall B \subseteq \mathbb{R}$:

$$P_X(B) = \int_B f(x)\,\mathrm{d}x.$$

B.4.3 Random vectors

We can generalise the above to random *vectors*. These can be seen as *vectors* of random variables. These are just random variables on some Cartesian product space, i.e. $X : \Omega \to \mathcal{V}$, with $\mathcal{V} = V_1 \times \cdots \times V_m$. Once more, there are two special cases of distributions for the random vector $X = (X_1, \ldots, X_m)$. The first is a vector of discrete random variables:

Discrete distributions

$$\mathbb{P}(X_1 = x_1, \ldots, X_m = x_m) = f(x_1, \ldots, x_m),$$

where f is *joint probability function*, with $x_i \in V_i$.

The second is a vector of continuous random variables.

Continuous distributions
For $B \subseteq \mathbb{R}^m$

$$\mathbb{P}\left\{(X_1, \ldots, X_m) \in B\right\} = \int_B f(x_1, \ldots, x_m)\,\mathrm{d}x_1 \cdots \mathrm{d}x_m$$

In general, it is possible that X has neither a continuous, nor a discrete distribution; for example if some V_i is discrete and some V_j are continuous. In that case it is convenient to use measure-theoretic notation, explained in the next section.

B.4.4 Measure-theoretic notation

The previously seen special cases of discrete and continuous variables can be handled with a unified notation if we take advantage of the fact that probability is only a particular type of measure. As a first step, we note that summation can also be seen as integration with respect to the counting measure and that Riemann integration is integration with respect to the Lebesgue measure.

Integral with respect to a measure μ

Introduce the common notation $\int \cdots \mathrm{d}\mu(x)$, where μ is a measure. Let some real function $g : \Omega \to \mathbb{R}$. Then for any subset $B \subseteq \Omega$ we can write

- Discrete case: f is the probability function and we choose the *counting measure* for μ, so:

$$\sum_{x \in B} g(x)f(x) = \int_B g(x)f(x)\,\mathrm{d}\mu(x)$$

  Roughly speaking, the counting measure $\mu(\Omega)$ is equal to the number of elements in Ω.

- Continuous case: f is the probability density function and we choose the *Lebesgue measure* for μ, so:

$$\int_B g(x)f(x)\,\mathrm{d}x = \int_B g(x)f(x)\,\mathrm{d}\mu(x)$$

  Roughly speaking, the Lebesgue measure $\mu(S)$ is equal to the volume of S.

In fact, since probability is a measure in itself, we do not need to complicate things by using f and μ at the same time! This allows us to use the following notation.

Lebesgue-Stiletjes notation
If P is a probability measure on $(\Omega, \mathcal{F})$ and $B \subseteq \Omega$, and g is $\mathcal{F}$-measurable, we write the probability that $g(x)$ takes the value B can be written equivalently as:

$$\mathbb{P}(g \in B) = P_g(B) = \int_B g(x)\,\mathrm{d}P(x) = \int_B g\,\mathrm{d}P. \tag{B.4.3}$$

Intuitively, $\mathrm{d}P$ is related to densities in the following way. If P is a measure on Ω and is absolutely continuous with respect to another measure μ, then $p \triangleq \frac{\mathrm{d}P}{\mathrm{d}\mu}$ is the (Radon-Nikodyn) derivative of P with respect to μ. We write the integral as $\int gp\,\mathrm{d}\mu$. If μ is the Lebesgue measure, then p coincides with the probability density function.

B.4.5 Marginal distributions and independence

Although this is a straightforward outcome of the set-theoretic definition of probability, we also define the marginal explicitly for random vectors.

Marginal distribution
The marginal distribution of $X_1, \ldots, X_k$ from a set of variables $X_1, \ldots, X_m$, is

$$\mathbb{P}(X_1, \ldots, X_k) \triangleq \int \mathbb{P}(X_1, \ldots, X_k, X_{k+1} = x_{k+1}, \ldots, X_m = x_m) \\ \mathrm{d}\mu(x_{k+1}, \ldots, x_m). \quad \text{(B.4.4)}$$

In the above, $\mathbb{P}(X_1, \ldots X_k)$ can be thought of as the probability measure for any events related to the random vector $(X_1, \ldots, X_k)$. Thus, it defines a probability measure over $\left(\mathbb{R}^k, \mathfrak{B}\left(\mathbb{R}^k\right)\right)$. In fact, let $Y = (X_1, \ldots, X_k)$ and $Z = (X_{k+1}, \ldots, X_m)$ for simplicity. Then define $Q(A) \triangleq \mathbb{P}(Z \in A)$, with $A \subseteq \mathbb{R}^{m-k-1}$. Then the above can be re-written as:

$$\mathbb{P}(Y \in B) = \int_{\mathbb{R}^{m-k-1}} \mathbb{P}(Y \in B \mid Z = z)\,\mathrm{d}Q(z).$$

Similarly, $\mathbb{P}(Y \mid Z = z)$ can be thought of as a function mapping from values of Z to probability measures. Let $P_z(B) \triangleq \mathbb{P}(Y \in B \mid Z = z)$ be this measure corresponding to a particular value of z. Then we can write

$$\mathbb{P}(Y \in B) = \int_{\mathbb{R}^{m-k-1}} \left(\int_B \mathrm{d}P_z(y) \right) \mathrm{d}Q(z).$$

Independence
If X_i is independent of X_j for all $i \neq j$:

$$\mathbb{P}(X_1, \ldots, X_m) = \prod_{i=1}^{M} \mathbb{P}(X_i), \qquad f(x_1, \ldots, x_m) = \prod_{i=1}^{M} g_i(x_i) \qquad \text{(B.4.5)}$$

B.4.6 Moments

There are some simple properties of the random variable under consideration which are frequently of interest in statistics. Two of those properties are *expectation* and *variance*. *expectation*

Expectation

Definition B.4.4. The expectation $\mathbb{E}(X)$ of any random variable $X : \Omega \to R$, where R is a vector space, with distribution P_X is defined by

$$\mathbb{E}(X) \triangleq \int_R t \, \mathrm{d}P_X(t), \tag{B.4.6}$$

as long as the integral exists.

Furthermore,

$$\mathbb{E}[g(X)] = \int g(t) \, \mathrm{d}P_X(t),$$

for any function g.

variance

Definition B.4.5. The *variance* $\mathbb{V}(X)$ of any random variable $X : \Omega \to \mathbb{R}$ with distribution P_X is defined by

$$\begin{aligned} \mathbb{V}(X) &\triangleq \int_{-\infty}^{\infty} [t - \mathbb{E}(X)]^2 \, \mathrm{d}P_X(t) \\ &= \mathbb{E}\left\{ [X - \mathbb{E}(X)]^2 \right\} \\ &= \mathbb{E}(X^2) - \mathbb{E}^2(X). \end{aligned} \tag{B.4.7}$$

covariance matrix

When $X : \Omega \to R$ with R an arbitrary vector space, the above becomes the *covariance matrix*:

$$\begin{aligned} \mathbb{V}(X) &\triangleq \int_{-\infty}^{\infty} [t - \mathbb{E}(X)] \, [t - \mathbb{E}(X)]^\top \, \mathrm{d}P_X(t) \\ &= \mathbb{E}\left\{ [X - \mathbb{E}(X)] \, [X - \mathbb{E}(X)]^\top \right\} \\ &= \mathbb{E}(XX^\top) - \mathbb{E}(X) \, \mathbb{E}(X)^\top. \end{aligned} \tag{B.4.8}$$

B.5 Divergences

Divergences are a natural way to measure how different two distributions are.

KL-Divergence

Definition B.5.1. The *KL-Divergence* is a non-symmetric divergence.

$$D\left(P \parallel Q\right) \triangleq \int \frac{\mathrm{d}P}{\mathrm{d}Q} \, \mathrm{d}P. \tag{B.5.1}$$

Another useful distance is the L_1 distance

Definition B.5.2. The L_1-distance betwen two measures is defined as:

$$\|P - Q\|_1 = \int |\frac{\mathrm{d}P}{\mathrm{d}\mu} - \frac{\mathrm{d}Q}{\mathrm{d}\mu}| \, \mathrm{d}\mu, \tag{B.5.2}$$

where μ is any measure dominating both P and Q.

B.6 Empirical distributions

rical distribution

When we have no model for a particular distribution, it is sometimes useful to construct the *empirical distribution*, which basically counts how many times we observe different outcomes.

Definition B.6.1. Let $x^n = (x_1, \ldots, x_n)$ drawn from a product measure $x^n \sim P^n$ on the measurable space $(\mathcal{X}^n, \mathcal{F}_n)$. Let $\mathfrak{S}$ be any σ-field on $\mathcal{X}$. Then empirical distribution of x^n is defined as

$$\hat{P}_n(B) \triangleq \frac{1}{n} \sum_{t=1}^{n} \mathbb{I}\{x_t \in B\}. \tag{B.6.1}$$

The problem with the empirical distribution is that does not capture the uncertainty we have about what the real distribution is. For that reason, it should be used with care, even though it does converge to the true distribution in the limit. A clever way to construct a measure of uncertainty is to perform *sub-sampling*, that is to create k random samples of size $n' < n$ from the original sample. Each sample will correspond to a different random empirical distribution. Sub-sampling is performed *without replacement* (i.e. for each sample, each observation x_i is only used once). When sampling with replacement and $n' = n$, the method is called *bootstrapping*.

sub-sampling

without replacement

bootstrapping

B.7 Further reading

Much of this material is based on DeGroot [1970]. See Kolmogorov and Fomin [1999] for a really clear exposition of measure, starting from rectangle areas (developed from course notes in 1957). Also see Savage [1972] for a verbose, but interesting and rigorous introduction to subjective probability. A good recent text on elementary probability and statistical inference is Bertsekas and Tsitsiklis [2008].

B.8 Exercises

EXERCISE 42 (5). Show that for any sets A, B, D:

$$A \cap (B \cup D) = (A \cap B) \cup (A \cap D).$$

Show that

$$(A \cup B)^{\complement} = A^{\complement} \cap B^{\complement}, \quad \text{and} \quad (A \cap B)^{\complement} = A^{\complement} \cup B^{\complement}$$

EXERCISE 43 (10). Prove that any probability measure P has the following properties:

1. $P(A^{\complement}) = 1 - P(A)$.
2. If $A \subset B$ then $P(A) \leq P(B)$.
3. For any sequence of events $A_1, \ldots, A_n$

$$P\left(\bigcup_{i=1}^{\infty} A_i\right) \leq \sum_{i=1}^{\infty} P(A_i) \qquad \text{(union bound)}$$

Hint: Recall that If $A_1, \ldots, A_n$ are disjoint *then $P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$ and that $P(\emptyset) = 0$*

Definition B.8.1. A random variable $X \in \{0, 1\}$ has Bernoulli distribution with parameter $p > [0, 1]$, written $X \sim \mathit{Bern}(p)$, if

$$p = \mathbb{P}(X = 1) = 1 - \mathbb{P}(X = 0).$$

The probability function of X can be written as

$$f(x \mid p) = \begin{cases} p^x (1-p)^{1-x}, & x \in \{0, 1\} \\ 0, & \text{otherwise.} \end{cases}$$

Definition B.8.2. A random variable $X \in \{0, 1\}$ has a binomial distribution with parameters $p > [0, 1]$, $n \in \mathbb{N}$ written $X \sim \mathit{Binom}(p, n)$, if the probability function of X is

$$f(x \mid n, p) = \begin{cases} \binom{n}{x} p^x (1-p)^{n-x}, & x \in \{0, 1, \ldots, n\} \\ 0, & \text{otherwise.} \end{cases}$$

If $X_1, \ldots, X_n$ is a sequence of Bernoulli random variables with parameter p, then $\sum_{i=1}^{n} X_i$ has a binomial distribution with parameters n, p.

EXERCISE 44 (10). Let $X \sim \mathit{Bern}(p)$

1. Show that $\mathbb{E}\, X = p$
2. Show that $\mathbb{V}\, X = p(1-p)$
3. Find the value of p for which X has the greatest variance.

EXERCISE 45 (10). In a few sentences, describe your views on the usefuleness of probability.

- Is it the only formalism that can describe both random events and uncertainty?
- Would it be useful to separate randomness from uncertainty?
- What would be desirable properties of an alternative concept?

Appendix C

Useful results

C.1 Functional Analysis

Definition C.1.1 (supremum)**.** When we say that

$$M = \sup_{x \in A} f(x),$$

then: (i) $M \geq f(x)$ for any $x \in A$. In other words, M is an upper bound on $f(x)$. (ii) for any $M' < M$, there exists some $x' \in A$ s.t. $M' < f(x')$.

In other words, there exists no smaller upper bound than M. When the function f has a maximum, then the supremum is identical to the maximum.

Definition C.1.2 (infimum)**.** When we say that

$$M = \sup_{x \in A} f(x),$$

then: (i) $M \geq f(x)$ for any $x \in A$. In other words, M is an upper bound on $f(x)$. (ii) for any $M' < M$, there exists some $x' \in A$ s.t. $M' < f(x')$.

Norms Let (S, Σ, μ) be a measure space. The L_p norm of a μ-measurable function f is defined as

$$\|f\|_p = \left(\int_S |f(x)|^p \, \mathrm{d}\mu(x) \right)^{1/p}. \tag{C.1.1}$$

Hölder inequality. Let (S, Σ, μ) be a measure space and let $1 \leq p, q \leq \infty$ with $1/p + 1/q = 1$ then for all μ-measurable f, g:

$$\|fg\|_1 \leq \|f\|_p \|g\| q. \tag{C.1.2}$$

The special case $p = q = 2$ results in the Cauchy-Schwarz inequality.

Lipschitz continuity We say that a function $f : X \to Y$ is Lipschitz, with respect to metrics d, ρ on X, Y respectively when

$$\rho(f(a) - f(b)) \leq d(a, b) \qquad \forall a, b \in X. \tag{C.1.3}$$

Special spaces. The n-dimensional Euclidean space is denoted by $\mathbb{R}^n$.

The n-dimensional simplex is denoted by $\mathbb{\Delta}^n$ and it holds that for any $\boldsymbol{x} \in \mathbb{\Delta}^n$, $\|\boldsymbol{x}\|_1 = 1$ and $x_k \geq 0$.

C.1.1 Series

Definition C.1.3 (The geometric series)**.** The sum $\sum_{k=0}^{n} x^k$ is called the geometric series and has the property

$$\sum_{k=0}^{n} x^k = \frac{x^{n+1} - 1}{x - 1}. \tag{C.1.4}$$

Taking derivatives with respect to x can result in other useful formulae.

C.1.2 Special functions

Definition C.1.4 (Gamma function)**.** For a positive integer n,

$$\Gamma(n) = (n-1)! \tag{C.1.5}$$

For a positive real numbers (or complex numbers with a postive real part), the gamma function is defined as

$$\Gamma(t) = \int_0^\infty x^{-1} e^{-x} \, \mathrm{d}x. \tag{C.1.6}$$

Appendix D

Index

Index

Bibliography

Shipra Agrawal and Randy Jia. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 1184–1194, 2017. URL http://papers.nips.cc/paper/6718-optimistic-posterior-sampling-for-reinforcement-learning-wo

Mauricio Álvarez, David Luengo, Michalis Titsias, and Neil Lawrence. Efficient multioutput gaussian processes through variational inducing kernels. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, pages 25–32, 2010.

A. Antos, C. Szepesvári, and R. Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008a.

Andrè Antos, Rémi Munos, and Csaba Szepesvari. Fitted Q-iteration in continuous action-space MDPs. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA, 2008b.

Robert B. Ash and Catherine A. Doleéans-Dade. *Probability & Measure Theory*. Academic Press, 2000.

Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *colt2009. Proceedings of the 22nd Annual Conference on Learning Theory*, pages 217–226, 2009.

Peter Auer and Ronald Ortner. UCB revisited: improved regret bounds for the stochastic multi-armed bandit problem. *Period. Math. Hungar.*, 61(1-2): 55–65, 2010.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2/3):235–256, 2002a.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002b. doi: 10.1137/S0097539701398375. URL http://dx.doi.org/10.1137/S0097539701398375.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the*

34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, pages 263–272, 2017. URL http://proceedings.mlr.press/v70/azar17a.html.

Andrew G Barto. Adaptive critics and the basal ganglia. *Models of information processing in the basal ganglia*, page 215, 1995.

Jonathan Baxter and Peter L. Bartlett. Reinforcement learning in POMDP's via direct gradient ascent. In *Proc. 17th International Conf. on Machine Learning*, pages 41–48. Morgan Kaufmann, San Francisco, CA, 2000. URL citeseer.nj.nec.com/baxter00reinforcement.html.

Richard Ernest Bellman. A problem in the sequential design of experiments. *Sankhya*, 16:221–229, 1957.

A. Bernstein. Adaptive state aggregation for reinforcement learning. Master's thesis, Technion – Israel Institute of Technolog01y, 2007.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

Dimitri P Bertsekas and John N Tsitsiklis. *Introduction to Probability: Dimitri P. Bertsekas and John N. Tsitsiklis*. Athena Scientific, 2008.

J. A. Boyan. Technical update: Least-squares temporal difference learning. *Machine Learning*, 49(2):233–246, 2002.

S. J. Bradtke and A. G. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1):33–57, 1996.

R. I. Brafman and M. Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231, 2003. ISSN 1532-4435.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012. doi: 10.1561/2200000024. URL http://dx.doi.org/10.1561/2200000024.

Apostolos N. Burnetas and Michael N. Katehakis. Optimal adaptive policies for Markov decision processes. *Math. Oper. Res.*, 22(1):222–255, 1997.

George Casella, Stephen Fienberg, and Ingram Olkin, editors. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer, 1999.

Herman Chernoff. Sequential design of experiments. *Annals of Mathematical Statistics*, 30(3):755–770, 1959.

Herman Chernoff. Sequential models for clinical trials. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol.4*, pages 805–812. Univ. of Calif Press, 1966.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. Technical Report 1610.07524, arXiv, 2016.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. Technical Report 1701.08230, arXiv, 2017.

K. Csilléry, M. G. B. Blum, O. E. Gaggiotti, O. François, et al. Approximate Bayesian computation (ABC) in practice. *Trends in ecology & evolution*, 25 (7):410–418, 2010.

Richard Dearden, Nir Friedman, and Stuart J. Russell. Bayesian Q-learning. In *AAAI/IAAI*, pages 761–768, 1998. URL `citeseer.ist.psu.edu/dearden98bayesian.html`.

J. J. Deely and D. V. Lindley. Bayes empirical Bayes. *Journal of the American Statistical Association*, 76(376):833–841, 1981. ISSN 01621459. URL `http://www.jstor.org/stable/2287578`.

Morris H. DeGroot. *Optimal Statistical Decisions*. John Wiley & Sons, 1970.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7-9):1508–1524, 2009.

Christos Dimitrakakis. *Ensembles for Sequence Learning*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2006a.

Christos Dimitrakakis. Nearly optimal exploration-exploitation decision thresholds. In *Int. Conf. on Artificial Neural Networks (ICANN)*, 2006b.

Christos Dimitrakakis. Tree exploration for Bayesian RL exploration. In *Computational Intelligence for Modelling, Control and Automation, International Conference on*, pages 1029–1034, Wien, Austria, 2008. IEEE Computer Society. ISBN 978-0-7695-3514-2. doi: http://doi.ieeecomputersociety.org/10.1109/CIMCA.2008.32.

Christos Dimitrakakis. Bayesian variable order Markov models. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *JMLR : W&CP*, pages 161–168, Chia Laguna Resort, Sardinia, Italy, 2010a.

Christos Dimitrakakis. Complexity of stochastic branch and bound methods for belief tree search in Bayesian reinforcement learning. In *2nd international conference on agents and artificial intelligence (ICAART 2010)*, pages 259–264, Valencia, Spain, 2010b. ISNTICC, Springer.

Christos Dimitrakakis. Robust bayesian reinforcement learning through tight lower bounds. In *European Workshop on Reinforcement Learning (EWRL 2011)*, number 7188 in LNCS, pages 177–188, 2011.

Christos Dimitrakakis. Monte-carlo utility estimates for bayesian reinforcement learning. In *IEEE 52nd Annual Conference on Decision and Control (CDC 2013)*, 2013. arXiv:1303.2506.

Christos Dimitrakakis and Michail G. Lagoudakis. Algorithms and bounds for rollout sampling approximate policy iteration. In *EWRL*, pages 27–40, 2008a.

Christos Dimitrakakis and Michail G. Lagoudakis. Rollout sampling approximate policy iteration. *Machine Learning*, 72(3):157–171, September 2008b. doi: 10.1007/s10994-008-5069-3. Presented at ECML'08.

Christos Dimitrakakis and Nikolaos Tziortziotis. ABC reinforcement learning. In *ICML 2013*, volume 28(3) of *JMLR W & CP*, pages 684–692, 2013. See also arXiv:1303.6977.

Christos Dimitrakakis and Nikolaos Tziortziotis. Usable ABC reinforcement learning. In *NIPS 2014 Workshop: ABC in Montreal*, 2014.

Christos Dimitrakakis, Yang Liu, David Parkes, and Goran Radanovic. Subjective fairness: Fairness is in the eye of the beholder. Technical Report 1706.00119, arXiv, 2017.

Michael O'Gordon Duff. *Optimal Learning Computational Procedures for Bayes-adaptive Markov Decision Processes*. PhD thesis, University of Massachusetts at Amherst, 2002.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226. ACM, 2012.

Yaakov Engel, Shie Mannor, and Ron Meir. Bayes meets bellman: The gaussian process approach to temporal difference learning. In *ICML 2003*, 2003.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 201–208. ACM, 2005.

Eyal Even-Dar and Yishai Mansour. Approximate equivalence of markov decision processes. In *Learning Theory and Kernel Machines. COLT/Kernel 2003*, Lecture notes in Computer science, pages 581–594, Washington, DC, USA, 2003. Springer.

Milton Friedman and Leonard J. Savage. The expected-utility hypothesis and the measurability of utility. *The Journal of Political Economy*, 60(6):463, 1952.

Thomas Furmston and David Barber. Variational methods for reinforcement learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *JMLR : W&CP*, pages 241–248, Chia Laguna Resort, Sardinia, Italy, 2010.

J. Geweke. Using simulation methods for Bayesian econometric models: inference, development, and communication. *Econometric Reviews*, 18(1):1–73, 1999.

Mohammad Ghavamzadeh and Yaakov Engel. Bayesian policy gradient algorithms. In *NIPS 2006*, 2006.

C. J. Gittins. *Multi-armed Bandit Allocation Indices*. John Wiley & Sons, New Jersey, US, 1989.

Robert Grande, Thomas Walsh, and Jonathan How. Sample efficient reinforcement learning with gaussian processes. In *International Conference on Machine Learning*, pages 1332–1340, 2014.

Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.

M. Hutter. Feature reinforcement learning: Part I: Unstructured MDPs. *Journal of Artificial General Intelligence*, 1:3–24, 2009.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010.

Tobias Jung and Peter Stone. Gaussian processes for sample-efficient reinforcement learning with RMAX-like exploration. In *ECML/PKDD 2010*, pages 601–616, 2010.

Sham Kakade. A natural policy gradient. *Advances in neural information processing systems*, 2:1531–1538, 2002.

Emilie Kaufmanna, Nathaniel Korda, and Rémi Munos. Thompson sampling: An optimal finite time analysis. In *ALT-2012*, 2012.

Michael Kearns and Satinder Singh. Finite sample convergence rates for Q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems*, volume 11, pages 996–1002. The MIT Press, 1999.

Niki Kilbertus, Mateo Rojas-Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. Technical Report 1706.02744, arXiv, 2017.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. Technical Report 1609.05807, arXiv, 2016.

AN Kolmogorov and SV Fomin. *Elements of the theory of functions and functional analysis*. Dover Publications, 1999.

M. Lagoudakis and R. Parr. Reinforcement learning as classification: Leveraging modern classifiers. In *ICML*, page 424, 2003a.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003b.

Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Adv. in Appl. Math.*, 6:4–22, 1985.

Nam M. Laird and Thomas A. Louis. Empirical Bayes confidence intervals based on bootstrap samples. *Journal of the American Statistical Association*, 82(399):739–750, 1987.

Tor Lattimore and Marcus Hutter. Near-optimal PAC bounds for discounted MDPs. *Theor. Comput. Sci.*, 558:125–143, 2014.

Howard Philips Lovecraft. History of the necronomicon, 1938.

T. Lwin and J. S. Maritz. Empirical Bayes approach to multiparameter estimation: with special reference to multinomial distribution. *Annals of the Institute of Statistical Mathematics*, 41(1):81–99, 1989.

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

J. M. Marin, P. Pudlo, C. P. Robert, and R. J. Ryder. Approximate Bayesian computational methods. *Statistics and Computing*, pages 1–14, 2011.

Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001a.

Thomas P. Minka. Bayesian linear regression. Technical report, Microsoft research, 2001b.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

R. Munos and C. Szepesvári. Finite-time bounds for fitted value iteration. *The Journal of Machine Learning Research*, 9:815–857, 2008.

Ronald Ortner, Daniil Ryabko, Peter Auer, and Rémi Munos. Regret bounds for restless Markov bandits. *Theor. Comput. Sci.*, 558:62–76, 2014. doi: 10.1016/j.tcs.2014.09.026. URL `http://dx.doi.org/10.1016/j.tcs.2014.09.026`.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *NIPS*, 2013.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, pages 4026–4034, 2016.

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

P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. In *ICML 2006*, pages 697–704. ACM Press New York, NY, USA, 2006.

Pascal Poupart and Nikos Vlassis. Model-based Bayesian reinforcement learning in partially observable domains. In *International Symposium on Artificial Intelligence and Mathematics (ISAIM)*, 2008.

Marting L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New Jersey, US, 1994.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. ISBN 13 978-0-262-18253-9.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

Herbert Robbins. An empirical Bayes approach to statistics. In Jerzy Neyman, editor, *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. University of California Press, Berkeley, CA, 1955.

Herbert Robbins. The empirical Bayes approach to statistical decision problems. *The Annals of Mathematical Statistics*, 35(1):1–20, 1964.

Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. Bayes-adaptive POMDPs. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, Cambridge, MA, 2008. MIT Press.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, et al., editors, *Parallel Distributed Processing: Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, 1987.

Leonard J. Savage. *The Foundations of Statistics*. Dover Publications, 1972.

Wolfram Schultz, Peter Dayan, and P. Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997. ISSN 0036-8075. doi: 10.1126/science.275.5306.1593. URL `http://science.sciencemag.org/content/275/5306/1593`.

S. Singh, T. Jaakkola, and M. I. Jordan. Reinforcement learning with soft state aggregation. *Advances in neural information processing systems*, pages 361–368, 1995.

M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

A. L. Strehl and M. L. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74 (8):1309–1331, 2008. ISSN 0022-0000.

Alexander L. Strehl and Michael L. Littman. A theoretical analysis of model-based interval estimation. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005)*, pages 857–864. ACM, 2005.

Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888. ACM, 2006.

Malcolm Strens. A Bayesian framework for reinforcement learning. In *ICML 2000*, pages 943–950, 2000.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPS 99*, 1999.

Ole Tange. Gnu parallel-the command-line power tool. *The USENIX Magazine*, 36(1):42–47, 2011.

Ambuj Tewari and Peter Bartlett. Optimistic linear programming gives logarithmic regret for irreducible MDPs. In *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, pages 1505–1512. MIT Press, 2008.

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.

T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P. H. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202, 2009.

Marc Toussaint, Stefan Harmelign, and Amos Storkey. Probabilistic inference for solving (PO)MDPs, 2006.

Paul Tseng. Solving h-horizon, stationary markov decision problems in time proportional to log (h). *Operations Research Letters*, 9(5):287–297, 1990.

John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

Nikolaos Tziortziotis and Christos Dimitrakakis. Bayesian inference for least squares temporal difference regularization. In *ECML*, 2017.

Nikolaos Tziortziotis, Christos Dimitrakakis, and Konstantinos Blekas. Cover tree Bayesian reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 2014.

J. Veness, K. S. Ng, M. Hutter, and D. Silver. A Monte Carlo AIXI approximation. Arxiv preprint arXiv:0909.0801, 2009.

Nikos Vlassis, Michael L. Littman, and David Barber. On the computational complexity of stochastic controller optimization in POMDPs. *TOCT*, 4(4): 12, 2012.

Tao Wang, Daniel Lizotte, Michael Bowling, and Dale Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *ICML '05*, pages 956–963, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: http://doi.acm.org/10.1145/1102351.1102472.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8 (3-4):279–292, 1992.

T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and M. J. Weinberger. Inequalities for the L_1 deviation of the empirical distribution. *Hewlett-Packard Labs, Tech. Rep*, 2003.

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

Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4):593–603, 2011.

Henry H Yin and Barbara J Knowlton. The role of the basal ganglia in habit formation. *Nature Reviews Neuroscience*, 7(6):464, 2006.

Martin Zinkevich, Amy Greenwald, and Michael Littman. Cyclic equilibria in markov games. In *Advances in Neural Information Processing Systems*, 2006.