Chapter 8

Partially Observable Markov Decision Processes

Confucious said: "To know that we know what we know, and that we do not know what we do not know, that is true knowledge."[1]

Henry David Thoreau, American naturalist and philosopher, 1817-1862.

This chapter studies a class of widely applicable models referred to as *partially observable Markov decision processes* or *POMDPs*. In a POMDP, the decision maker cannot observe the state of the system but instead receives a *signal* that provides noisy information about the *hidden* system state. Since the state is unobservable, actions must be chosen on the basis of the only available information, namely the sequence of previously observed signals and actions. As in the previously discussed Markov decision process model, the decision maker chooses actions so as to maximize a performance measure such as the expected total reward or discounted reward over a finite or infinite horizon.

The key idea in analyzing a POMDP is summarized in the following sequence of events:

1. Given a prior estimate (probability distribution) of the hidden state, the decision maker chooses an action and observes a signal.

2. Based on this action and signal, Bayes' Theorem[2] is used to obtain a posterior distribution of the hidden state.

[1] Thoreau [1854], Chapter 1.

[2] The process of using Bayes' Theorem in this way is sometimes referred to as *Bayesian filtering*.

3. This posterior distribution becomes the prior distribution at the next decision epoch and provides the basis for choosing the next action.

These probability distributions are referred to as *belief states*. After establishing that the information encoded in a belief state is sufficient for decision making and Markovian, the POMDP can be transformed into a related Markov decision process with state space equal to the set of all probability distributions on S or equivalently the $(|S|-1)$-dimensional unit simplex[3].

The model resulting from this transformation has an uncountable compact state space. On the surface, this makes the model intractable but a remarkable result of Smallwood and Sondik [1973] shows that optimal value functions are piecewise linear and convex. Since such functions can be represented by the maximum of a *finite* number of linear functions, exact and approximate algorithms can be developed to exploit the structure.

Note that modern applications, such as autonomous vehicle navigation or robotic control, base action choice on the observed signal instead of the belief state. Although sub-optimal, such an approach is the only option available when a system model is not available or too complex to represent. This chapter will assume a known model and focus on methods based on belief states.

8.1 Model overview

Consider a Markov decision process in which the decision maker cannot directly observe the system state, but instead observes a *signal* from the system that provides some information about the true system state but does not completely describe it. Since action choice can only depend on available information, it must be based on previously observed signals and actions, and not the hidden system state.

Such a model is widely applicable. For example:

1. In medicine, clinicians use diagnostic tests such as x-rays or blood tests to provide information regarding the unobservable health state of a patient. Test results inform future treatment decisions and the need for further tests.

2. In equipment maintenance, the operator of a machine, in the face of uncertainty about its true working condition, bases preventive maintenance decisions on signals obtained through sensors or on the number of defects in a sample of the machine's output.

3. In robot navigation, a robot uses sensor information such as camera images to estimate its true location and make navigational decisions.

[3]The $(N-1)$-dimensional unit simplex is the set of N-dimensional non-negative vectors $(x_1, \ldots, x_N)$ satisfying $\sum_{i=1}^{N} x_i = 1$

Information acquisition

The observation process may be conceptualized in two different ways:

1. *Passive information acquisition*: From this perspective, action choice does not directly influence the information gathering process.

2. *Active information acquisition*: Actions may be employed to gather information about the system state. These actions can vary in accuracy and cost, and may or may not affect the system state.

While this distinction will not impact analysis, it provides a framework for understanding and modeling the information acquisition process in a POMDP.

The following two examples shed light on this distinction. As a model of passive information acquisition, consider a robot navigating in a Gridworld that acquires information about its location through a noisy sensor readings. The robot chooses its next move based on these noisy observations, together with past actions and observations. This represents passive information acquisition, since the robot's actions do not directly influence the information gathering process.

Active information acquisition is prevalent in medical decision making in which a clinician must decide which of several tests to use to assess a patient's health status. Usually the test does not alter the patent's health state, but instead provides important information about it. For instance, consider the challenge of monitoring and managing the blood-oxygen level in a newborn child. Using a clip attached to the baby's finger provides a noisy but non-invasive measure. Alternatively, the clinician may decide to draw a blood sample so as to obtain a more accurate measure. Because the baby's blood supply is limited, the more invasive test must be used judiciously and only when the degree of uncertainty about the blood oxygen level is high.

These examples as described above can be easily modified to include both passive and active features.

8.1.1 POMDP dynamics

In the models analyzed here, the signal is an element of a finite discrete *signal space* O. Let the random variable, Z_n, denote the signal received at decision epoch n. Its probability distribution is represented by[4]

$$u(o|s,a) := P[Z_{n+1} = o \,|X_{n+1} = s, Y_n = a] \tag{8.1}$$

for $o \in O$, $a \in A_s$, $s \in S$ and $n \geq 1$. Note that the signal distribution is conditional on the action chosen at the *preceding* decision epoch and the state at the *current* decision

[4]If the signal space were continuous, $u(o|s,a)$ could represent a probability density or distribution function.

epoch. Also assume $u(o|s,a)$ is stationary, that is, it does not vary with decision epoch. In passive-information acquisition models, u will also be independent of a and only a function of the state. Note also that a Markov decision process is a special case of a POMDP in which there is a one-to-one mapping between observations and states, namely, $u(o|s,a) = 1$ when $o = s$ and 0 otherwise.

A POMDP evolves as follows. At decision epoch n, the system is in unobservable state s^n and the decision maker observes a signal, o^n. Using this and all past information, the decision maker chooses an action a^n. The consequence of choosing this action is an (unobserved) system transition to state s^{n+1} and a reward $r^n = r_n(s^n, a^n, s^{n+1})$ (or $r_n(s^n, a^n)$) being accrued. A new signal o^{n+1} becomes available at the start of the next decision epoch. Figure 8.1 illustrates the timing of events between two decision epochs.

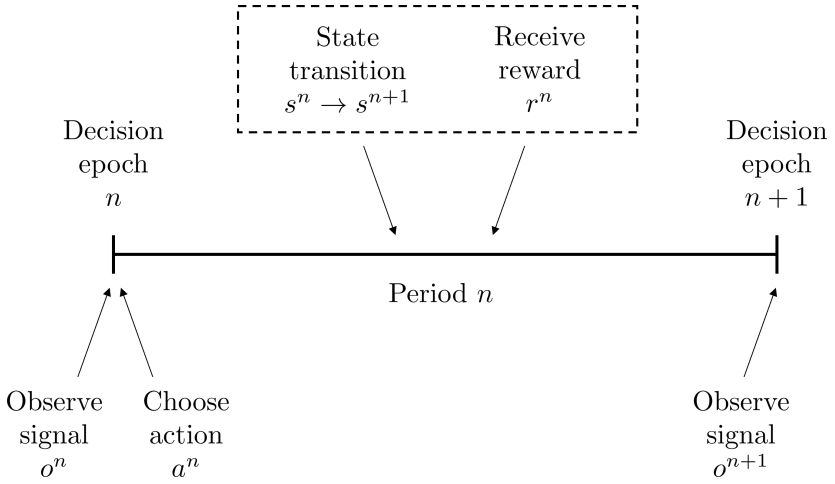


Figure 8.1: Timeline of events in period $n > 1$ of a POMDP. The decision maker observes the signal and the action choice. Events in the dashed box, namely the state transition and reward, are unobservable. Note that the first period ($n = 1$) begins with the decision maker's prior assessment of the state of the unobservable Markov decision process, so a signal is unnecessary.

The POMDP model assumes that the decision maker observes the past sequence of actions and signals, but not the sequence of state transitions or rewards since the system state is unobservable[5]. Note that the first decision epoch is distinguished because this iterative process begins with a prior assessment of the system state $b_1(s) := P[X_1 = s]$ instead of a signal[6].

[5]In some applications, the rewards may be observable, but in such a case must not provide any extra information about the current hidden state. For example, in Gridworld navigation, all transitions might incur a cost of one unit.

[6]This is consistent with the notion of a Markov decision process beginning in a random state with a known probability distribution $\rho(s)$.

A comment on action choice

We emphasize that there is some subtlety that is often overlooked regarding how actions are implemented in a POMDP. In most POMDP applications, $A_s = A$ for all $s \in S$ so that any action can be used in *any* unobservable state. For example a robot may try to move north, south, east or west at any location or a doctor may choose one of several tests in any health state.

However, it is also possible that A_s varies with s, which means that different actions are available in different (hidden) states such as in the model in Section 8.1.2. In this situation the POMDP decision maker must inform the Markov decision process controller[7] how to select an action in each possible state. What this means is that:

1. An action in a POMDP is a decision rule in the hidden Markov decision process.
2. A decision rule in the POMDP is a function mapping the current signal and past signals and actions to decison rules in the hidden Markov decision process.

To emphasize this distinction, consider first a model in which $A_s = A$ for all $s \in S$ and restrict attention to Markovian deterministic decision rules. If this were a Markov decision process, a decision rule might specify a different action in each state. However in a POMDP, this is not possible since the state of the underlying Markov decision process is unknown. Therefore the only decision rules that can be specified for use in the hidden Markov decision process are of the form $d(s) = a$ for all $s \in S$. Therefore a decision rule in the POMDP maps each possible current signal and past signals and actions to a single action in the underlying Markov decision process that can be applied in every state. For example "use test B" in a clinical decision making problem. This argument applies as well to using Markovian randomized decision rules in the underlying Markov decision process.

When A_s varies with s, the distinction and control mechanism is more problematic. For this model to be implementable, there must exist a controller in the hidden Markov decision process that can implement *any* decision rule handed down from the decision maker in the POMDP. In this case an action in the POMDP corresponds to a Markovian decision rule in the hidden Markov decision process. The example in Section 8.1.2 illustrates such a model.

[7] To distinguish the two levels of this model, the entity making decisions in the POMDP is referred to as the *decision maker* and the entity making decisions in the Markov decision process is referred to as the *controller*.

The POMDP model

In summary, a POMDP consists of an underlying[8] Markov decision process with unobservable states and rewards, plus a set of possible observations O and a corresponding probability distribution $u(\cdot|s,a)$. More formally, a POMDP can be represented by the collection of objects

$$\big\{T, S, \{A_s : s \in S\}, O, r(s,a), p(j|s,a)\big\},$$

where T is the horizon length and the rest of these quantities have been defined above.

While the added dynamics associated with partial observability are straightforward to understand, the mathematics of the model are considerably more complex than in a fully observable Markov decision process.

An illustrative example

The following is a high-level description of a prototypical clinical decision problem to illustrate some model features. The decision is whether to treat or test a patient when the true disease state is not known. Assume the disease state can be represented by

$$S = \{\text{none}, \text{mild}, \text{moderate}, \text{severe}\}$$

and

$$A_s = \{\text{do nothing}, \text{test}, \text{treat}\}$$

for each $s \in S$. If not treated the (non-fatal) disease may progress and if treated the disease may improve. Let $p(j|s,a)$ represent the probability of change in disease state when the patient is treated or untreated (corresponding to the actions "do nothing" or "test"). This becomes a meaningful problem when the treatment is effective in controlling the disease, expensive, has serious side effects and can only be administered once. So in essence this is a partially observable optimal stopping problem that terminates upon treatment.

In most settings, test results will be imprecise with possible outcomes

$$O = \{\text{negative}, \text{inconclusive}, \text{positive}\},$$

which are observed with probability $u(o|s, \text{test})$ upon testing.

In contrast to the model in the next section, *each action can be applied in each state.* That is the $A_s = A$ for all $s \in S$. Moreover, the model allows for both active and passive information acquisition.

[8] Also referred to as a hidden, unobservable or core MDP in the literature.

8.1.2 A two-state POMDP model

To provide a model that illustrates the generality of the POMDP model, consider a modified version of the two-state model in Section 2.5 that adds a signal space $O = \{o_1, o_2\}$ and signal probabilities $u(o_1|s_1) = 0.8$, $u(o_2|s_1) = 0.2$, $u(o_1|s_1) = 0.4$ and $u(o_2|s_2) = 0.6$ that are independent of decision epoch and action choice. Note that it is more likely to receive signal o_1 when the hidden state is s_1 and o_2 when the hidden state is s_2. This may be regarded as a passive information acquisition model because the observation probability does not depend on action choice.

Assume an initial state distribution $b_1(s_1) = P[X_1 = s_1]$ and $b_1(s_2) = P[X_1 = s_2]$. Let $\mathbf{b}_1 = (b_1(s_1), b_1(s_2))$, denote a column vector with components $b_1(s_1)$ and $b_1(s_2)$. Since there are only two states, letting $b_1(s_1) = q$ implies $b_1(s_2) = 1 - q$ for $0 \leq q \leq 1$. Note that in this example, the unit simplex is $\{(x_1, x_2)|x_1 + x_2 = 1, x_i \geq 0, x_2 \geq 0\}$.

Figure 8.2 provides a visual representation of the model.

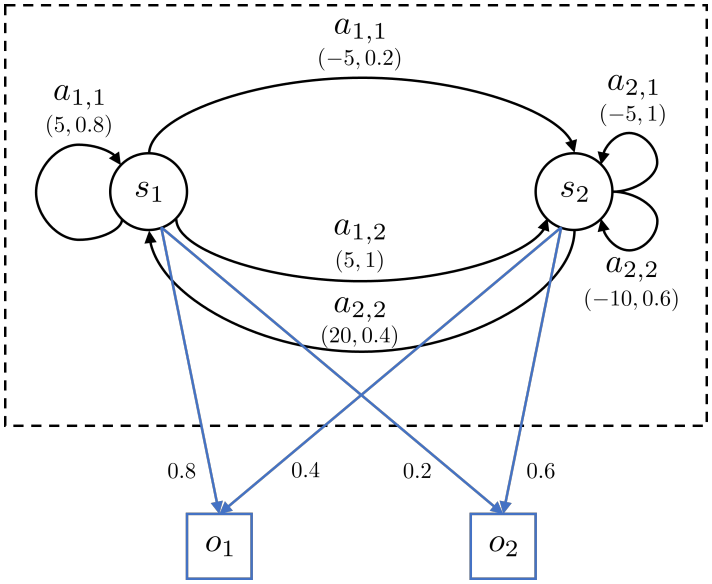


Figure 8.2: Graphical representation of the two-state POMDP model. The entities in the dashed box represents the hidden (unobservable) Markov decision process. The square boxes indicate the observable signals, with probabilities of observation on the arcs originating in the hidden states. Note the model can be generalized to allow the signal to depend on the action as well.

Computing the optimal value function in a one-period model

The following calculations for a one-period model illustrate some of the key steps in the analysis of a POMDP and the subtlety of decision rule specification. Suppose (for

illustrative purposes) that the POMDP decision maker wishes to restrict attention to *only* the two Markovian deterministic decision rules, f and g, in the underlying Markov decision process defined by:

$$f(s) := \begin{cases} a_{1,1}, & \text{if } s = s_1 \\ a_{2,1}, & \text{if } s = s_2 \end{cases} \tag{8.2}$$

and

$$g(s) := \begin{cases} a_{1,2}, & \text{if } s = s_1 \\ a_{2,2}, & \text{if } s = s_2. \end{cases} \tag{8.3}$$

Note that f can be interpreted as choosing the "first action" in each state and g can be interpreted as choosing the "second action" in each state. Moreover, there are other decision rules available for consideration such as choosing the first action in state s_1 and the second action in state s_2, or even randomizing between actions in each hidden state. However, restricting attention to f and g is sufficient to illustrate the key points in the calculations below.

In this one-period model, the only information available to the POMDP decision maker is $\mathbf{b}_1$ the POMDP decision maker's belief that the system state at decision epoch 1 is s_1 or s_2. Hence a decision rule in the POMDP must specify when to choose f and g as a function of $\mathbf{b}_1$. As an example consider a POMDP decision rule of the form:

$$d(\mathbf{b}_1) = \begin{cases} f, & b_1(s_1) \geq \overline{q} \\ g, & b_1(s_1) < \overline{q} \end{cases} \tag{8.4}$$

for some pre-specified $\overline{q} \in [0, 1]$. Alternatively the same decision rule can be expressed in terms of Markov decision process primitives as:

$$d(\mathbf{b}_1) = \begin{cases} a_{1,1}, & s = s_1, b_1(s_1) \geq \overline{q} \\ a_{1,2}, & s = s_1, b_1(s_1) < \overline{q} \\ a_{2,1}, & s = s_2, b_1(s_1) \geq \overline{q} \\ a_{2,2}, & s = s_2, b_1(s_1) < \overline{q}. \end{cases} \tag{8.5}$$

Suppose $\overline{q} = 0.5$. Then d chooses decision rule f if the probability of being in state s_1 is at least 0.5 and chooses decision rule g if the probability of being in state s_1 is less than 0.5. The expression for the decision rule d in (8.5) makes clear that while decision rules in the POMDP are functions of belief state, they specify what action to choose in each hidden state.

POMDP decision rules that always choose f and g represent extreme cases. Using either one of them for every belief state is most likely sub-optimal. The question then becomes whether there is an optimal value for $\overline{q}$, or even if there is a better decision rule with multiple breakpoints $\overline{q}_1, \ldots, \overline{q}_k$.

To address this question, computation of the expected total reward under decision rules f and g is illustrated below. The goal is to understand under what values for

$\mathbf{b}_1$ does f produce higher expected total reward than g. Note that even though these decision rules do not depend on $\mathbf{b}_1$, the expected total reward will depend on $\mathbf{b}_1$ since the total reward will depend on the starting state of the system. Furthermore, since this is a two-state model, a single parameter q fully specifies $\mathbf{b}_1$.

Assume a terminal reward $r_2(s_1) = x$ and $r_2(s_2) = y$. Let $v_f(q)$ and $v_g(q)$ denote the expected total reward obtained using f and g when $b_1(s_1) = q = 1 - b_1(s_2)$. Then,

$$\begin{aligned} v_f(q) &= (r(s_1, a_{1,1}, s_1) + r_2(s_1))p(s_1|s_1, a_{1,1})b_1(s_1) + (r(s_1, a_{1,1}, s_2) + r_2(s_2))p(s_2|s_1, a_{1,1})b_1(s_1) \\ &\quad + (r(s_2, a_{2,1}, s_1) + r_2(s_1))p(s_1|s_2, a_{2,1})b_1(s_2) + (r(s_2, a_{2,1}, s_2) + r_2(s_2))p(s_1|s_2, a_{2,1})b_1(s_2) \\ &= (5 + x)0.8q + (-5 + y)0.2q + (0 + x)0(1 - q) + (-5 + y)1(1 - q) \\ &= (3 + 0.8x + 0.2y)q + (-5 + y)(1 - q) \\ &= -5 + y + (8 + 0.8x - 0.8y)q, \end{aligned} \tag{8.6}$$

and

$$\begin{aligned} v_g(q) &= (r(s_1, a_{1,2}, s_1) + r_2(s_1))p(s_1|s_1, a_{1,2})b_1(s_1) + (r(s_1, a_{1,2}, s_2) + r_2(s_2))p(s_2|s_1, a_{1,2})b_1(s_1) \\ &\quad + (r(s_2, a_{2,2}, s_1) + r_2(s_1))p(s_1|s_2, a_{2,2})b_1(s_2) + (r(s_2, a_{2,2}, s_2) + r_2(s_2))p(s_1|s_2, a_{2,2})b_1(s_2) \\ &= (0 + x)0q + (5 + y)1q + (20 + x)0.4(1 - q) + (-10 + y)0.6(1 - q) \\ &= (5 + y)q + (2 + 0.4x + 0.6y)(1 - q) \\ &= 2 + 0.4x + 0.6y + (3 - 0.4x + 0.4y)q. \end{aligned} \tag{8.7}$$

Figure 8.3 shows how $v_f(q)$ and $v_g(q)$ vary as a function of q when $x = 10$ and $y = 0$. The two functions intersect at $q = 11/17$, $v_f(q) > v_g(q)$ for $q > 11/17$ and $v_g(q) > v_f(q)$ for $q < 11/17$. Thus if the POMDP decision maker believes $q < 11/17$ then the decision maker will inform the Markov decision process controller to use decision rule g. Accordingly, the Markov decision process controller will use action $a_{1,2}$ if the Markov decision process is in state s_1 and $a_{2,2}$ if the Markov decision process is in state s_2.

It is left as an exercise to show that for other choices of x and y, $v_f(q)$ may lie above or below $v_g(q)$ for all values of q.

Some observations that motivate the general analysis of POMDP models follow:

1. The function $\max\{v_f(q), v_g(q)\}$ is piecewise linear and convex in q.

2. The distribution of the state at decision epoch 2, X_2, is required to evaluate the expected total reward. It involves both the prior distribution $\mathbf{b}_1$ and the transition probabilities in the hidden Markov decision process, $p(j|s, a)$.

3. The expected total reward under f and g are linear functions of $\mathbf{b}_1$. For example, $v_f(q) = \boldsymbol{\gamma}^\mathsf{T}\mathbf{b}_1$, where $\boldsymbol{\gamma} = (3 + 0.8x + 0.2y, -5 + y)$ and $\mathbf{b}_1 = (q, 1 - q)$.

4. Figure 8.3 shows that the optimal policy partitions the interval $[0, 1]$ into two disjoint intervals.

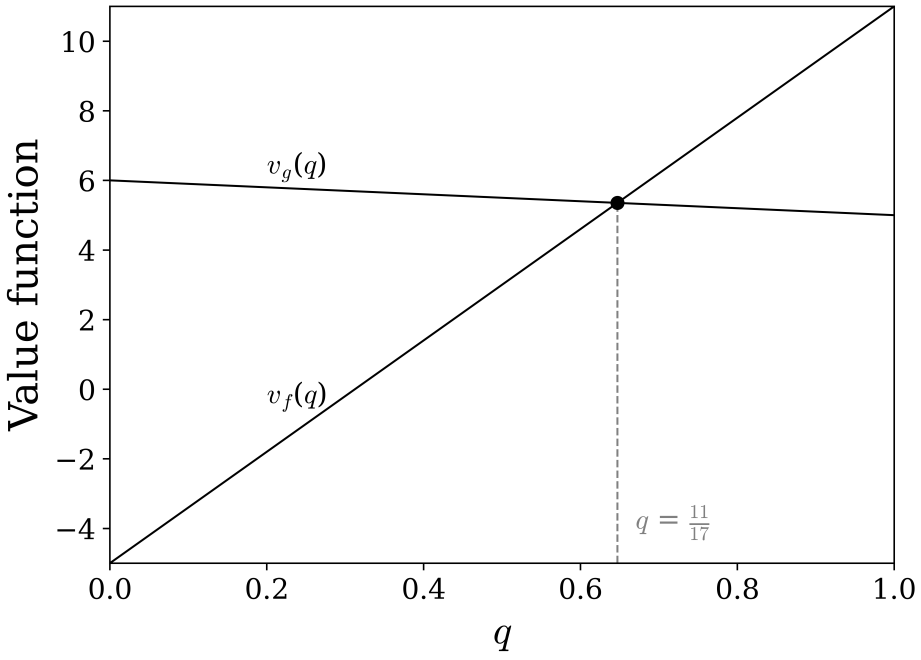


Figure 8.3: Plot of v_f and v_g as functions of q for $q \in [0, 1]$ for terminal reward $x = 10$ and $y = 0$.

5. Since the decision maker cannot observe the hidden state, the choice of decision rule must be based on the probability distribution $\mathbf{b}_1$, or equivalently, q.

6. The above calculations do not use any information regarding the observed signal, which does not become known to the decision maker until the second decision epoch, after the transition from X_1 to X_2. This signal will be used to update the subsequent belief state.

Updating the belief probability at decision epoch 2

The probability distribution $\mathbf{b}_1$ reflects the decision maker's "belief" as to what the current state is. The probability distribution of the hidden state in decision epoch 2, $\mathbf{b}_2$ can be easily calculated assuming that $\mathbf{b}_1$ and the signal at decision epoch 2 are known. In this example, that means that the calculation depends on the observed signal o_1 or o_2 and whether the POMDP decision rule informed the Markov decision process controller to chose the decision rule f or g at decision epoch 1.

Let X_1 be a random variable with distribution $\mathbf{b}_1 = (q, 1 - q)$ and $b_2(s)$ be the probability that the hidden state is s at decision epoch 2. Suppose that for a specific value of $\mathbf{b}_1$ the decision maker uses decision rule f at decision epoch 1 and observes o_1

at decision epoch 2. Then $b_2(s_1)$ and $b_2(s_2)$ can be calculated as follows:

$$\begin{aligned} b_2(s_1) &= P[X_2 = s_1 | Z_2 = o_1, Y_1 = f(X_1), X_1] \\ &= \frac{P[Z_2 = o_1 | X_2 = s_1, Y_1 = f(X_1), X_1] P[X_2 = s_1 | Y_1 = f(X_1), X_1]}{P[Z_2 = o_1 | Y_1 = f(X_1), X_1]} \\ &= \frac{u(o_1|s_1)p(s_1|s_1, a_{1,1})b_1(s_1) + u(o_1|s_1)p(s_1|s_2, a_{2,1})b_1(s_2)}{\sum_{s \in \{s_1, s_2\}} (u(o_1|s)p(s|s_1, a_{1,1})b_1(s_1) + u(o_1|s)p(s|s_2, a_{2,1})b_1(s_2))} \\ &= \frac{0.8 \cdot 0.8q + 0.8 \cdot 0(1-q)}{\big(0.8 \cdot 0.8q + 0.8 \cdot 0(1-q)\big) + \big(0.4 \cdot 0.2q + 0.4 \cdot 1(1-q)\big)} \\ &= \frac{0.64q}{0.4 + 0.32q} \end{aligned} \tag{8.8}$$

and

$$b_2(s_2) = 1 - b_2(s_1) = \frac{0.4 - 0.32q}{0.4 + 0.32q}.$$

To make this concrete, when $b_1(s_1) = q = 0.2$, $b_2(s_1) = 0.128/0.464 = 0.276$. This means that after using f at decision epoch 1 and observing o_1 at decision epoch 2, the belief that the hidden state is s_1 has increased from 0.2 to 0.276. Similarly if $q = 1$, $b_2(s_1) = 0.64/0.72 = 0.889$. Note that when $q = 0$, $b_2(s_1) = 0$ regardless of what signal is observed.

Similar calculations apply when o_2 is observed, or when decision rule g is used instead of f. These calculations are left as exercises for the reader. Thus, for each region in Figure 8.3, corresponding to either decision rule f or g, $\mathbf{b}_2$ assumes one of two potential values. This value is determined by the observed outcome, o_1 or o_2.

The astute reader will note that the calculations shown in (8.8) are precisely those described by *Bayes' Theorem*. Moreover, conditional on the observation in decision epoch 2, the probability distribution of the state in decision epoch 2 is a deterministic function of the probability distribution in decision epoch 1. This process is referred to as *Bayesian updating*. It use is formalized in the next section.

8.2 Information and belief states

If the POMDP decision maker chooses an action based on *all* information that is available at a given epoch, including all prior signals and actions, that would be analogous to using a history-dependent decision rule in a fully observable Markov decision process. As the process evolves, this information vector would grow and quickly make analysis unwieldy.

Instead, a succinct representation of this information is needed, so that decision making in a POMDP can be done in a similar fashion as using a Markovian decision rule in a MDP, in which the current state alone contains the relevant information needed to choose an action. It turns out that the information encoded in the belief state is sufficient for decision making.

Information states

Up to now the discussion of belief states has been quite informal. The following definitions makes this concept precise.

Definition 8.1. Let $a^1, \ldots, a^{n-1}$ and $o^2, \ldots, o^n$ respectively denote the sequence of actions chosen and signals observed by the POMDP decision maker. Define an *information state* to be a vector I_n for $n \geq 1$ represented by

$$I_n := \begin{cases} \mathbf{b}_1 & n = 1 \\ (o^n, a^{n-1}, \ldots, o^2, a^1, \mathbf{b}_1) & n \geq 2. \end{cases} \tag{8.9}$$

Note that:

1. The information state I_n contains all knowledge available to the POMDP decision maker at decision epoch n.

2. The information state I_n assumes values in the set of all realizations of the signal and action choice processes $(Z_2, \ldots, Z_n)$ and $(Y_1, \ldots, Y_{n-1})$. Moreover, since action choice at the first decision is based on $\mathbf{b}_1$, this quantity is appended to the information vector.

3. Note that I_n satisfies the recursion

$$I_{n+1} = (o^{n+1}, a^n, I_n) \tag{8.10}$$

for $n = 1, 2, \ldots$.

4. The information vector in a POMDP differs from the history vector in a fully observable Markov decision processin that the observed state is replaced by the observed signal. Therefore, I_n replaces the history H_n in this setting[9].

5. Most descriptions of POMDPs in the literature include $\mathbf{b}_1$ in the model formulation.

Belief states

Belief states describe the POMDP decision maker's assessment of the distribution of states in the hidden Markov decision process at each decision epoch.

[9] Although I_n is a vector, it is unbolded so as to avoid confusion with the identity matrix $\mathbf{I}$, as well as to remain consistent with the notation for the history H_n, which plays an analogous role in the fully observable Markov decision process model.

Definition 8.2. For each $s \in S$ and probability distribution on S, $\mathbf{b}_1$, define the *belief state* $\mathbf{b}_n$ to be a vector with components

$$b_n(s) := P[X_n = s | I_n] \tag{8.11}$$

for $s \in S$ and $n \geq 2$.

Some comments about the belief state follow:

1. Suppose $a^1, \ldots, a^{n-1}$ and $o^2, \ldots, o^n$ denote the sequence of actions and signals available to the POMDP decision maker at decision epoch $n \geq 2$. Then

$$b_n(s) = P[X_n = s | Z_n = o^n, Y_{n-1} = a^{n-1}, \ldots, Z_2 = o^2, Y_1 = a^1, \mathbf{b}_1]$$

2. The belief state summarizes the decision maker's beliefs about what the system state is at a given decision epoch. It is a function of past actions and observations, and not the hidden states. In other words, it is a function of only the observable parts of the process to the decision maker.

3. Although $b_n(s)$ explicitly depends on the past sequence of actions and observations, they are suppressed in the notation.

4. The first signal is observed prior to decision epoch 2. There is no signal at decision epoch 1; that decision is based entirely on the initial state distribution $b_1(s) = P[X_1 = s]$.

5. For each n and $s \in S$, $0 \leq b_n(s) \leq 1$ and $\sum_{s \in S} b_n(s) = 1$. When S is a finite set with M elements, $b_n(\cdot)$ is an element of the $(M-1)$-dimensional unit simplex, $\mathcal{S}_{M-1}$, defined by

$$\mathcal{S}_{M-1} := \left\{ (x_1, \ldots, x_M) \;\middle|\; \sum_{i=1}^{M} x_i = 1, 0 \leq x_i \leq 1 \text{ for } i = 1, 2, \ldots, M \right\}. \tag{8.12}$$

6. In the two-state example above, $\mathcal{S}_1 = [0, 1]$, so $b_1(s_1) = q$ for a scalar $q \in [0, 1]$. In a three-state model, $\mathcal{S}_2$ is an equilateral triangle with sides of length 1 and in a four-state model, it is a tetrahedron.

A belief state recursion

The following fundamental result shows that in a POMDP:

1. The current belief state is a function of the previous belief state, the previous action and the current signal.

2. There is an explicit formula for updating the belief state.

What this means is that conditional on the belief state, no extra information about the hidden state of the system can be obtained by including the entire sequence of past actions and observations.

Theorem 8.1. For $n \geq 1$ the components of $\mathbf{b}_{n+1}$ satisfy

$$b_{n+1}(s) = P[X_{n+1} = s|I_{n+1}] = P[X_{n+1} = s|Z_{n+1} = o^{n+1}, Y_n = a^n, I_n] \quad (8.13)$$

for all $s \in S$. Moreover,

$$b_{n+1}(s) = \frac{u(o^{n+1}|s, a^n) \sum_{j \in S} p(s|j, a^n) b_n(j)}{\sum_{s' \in S} u(o^{n+1}|s', a^n) \sum_{j \in S} p(s'|j, a^n) b_n(j)}. \quad (8.14)$$

Proof. Choose n and apply Bayes' Theorem to obtain,

$$\begin{aligned} b_{n+1}(s) &= P[X_{n+1} = s|I_{n+1}] = P[X_{n+1} = s|Z_{n+1} = o^{n+1}, Y_n = a^n, I_n] \\ &= \frac{P[Z_{n+1} = o^{n+1}|X_{n+1} = s, Y_n = a^n, I_n] P[X_{n+1} = s|Y_n = a^n, I_n]}{P[Z_{n+1} = o^{n+1}|Y_n = a^n, I_n]} \end{aligned}$$

By assumption, Z_{n+1} does not depend on I_n, so that

$$P[Z_{n+1} = o^{n+1}|X_{n+1} = s, Y_n = a^n, I_n] = u(o^{n+1}|s, a^n).$$

From the law of total probability $P[X_{n+1} = s|Y_n = a^n, I_n]$ can be written as

$$\begin{aligned} P[X_{n+1} = s|Y_n = a^n, I_n] &= \sum_{j \in S} P[X_{n+1} = s|X_n = j, Y_n = a^n, I_n] P[X_n = j|Y_n = a^n, I_n] \\ &= \sum_{j \in S} p(s|j, a^n) b_n(j), \end{aligned}$$

where

$$P[X_{n+1} = s|X_n = j, Y_n = a^n, I_n] = p(s|j, a^n)$$

since the transition probability in the hidden Markov decision process does not depend on I_n. The second expression in this summation is the probability distribution of the hidden state at decision epoch n which does not depend on the subsequent action Y_n. By definition

$$P[X_n = j|Y_n = a^n, I_n] = P[X_n = j|I_n] = b_n(j)$$

so putting all of this together gives,

$$b_{n+1}(s) = \frac{u(o^{n+1}|s, a^n) \sum_{j \in S} p(s|j, a^n) b_n(j)}{\sum_{s' \in S} u(o^{n+1}|s', a^n) \sum_{j \in S} p(s'|j, a^n) b_n(j)}.$$

which establishes both results in the theorem. □

Thus, computing $\mathbf{b}_{n+1}$ requires only the most recent signal, action, and belief state[10]. Equation (8.14) shows that the updated belief state is computed by the following four-step process.

1. The decision maker chooses action a^n on the basis of the belief state $\mathbf{b}_n$ at decision epoch n.

2. The hidden process moves to state s according to the probability distribution $\sum_{j\in S} p(s|j,a^n)b_n(j)$.

3. The system generates the signal o^{n+1} from the probability distribution $u(\cdot|s,a^n)$.

4. The decision maker updates the belief state to $\mathbf{b}_{n+1}$ using (8.14).

Note that steps 2 and 3 are probabilistic, while step 4 is deterministic. Moreover, the numerator of (8.14) equals the probability that the hidden state is s and that the decision maker observes o^{n+1}, while the denominator equals the probability that the decision maker observes o^{n+1}, independent of s. These computations also show that $b_{n+1}(s)$ takes into account all the information available to the decision maker at decision epoch $n+1$, through $\mathbf{b}_n$, the action at decision epoch n and the observation o^{n+1} at decision epoch $n+1$.

As a consequence of this result, the POMDP dynamics can be expressed in terms of the belief state as follows. After observing signal o^n in decision epoch $n \geq 2$, the decision maker updates the belief state using equation (8.14), and then chooses an action. The rest of the dynamics remain unchanged. Figure 8.4 illustrates how the belief state enters into the sequence of events in period n. At decision epoch n, the decision maker observes signal o^n, updates the belief state $\mathbf{b}_n$ using (8.14) and chooses action a^n. Then prior to decision epoch $n+1$ the hidden system transitions to s^{n+1} and generates a signal o^{n+1}, which the decision maker then uses to update the belief state to $\mathbf{b}_{n+1}$.

8.3 Transforming a POMDP into a Markov decision process

This section shows how to transform a POMDP with a finite state space into an a fully observable MDP with a continuous state space, namely, the $(|S|-1)$-dimensional unit simplex, or equivalently, the set of all probability distributions on S. To evaluate policies and find optimal policies in a POMDP, the POMDP is converted into an MDP in which the belief state of the POMDP becomes the state of the equivalent MDP. Accordingly, the action sets, rewards, and transition probabilities must be redefined in terms of the belief states.

[10]This is related to the concept of a *sufficient statistic*. More formally one needs to establish that optimal value functions in a POMDP only depend on the history through the belief state.

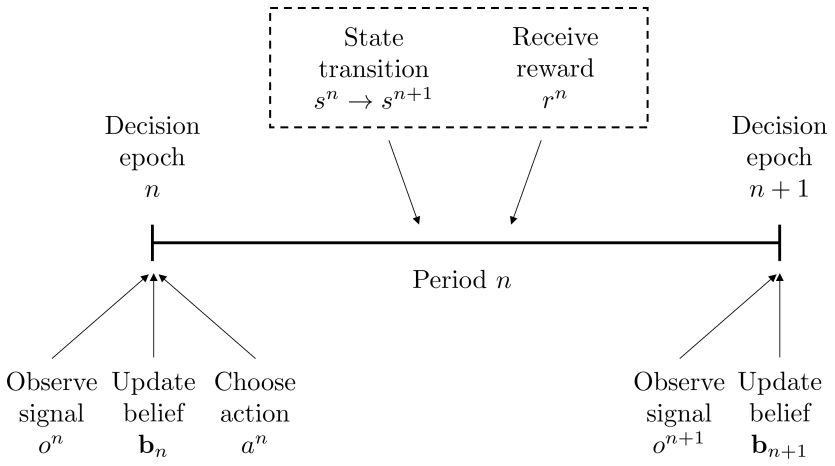


Figure 8.4: Timeline for a POMDP in period $n > 1$. The decision maker observes the signal, the updated belief, and the action choice. Events in the dashed box, namely the state transition and reward, are unobservable. Note that at decision epoch 1, o^1 is superfluous.

Note that there are two interrelated Markov decision processes under consideration, the original hidden model underlying the POMDP model, and the equivalent Markov decision process formulation of the POMDP model. The former is referred to as the *underlying Markov decision process* and the latter the *derived Markov decision process*. A tilde will be used to distinguish entities in the derived Markov decision process from those in the underlying Markov decision process. For example, the derived state space will be written $\tilde{S}$.

8.3.1 The derived Markov decision process

Recall that a Markov decision process is defined by its decision epochs, states, actions, rewards and transitions probabilities. In constructing the derived Markov decision process of a POMDP, assume S and O are discrete and finite, and written as $S = \{s_1, \ldots, s_M\}$ and $O = \{o_1, \ldots, o_K\}$.

Decision epochs: Decision epochs correspond to the point in time immediately after the decision maker updates the belief state. The model may be either finite or infinite horizon.

$$\tilde{T} = \{1, 2, \ldots, N\}, \quad N \leq \infty.$$

States: Assuming S has M elements, then $\tilde{S}$ is the $(M-1)$-dimensional unit simplex, that is

$$\tilde{S} = \mathcal{S}_{M-1}.$$

The states of the derived Markov decision process correspond to the belief states in the POMDP and are denoted by $\mathbf{b}$. Thus, $\tilde{S}$ is an uncountable and compact set even when S is finite.

When the underlying Markov decision process has two states, $\tilde{S}$ is the unit interval, when it has three states, it is equivalent to an equilateral triangle in the plane and when S has four states, $\tilde{S}$ is equivalent to a tetrahedron with equal sides.

Actions: Since the POMDP decision maker is unable to observe the hidden state, an action in the POMDP must correspond to a decision rule that *can be implemented* in the underlying Markov decision process.

In most applications, A_s will not vary with s, that is $A_s = A$ for all $s \in S$, so that selecting an action in the POMDP will result in implementing the same action in all hidden states of the underlying Markov decision process. In this case,

$$\tilde{A} = A.$$

This specification applies to the clinical example described above in which the true health state of a patient is unknown to the POMDP decision maker and the inspection problem in Section 8.6.1 below.

In contrast, when A_s varies with s such as in Example 8.1.2, a POMDP action must specify what action the MDP controller should take in each hidden state. In other words, an action in the derived Markov decision process is a decision rule in the hidden MDP. This means that it is a vector of length M that specifies an action for each hidden state. In this case,

$$\tilde{A} \subseteq \prod_{s \in S} A_s = \{(a_{s_1}, a_{s_2}, \ldots, a_{s_M}) \mid a_{s_m} \in A_{s_m}, m = 1, 2, \ldots, M\}.$$

Note that $\tilde{A}$ is specified to be a subset of $\Pi_{s \in S} A_s$ to emphasize that it may only contain a subset of all possible decision rules in the underlying Markov decision process. Since $\tilde{A}$ consists of vectors of length M, actions in the derived Markov decision process will be written as vectors $\mathbf{a}$, with components a_s for each $s \in S$. The two-state model in Section 8.1.2 provides an example in which an action in the POMDP corresponds to a (Markovian deterministic) decision rule in the underlying MDP and the set of possible actions in the derived MDP is a proper subset of the set of all decision rules in the hidden MDP.

The development in this chapter will continue with the specification that actions in the POMDP are decision rules in the MDP. However, examples where $A_s = A$ for all $s \in S$ will be highlighted to illustrate the simpler formulation.

Rewards: In the derived Markov decision process, rewards and transition probabilities of the hidden MDP must be converted to functions of the belief state $\mathbf{b}$ which is the state of the derived MDP. This is done as follows.

Since the reward in the underlying MDP, $r(s, a, j)$, is a function of hidden states s and j, it can be converted to a function of the belief state $\mathbf{b}$ as follows. Define $\tilde{r}(\mathbf{b}, \mathbf{a})$ for $\mathbf{b} \in \tilde{S}$ and $\mathbf{a} \in \tilde{A}$ by

$$\tilde{r}(\mathbf{b}, \mathbf{a}) = \sum_{s \in S} \sum_{j \in S} r(s, a_s, j) p(j|s, a_s) b(s). \tag{8.15}$$

In the above expression, the inner sum (over j) gives the expected reward conditional on the hidden system occupying state s and choosing action a_s. The outer sum (over s) takes into account that s is unobservable and that the decision maker enters the decision epoch with the belief state $\mathbf{b}$. The example in Section 8.1.2 carries out the calculation implicit in (8.15) in two steps.

When the underlying reward function is independent of j, (8.15) simplifies to

$$\tilde{r}(\mathbf{b}, \mathbf{a}) = \sum_{s \in S} r(s, a_s) b(s). \tag{8.16}$$

Equations (8.15) and (8.16) show that $\tilde{r}(\mathbf{b}, \mathbf{a})$ is **linear** in $\mathbf{b}$. This observation will play a key role in subsequent computations. Note also that even when the underlying reward depends on the subsequent state of the hidden process as in (8.15), $\tilde{r}(\mathbf{b}, \mathbf{a})$ depends only on the belief state at the current decision epoch and not on the belief state at the subsequent decision epoch.

For $\mathbf{b} \in \tilde{S}$, the terminal reward in a finite horizon model becomes

$$\tilde{r}_N(\mathbf{b}) = \sum_{s \in S} r_N(s) b(s). \tag{8.17}$$

Transition probabilities: Introducing some additional notation simplifies and provides insight into some of the expressions below. For each $\mathbf{b} \in \tilde{S}$ and $\mathbf{a} \in \tilde{A}$, define the probability of observing $o \in O$ by

$$\eta(o|\mathbf{b}, \mathbf{a}) := \sum_{s' \in S} \sum_{s \in S} u(o|s', a_s) p(s'|s, a_s) b(s). \tag{8.18}$$

Let the updated belief state be represented by $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$. It is a vector of dimension M and depends on o, $\mathbf{a}$, and $\mathbf{b}$. Therefore, it follows from (8.14) that the j-th component of the updated belief state when observing $o \in O$ is given by

$$\tau_j(o, \mathbf{a}, \mathbf{b}) := \frac{\sum_{s \in S} u(o|j, a_s) p(j|s, a_s) b(s)}{\sum_{s' \in S} \sum_{s \in S} u(o|s', a_s) p(s'|s, a_s) b(s)} = \frac{\sum_{s \in S} u(o|j, a_s) p(j|s, a_s) b(s)}{\eta(o|\mathbf{b}, \mathbf{a})}. \tag{8.19}$$

Consequently

$$\tilde{p}(\mathbf{b}'|\mathbf{b}, \mathbf{a}) = \begin{cases} \eta(o|\mathbf{b}, \mathbf{a}) & \text{when } b'(j) = \tau_j(o, \mathbf{a}, \mathbf{b}) \text{ for all } j \in S \text{ and for each } o \in O \\ 0 & \text{otherwise.} \end{cases} \tag{8.20}$$

Hence, the probability of observing a particular belief state is equal to the probability of observing the specific signal that leads to that belief state.

Observe that when O has K elements, $\tilde{p}(\mathbf{b}'|\mathbf{b}, \mathbf{a})$ assumes at most K non-zero values, at most one for each $o \in O$. Moreover it follows from (8.14) that the next belief state is a deterministic function of the current belief state and signal.

Table 8.1 summarizes the relationship between the elements in the hidden MDP and the derived MDP.

Element	Hidden MDP	Derived MDP
State	s	$\mathbf{b}$
Action (A_s independent of s)	a	a
Action (A_s varies with s)	a	$\mathbf{a} = (a_{s_1}, \ldots, a_{s_M})$
Rewards	$r(s, a)$	$\tilde{r}(\mathbf{b}, \mathbf{a})$
Transition probabilities	$p(j\|s, a)$	$\tilde{p}(\mathbf{b}'\|\mathbf{b}, \mathbf{a})$

Table 8.1: Relationship between entities in the hidden and derived Markov decision processes.

Matrix notation

Much of the literature represents the rewards and transition probabilities using the following matrix notation. It simplifies calculations and makes the dimensions of various objects more apparent.

$\mathbf{b}$: Column vector with components $b_{s_1}, \ldots, b_{s_M}$.

$\mathbf{P_a}$: $M \times M$ matrix with its (s, j)-th component equal to $p(j|s, a_s)$[a].

$\mathbf{R_a}$: $M \times M$ matrix with (s, j)-th entry $r(s, a_s, j)$.

$\mathbf{r_a}$: M-dimensional column vector with components $r(s, a_s)$ for all $s \in S$.

$\mathbf{r}_N$: M-dimensional column vector with components $r_N(s)$ for all $s \in S$ (for the finite horizon setting).

$\mathbf{U}^o_{\mathbf{a}}$: $M \times M$ **diagonal** matrix with entries $u(o|s_1, a_{s_1}), \ldots, u(o|s_M, a_{s_M})$.

$\mathbf{e}$: M-component column vector with all entries equal to 1.

[a] This definition assumes the use of deterministic decision rules in the derived MDP. It can be extended to allow randomized decision rules. Randomized rules arise when applying policy gradient algorithms (Chapter 11) to a POMDP.

When the reward function of the underlying MDP is $r(s, a, j)$, then from (8.15)

$$\tilde{r}(\mathbf{b}, \mathbf{a}) = \text{diag}(\mathbf{R_a P_a^\mathsf{T}})\mathbf{b}, \tag{8.21}$$

where $\text{diag}(\mathbf{R_a P_a^\mathsf{T}})$ denotes an $M \times M$ diagonal matrix containing the diagonal elements of $\mathbf{R_a P_a^\mathsf{T}}$, that is, the elements $(\mathbf{R_a P_a^\mathsf{T}})_{j,j}$ for $j = 1, \ldots, M$. When the underlying reward function $r(s, a)$ does not depend on j, then from (8.16)

$$\tilde{r}(\mathbf{b}, \mathbf{a}) = \mathbf{r_a^\mathsf{T}}\mathbf{b}. \tag{8.22}$$

In this notation,

$$\eta(o|\mathbf{b}, \mathbf{a}) = \mathbf{e}^\mathsf{T}\mathbf{U_a^o P_a^\mathsf{T}}\mathbf{b} \quad \text{and} \quad \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}) = \frac{\mathbf{U_a^o P_a^\mathsf{T}}\mathbf{b}}{\eta(o|\mathbf{b}, \mathbf{a})}. \tag{8.23}$$

It is important to emphasize that $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ is a vector with $|S|$-components and $\eta(o|\mathbf{b}, \mathbf{a})$ is a scalar in $[0, 1]$.

The following expression summarizes this calculation:

$$\tilde{p}(\mathbf{b}'|\mathbf{b}, \mathbf{a}) = \begin{cases} \eta(o|\mathbf{b}, \mathbf{a}) & \text{when } \mathbf{b}' = \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}) \text{ for each } o \in O \\ 0 & \text{otherwise.} \end{cases} \tag{8.24}$$

Equation (8.24) shows that the only values for $\mathbf{b}'$ with non-zero probability are those represented by $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ for $o \in O$. When O has K elements, there are at most K distinct vectors $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$.

8.3.2 The two-state model revisited

This section illustrates the derived MDP model for the two-state POMDP.

Decision epochs:

$$\tilde{T} = \{1, 2, \ldots, N\}, \quad N \leq \infty$$

States:

$$\tilde{S} = \{(q_1, q_2) \mid q_1 + q_2 = 1, q_1 \geq 0, q_2 \geq 0\} = \mathcal{S}_1$$

For each i, $i = 1, 2$, q_i represents the probability that the hidden state is s_i. Note that the state space can be equivalently represented as

$$\{q \mid 0 \leq q \leq 1\} = [0, 1],$$

where q is the probability that the hidden state is s_1, as shown in Section 8.1.2. Below, $\mathcal{S}_1$ is used to emphasize the fact that the belief state is a probability distribution with two components.

Actions: Consider an action set corresponding to the two decision rules defined in equations (8.2) and (8.3).

$$\tilde{A} = \{(a_{1,1}, a_{2,1}), (a_{1,2}, a_{2,2})\} = \{f, g\}.$$

The first component of each action in $\tilde{A}$ denotes the action used when the hidden process is in state s_1 and the second component denotes the action used if the hidden process is in state s_2. Note that this $\tilde{A}$ is a proper subset of the set of Markovian deterministic decision rules in the underlying two-state Markov decision process. That is, the decision rules $(a_{1,1}, a_{2,2})$ and $(a_{1,2}, a_{2,1})$ are excluded.

Rewards: The reward when the decision maker chooses action $(a_{1,1}, a_{2,1}) \in \tilde{A}$ after computing $\mathbf{b} = (q_1, q_2) \in \tilde{S}$ is

$$\begin{aligned}\tilde{r}((q_1, q_2), (a_{1,1}, a_{2,1})) &= (r(s_1, a_{1,1}, s_1)p(s_1|s_1, a_{1,1}) + r(s_1, a_{1,1}, s_2)p(s_2|s_1, a_{1,1}))q_1 \\ &\quad + (r(s_2, a_{2,1}, s_1)p(s_1|s_2, a_{2,1}) + r(s_2, a_{2,1}, s_2)p(s_2|s_2, a_{2,1}))q_2 \\ &= (5 \cdot 0.8 + (-5)0.2)q_1 + (0 \cdot 0 + (-5)1)q_2 \\ &= 3q_1 - 5q_2\end{aligned} \tag{8.25}$$

Observe that the reward is a linear function of the vector (q_1, q_2). Since $q_2 = 1 - q_1$ the reward reduces to $-5 + 8q_1$. It is left to the reader to show that $\tilde{r}\big((q_1, q_2), (a_{1,2}, a_{2,2})\big) = 5q_1 + 2q_2$.

Transition probabilities: The probability that the hidden state occupies s_1 when the decision maker chooses action $(a_{1,1}, a_{2,1}) \in \tilde{A}$ given belief state $\mathbf{b} = (q_1, q_2) \in \tilde{S}$ is calculated as follows. From (8.18), the probability of observing o_1 is given by

$$\begin{aligned}\eta(o_1|(q_1, q_2), (a_{1,1}, a_{2,1})) &= u(o_1|s_1)(p(s_1|s_1, a_{1,1})q_1 + p(s_1|s_2, a_{2,1})q_2) \\ &\quad + u(o_1|s_2)(p(s_2|s_1, a_{1,1})q_1 + p(s_2|s_2, a_{2,1})q_2) \\ &= 0.8(0.8q_1 + 0q_2) + 0.4(0.2q_1 + 1q_2) = 0.72q_1 + 0.4q_2\end{aligned}$$

and the probability of observing o_2 equals $0.28q_1 + 0.6q_2$. Then, from (8.19), the probability the hidden state equals s_1 is given by

$$\begin{aligned}\tau_{s_1}(o_1, (a_{1,1}, a_{2,1}), (q_1, q_2)) &= \frac{u(o_1|s_1)(p(s_1|s_1, a_{1,1})q_1 + p(s_1|s_2, a_{2,1})q_2)}{\eta(o_1(q_1, q_2), (a_{1,1}, a_{2,2}))} \\ &= \frac{0.8(0.8q_1 + 0q_2)}{0.72q_1 + 0.4q_2} = \frac{0.64q_1}{0.72q_1 + 0.4q_2}.\end{aligned} \tag{8.26}$$

The probability the hidden state equals s_2 is given by $1 - \tau_{s_1}(o_1, (a_{1,1}, a_{2,1}), (q_1, q_2))$, which becomes

$$\tau_{s_2}(o_1, (a_{1,1}, a_{2,1}), (q_1, q_2)) = \frac{0.08q_1 + 0.4q_2}{0.72q_1 + 0.4q_2}. \tag{8.27}$$

Hence the updated belief state when observing o_1 is

$$\boldsymbol{\tau}(o_1, (a_{1,1}, a_{2,1}), (q_1, q_2)) = \left(\frac{0.64q_1}{0.72q_1 + 0.4q_2}, \frac{0.08q_1 + 0.4q_2}{0.72q_1 + 0.4q_2} \right) \tag{8.28}$$

and when observing o_2 is

$$\boldsymbol{\tau}(o_2, (a_{1,1}, a_{2,1}), (q_1, q_2)) = \left(\frac{0.16q_1}{0.28q_1 + 0.6q_2}, \frac{0.12q_1 + 0.6q_2}{0.28q_1 + 0.6q_2} \right). \tag{8.29}$$

Putting it all together, the transition probabilities in the derived model satisfy

$$\tilde{p}((q_1', q_2')|(q_1, q_2), (a_{1,1}, a_{2,1})) = \begin{cases} 0.72q_1 + 0.4q_2 & \text{for } (q_1', q_2') = \boldsymbol{\tau}(o_1, (a_{1,1}, a_{2,1}), (q_1, q_2)) \\ 0.28q_1 + 0.6q_2 & \text{for } (q_1', q_2') = \boldsymbol{\tau}(o_2, (a_{1,1}, a_{2,1}), (q_1, q_2)) \\ 0 & \text{otherwise,} \end{cases} \tag{8.30}$$

so that $\tilde{p}(\cdot|(q_1, q_2), (a_{1,1}, a_{2,1}))$ is indeed a transition probability. It only takes two positive values, one for each observation.

To provide more insight into these calculations, suppose the belief state is $(0.2, 0.8)$, the decision maker chooses action $(a_{1,1}, a_{2,1})$, and observes o_1. Figure 8.5 shows that the subsequent belief state, given by (8.28), equals $(0.276, 0.724)$. This belief state is observed by the decision maker with probability 0.464. If instead the decision maker observed o_2, the subsequent belief state becomes $(0.06, 0.94)$ and is observed with probability 0.536. It is left to the reader to compute the values of the belief state, corresponding to using action $(a_{1,2}, a_{2,2})$.

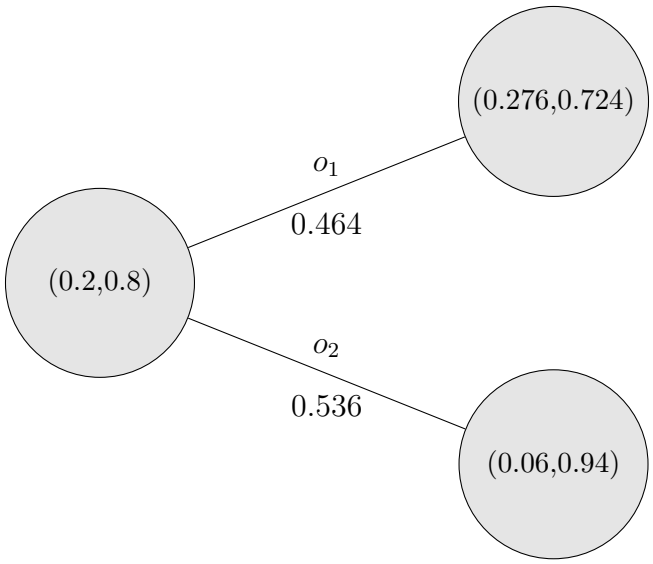


Figure 8.5: Change in belief states when using action $(a_{1,1}, a_{2,1})$.

Matrix representation

The matrix representation for the entities in the above model follow. For action $\mathbf{a} = \mathbf{a}_1 := (a_{1,1}, a_{2,1})$:

$$\mathbf{P}_{\mathbf{a}_1} = \begin{bmatrix} 0.8 & 0.2 \\ 0 & 1 \end{bmatrix}, \ \mathbf{U}^{o_1}_{\mathbf{a}_1} = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.4 \end{bmatrix}, \ \mathbf{U}^{o_2}_{\mathbf{a}_1} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.6 \end{bmatrix}, \ \mathbf{R}_{\mathbf{a}_1} = \begin{bmatrix} 5 & -5 \\ 0 & -5 \end{bmatrix}. \quad (8.31)$$

When $\mathbf{a} = \mathbf{a}_2 := (a_{1,2}, a_{2,2})$,

$$\mathbf{P}_{\mathbf{a}_2} = \begin{bmatrix} 0 & 1 \\ 0.4 & 0.6 \end{bmatrix}, \ \mathbf{U}^{o_1}_{\mathbf{a}_2} = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.4 \end{bmatrix}, \ \mathbf{U}^{o_2}_{\mathbf{a}_2} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.6 \end{bmatrix}, \ \mathbf{R}_{\mathbf{a}_2} = \begin{bmatrix} 0 & 5 \\ 20 & -10 \end{bmatrix}. \quad (8.32)$$

Note that $\mathbf{U}^o_{\mathbf{a}}$ is the same for both actions since the observation depends on the hidden state and not the action. Using these quantities, it is left to the reader to evaluate $\eta(o|\mathbf{b}, \mathbf{a})$ and $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ from (8.23).

As noted above, the observation process in the two-state POMDP model is passive. Suppose there is a costly action $\bar{\mathbf{a}}$ defined by

$$\mathbf{P}_{\bar{\mathbf{a}}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \ \mathbf{U}^{o_1}_{\bar{\mathbf{a}}} = \begin{bmatrix} 0.9 & 0 \\ 0 & 0.05 \end{bmatrix}, \ \mathbf{U}^{o_2}_{\bar{\mathbf{a}}} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.95 \end{bmatrix}, \ \mathbf{r}_{\bar{\mathbf{a}}} = \begin{bmatrix} -15 \\ -20 \end{bmatrix}$$

The presence of this action adds an active information acquisition possibility.

8.4 A finite horizon model

This section provides a framework for optimization in a finite horizon POMDP model. It presents the Bellman equation and a backwards induction algorithm to determine optimal values and policies. Using the assumption that the set of observations is finite, it establishes the piecewise linearity and convexity of the optimal value function and describes some implementable solution methods. This development uses results from Chapter 4. It assumes that S and A_s for each $s \in S$ are finite (and discrete) and stationary to simplify exposition.

8.4.1 Optimality criterion

As in the fully observable model, the decision maker seeks to maximize

$$v^\pi(\mathbf{b}) := E^\pi \left[\sum_{n=1}^{N-1} r_n(X_n, Y_n) + r_N(X_N) \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right] \quad (8.33)$$

over history-dependent randomized policies $\pi = (d_1, \ldots, d_{N-1})$. We emphasize that in this definition, d_n is a *history-dependent randomized decision rule in the POMDP*, which means it is a function of the information at decision epoch n, namely $I_n = (o^n, a^{n-1}, \ldots, o^2, a^1, \mathbf{b}_1)$ as defined in (8.9).

Note that in this definition, the policy is based on the entities that are observable in the POMDP, while the quantity in the expectation is based on unobservable hidden states. The following theorem addresses this apparent mismatch.

Fundamentals

The following important result shows that the above expression is equivalent to the related expression in the derived Markov decision process.

Theorem 8.2. Suppose for $\pi \in \Pi^{\text{HR}}$, $v^\pi(\mathbf{b})$ is defined by (8.33). Then

$$v^\pi(\mathbf{b}) = E^\pi \left[\sum_{n=1}^{N-1} \tilde{r}(\mathbf{b}_n, Y_n) + \tilde{r}_N(\mathbf{b}_N) \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right]. \tag{8.34}$$

The proof follows by noting that

$$\begin{aligned} E^\pi \left[r_n(X_n, Y_n) \,|\, \mathbf{b}_1 = \mathbf{b} \right] =& E^\pi \left[E^\pi \left[r_n(X_n, Y_n) \,|\, I_n \right] \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right] \\ =& E^\pi \left[\sum_{s \in S} r(s, Y_n)\ b_n(s) \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right] \\ =& E^\pi [\, \tilde{r}(\mathbf{b}_n, Y_n) \,|\, \mathbf{b}_1 = \mathbf{b} \,], \end{aligned} \tag{8.35}$$

where the equality in (8.35) is a result of Theorem 8.1. Equivalently,

$$E^\pi \left[r_N(X_N) \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right] = E^\pi \left[\tilde{r}_N(\mathbf{b}_N) \,\middle|\, \mathbf{b}_1 = \mathbf{b} \right].$$

From a different perspective, this theorem can be interpreted as saying that $\mathbf{b}_n$ is a *sufficient statistic* for I_n with respect to the finite horizon policy value function.

Optimal policies and values

An optimal policy and value are defined similarly to fully observable models. The key distinction is in the definition of Π^{HR}, the class of history-dependent randomized policies. In a POMDP, such policies are composed of decision rules that are functions from the set of information states (previous observations and actions) to the set of probability distributions over the action set.

Definition 8.3. An *optimal policy* $\pi^* \in \Pi^{\text{HR}}$ satisfies

$$v^{\pi^*}(\mathbf{b}) \geq v^\pi(\mathbf{b}) \tag{8.36}$$

for all $\pi \in \Pi^{\text{HR}}$ and $\mathbf{b} \in \tilde{S}$.

Definition 8.4. The *value* of the POMDP is defined by

$$v^*(\mathbf{b}) := \sup_{\pi \in \Pi^{\mathrm{HR}}} v^\pi(\mathbf{b}) \tag{8.37}$$

for all $\mathbf{b} \in \tilde{S}$.

The optimal value from decision epoch $n = 1, \ldots, N-1$ to the end of the planning horizon is defined by

$$v_n^*(\mathbf{b}) := \sup_{\pi \in \Pi^{\mathrm{HR}}} v_n^\pi(\mathbf{b}), \tag{8.38}$$

where

$$v_n^\pi(\mathbf{b}) := E^\pi \left[\sum_{i=n}^{N-1} \tilde{r}(\mathbf{b}_i, Y_i) + \tilde{r}_N(\mathbf{b}_N) \,\middle|\, \mathbf{b}_n = \mathbf{b} \right]. \tag{8.39}$$

Moreover, $v_1^*(\mathbf{b}) = v^*(\mathbf{b})$ by definition.

Note that in the derived MDP, history-dependent randomized decision rules are functions of the history $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ while Markovian decision rules are functions of $\mathbf{b}_n$ only.

As a result of Theorem 4.2[11], it suffices to consider Markovian deterministic policies to obtain optimal policies.

Theorem 8.3. There exists $\pi^* \in \Pi^{\mathrm{MD}}$ that achieves the optimal value $v^*(\mathbf{b})$ for all $\mathbf{b} \in \tilde{S}$. That is,

$$v^*(\mathbf{b}) = \max_{\pi \in \Pi^{\mathrm{MD}}} v^\pi(\mathbf{b}). \tag{8.40}$$

8.4.2 The Bellman equation and its solution

In the derived MDP, given a belief state $\mathbf{b}$, choosing an action $\mathbf{a}$ and observing o, the process transitions to a new belief state $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ with probability $\eta(o|\mathbf{b}, \mathbf{a})$ and accrues reward $\tilde{r}(\mathbf{b}, \mathbf{a})$. Define the finite horizon Bellman equation for the derived MDP as follows.

[11] Note that the proof of this result in Chapter 4 applies as well to models with compact state spaces and finite action spaces.

Definition 8.5. In a POMDP, the *Bellman equation*[a] for the derived MDP refers to the system of equations:

$$u_n(\mathbf{b}) = \max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) u_{n+1}(\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})) \right\} \tag{8.41}$$

for $n = 1,\ldots,N-1$ together with the boundary condition $u_N(\mathbf{b}) = \tilde{r}_N(\mathbf{b})$.

[a]Following the convention used throughout this book, the singular expression "Bellman equation" is used to refer to the system of equations (8.41) and its boundary condition.

A conceptual algorithm

Since $\tilde{A}$ is finite, the maximum in (8.41) is attained. Hence values that satisfy the Bellman equation can be computed by the following backwards induction algorithm. Of course, since the unit simplex $\tilde{S}$ is a non-finite compact set, the algorithm cannot be implemented directly.

One possible implementation is to evaluate (8.41) on a grid of values and interpolate as necessary. Unfortunately, such an approach can be prohibitive in high dimensions. More practical implementable computational algorithms are described later.

Algorithm 8.1. Finite horizon policy optimization for a POMDP

1. **Initialize:** Set $n \leftarrow N-1$ and $u_N(\mathbf{b}) \leftarrow \tilde{r}_N(\mathbf{b})$ for all $\mathbf{b}\in\tilde{S}$.
2. **Iterate:** While $n \geq 1$:
   (a) For all $\mathbf{b}\in\tilde{S}$, evaluate $u_n(\mathbf{b})$ according to

$$u_n(\mathbf{b}) \leftarrow \max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) u_{n+1}(\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})) \right\}, \tag{8.42}$$

   set

$$\tilde{A}^*_{n,\mathbf{b}} \leftarrow \arg\max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) u_{n+1}(\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})) \right\} \tag{8.43}$$

   and select $d_n(\mathbf{b}) \in \tilde{A}^*_{n,\mathbf{b}}$.
   (b) $n \leftarrow n-1$.
3. **Terminate:** Return $u_1(\mathbf{b})$ for all $\mathbf{b}\in\tilde{S}$ and $\pi = (d_1,\ldots,d_{N-1})$.

Some comments about this algorithm follow:

1. The main challenge in implementing this algorithm is that the right hand side of (8.42) must be evaluated for a non-finite number of belief states $\mathbf{b}$. This is addressed in several ways below.

2. Observe that the summation in (8.42) is indexed by the observation set O. This is because transition probabilities in this model, as given by (8.20), only assume a *finite* number of positive values indexed by elements of O.

3. Referring to the fully observable Markov decision process model in Chapter 4, where the backwards induction recursion is of the form

$$u_n(s) = \max_{a \in A_s} \left\{ r_n(s,a) + \sum_{j \in S} p_n(j|s,a) u_{n+1}(j) \right\},$$

one observes that $\eta(o|\mathbf{b},\mathbf{a})$ replaces the transition probability and $\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})$ replaces the subsequent state j.

The following result is the analog of Theorem 4.3 applied to the derived MDP. It is a fundamental result in the theory of POMDPs.

Theorem 8.4. Suppose $\tilde{A}$ is finite and $u_n(\mathbf{b})$ for $n = 1, \ldots, N$ is obtained by applying Algorithm 8.1. Then

1. For $n = 1, \ldots, N$ and $\mathbf{b} \in \tilde{S}$, $u_n(\mathbf{b}) = v_n^*(\mathbf{b})$ and $u_1(\mathbf{b}) = v^*(\mathbf{b})$.
2. Suppose $d_n^*(\mathbf{b}) \in \tilde{A}_{n,\mathbf{b}}^*$ for $n = 1, \ldots, N-1$ and $\mathbf{b} \in \tilde{S}$. Then $\pi^* = (d_1^*, \ldots, d_{N-1}^*)$ is an optimal policy.
3. Moreover, $v^{\pi^*}(\mathbf{b}) = v^*(\mathbf{b})$, for all $\mathbf{b} \in \tilde{S}$ and $v_n^{\pi^*}(\mathbf{b}) = v_n^*(\mathbf{b})$, for all $\mathbf{b} \in \tilde{S}$ and $n = 1, \ldots, N$.

Policy evaluation

Note by replacing the maximum in (8.42) by actions corresponding $\pi = (d_1, d_2, \ldots, d_{N-1}) \in \Pi^{\text{MD}}$, it follows that $v_n^\pi(\mathbf{b})$ can be determined by the recursion

$$u_n(\mathbf{b}) = \tilde{r}(\mathbf{b}, d_n(\mathbf{b})) + \sum_{o \in O} \eta(o|\mathbf{b}, d_n(\mathbf{b})) u_{n+1}(\boldsymbol{\tau}(o, d_n(\mathbf{b}), \mathbf{b})). \tag{8.44}$$

subject to $u_N(\mathbf{b}) = \tilde{r}_N(\mathbf{b})$ for $n = 1, \ldots, N-1$ and $\mathbf{b} \in \tilde{S}$.

When π is stationary, this recursion (with the inclusion of a discount factor befoe the summation) may be used to evaluate a fixed policy in an infinite horizon discounted model.

8.4.3 Structure of optimal value functions in a POMDP

This section establishes the elegant[12] result that $v_n^*(\mathbf{b})$ is piecewise linear and convex.

Piecewise linear convex functions

This brief section defines and describes the properties of piecewise linear convex functions that are fundamental in the following sections.

Definition 8.6. Let X be a subset of $\mathbb{R}^n$. A real function $f(\mathbf{x})$ on X is said to be *piecewise linear convex (PLC)* if for some finite set, Γ, of n-dimensional vectors:

$$f(\mathbf{x}) = \max_{\boldsymbol{\gamma}\in\Gamma}\{\boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}\}. \tag{8.45}$$

The above definition means that piecewise linear convex functions may be written as the maximum of a finite number of linear functions[13]. Useful properties of piecewise linear convex functions follow, proofs are left as an exercise:

1. If $f(\mathbf{x})$ and $g(\mathbf{x})$ are PLC, then $f(\mathbf{x}) + g(\mathbf{x})$ is PLC.
2. If $f(\mathbf{x})$ and $g(\mathbf{x})$ are PLC, then $\max\{f(\mathbf{x}), g(\mathbf{x})\}$ is PLC.
3. Let $\boldsymbol{\gamma}_{\mathbf{x}} \in \arg\max_{\boldsymbol{\gamma}\in\Gamma}\{\boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}\}$. If $f(\mathbf{x})$ is PLC, then $f(\mathbf{x}) = (\boldsymbol{\gamma}_{\mathbf{x}})^{\mathsf{T}}\mathbf{x}$ for each $\mathbf{x} \in X$.
4. For each $\boldsymbol{\gamma} \in \Gamma$, $G_{\boldsymbol{\gamma}} := \{\mathbf{x} \in X | f(\mathbf{x}) = \boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}\}$ is a closed convex subset of X or the empty set.
5. If Γ is finite, there are finitely many $G_{\boldsymbol{\gamma}}$.

[12] This result appears in Smallwood and Sondik [1973] and was established in Sondik's PhD dissertation.

[13] In $\mathbb{R}^1$, this is equivalent to

$$f(x) = \begin{cases} c_1 x + d_1, & x \in [a_1, b_1) \\ c_2 x + d_2, & x \in [a_2, b_2) \\ \dots & \\ c_m x + d_m, & x \in [a_m, b_m). \end{cases}$$

The fourth property does not exclude the possibility that there may be some $\boldsymbol{\gamma}' \in \Gamma$ that do not attain the maximum in (8.45) for any $\mathbf{x} \in X$. In other words $(\boldsymbol{\gamma}')^{\mathsf{T}}\mathbf{x} < f(\mathbf{x})$ for all $\mathbf{x} \in X$. Figure 8.7 illustrates this observation.

The following additional terminology will facilitate exposition.

Definition 8.7. Refer to elements in Γ as $\boldsymbol{\gamma}$-*vectors* and a vector $\boldsymbol{\gamma}^*$ chosen according to

$$\boldsymbol{\gamma}^* \in \arg\max_{\boldsymbol{\gamma}\in\Gamma}\{\boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}^*\}$$

as a *support*[a] of the convex function $f(\mathbf{x}) = \max_{\boldsymbol{\gamma}\in\Gamma}\{\boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}\}$ at $\mathbf{x}^*$.

[a]Some authors refer to such $\boldsymbol{\gamma}$ as a *subgradient* or *gradient* and $\boldsymbol{\gamma}^{\mathsf{T}}\mathbf{x}$ as a *supporting hyperplane*.

The key structural result

The following theorem establishes the fundamental result regarding the structure of the optimal value function in a POMDP. It provides the basis for all POMDP algorithms based on belief states[14]. Its straightforward proof is informative.

[14]Some simulation-based algorithms seek policies that are functions of the observations only.

Theorem 8.5. Suppose $v_n^*(\mathbf{b})$ satisfies the Bellman equation (8.41) for $n = 1, \ldots, N$ and all $\mathbf{b} \in \tilde{S}$ and $\mathbf{r}_N$, $\mathbf{r}_\mathbf{a}$, $\mathbf{P}_\mathbf{a}$, $\mathbf{U}_\mathbf{a}^o$, and $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ are as defined in Section 8.3.1.

1. Then $v_n^*(\mathbf{b})$ is a piecewise linear convex function of $\mathbf{b}$ for $n = 1, \ldots, N$.
2. For $n = 1, \ldots, N$
$$v_n^*(\mathbf{b}) = \max_{\boldsymbol{\gamma} \in \Gamma_n} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}\}, \tag{8.46}$$
where $\Gamma_N = \{\mathbf{r}_N\}$,
$$\Gamma_n := \left\{ \mathbf{r}_\mathbf{a} + \sum_{o \in O} \mathbf{P}_\mathbf{a} \mathbf{U}_\mathbf{a}^o \boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})} \;\middle|\; \mathbf{a} \in \tilde{A} \right\} \tag{8.47}$$
is finite for $n = 1, \ldots, N-1$, and
$$\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})} \in \arg\max_{\boldsymbol{\gamma} \in \Gamma_{n+1}} \{\boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})\} \tag{8.48}$$
3. For each $n = 1, \ldots, N$, the state space $\tilde{S}$ can be partitioned into at most $|\Gamma_n|$ polyhedra[a]. Furthermore, all states in each polyhedron have the same optimal action.

[a] Recall that a polyhedron is the set of solutions of a finite number of linear inequalities or equivalently the intersection of a finite number of half-spaces. The expression *polyhedra* refers to more than one polyhedron.

Proof. The proof of parts 1 and 2 is by (backwards) induction on n. For $n = N$, $v_N^*(\mathbf{b}) = \mathbf{r}_N^\mathsf{T} \mathbf{b}$, so it is PLC[15].

Assume that $v_m^*(\mathbf{b})$ is PLC and (8.46)-(8.48) hold for $m = n+1, \ldots, N-1$. This means in particular that for each $\mathbf{b}' \in \tilde{S}$
$$v_{n+1}^*(\mathbf{b}') = \max_{\boldsymbol{\gamma} \in \Gamma_{n+1}} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}'\}$$
for a *finite* set Γ_{n+1}. From (8.20), for each $\mathbf{b} \in \tilde{S}$ and $\mathbf{a} \in \tilde{A}$, there are only finitely many values for the subsequent belief state $\mathbf{b}' = \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ at decision epoch n, each corresponding to a different $o \in O$. Thus
$$v_n^*(\mathbf{b}) = \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v_{n+1}^*(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}$$

[15] Actually it is linear.

$$= \max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) \left(\max_{\boldsymbol{\gamma}\in\Gamma_{n+1}} \boldsymbol{\gamma}^{\mathsf{T}} \boldsymbol{\tau}(o,\mathbf{a},\mathbf{b}) \right) \right\} \tag{8.49}$$

$$= \max_{\mathbf{a}\in\tilde{A}} \left\{ \mathbf{r}_{\mathbf{a}}^{\mathsf{T}}\mathbf{b} + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) \left(\max_{\boldsymbol{\gamma}\in\Gamma_{n+1}} \boldsymbol{\gamma}^{\mathsf{T}} \left(\frac{\mathbf{U}_{\mathbf{a}}^{o}\mathbf{P}_{\mathbf{a}}^{\mathsf{T}}\mathbf{b}}{\eta(o|\mathbf{b},\mathbf{a})} \right) \right) \right\} \tag{8.50}$$

$$= \max_{\mathbf{a}\in\tilde{A}} \left\{ \mathbf{r}_{\mathbf{a}}^{\mathsf{T}}\mathbf{b} + \sum_{o\in O} \left(\max_{\boldsymbol{\gamma}\in\Gamma_{n+1}} \boldsymbol{\gamma}^{\mathsf{T}}\mathbf{U}_{\mathbf{a}}^{o}\mathbf{P}_{\mathbf{a}}^{\mathsf{T}}\mathbf{b} \right) \right\} \tag{8.51}$$

$$= \max_{\mathbf{a}\in\tilde{A}} \left\{ \sum_{o\in O} \max_{\boldsymbol{\gamma}\in\Gamma_{n+1}} \left(\frac{\mathbf{r}_{\mathbf{a}}^{\mathsf{T}}}{|O|} + \boldsymbol{\gamma}^{\mathsf{T}}\mathbf{U}_{\mathbf{a}}^{o}\mathbf{P}_{\mathbf{a}}^{\mathsf{T}} \right) \mathbf{b} \right\}$$

$$= \max_{\mathbf{a}\in\tilde{A}} \left\{ \sum_{o\in O} \max_{\boldsymbol{\gamma}\in\Gamma_{n+1}} \left(\frac{\mathbf{r}_{\mathbf{a}}}{|O|} + \mathbf{P}_{\mathbf{a}}\mathbf{U}_{\mathbf{a}}^{o}\boldsymbol{\gamma} \right)^{\mathsf{T}} \mathbf{b} \right\}, \tag{8.52}$$

where (8.49) follows from the induction hypothesis and (8.50) follows from (8.23) and (8.24). The next expression (8.51) follows by canceling out the probability $\eta(o|\mathbf{b},\mathbf{a})$ and the last two expressions follow by rearranging terms. Note that moving $\mathbf{r}_{\mathbf{a}}$ inside the summation and "max" requires dividing it by the number of terms in the summation $|O|$. Since

$$\frac{\mathbf{r}_{\mathbf{a}}}{|O|} + \mathbf{P}_{\mathbf{a}}\mathbf{U}_{\mathbf{a}}^{o}\boldsymbol{\gamma} \tag{8.53}$$

is a vector, each term in (8.52) is PLC. Thus, the expression inside the braces is PLC because the sum of PLC functions is PLC. Finally since the max of PLC functions is PLC, $v_n^*(\mathbf{b})$ is PLC for $n = 1, \ldots, N$.

Defining $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$ by (8.48), substituting it into (8.53) and rearranging terms establishes part 2.

To prove the result in part 3, represent the finite set of vectors in Γ_n by $\{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_K\}$. For an arbitrary $\boldsymbol{\gamma}_i$ in this set, consider the set B_i, which contains all $\mathbf{b}$ for which $\mathbf{v}_n^*(\mathbf{b}) = \boldsymbol{\gamma}_i^{\mathsf{T}}\mathbf{b}$. Thus

$$B_i := \{\mathbf{b} \in \tilde{S} \mid \boldsymbol{\gamma}_i^{\mathsf{T}}\mathbf{b} \geq \boldsymbol{\gamma}^{\mathsf{T}}\mathbf{b},\ \forall \boldsymbol{\gamma} \in \Gamma_n\} = \bigcap_{j\neq i} \{\mathbf{b} \in \tilde{S} \mid (\boldsymbol{\gamma}_i - \boldsymbol{\gamma}_j)^{\mathsf{T}}\mathbf{b} \geq \mathbf{0}\}. \tag{8.54}$$

Consequently, B_i is a polyhedron since it is the intersection of a finite number of half-spaces. Since $\boldsymbol{\gamma}_i$ corresponds to a particular action, that action is optimal for all belief states in B_i.

□

The following interpretation of the quantities in (8.47) and (8.48) provides motivation for algorithms below. The set Γ_n contains at most $|\tilde{A}| \times |\Gamma_{n+1}|$ elements, one for each action and element in Γ_{n+1}. The quantity $\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})$ is the belief state at decision epoch $n+1$ if the belief state at decision epoch n is $\mathbf{b}$ and action $\mathbf{a}$ is chosen at decision epoch n after observing o. The quantity $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$ is the support of $v_{n+1}^*(\cdot)$ at $\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})$. In light of these observations, the following definition is provided for convenience.

Definition 8.8. Define $\boldsymbol{\gamma}_{\mathbf{a}}(\mathbf{b})$ by

$$\boldsymbol{\gamma}_{\mathbf{a}}(\mathbf{b}) := \mathbf{r}_{\mathbf{a}} + \sum_{o \in O} \mathbf{P}_{\mathbf{a}} \mathbf{U}_{\mathbf{a}}^{o} \boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})} \tag{8.55}$$

for all $\mathbf{a} \in \tilde{A}$.

Note that $\boldsymbol{\gamma}_{\mathbf{a}}(\mathbf{b})$ can take on at most $|\tilde{A}| \times |\Gamma_{n+1}|$ values because $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$ is chosen from the finite set Γ_{n+1}. Hence $v_n^*(\mathbf{b})$ can be evaluated inductively using any of the following expressions.

Equivalent representations for $v_n^*(\mathbf{b})$

$$\begin{aligned} v_n^*(\mathbf{b}) &= \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v_{n+1}^*(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\ &= \max_{\mathbf{a} \in \tilde{A}} \left\{ \left(\mathbf{r}_{\mathbf{a}} + \sum_{o \in O} \mathbf{P}_{\mathbf{a}} \mathbf{U}_{\mathbf{a}}^{o} \boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})} \right)^{\mathsf{T}} \mathbf{b} \right\} \qquad (8.56) \\ &= \max_{\mathbf{a} \in \tilde{A}} \{ \boldsymbol{\gamma}_{\mathbf{a}}(\mathbf{b})^{\mathsf{T}} \mathbf{b} \} \qquad (8.57) \\ &= \max_{\boldsymbol{\gamma} \in \Gamma_n} \{ \boldsymbol{\gamma}^{\mathsf{T}} \mathbf{b} \}. \end{aligned}$$

The challenge in applying these results on all of $\tilde{S}$ is to efficiently determine Γ_n for $n = 1, \ldots, N-1$. Referring to these equivalent representations will enhance your understanding of the algorithms below.

Some further remarks about Theorem 8.5 follow:

1. Although the state space is infinite dimensional, the optimal value function has a finite dimensional characterization, where it is defined as the maximum of a finite set of linear functions on the unit simplex in $\mathbb{R}^{|S|}$. Note that this result holds when there is a finite observation set and a finite action set.

2. While each $\boldsymbol{\gamma} \in \Gamma_n$ is associated with a single action, the same action may be associated with different values of $\boldsymbol{\gamma}$ in different regions of $\tilde{S}$, as shown in a later example.

3. This theorem is in the spirit of results in Section 4.4 on the optimality of structured policies. Here, the special structure of $v_n^*(\mathbf{b})$ being piecewise linear convex is used to show that an optimal policy is constant over the polyhedrons in (8.54) that define $v_n^*(\mathbf{b})$.

4. In an expressions of the form $f(\mathbf{b}) = \boldsymbol{\gamma}^\mathsf{T}\mathbf{b}$, the ith component of $\boldsymbol{\gamma}$ is the marginal rate of change of $f(\mathbf{b})$ with respect to b_i. This description does not account for the condition that the components of $\mathbf{b}$ sum to 1.

Figure 8.3 shows the consequences of this theorem in a one-period model ($N = 2$) with two states. The optimal value function is piecewise linear and the one-dimensional belief space (the probability of being in the first state) is separated into two intervals, one in which the first action is optimal and one in which the second action is optimal. Note that where these intervals meet, at $q = 11/17$, both actions are optimal. A multi-period example below provides additional insights.

8.4.4 An exact algorithm for finite horizon POMDPs

As noted above, the non-finiteness of $\tilde{S}$ makes direct implementation of Algorithm 8.1 impossible. The PLC representation for $v_n(\mathbf{b})$ in Theorem 8.5 introduces a sense of finiteness that underlies several implementable algorithms. We believe the algorithm proposed by Monahan [1982] is the most transparent and provides good insight into the issues that arise when solving a POMDP exactly.

Motivation

Theorem 8.5 established that

$$v_n^*(\mathbf{b}) = \max_{\boldsymbol{\gamma}\in\Gamma_n}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}$$

for a particular finite set Γ_n where each element of Γ_n corresponds to an action in $\tilde{A}$. Thus if Γ_n is known, for any belief state $\mathbf{b}$, the optimal action at decision epoch n can be recovered by choosing

$$\gamma_\mathbf{b} \in \arg\max_{\boldsymbol{\gamma}\in\Gamma_n}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}$$

and finding an action $\mathbf{a}$ that corresponds to $\gamma_\mathbf{b}$ or equivalently $\boldsymbol{\gamma}_\mathbf{a}(\mathbf{b}) = \gamma_\mathbf{b}$. Moreover, as a result of part 3 of Theorem 8.5 the belief space can be partitioned into polyhedra in which a single action is optimal on the interior of each polyhedra so that an optimal decision rule can be specified explicitly.

The following algorithm generates Γ_n as follows. Assuming that Γ_{n+1} is available, Γ_n for each n can be constructed using (8.47). However, the computations quickly become complex given that $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$, which is required to compute Γ_n, depends on $o, \mathbf{a}$, and $\mathbf{b}$. In other words, computing Γ_n exactly would require determining which elements of Γ_{n+1} attain the "argmax" in equation (8.48) for *some* value of $o, \mathbf{a}$, and $\mathbf{b}$.

Rather than generating Γ_n exactly using $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$, the algorithm below generates a larger set $G_n \supseteq \Gamma_n$, and then *prunes* the unnecessary elements. Generating Γ_n exactly may require further pruning anyway, since some of the elements may not be supports of $v_n^*(\mathbf{b})$ for any $\mathbf{b}$. Thus, if G_n can be generated and pruned efficiently, it is possible to avoid computing $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})}$ for each value of $o, \mathbf{a}$, and $\mathbf{b}$.

Define G_n inductively by

$$G_n := \left\{ \mathbf{r_a} + \sum_{o \in O} \mathbf{P_a U_a^o} \bar{\boldsymbol{\gamma}}_o \;\middle|\; \mathbf{a} \in \tilde{A}, \bar{\boldsymbol{\gamma}}_o \in \Gamma_{n+1} \right\}, \tag{8.58}$$

where the recursion is initialized by $\Gamma_N = \{\mathbf{r}_N\}$. The subtle difference is to simply use *all* elements of Γ_{n+1}, to define G_n, even if they would not have attained the maximum in (8.48) for a particular $o, \mathbf{a}$, and $\mathbf{b}$.

Pruning G_n

Pruning is a key step in a POMDP algorithm. That is, elements of G_n that do not achieve the maximum in $\max_{\bar{\boldsymbol{\gamma}} \in G_n} \bar{\boldsymbol{\gamma}}^\mathsf{T} \mathbf{b}$ for any $\mathbf{b}$ need to be pruned (removed from consideration). One way to do this is to solve the following linear program for *each* $\bar{\boldsymbol{\gamma}} \in G_n$:

$$\begin{aligned} \underset{z, \mathbf{b}}{\text{minimize}} \quad & z - \bar{\boldsymbol{\gamma}}^\mathsf{T} \mathbf{b} \\ \text{subject to} \quad & z \geq \boldsymbol{\gamma}^\mathsf{T} \mathbf{b}, \quad \forall\, \boldsymbol{\gamma} \in G_n \\ & \mathbf{b} \in \tilde{S}, z \in \mathbb{R}^1. \end{aligned} \tag{8.59}$$

Note that in the objective function $\bar{\gamma}$ is *fixed* and $\mathbf{b}$ and z are variable. As a result of the first set of constraints, the objective function is bounded below by zero. The second set of constraints ensure that $\mathbf{b}$ is a belief state, that is, it lies in the unit simplex, and that z is a scalar.

The following lemma shows that solutions of this linear program identifies dominated elements of G_n.

Lemma 8.1. Suppose for some $\bar{\boldsymbol{\gamma}} \in G_n$ that the optimal objective function value of the above linear program is positive. Then $\bar{\boldsymbol{\gamma}} \notin \Gamma_n$.

Proof. As a result of constraint (8.59), the objective function value is non-negative. Furthermore if

$$v_n(\bar{\mathbf{b}}) = \max_{\boldsymbol{\gamma} \in G_n} \boldsymbol{\gamma}^\mathsf{T} \bar{\mathbf{b}} = \bar{\boldsymbol{\gamma}}^\mathsf{T} \bar{\mathbf{b}}$$

for some $\mathbf{b} \in \tilde{S}$ then the objective function of the above linear program will be zero. Hence if it is positive, $\bar{\boldsymbol{\gamma}} \notin \Gamma_n$. □

An algorithm to solve a finite horizon POMDP

The following algorithm due to Monahan [1982] implements the process described above. For each n it starts with a set of potential supports for $v_n(\mathbf{b})$ denote by G_n and

eliminates $\boldsymbol{\gamma}$-vectors one at a time using the result in Lemma 8.1. It returns the set Γ_n of $\boldsymbol{\gamma}$-vectors that are supports for $v_n(\mathbf{b})$ for at least one $\mathbf{b} \in \tilde{S}$ for $n = 1, \ldots, N$.

Algorithm 8.2. A support-based pruning algorithm

1. **Initialize:** Set $n \leftarrow N-1$ and $\Gamma_N \leftarrow \{\mathbf{r}_N\}$.
2. **Iterate:** While $n \geq 1$:
   (a) **Generate candidate vectors:**

$$G_n \leftarrow \left\{ \mathbf{r}_\mathbf{a} + \sum_{o \in O} \mathbf{P}_\mathbf{a} \mathbf{U}_\mathbf{a}^o \bar{\boldsymbol{\gamma}}_o \;\middle|\; \mathbf{a} \in \tilde{A}, \bar{\boldsymbol{\gamma}}_o \in \Gamma_{n+1} \right\}. \tag{8.60}$$

   (b) **Prune:** For each $\bar{\boldsymbol{\gamma}} \in \Gamma_n$, solve

$$\begin{aligned} \underset{z,\mathbf{b}}{\text{minimize}} \quad & z - \bar{\boldsymbol{\gamma}}^\mathsf{T} \mathbf{b} \\ \text{subject to} \quad & z \geq \boldsymbol{\gamma}^\mathsf{T} \mathbf{b}, \quad \forall \boldsymbol{\gamma} \in G_n \\ & \mathbf{b} \in \tilde{S}, z \in \mathbb{R} \end{aligned} \tag{8.61}$$

   and let $(z^*, \mathbf{b}^*)$ be an optimal solution. If $z^* - \bar{\boldsymbol{\gamma}}^\mathsf{T} \mathbf{b}^* > 0$,

$$G_n \leftarrow G_n \setminus \{\bar{\boldsymbol{\gamma}}\}.$$

   (c) Set $\Gamma_n \leftarrow G_n$ and $n \leftarrow n-1$.
3. **Terminate:** Return Γ_n for $n = 1, \ldots, N$.

Using the Γ_n found by the above algorithm, the optimal value in state $\mathbf{b}$ is given by

$$v_n^*(\mathbf{b}) = \max_{\boldsymbol{\gamma} \in \Gamma_n} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}\}. \tag{8.62}$$

To determine an optimal action in $\mathbf{b}$, first choose $\boldsymbol{\gamma}_\mathbf{b}$ to satisfy

$$\boldsymbol{\gamma}_\mathbf{b} \in \arg\max_{\boldsymbol{\gamma} \in \Gamma_n} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}\}, \tag{8.63}$$

and then determine action $\mathbf{a}^*$ for which:

$$\boldsymbol{\gamma}_\mathbf{b} = \mathbf{r}_{\mathbf{a}^*} + \sum_{o \in O} \mathbf{P}_{\mathbf{a}^*} \mathbf{U}_\mathbf{a}^o \boldsymbol{\gamma}_{\boldsymbol{\tau}(o, \mathbf{a}^*, \mathbf{b})}. \tag{8.64}$$

Some comments follow:

1. Prior to implementing the pruning step, dominated constraints can be removed to reduce the size of G_n. This is a standard procedure in modern LP solvers and illustrated in the example below.

2. Note that in implementing step 2(b), the number of constraints in (8.61) is decreasing as vectors are eliminated from G_n.

3. If one is interested in the structure of the optimal policy, one can use the result in part 3 of Theorem 8.5 to partition $\tilde{S}$ into polyhedra in which a single action is optimal. Of course more than one action may be optimal on the boundaries.

4. The calculations described in (8.62)-(8.64) can be avoided by *maintaining a list of which actions correspond to which elements of* Γ_n.

Other approaches

POMDP algorithms can be classified[16] as support-based, belief-based or approximate.

As noted above, the above algorithm is *support-based*; at each iteration it starts with a set of possible supports and prunes it to obtain a reduced set of supports that define the optimal value function at that iteration. From these supports the polyhedra defining an optimal policy can be determined.

An alternative class of algorithms is *belief-based*. They start with a belief state $\mathbf{b}'$ and seek to define the region in which the support of $v_n(\mathbf{b})$ at $\mathbf{b}'$, denote by $\boldsymbol{\gamma}_{\mathbf{b}'}$ is optimal. The intent is to perform this calculation at sufficiently many such $\mathbf{b}'$ so as to characterize $v_n(\mathbf{b})$ for all $\mathbf{b} \in \tilde{S}$. These points are sometimes referred to as *witness points* because they "testify" to the existence of a polyhedron in which $\boldsymbol{\gamma}_{\mathbf{b}'}$ is a support of $v_n(\mathbf{b})$.

Approximate methods focus on extending exact calculations on finite subsets of $\tilde{S}$ to approximate $v_n(\mathbf{b})$ for all belief states. These subsets can be fixed as described in the next section or generated throughout the computational process.

An application of Algorithm 8.2

The calculations in this section demonstrate the application of Algorithm 8.2 to the two-state example from Section 8.3.2 with $N = 3$. In this example, express $\mathbf{b}$ by (q_1, q_2) where $q_1 \geq 0$, $q_2 \geq 0$ and $q_1 + q_2 = 1$ and let $\mathbf{a}_1 = (a_{1,1}, a_{2,1})$ and $\mathbf{a}_2 = (a_{1,2}, a_{2,2})$.

To make the example more insightful, set $r(s_2, a_{2,1}, s_2) = 4$ instead of -5. This means that the reward matrix in (8.31) changes to

$$\mathbf{R}_{\mathbf{a}_1} = \begin{bmatrix} 5 & -5 \\ 0 & 4 \end{bmatrix}.$$

With this change, $r(s_2, a_{2,1}) = 4$ so that $\mathbf{r}_{\mathbf{a}_1} = (3, 4)$. Moreover set $r_3(s) = 0$ for all $s \in S$ so that $\tilde{\mathbf{r}}_3(\mathbf{b}) = \mathbf{0}$ for all $\mathbf{b} \in \tilde{S}$.

[16] Spaan and Vlassis [2005] provides a nice overview of POMDP algorithms.

The iterates of the algorithm follow:

1. For $n = 3$, $\Gamma_3 = \{\mathbf{r}_3\} = \{(0, 0)\}$ so that $v_3^*(\mathbf{b}) = \max_{\boldsymbol{\gamma}\in\Gamma_3}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\} = 0$.

2. For $n = 2$, since $\Gamma_3 = \{\mathbf{0}\}$,

$$G_2 = \{\mathbf{r}_{(a_{1,1},a_{2,1})}, \mathbf{r}_{(a_{1,2},a_{2,2})}\} = \{(3, 4), (5, 2)\}.$$

It is clear from Figure 8.6 that both vectors in G_2 are required to define $v_2(\mathbf{b})$ so that $\Gamma_2 = G_2$.

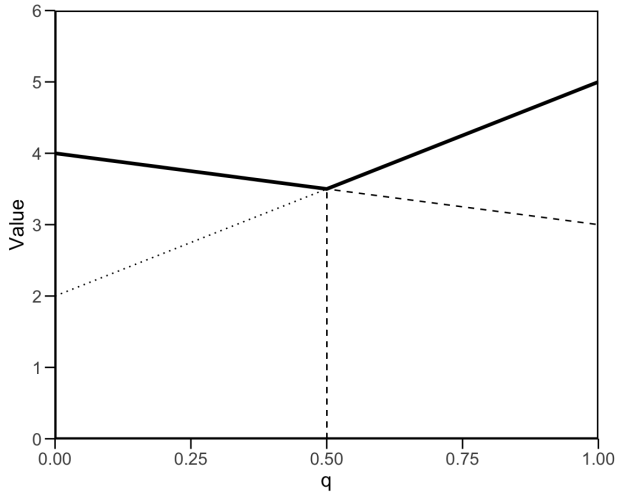


Figure 8.6: Graphical representation of $v_2^*(\mathbf{b})$. Note that $q_1 = q$ and $q_2 = 1 - q$.

Hence

$$v_2^*(\mathbf{b}) = \max_{\boldsymbol{\gamma}\in\Gamma_2}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\} = \max\{3q_1 + 4q_2, 5q_1 + 2q_2\},$$

where the first term in the max corresponds to action $(a_{1,1}, a_{2,1})$ and the second term corresponds to action $(a_{1,2}, a_{2,2})$. Using the equivalent representation $q_1 = q$ and $q_2 = 1 - q$, the optimal value function can be written as

$$v_2^*(\mathbf{b}) = \max\{4 - q, 2 + 3q\},$$

and represented in Figure 8.6. Thus,

$$d_2^*(\mathbf{b}) = \arg\max_{\boldsymbol{\gamma}\in\Gamma_2}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\} = \begin{cases}(a_{1,1}, a_{2,1}), & 0 \le q \le 0.5\\ (a_{2,1}, a_{2,2}), & 0.5 < q \le 1.\end{cases} \tag{8.65}$$

That is, for belief states where the probability of being in state s_1 is less than 0.5, action $(a_{1,1}, a_{2,1})$ should be chosen. Otherwise, choose action $(a_{2,1}, a_{2,2})$. Hence $\tilde{S}$ is partitioned into two polyhedra, which in this case are intervals.

3. The case $n = 1$ provides more insight into the workings of the algorithm. In it

$$G_1 = \Big\{ \mathbf{r}_{(a_{1,1},a_{2,1})} + \mathbf{P}_{(a_{1,1},a_{2,1})}\mathbf{U}^{o1}_{(a_{1,1},a_{2,1})}\boldsymbol{\gamma}_1 + \mathbf{P}_{(a_{1,1},a_{2,1})}\mathbf{U}^{o2}_{(a_{1,1},a_{2,1})}\boldsymbol{\gamma}_2,$$
$$\mathbf{r}_{(a_{1,2},a_{2,2})} + \mathbf{P}_{(a_{1,2},a_{2,2})}\mathbf{U}^{o1}_{(a_{1,2},a_{2,2})}\boldsymbol{\gamma}_1 + \mathbf{P}_{(a_{1,2},a_{2,2})}\mathbf{U}^{o2}_{(a_{1,2},a_{2,2})}\boldsymbol{\gamma}_2 \;\Big|\; (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) \in \Gamma_2 \times \Gamma_2 \Big\}.$$

A comment about the above expression may be helpful. Note that since Γ_2 has two elements, $(3, 4)$ and $(5, 2)$, the pair (γ_1, γ_2) can take on **four** values, that is

$$\Gamma_2 \times \Gamma_2 = \big\{\big((3,4),(3,4)\big), \big((3,4),(5,2)\big), \big((5,2),(3,4)\big), \big((5,2),(5,2)\big)\big\}.$$

Since there are two actions, G_1 has eight elements. For example, when $(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2) = \big((3,4),(3,4)\big)$, then the element of G_1 corresponding to action $\mathbf{a}_1 = (a_{1,1}, a_{2,1})$ is

$$\mathbf{r}_{(a_{1,1},a_{2,1})} + \mathbf{P}_{(a_{1,1},a_{2,1})}\mathbf{U}^{o1}_{(a_{1,1},a_{2,1})}\gamma_1 + \mathbf{P}_{(a_{1,1},a_{2,1})}\mathbf{U}^{o2}_{(a_{1,1},a_{2,1})}\boldsymbol{\gamma}_2 =$$

$$\begin{bmatrix}3\\4\end{bmatrix} + \begin{bmatrix}0.8 & 0.2\\ 0 & 1\end{bmatrix}\begin{bmatrix}0.8 & 0\\ 0 & 0.4\end{bmatrix}\begin{bmatrix}3\\4\end{bmatrix} + \begin{bmatrix}0.8 & 0.2\\ 0 & 1\end{bmatrix}\begin{bmatrix}0.2 & 0\\ 0 & 0.6\end{bmatrix}\begin{bmatrix}3\\4\end{bmatrix} = \begin{bmatrix}6.2\\8\end{bmatrix}.$$

Similar calculations show that

$$G_1 = \left\{\begin{bmatrix}6.2\\8\end{bmatrix}, \begin{bmatrix}6.28\\6.8\end{bmatrix}, \begin{bmatrix}7.32\\7.2\end{bmatrix}, \begin{bmatrix}7.4\\6\end{bmatrix}, \begin{bmatrix}9\\5.6\end{bmatrix}, \begin{bmatrix}7.8\\5.04\end{bmatrix}, \begin{bmatrix}8.2\\5.76\end{bmatrix}, \begin{bmatrix}7\\5.2\end{bmatrix}\right\}, \tag{8.66}$$

where the first four vectors correspond to action $\mathbf{a}_1 = (a_{1,1}, a_{2,1})$ and the last four correspond to action $\mathbf{a}_2 = (a_{1,2}, a_{2,2})$. Observe that there are three dominated elements (the second, sixth, and eighth vectors) so that G_1 can be reduced to

$$G_1 = \left\{\begin{bmatrix}6.2\\8\end{bmatrix}, \begin{bmatrix}7.32\\7.2\end{bmatrix}, \begin{bmatrix}7.4\\6\end{bmatrix}, \begin{bmatrix}9\\5.6\end{bmatrix}, \begin{bmatrix}8.2\\5.76\end{bmatrix}\right\}.$$

To obtain Γ_1 requires solving five linear programs. Setting $\bar{\gamma} = (6.2, 8)$, the first linear program becomes

$$\begin{aligned} \underset{z,q_1\geq 0,q_2\geq 0}{\text{minimize}} \quad & z - (6.2q_1 + 8q_2) \\ \text{subject to} \quad & z \geq 6.2q_1 + 8q_2, \\ & z \geq 7.32q_1 + 7.2q_2, \\ & z \geq 7.4q_1 + 6q_2, \\ & z \geq 9q_1 + 5.6q_2, \\ & z \geq 8.2q_1 + 5.76q_2, \\ & 1 = q_1 + q_2. \end{aligned}$$

Since the objective function is positive, $(6.2, 8)$ can be eliminated from G_1 so the next linear program uses $\bar{\gamma} = (7.32, 7.20)$ in the objective function and has one fewer constraint. Continuing in this way establishes that

$$\Gamma_1 = \{(6.2, 8), (7.32, 7.2), (9, 5.6)\},$$

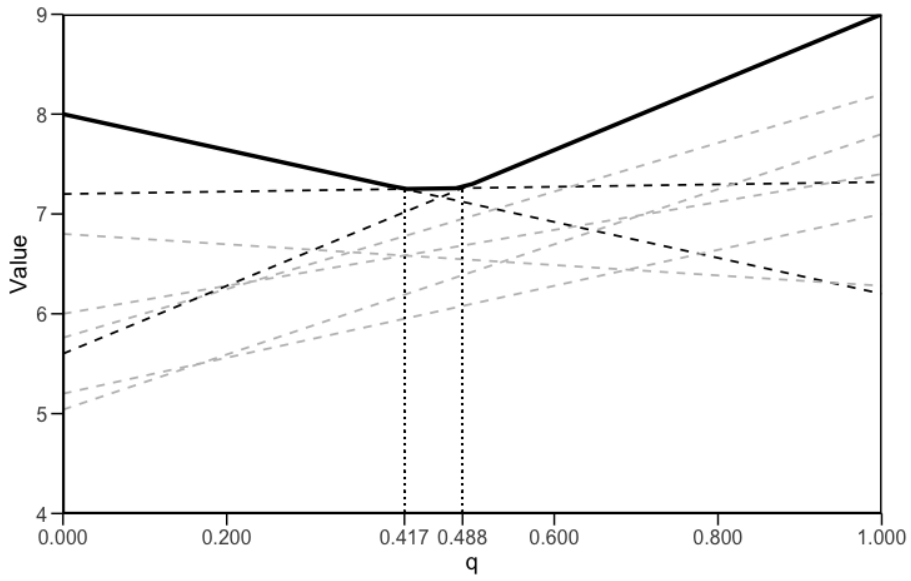


Figure 8.7: Graphical representation of $v_1^*(\mathbf{b})$. It shows $\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}$ for all $\boldsymbol{\gamma} \in G_1$ and that $q_1 = q$ and $q_2 = 1 - q$. Note that lines corresponding to $\boldsymbol{\gamma} \notin \Gamma_1$ are represented by grey dashed lines and those corresponding to $\boldsymbol{\gamma} \in \Gamma_1$ by black dashed lines. The bold line represents the piecewise linear convex function $v_1^*(\mathbf{b})$.

where the first two vectors correspond to $\mathbf{a}_1$ and the third vector corresponds to $\mathbf{a}_2$.

Thus Figure 8.7 shows that

$$\begin{aligned} v_1^*(\mathbf{b}) &= \max_{\boldsymbol{\gamma}\in\Gamma_1}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\} \\ &= \max\{6.2q_1 + 8q_2, 7.32q_1 + 7.2q_2, 9q_1 + 5.6q_2\} \\ &= \max\{8 - 1.8q, 7.2 + 0.12q, 5.6 + 3.4q\} \\ &= \begin{cases} 8 - 1.8q & \text{for } 0 \le q \le 0.417 \\ 7.2 + 0.12q & \text{for } 0.417 \le q \le 0.488 \\ 5.6 + 3.4q & \text{for } 0.488 \le q \le 1. \end{cases} \end{aligned}$$

Hence $v_1^*(\mathbf{b})$ is PLC and it can be shown that for

$$d_1^*(\mathbf{b}) = \begin{cases} \mathbf{a}_1, & 0 \le q \le 0.488 \\ \mathbf{a}_2, & 0.488 < q \le 1. \end{cases} \tag{8.67}$$

Figure 8.7 shows that $v_1^*(\mathbf{b})$ is piecewise linear convex with three distinct supports $(6.2, 8)$, $(7.32, 7.2)$, and $(9, 5.6)$. The first breakpoint occurs at $q = 5/12 = 0.417$ and the second breakpoint occurs at $q = 20/41 = 0.488$. On both sides of the first breakpoint, the optimal action is $(a_{1,1}, a_{2,1})$, so the fact that a breakpoint exists there

is due to the fact that $\boldsymbol{\gamma}_{\boldsymbol{\tau}(o_1,(a_{1,1},a_{2,1}),(q_1,q_2))}$ changes from $(3,4)$ to $(5,2)$. In other words, it is due to the breakpoint in $v_2^*(\mathbf{b})$. The second breakpoint is due to a change in the optimal action, where for $q < 0.488$, $\mathbf{a}_1 = (a_{1,1}, a_{2,1})$ is optimal, and for $q > 0.488$, $\mathbf{a}_2 = (a_{1,2}, a_{2,2})$ is optimal.

8.4.5 A finite-grid approximation for finite horizon POMDPs

This section outlines a straightforward and implementable approach for approximating solutions to a POMDP. The core principle involves evaluating the optimal value function at a *finite grid* of points within the belief space $\tilde{S}$ rather than across the entire continuous space. This yields a set of $\boldsymbol{\gamma}$-vectors, which can be used to:

1. Generate upper and lower bounds on the unknown true optimal value function.
2. Derive a policy that is guaranteed to be within a known distance of the optimal policy (based on the bounds).

This provides a practical method for tackling the computational challenges associated with solving POMDPs and has provided the basis for several approximate algorithms for solving[17] larger models.

A lower bound

Generating a lower bound is straightforward. Since the optimal value function at decision epoch n can be written as

$$v_n^*(\mathbf{b}) = \max_{\boldsymbol{\gamma}\in\Gamma_n}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}, \tag{8.68}$$

a lower bound on it can be obtained by instead choosing the maximum over a **subset** of Γ_n. To see why this is true, let H be a finite subset set of $\tilde{S}$ and let Γ_n^H be the set of $\boldsymbol{\gamma} \in \Gamma_n$ that define the optimal value function on H. That is,

$$\Gamma_n^H := \left\{ \boldsymbol{\gamma}_\mathbf{b} \;\middle|\; \boldsymbol{\gamma}_\mathbf{b} \in \arg\max_{\boldsymbol{\gamma}\in\Gamma_n}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}, \mathbf{b} \in H \right\}. \tag{8.69}$$

Then

$$\underline{v}_n(\mathbf{b}) := \max_{\boldsymbol{\gamma}\in\Gamma_n^H}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\} \tag{8.70}$$

is a lower bound on $v_n^*(\mathbf{b})$ for all $\mathbf{b}$. In particular

$$\underline{v}_n(\mathbf{b}) = \begin{cases} v_n^*(\mathbf{b}) & \text{for } \mathbf{b} \in H \\ \leq v_n^*(\mathbf{b}) & \text{for } \mathbf{b} \in \tilde{S} \setminus H. \end{cases} \tag{8.71}$$

Note that if Γ_n is not available, then (8.69) can be applied with any subset of Γ_n. In this case there may be an inequality for $\mathbf{b} \in H$ in (8.71) if a $\boldsymbol{\gamma}$-vector that supports $v_n^*(\mathbf{b})$ has been omitted. In this case, $\underline{v}_n(\mathbf{b}) \leq v_n^*(\mathbf{b})$.

[17] See the Bibliographic Remarks section for references.

An upper bound

To generate an upper bound, find $v_n^*(\mathbf{b})$ for $\mathbf{b} \in H$ and interpolate the optimal value function between points in H.

Let $H = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$. Suppose $\mathbf{b} \in \tilde{S} \setminus H$ can be written as a convex combination of elements in H[18]. Then, $v_n^*(\mathbf{b})$ for $\mathbf{b} \in \tilde{S} \setminus H$ can be bounded above by

$$\overline{v}_n(\mathbf{b}) := \sum_{i=1}^{m} \alpha_i v_n^*(\mathbf{b}_i), \tag{8.72}$$

where $\mathbf{b} = \sum_{i=1}^m \alpha_i \mathbf{b}_i$, $\alpha_i \geq 0, i = 1, \ldots, m$, and $\sum_{i=1}^m \alpha_i = 1$. The convexity of $v_n^*(\mathbf{b})$ and Jensen's inequality implies that

$$\sum_{i=1}^{m} \alpha_i v_n^*(\mathbf{b}_i) \geq v_n^* \left(\sum_{i=1}^{m} \alpha_i \mathbf{b}_i \right) = v_n^*(\mathbf{b}),$$

so that

$$\overline{v}_n(\mathbf{b}) = \begin{cases} v_n^*(\mathbf{b}), & \mathbf{b} \in H \\ \sum_{i=1}^m \alpha_i v_n^*(\mathbf{b}_i), & \mathbf{b} \in \tilde{S} \setminus H \end{cases} \tag{8.73}$$

is an upper bound on $v_n^*(\mathbf{b})$ for all $\mathbf{b}$.

Obtaining a decision rule

One approach to obtain a policy for an arbitrary $\mathbf{b} \in \tilde{S}$ is to use the lower bound as follows.

$$d_n(\mathbf{b}) \in \arg\max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \underline{v}_{n+1}\big(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})\big) \right\}. \tag{8.74}$$

The reason for using the lower bound is that the difference between the value of this policy and the optimal value can be bounded using (8.77) below.

To evaluate the above requires computing $\eta(o|\mathbf{a}, \mathbf{b})$ and $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ using (8.24) and then evaluating

$$\underline{v}_{n+1}\big(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})\big) = \max_{\boldsymbol{\gamma} \in \Gamma_{n+1}^H} \{\boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})\}. \tag{8.75}$$

To do this requires only storing Γ_{n+1}^H. Note that since $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ most likely will not be an element of H, $\underline{v}_{n+1}\big(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})\big)$ must be evaluated as in (8.75). With this representation, an action that achieves the arg max in (8.74) can be tracked via the corresponding $\boldsymbol{\gamma}$-vector.

[18] To ensure all points in $\tilde{S}$ can be represented in this way requires that H contain the extreme points of the unit simplex.

The policy $\pi = (d_1, \ldots, d_{N-1})$ where each $d_n(\mathbf{b})$ is defined by (8.74) is referred to as the *lower bound approximate policy.*

Properties of the bounds

Using the upper and lower bounding functions, the value of the policy defined by (8.74) and the optimal policy can be bounded as follows.

Theorem 8.6. For any $\mathbf{b} \in \tilde{S}$, suppose $\underline{v}_n(\mathbf{b})$ and $\overline{v}_n(\mathbf{b})$ are computed using (8.70) and (8.72) for n=$1, \ldots, N$ and let $\pi = (d_1, d_2, \ldots, d_{N-1})$ be a policy that uses decision rules d_n obtained from (8.74). Then

$$\underline{v}_n(\mathbf{b}) \leq v_n^{\pi}(\mathbf{b}) \leq v_n^*(\mathbf{b}) \leq \overline{v}_n(\mathbf{b}). \tag{8.76}$$

for any $n = 1, \ldots, N$.

Proof. The proof is by induction. For $n = N$, the inequalities hold by definition. Assume

$$\underline{v}_j(\mathbf{b}) \leq v_j^{\pi}(\mathbf{b}) \leq v_j^*(\mathbf{b}) \leq \overline{v}_j(\mathbf{b})$$

for $j = n+1, \ldots, N$. Then for all $\mathbf{b} \in \tilde{S}$

$$\begin{aligned}
\underline{v}_n(\mathbf{b}) &= \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \underline{v}_{n+1}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\
&\leq \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v_{n+1}^{\pi}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\
&= v_n^{\pi}(\mathbf{b}) \leq v_n^*(\mathbf{b}) \\
&= \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v_{n+1}^*(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\
&\leq \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \overline{v}_{n+1}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\
&= \overline{v}_n(\mathbf{b}),
\end{aligned}$$

where the first inequality and last inequalities are a result of the induction hypothesis. □

As a consequence of this theorem, one obtains the following bound.

Corollary 8.1. Under the hypothesis of Theorem 8.6,

$$0 \leq v_n^*(\mathbf{b}) - v_n^\pi(\mathbf{b}) \leq \overline{v}_n(\mathbf{b}) - \underline{v}_n(\mathbf{b}) \tag{8.77}$$

for $n = 1, \ldots, N$ and all $\mathbf{b} \in \tilde{S}$.

Another upper bound

The following provides a different and potentially useful upper bound on the optimal value function. For all $\mathbf{b} \in \tilde{S}$ and $n = 1, \ldots, N-1$, inductively define

$$\overline{v}'_n(\mathbf{b}) := \sum_{i=1}^m \alpha_i \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}_i, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}_i, \mathbf{a}) \overline{v}'_{n+1}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}_i)) \right\}, \tag{8.78}$$

where $\mathbf{b} = \sum_{i=1}^m \alpha_i \mathbf{b}_i$, $\alpha_i \geq 0, i = 1, \ldots, m$, and $\sum_{i=1}^m \alpha_i = 1$, with $\overline{v}'_N(\mathbf{b}) = \tilde{r}_N(\mathbf{b})$.

Proposition 8.1. For all $\mathbf{b} \in \tilde{S}$ and $n = 1, \ldots, N$,

$$v_n^*(\mathbf{b}) \leq \overline{v}'_n(\mathbf{b}). \tag{8.79}$$

Proof. For $n = N$, the inequality holds by definition. For a given $n < N$, assume the result holds for $n+1, \ldots, N$. Then,

$$\begin{aligned}
\overline{v}'_n(\mathbf{b}) &= \sum_{i=1}^m \alpha_i \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}_i, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}_i, \mathbf{a}) \overline{v}'_{n+1}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}_i)) \right\} \\
&\geq \sum_{i=1}^m \alpha_i \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}_i, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}_i, \mathbf{a}) v^*_{n+1}(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}_i)) \right\} \\
&= \sum_{i=1}^m \alpha_i v_n^*(\mathbf{b}_i) \geq v_n^* \left(\sum_{i=1}^m \alpha_i \mathbf{b}_i \right), = v_n^*(\mathbf{b})
\end{aligned}$$

where the last inequality is due to Jensen's inequality. □

Since both (8.73) and (8.78) generate upper bounds on the optimal value function, the minimum of the two is also an upper bound.

Choosing a finite grid

To generate the lower bound, any set $H \subseteq \tilde{S}$ suffices. However, to generate the upper bound, all points in $\tilde{S}$ need to be expressable as convex combinations of elements of H. Thus, H must include the vertices of $\tilde{S}$. Furthermore, the set H should balance being

fine enough so the approximation is of sufficient quality while being coarse enough to remain computationally tractable.

One particular set H that satisfies these requirements is based on the *Freundenthal triangulation* of $\mathbb{R}^m$, which induces a grid over the unit simplex (i.e., the belief space) and allows the modeler to adjust its granularity. Let K be a positive integer and consider the set[19]

$$H = \left\{ \frac{\mathbf{k}}{K} \;\middle|\; \mathbf{k} \in \mathbb{Z}_+^m, \sum_{i=1}^{m} k_i = K \right\}. \tag{8.80}$$

This set H generates a grid (and a corresponding partition) over the unit simplex that becomes increasingly fine as K increases.

Another choice is to choose H randomly by simulating belief states. Ideally these points should in some way represent beliefs that will be observed in practice and as well cover a large portion of $\tilde{S}$. These belief states can be based on random sampling from the prior or on updates resulting from implementing a policy. Several approximate algorithms are based on different approaches to choosing and modifying the grid points.

Algorithms to compute bounds and approximate policies

Computations based on Lovejoy [1991a] and Lovejoy [1991b] are summarized in the following two algorithms. The first iteratively constructs the elements required to derive lower and upper bounds on the optimal value function and the lower bound approximate policy by restricting attention to belief states in the finite grid H. The second algorithm uses the output of the first algorithm to compute bounds and the lower bound approximate policy at an arbitrary belief state $\mathbf{b}'$.

Note that two algorithms are described when a single statement would suffice. This is so as to distinguish objectives and calculations. In practice one would combine them at each decision epoch.

Algorithm 8.3. Bounding the value of a finite horizon POMDP on a grid

1. **Initialize:**
   (a) Specify a finite set $H \leftarrow \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subseteq \tilde{S}$.
   (b) $\Gamma_N^H \leftarrow \{\mathbf{r}_N\}$ and $\Gamma_N \leftarrow \{\mathbf{r}_N\}$.
   (c) For all $\mathbf{b} \in H$, $\underline{v}_N(\mathbf{b}) \leftarrow \tilde{r}_N(\mathbf{b})$, $v_N^*(\mathbf{b}) \leftarrow \tilde{r}_N(\mathbf{b})$, and $\overline{v}_N(\mathbf{b}) \leftarrow \tilde{r}_N(\mathbf{b})$.
   (d) $n \leftarrow N - 1$.
2. **Iterate:** While $n \geq 1$:

[19] Recall that $\mathbb{Z}_+^m$ denotes the m-dimensional space of positive integers.

(a) **Generate lower bound on H:**

i.
$$\Gamma_n \leftarrow \left\{ \mathbf{r_a} + \sum_{o\in O} \mathbf{P_a U_a^o} \bar{\boldsymbol{\gamma}}_o \;\middle|\; \mathbf{a} \in \tilde{A}, \bar{\boldsymbol{\gamma}}_o \in \Gamma_{n+1}^H \right\}. \tag{8.81}$$

ii.
$$\Gamma_n^H \leftarrow \left\{ \boldsymbol{\gamma}_\mathbf{b} \;\middle|\; \boldsymbol{\gamma}_\mathbf{b} \in \arg\max_{\boldsymbol{\gamma}\in\Gamma_n} \{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}, \mathbf{b} \in H \right\}. \tag{8.82}$$

iii. For $\mathbf{b} \in H$[a]
$$\underline{v}_n(\mathbf{b}) \leftarrow \max_{\boldsymbol{\gamma}\in\Gamma_n^H} \{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}. \tag{8.83}$$

(b) **Generate upper bound on H:** For all $\mathbf{b} \in H$,
$$\overline{v}_n(\mathbf{b}) \leftarrow \max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) \overline{v}_{n+1}(\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})) \right\}. \tag{8.84}$$

(c) $n \leftarrow n-1$.

3. **Terminate:** Return Γ_n^H and $\overline{v}_n(\mathbf{b})$ for $\mathbf{b} \in H$ and $n = 1, \ldots, N$.

[a] This calculation applies to $\mathbf{b} \in \tilde{S} \setminus H$ as well. It is not necessary to include this in the algorithm; computing Γ_n^H suffices.

The above algorithm summarizes calculations on the grid. The following algorithm applies them to obtain bounds and approximate policies for any belief state.

Algorithm 8.4. Extending bounds to arbitrary belief states.

1. **Initialize:**
   (a) Specify $\mathbf{b}' \in \tilde{S} \setminus H$.
   (b) Apply Algorithm 8.3.
   (c) Set $n \leftarrow N-1$.
2. **Iterate:** While $n \geq 1$:
   (a) **Generate lower bound at $\mathbf{b}'$:**
$$\underline{v}_n(\mathbf{b}') \leftarrow \max_{\boldsymbol{\gamma}\in\Gamma_n^H} \{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}'\}. \tag{8.85}$$

(b) **Generate upper bound at $\mathbf{b}'$:**

$$\overline{v}_n(\mathbf{b}') \leftarrow \sum_{i=1}^{m} \alpha_i \overline{v}_n(\mathbf{b}_i), \tag{8.86}$$

where $\mathbf{b}' = \sum_{i=1}^{m} \alpha_i \mathbf{b}_i, \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geq 0$ for $i = 1, \ldots, m$ and $\mathbf{b}_i \in H$ are the closest points to $\mathbf{b}'$ that allow $\mathbf{b}'$ to be written as their convex combination.

(c) **Generate lower bound approximate policy at $\mathbf{b}'$**: Set

$$d_n(\mathbf{b}') \in \arg\max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}', \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}', \mathbf{a}) \underline{v}_{n+1}\big(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}')\big) \right\}, \tag{8.87}$$

where

$$\underline{v}_{n+1}\big(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}')\big) = \max_{\boldsymbol{\gamma} \in \Gamma_{n+1}^H} \{\boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b}')\}.$$

(d) $n \leftarrow n - 1$.

3. **Terminate:** Return $\underline{v}_n(\mathbf{b}')$, $\overline{v}_n(\mathbf{b}')$ and $d_n(\mathbf{b}')$ for $n = 1, \ldots, N$.

Some comments on these algorithms follow:

1. When applying these algorithms in practice, the calculations can be combined for each n when structural results or a policy are required. Alternatively, they can be applied online by first applying Algorithm 8.3 and then applying Algorithm 8.4 when $\mathbf{b}'$ is observed at decision epoch n.

2. Using $\Gamma_n^H = \{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_{|H|}\}$, the structure of the lower bound approximate policy can be determined by partitioning $\tilde{S}$ into $|H|$ polyhedra defined by (breaking ties arbitrarily)

$$\left\{ \mathbf{b} \in \tilde{S} \;\middle|\; \boldsymbol{\gamma}_i = \arg\max_{\boldsymbol{\gamma} \in \Gamma_n^H} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}\} \right\}.$$

3. For each n, $|\Gamma_n^H| \leq |H|$. This avoids the exponential growth of $|\Gamma_n|$ in Algorithm 8.2.

4. Note that (8.81) differs from (8.60) in Algorithm 8.2 by replacing Γ_n by the smaller set Γ_n^H. Consequently Γ_n^H may not contain $\boldsymbol{\gamma}$-vectors corresponding to the optimal value function.

5. The quality of the approximation depends on H. Note that H can vary with n and be defined adaptively. Moreover, different specifications of H can be used to obtain the upper and lower bound.

6. The upper bound for arbitrary $\mathbf{b} \in \tilde{S} \setminus H$ is the interpolation of the values $\overline{v}_n(\mathbf{b}_i)$, where $\mathbf{b}_i \in H$ are the nearest points to $\mathbf{b}$ that allow $\mathbf{b}$ to be written as a convex combination of those points. If H corresponds to the Freundenthal triangulation, then for any point $\mathbf{b} \in \tilde{S}$, it is straightforward to find the closest integer lattice points in $\mathbb{Z}_+^m$ around $K\mathbf{b}$.

An example

The following example applies Algorithms 8.3 and 8.4 to an $N = 4$ version of the two-state model analyzed above. However, to avoid using different notation to refer to the same constructs in the exact calculations above, $N = 3$ represents the final decision epoch and the decision epoch $n = 0$ is added prior to decision epoch 1 to represent the extra period.

Choose the finite grid $H = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\} = \{(0,1), (0.4, 0.6), (1,0)\}$.

1. For $n = 3$, $\Gamma_3^H = \Gamma_3 = \{\mathbf{r}_3\} = \{(0,0)\}$. Then, $\underline{v}_3(\mathbf{b}) = \overline{v}_3(\mathbf{b}) = 0$ for $\mathbf{b} \in H$.

2. For $n = 2$, obtain Γ_2 using (8.81). Since $\Gamma_3^H = \Gamma_3$, $\Gamma_2 = \{(3,4), (5,2)\}$ as above. Following (8.82), $\Gamma_2^H = \{(3,4), (5,2)\}$. Then

$$\underline{v}_2(\mathbf{b}) = \max\{3q_1 + 4q_2, 5q_1 + 2q_2\},$$

where $\mathbf{b} = (q_1, q_2)$. Thus, $\underline{v}_2((0,1)) = 4$, $\underline{v}_2((0.4, 0.6)) = 3.6$ and $\underline{v}_2((1,0)) = 5$. Moreover,

$$\underline{v}_2(\mathbf{b}) = \max_{\boldsymbol{\gamma} \in \Gamma_2^H} \{\boldsymbol{\gamma}^\mathsf{T} \mathbf{b}\}$$

for $\mathbf{b} \in \tilde{S}$.

To obtain the upper bound using (8.84) for $\mathbf{b} \in H$,

$$\overline{v}_2(\mathbf{b}) = \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \overline{v}_3(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}, \tag{8.88}$$

but since $\overline{v}_3(\mathbf{b}) = 0$ for all $\mathbf{b}$, $\overline{v}_2(\mathbf{b}) = \max_{\mathbf{a} \in \tilde{A}} \{\tilde{r}(\mathbf{b}, \mathbf{a})\} = \max\{3q_1 + 4q_2, 5q_1 + 2q_2\}$. Thus, $\overline{v}_2((0,1)) = 4$, $\overline{v}_2((0.4, 0.6)) = 3.6$ and $\overline{v}_2((1,0)) = 5$, identical to the corresponding lower bounds.

For $\mathbf{b} \in \tilde{S} \setminus H$ and $\alpha_i \geq 0$, $i = 1, 2, 3$,

$$\overline{v}_2(\mathbf{b}) = \begin{cases} \alpha_1 \overline{v}_2(\mathbf{b}_1) + \alpha_2 \overline{v}_2(\mathbf{b}_2), & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \leq q_1 \leq 0.4, \\ \alpha_2 \overline{v}_2(\mathbf{b}_2) + \alpha_3 \overline{v}_2(\mathbf{b}_3) & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \leq 1 \end{cases}$$

$$= \begin{cases} 4\alpha_1 + 3.6\alpha_2, & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \le q_1 \le 0.4, \\ 3.6\alpha_2 + 5\alpha_3, & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \le 1. \end{cases} \tag{8.89}$$

These bounds are shown for all $\mathbf{b}$ in Figure 8.8a. Note that since the belief state is 1-dimensional, at most two of the α_i need to be positive for any given $\mathbf{b}'$. In particular, if $\mathbf{b}'$ is to the left of the breakpoint at 0.4, only α_1 and α_2 are positive, otherwise α_2 and α_3 are positive.

Note also that the lower bound equals the optimal value found earlier. So, the lower bound approximate policy matches the optimal policy given in (8.65).

3. For $n = 1$, obtain Γ_1 using (8.81) and (8.82). Since $\Gamma_2^H = \Gamma_2$,

$$\Gamma_1 = \{(6.2, 8), (7.32, 7.2), (9, 5.6)\},$$

from calculations using the exact algorithm. It is easy to see from (8.82) that

$$\Gamma_1^H = \{\boldsymbol{\gamma}' \mid \boldsymbol{\gamma}' \in \arg\max_{\boldsymbol{\gamma} \in \Gamma_1} \{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}\}, \mathbf{b} \in \{(0,1), (0.4, 0.6), (1,0)\}\} = \{(6.2, 8), (9, 5.6)\}.$$

As noted in the example of applying Algorithm 8.2, the first $\boldsymbol{\gamma}$-vector corresponds to $\mathbf{a}_1$ and the second to $\mathbf{a}_2$. Moreover observe that Γ_1^H is a **proper subset** of Γ_1. Hence from (8.85),

$$\underline{v}_1(\mathbf{b}) = \max\{6.2q_1 + 8q_2, 9q_1 + 5.6q_2\}$$

for all $\mathbf{b} = (q_1, q_2) \in \tilde{S}$ and $\underline{v}_1((0,1)) = 8$, $\underline{v}_1((0.4, 0.6)) = 7.28$ and $\underline{v}_1((1,0)) = 9$.

To obtain the upper bound for $\mathbf{b} \in H$, from (8.84)

$$\overline{v}_1(\mathbf{b}) = \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \overline{v}_2(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}. \tag{8.90}$$

This requires computing $\eta(o|\mathbf{b}, \mathbf{a})$ and $\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})$ for all $\mathbf{b} \in H$ and $o \in O$. For $\mathbf{b} = (0,1)$, $\eta(o_1|\mathbf{b}, \mathbf{a}_1) = 0.4$, $\eta(o_2|\mathbf{b}, \mathbf{a}_1) = 0.6$, $\eta(o_1|\mathbf{b}, \mathbf{a}_2) = 0.56$, $\eta(o_2|\mathbf{b}, \mathbf{a}_2) = 0.44$, $\boldsymbol{\tau}(o_1, \mathbf{a}_1, \mathbf{b}) = (0,1)$, $\boldsymbol{\tau}(o_2, \mathbf{a}_1, \mathbf{b}) = (0,1)$, $\boldsymbol{\tau}(o_1, \mathbf{a}_2, \mathbf{b}) = (4/7, 3/7)$, and $\boldsymbol{\tau}(o_2, \mathbf{a}_2, \mathbf{b}) = (2/11, 9/11)$. Thus,

$$\begin{aligned} \overline{v}_1((0,1)) &= \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) \overline{v}_2(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\} \\ &= \max\{(0,1)^\mathsf{T}(3,4) + 0.4\overline{v}_2((0,1)) + 0.6\overline{v}_2((0,1)), \\ &\qquad (0,1)^\mathsf{T}(5,2) + 0.56\overline{v}_2((4/7, 3/7)) + 0.44\overline{v}_2((2/11, 9/11))\}. \end{aligned}$$

Note that the second term in the braces requires $\overline{v}_2((4/7, 3/7))$ and $\overline{v}_2((2/11, 9/11))$, that is, at values of $\mathbf{b}' \notin H$. From (8.89) and noting that:

$$\left(\frac{4}{7}, \frac{3}{7}\right) = \frac{5}{7}(0.4, 0.6) + \frac{2}{7}(1,0) \quad \text{and} \quad \left(\frac{2}{11}, \frac{9}{11}\right) = \frac{6}{11}(0,1) + \frac{5}{11}(0.4, 0.6),$$

it follows that

$$\begin{aligned}\overline{v}_1((0,1)) = \max\{&4 + 0.4\overline{v}_2((0,1)) + 0.6\overline{v}_2((0,1)),\\ &2 + 0.56\left(\frac{5}{7}\overline{v}_2((0.4,0.6)) + \frac{2}{7}\overline{v}_2((1,0))\right)\\ &+ 0.44\left(\frac{6}{11}\overline{v}_2((0,1)) + \frac{5}{11}\overline{v}_2((0.4,0.6))\right)\Big\}.\end{aligned}$$

Substituting in the values above results in $\overline{v}_1((0,1)) = 8$. Carrying out similar computations for the other values of $\mathbf{b} \in H$, $\overline{v}_1((0.4,0.6)) = 7.429$ and $\overline{v}_1((1,0)) = 9$.

For $\mathbf{b} \in \tilde{S} \setminus H$ and $\alpha_i \geq 0$, $i = 1,2,3$,

$$\begin{aligned}\overline{v}_1(\mathbf{b}) &= \begin{cases}\alpha_1\overline{v}_1(\mathbf{b}_1) + \alpha_2\overline{v}_1(\mathbf{b}_2), & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \leq q_1 \leq 0.4,\\ \alpha_2\overline{v}_1(\mathbf{b}_2) + \alpha_3\overline{v}_1(\mathbf{b}_3) & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \leq 1\end{cases}\\ &= \begin{cases}8\alpha_1 + 7.429\alpha_2, & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \leq q_1 \leq 0.4,\\ 7.429\alpha_2 + 9\alpha_3, & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \leq 1.\end{cases}\end{aligned} \tag{8.91}$$

Figure 8.8b displays the bounds and the optimal value function. Since Γ_1^H is a proper subset of Γ_1, the lower bound lies below the optimal value function on the region $q_1 \in [0.417, 0.488]$ as result of the calculations displayed in Figure 8.7.

Noting the relationship between supports and actions referred to above, the lower bound approximate policy can be shown to be (the breakpoint in the lower bound occurs at 0.462):

$$d_1(\mathbf{b}) = \begin{cases}\mathbf{a}_1, & 0 \leq q \leq 0.462\\ \mathbf{a}_2, & 0.462 < q \leq 1.\end{cases} \tag{8.92}$$

Note that this policy differs from the optimal policy for $0.462 \leq q < 0.488$. The bound in (8.77) can be used to assess the quality of this approximate policy.

4. For $n = 0$,

$$\Gamma_0^H = \{(9.56, 12), (11.16, 11.04), (13, 9.28)\}.$$

Thus, $\underline{v}_0((0,1)) = 12$, $\underline{v}_0((0.4,0.6)) = 11.088$ and $\underline{v}_0((1,0)) = 13$. Thus for $\mathbf{b}' = (q_1, q_2) \notin H$

$$\underline{v}_0(\mathbf{b}') = \max_{\boldsymbol{\gamma} \in \Gamma_0^H}\{\boldsymbol{\gamma}^\mathsf{T}\mathbf{b}'\} = \max\{9.56q_1 + 12q_2, 11.16q_1 + 11.04q_2, 13q_1 + 9.28q_2\}.$$

Following the same approach as above, the upper bound is

$$\begin{aligned}\overline{v}_0((0,1)) = \max\{&1 + 0.4\overline{v}_1((0,1)) + 0.6\overline{v}_1((0,1)),\\ &2 + 0.56\overline{v}_1((4/7,3/7)) + 0.44\overline{v}_1((2/11,9/11))\} = 12\end{aligned}$$

$\overline{v}_0((0.4,0.6)) = 11.325$ and $\overline{v}_1((1,0)) = 13$.

For $\mathbf{b} \in \tilde{S} \setminus H$ and $\alpha_i \geq 0$, $i = 1, 2, 3$,

$$\begin{aligned}\overline{v}_0(\mathbf{b}) &= \begin{cases} \alpha_1\overline{v}_0(\mathbf{b}_1) + \alpha_2\overline{v}_0(\mathbf{b}_2), & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \leq q_1 \leq 0.4, \\ \alpha_2\overline{v}_0(\mathbf{b}_2) + \alpha_3\overline{v}_0(\mathbf{b}_3) & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \leq 1 \end{cases} \\ &= \begin{cases} 12\alpha_1 + 11.325\alpha_2, & \alpha_1 + \alpha_2 = 1, 0.4\alpha_2 = q_1, 0 \leq q_1 \leq 0.4, \\ 11.325\alpha_2 + 13\alpha_3, & \alpha_2 + \alpha_3 = 1, 0.4\alpha_2 + \alpha_3 = q_1, 0.4 < q_1 \leq 1. \end{cases} \end{aligned} \tag{8.93}$$

The lower bound approximate policy can be shown to be

$$d_1(\mathbf{b}) = \begin{cases} \mathbf{a}_1, & 0 \leq q \leq 0.375 \\ \mathbf{a}_1, & 0.375 < q \leq 0.489 \\ \mathbf{a}_2, & 0.489 < q \leq 1. \end{cases} \tag{8.94}$$

Let Γ^{opt} denote the set of $\boldsymbol{\gamma}$-vectors generated by Algorithm 8.2. Applying Algorithm 8.2 for an additional decision epoch it can be shown that optimal value function for $n = 0$ is defined by

$$\Gamma_0^{\text{opt}} = \{(9.56, 12), (10.213, 11.68), (11.16, 11.04), (13, 9.28)\}.$$

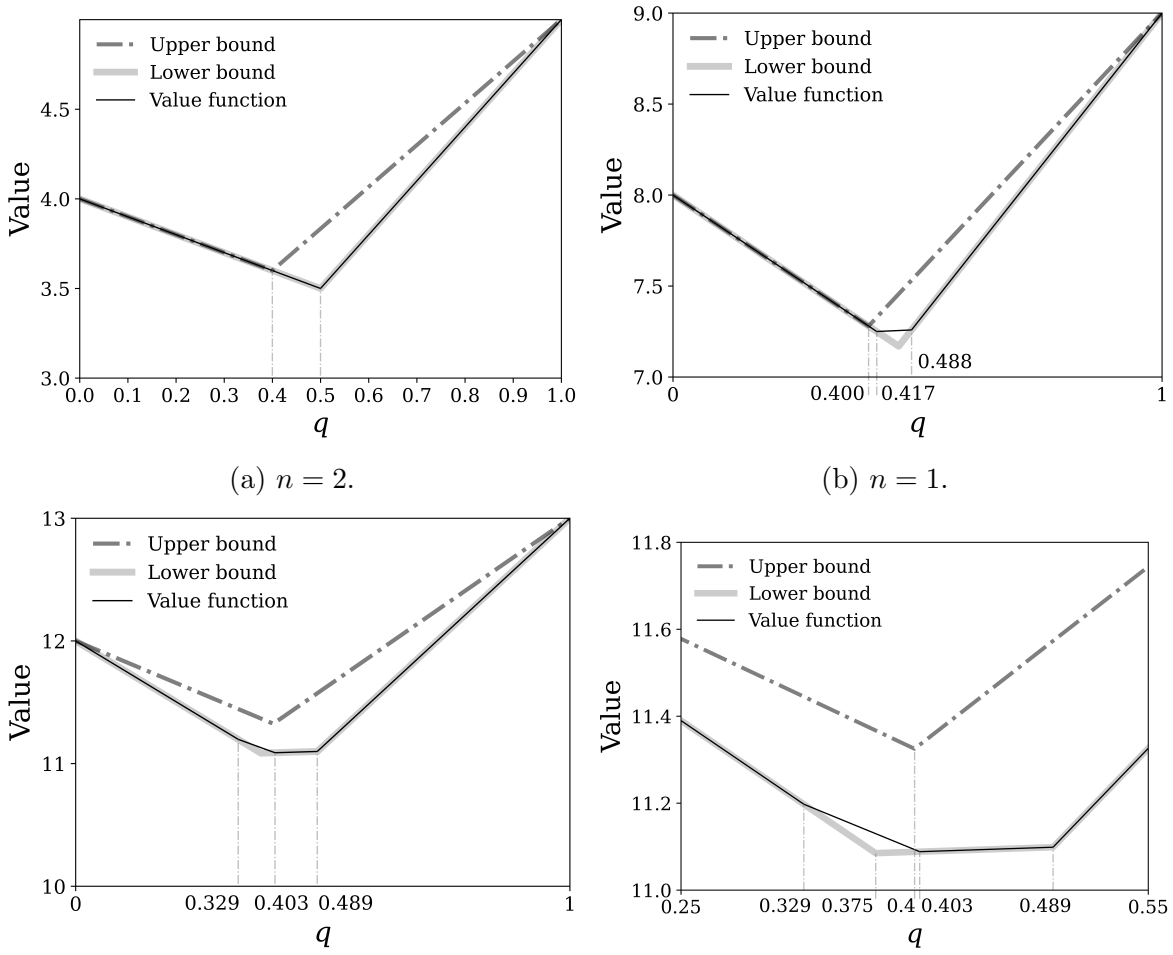
and the corresponding optimal policy is

$$d_1(\mathbf{b}) = \begin{cases} \mathbf{a}_1, & 0 \leq q \leq 0.329 \\ \mathbf{a}_1, & 0.329 < q \leq 0.403 \\ \mathbf{a}_1, & 0.403 < q \leq 0.489 \\ \mathbf{a}_2, & 0.489 < q \leq 1. \end{cases} \tag{8.95}$$

Thus, the lower bound approximate policy matches the optimal policy.

To clearly illustrate the workings of this algorithm, the optimal value function and the bounds at each iteration are plotted below. Again, let $q = q_1 = 1 - q_2$ denote the probability of being in the first state. Figure 8.8 illustrates the bounds for $n = 0, 1, 2$. Figure 8.8a shows that optimal value function for $n = 2$ is defined by $\Gamma_2^{\text{opt}} = \{(3, 4), (5, 2)\}$. The lower bound matches the optimal value function, since $\Gamma_2^{\text{opt}} = \Gamma_2^H$. The upper bound matches the optimal value function for $q \in [0, 0.4]$ and lies above the optimal value function for $q \in (0.4, 1)$. Figure 8.8b shows that the optimal value function for $n = 1$ is defined by $\Gamma_1^{\text{opt}} = \{(6.2, 8), (7.32, 7.2), (9, 5.6)\}$. Since $\Gamma_1^H = \{(6.2, 8), (9, 5.6)\}$, the lower bound lies below the optimal value function for q between 0.417 and 0.488. The upper and lower bounds continue to match the optimal value function on H.

Finally, Figures 8.8c and 8.8d, show that the optimal value function for $n = 0$ lies strictly between the upper and lower bounds: in contrast to the previous iterations, there is a gap between the upper and lower bound at $q = 0.4$. However, as shown above, the lower bound approximate policy matches the optimal policy.



(a) $n = 2$.

(b) $n = 1$.

(c) $n = 0$.

(d) $n = 0$ (zoomed in).

Figure 8.8: Optimal value function and its upper and lower bounds for the two-state POMDP model.

8.5 Infinite horizon models

This section presents a brief discussion of infinite horizon POMDPs focused on the discounted model.

8.5.1 Optimality criterion

For $0 \leq \lambda < 1$, let $v_\lambda^\pi(\mathbf{b})$ be the expected discounted reward of a policy $\pi = (d_1, d_2, \ldots)$, given an initial belief state $\mathbf{b} \in \tilde{S}$:

$$v_\lambda^\pi(\mathbf{b}) := E^\pi \left[\sum_{n=1}^{\infty} \lambda^{n-1} \tilde{r}(\mathbf{b}_n, d_n(\mathbf{b}_n)) \;\middle|\; \mathbf{b}_1 = \mathbf{b} \right]. \tag{8.96}$$

Definition 8.9. An *optimal policy* $\pi^* \in \Pi^{\text{HR}}$ satisfies

$$v_\lambda^{\pi^*}(\mathbf{b}) \geq v_\lambda^{\pi}(\mathbf{b}) \tag{8.97}$$

for all $\pi \in \Pi^{\text{HR}}$ and $\mathbf{b} \in \tilde{S}$.

Definition 8.10. The *value* of the POMDP is defined by

$$v_\lambda^*(\mathbf{b}) := \sup_{\pi \in \Pi^{\text{HR}}} v_\lambda^{\pi}(\mathbf{b}). \tag{8.98}$$

for all $\mathbf{b} \in \tilde{S}$.

Similar to the fully observable case, stationary deterministic policies are optimal within the class of all polices for a POMDP.

Theorem 8.7. There exists $\pi^* \in \Pi^{\text{SD}}$ that achieves the optimal value $v_\lambda^*(\mathbf{b})$ for all $\mathbf{b} \in \tilde{S}$. That is,

$$v_\lambda^{\pi^*}(\mathbf{b}) = v_\lambda^*(\mathbf{b}). \tag{8.99}$$

Going forward, focus is restricted to stationary deterministic policies.

8.5.2 Computing optimal values and finding optimal policies

Extending equation (8.44) to the discounted infinite horizon setting, the value of a stationary deterministic policy $\pi = (d, d, \ldots)$ can be written as

$$v_\lambda^{\pi}(\mathbf{b}) = \tilde{r}(\mathbf{b}, d(\mathbf{b})) + \lambda \sum_{o \in O} \eta(o|\mathbf{b}, d(\mathbf{b})) v_\lambda^{\pi}(\boldsymbol{\tau}(o, d(\mathbf{b}), \mathbf{b})). \tag{8.100}$$

Thus, the optimal value function can be written as

$$v_\lambda^*(\mathbf{b}) = \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \lambda \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v_\lambda^*(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}. \tag{8.101}$$

If $d^*(\mathbf{b})$ achieves the maximum in equation (8.101), then $\pi^* = (d^*, d^*, \ldots)$ is a stationary optimal policy with value $v_\lambda^*(\mathbf{b})$ for each $\mathbf{b} \in \tilde{S}$.

Similar to the fully observable case, the Bellman operator is a contraction mapping, which means that the Banach fixed-point theorem (Theorem 5.5) applies and there exists a unique solution to (8.101) that can be obtained by value iteration. Because the limit of PLC functions need not be piecewise linear, all that can be guaranteed is that $v_\lambda^*(\mathbf{b})$ is convex.

As frequently pointed out above there are challenges to applying value iteration. They will be discussed after describing a value iteration algorithm conceptually.

Algorithm 8.5. Value iteration for an infinite horizon discounted POMDP

1. **Initialize:**
   (a) Specify $v'(\mathbf{b})$ for all $\mathbf{b} \in \tilde{S}$.
   (b) Specify $\epsilon > 0$ and $\sigma > \epsilon$.
2. **Iterate:** While $\sigma \geq \epsilon$:
   (a) $v(\mathbf{b}) \leftarrow v'(\mathbf{b})$ for all $\mathbf{b} \in \tilde{S}$.
   (b) For all $\mathbf{b} \in \tilde{S}$, compute
   $$v'(\mathbf{b}) \leftarrow \max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \lambda \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}. \tag{8.102}$$
   (c) $\sigma \leftarrow \|\mathbf{v}' - \mathbf{v}\|$.
3. **Terminate:** For all $\mathbf{b} \in \tilde{S}$, return
   $$d_\epsilon(\mathbf{b}) \in \arg\max_{\mathbf{a} \in \tilde{A}} \left\{ \tilde{r}(\mathbf{b}, \mathbf{a}) + \lambda \sum_{o \in O} \eta(o|\mathbf{b}, \mathbf{a}) v(\boldsymbol{\tau}(o, \mathbf{a}, \mathbf{b})) \right\}.$$

Stopping criteria

Most approaches use a stopping criterion of the form

$$\|\mathbf{v}' - \mathbf{v}\| := \max_{\mathbf{b} \in \tilde{S}} |v'(\mathbf{b}) - v(\mathbf{b})| < \epsilon.$$

Note that the "max" rather than the "sup" is appropriate since for a finite number of iterations, $v(\mathbf{b})$ is continuous and $\tilde{S}$ is compact so the "max" is attained. The challenge in applying this criterion is to compute the "max", which itself involves solving an optimization problem. In small instances this may not be a problem because the iterates can be represented in closed form. In larger instances an option is to evaluate this criterion at a large number of states. Alternatives include stopping when there are small changes in the $\boldsymbol{\gamma}$-vectors or in the policy itself.

Optimal value function updates

For problems with a few states and actions, the update can be implemented exactly as in Algorithm 8.2. This approach may not be suitable for large problems since each step is computationally intensive especially when G_n is large.

Most workable alternatives are based on the ideas underlying Algorithms 8.3 and 8.4, namely, to restrict calculation of supports to a limited number of belief states. Such algorithms are referred to as *point-based value iteration*.

Algorithms differ on how the points at which to compute updates are chosen. Algorithms 8.3 and 8.4 use a fixed grid of points, some are based on sampling points, and some are based on using a tree to represent the possible realizations of the belief states. The details are beyond the scope of this book and the reader is referred to references in the Bibliographic Remarks section.

Large problems

The most promising approaches for very large problems appear to be those that are based on reinforcement learning methods (Chapter 11). Policy gradient type methods do not use the belief state approach and instead derive a randomized policy based on observations. In such methods, policies are determined by neural networks.

Bounds

The bound from Theorem 5.12 in Chapter 5 may be useful in assessing the quality of the greedy policy obtained by the above algorithm. In the notation of this chapter, for a vector $\mathbf{v}$, $\overline{\mathbf{v}} := \max_{\mathbf{b}\in\tilde{S}} v(\mathbf{b})$ and $\underline{\mathbf{v}} := \min_{\mathbf{b}\in\tilde{S}} v(\mathbf{b})$.

Theorem 8.8. For all $\mathbf{b} \in \tilde{S}$,

$$\begin{aligned} v(\mathbf{b}) + & (1-\lambda)^{-1}(\underline{\mathbf{v}' - \mathbf{v}}) \\ & \leq v'(\mathbf{b}) + \lambda(1-\lambda)^{-1}(\underline{\mathbf{v}' - \mathbf{v}}) \\ & \leq v_\lambda^{d^\infty}(\mathbf{b}) \leq v_\lambda^*(\mathbf{b}) \\ & \leq v'(\mathbf{b}) + \lambda(1-\lambda)^{-1}(\overline{\mathbf{v}' - \mathbf{v}}) \\ & \leq v(\mathbf{b}) + (1-\lambda)^{-1}(\overline{\mathbf{v}' - \mathbf{v}}), \end{aligned} \tag{8.103}$$

where

$$d(\mathbf{b}) \in \arg\max_{\mathbf{a}\in\tilde{A}} \left\{ \tilde{r}(\mathbf{b},\mathbf{a}) + \lambda \sum_{o\in O} \eta(o|\mathbf{b},\mathbf{a}) v\big(\boldsymbol{\tau}(o,\mathbf{a},\mathbf{b})\big) \right\}.$$

The challenge in applying these bounds is computing the quantities $(\overline{\mathbf{v}' - \mathbf{v}})$ and $(\underline{\mathbf{v}' - \mathbf{v}})$, which involve computing the maximum and minimum over the uncountable set of belief states.

Note that if $v(\mathbf{b}) = 0$, then $v'(\mathbf{b}) = \max_{a\in\tilde{A}} \tilde{r}(\mathbf{b},\mathbf{a})$ so that

$$\underline{r} := \min_{s\in s, a\in A_s} r(s,a) \leq v'(\mathbf{b}) \leq \max_{s\in s, a\in A_s} r(s,a) := \overline{r}$$

and the outer bounds become the obvious inequalities

$$\frac{\underline{r}}{1-\lambda} \leq v_\lambda^{d^\infty}(\mathbf{b}) \leq v_\lambda^*(\mathbf{b}) \leq \frac{\overline{r}}{1-\lambda}$$

for all $\mathbf{b} \in \tilde{S}$.

Note that Lovejoy [1991a] provided the following bound on the difference between the optimal n-period value function $v_n^*(\mathbf{b})$ and the optimal infinite horizon value function:

$$\frac{\underline{r}\,\lambda^n}{1-\lambda} \leq v_n^*(\mathbf{b}) \leq v_\lambda^*(\mathbf{b}) \leq \frac{\overline{r}\,\lambda^n}{1-\lambda}.$$

Hence the inequality

$$0 \leq v_\lambda^*(\mathbf{b}) - v_n^*(\mathbf{b}) \leq \frac{(\overline{r}-\underline{r})\,\lambda^n}{1-\lambda}$$

can determine how many value iteration steps to apply to obtain a good approximation of the infinite horizon optimal value function.

8.6 Examples

This section formulates POMDP models in several applications. In contrast to an MDP formulation, a POMDP formulation requires the additional specification of the observations. Derived MDP formulations for several examples are also provided.

8.6.1 Preventive maintenance and inspection

A machine that manufactures a product can be in one of two conditions, "good" or "bad". At any decision epoch, the decision maker can do one of three things: 1) minimal observation of the output, 2) actively inspect the output or 3) repair the machine. As a consequence of choosing an action, the decision maker receives a signal regarding the status of the equipment. Assume all actions each require one period and generate a signal at the end of the period. The first two actions provide a signal of the machine's status with active inspection providing a more accurate signal; neither affects machine state. The repair action results in the machine being in the "good" state by the start of the next period.

Different costs are associated with observing and repairing the machine. Revenue depends on the quality of the product, and thus depends on the machine state. This model has both passive and active information acquisition actions. The first action is passive and amounts to a visual inspection of the product. The second action is active and could come in the form of a diagnostic of the product's functionality. A POMDP formulation follows.

Decision epochs: The horizon may be either finite or infinite.

$$T = \{1, 2, \ldots, N\}, \quad N \leq \infty.$$

Hidden States:

$$S = \{s_1, s_2\},$$

where s_1 = "good" and s_2 = "bad".

Actions:

$$A = \{a_1, a_2, a_3\},$$

where a_1 = "do nothing", a_2 = "inspect" and a_3 = "repair". Note that all actions are applicable in each state.

Observations:

$$O = \{o_1, o_2\},$$

where o_1 = "normal" and o_2 = "abnormal". The probability of these observations depends on the hidden state and the action at the previous decision epoch. Since there are only two signals, it suffices to specify the probability of observing o_1:

$$u(o_1|s,a) = \begin{cases} \alpha_1 & s = s_1, a = a_1 \\ \alpha_2 & s = s_2, a = a_1 \\ \beta_1 & s = s_1, a = a_2 \\ \beta_2 & s = s_2, a = a_2 \\ 1 & s \in \{s_1, s_2\}, a = a_3. \end{cases}$$

The expression for $u(o_1|s,a)$ assumes that it is more likely to observe a "normal" signal in the "good" state than the "bad" state ($\alpha_1 > \alpha_2$, $\beta_1 > \beta_2$) and that inspection provides more accurate information than observation ($\beta_1 > \alpha_1$, $\beta_2 > \alpha_2$). If the decision maker decides to repair the equipment, it takes one period, the hidden state becomes "good" at the next decision epoch and the observed signal will be "normal" with probability 1.

Rewards: Rewards depend on assumptions regarding when the equipment changes state between decision epochs. For ease of exposition assume that a transition between states occurs at the end of a period, immediately preceding the subsequent decision epoch. Under this assumption, rewards are accrued prior to, and thus are independent of, the subsequent state. The reward function is given by

$$r(s,a) = \begin{cases} r_1 & s = s_1, a = a_1 \\ r_2 & s = s_2, a = a_1 \\ r_1 - k & s = s_1, a = a_2 \\ r_2 - k & s = s_2, a = a_2 \\ -K & s \in \{s_1, s_2\}, a = a_3. \end{cases}$$

Assume that the equipment generates greater revenue in the "good" state ($r_1 > r_2$) and that it costs more to repair than to inspect ($K > k$). When the machine is being repaired it does not produce any product, hence there is no revenue generated during that period.

Transition probabilities:

$$p(s'|s,a) = \begin{cases} 1-\gamma & s' = s_1, s = s_1, a \in \{a_1, a_2\} \\ \gamma & s' = s_2, s = s_1, a \in \{a_1, a_2\} \\ 0 & s' = s_1, s = s_2, a \in \{a_1, a_2\} \\ 1 & s' = s_2, s = s_2, a \in \{a_1, a_2\} \\ 1 & s' = s_1, s \in \{s_1, s_2\}, a = a_3 \\ 0 & s' = s_2, s \in \{s_1, s_2\}, a = a_3 \end{cases} \tag{8.104}$$

When the decision maker decides to observe or inspect, (8.104) shows that the probability that the equipment changes status from "good" to "bad" in one period equals γ and that it remains in the "bad" state with certainty once it reaches there. When the decision maker decides to repair, the equipment moves to the "good" state with certainty.

The dynamics are depicted graphically in Figure 8.9. Note that the observation probabilities depend on action choice.

The derived MDP

Next, the derived MDP for this example is formulated. Decision epochs remain unchanged.

States: Since there are two hidden states,

$$\tilde{S} = \{(q_1, q_2) \mid q_1 + q_2 = 1, q_1 \geq 0, q_2 \geq 0\} = \mathcal{S}_1,$$

where q_1 is the probability the system is in the "good" state and q_2 is the probability the system is in the "bad" state.

Actions: In this example, the action set is the same for each hidden state and can be implemented in each state. Thus, in the derived MDP, the action set simplifies to

$$\tilde{A} = A = \{a_1, a_2, a_3\},$$

where a_1, a_2, a_3 are as defined in the POMDP.

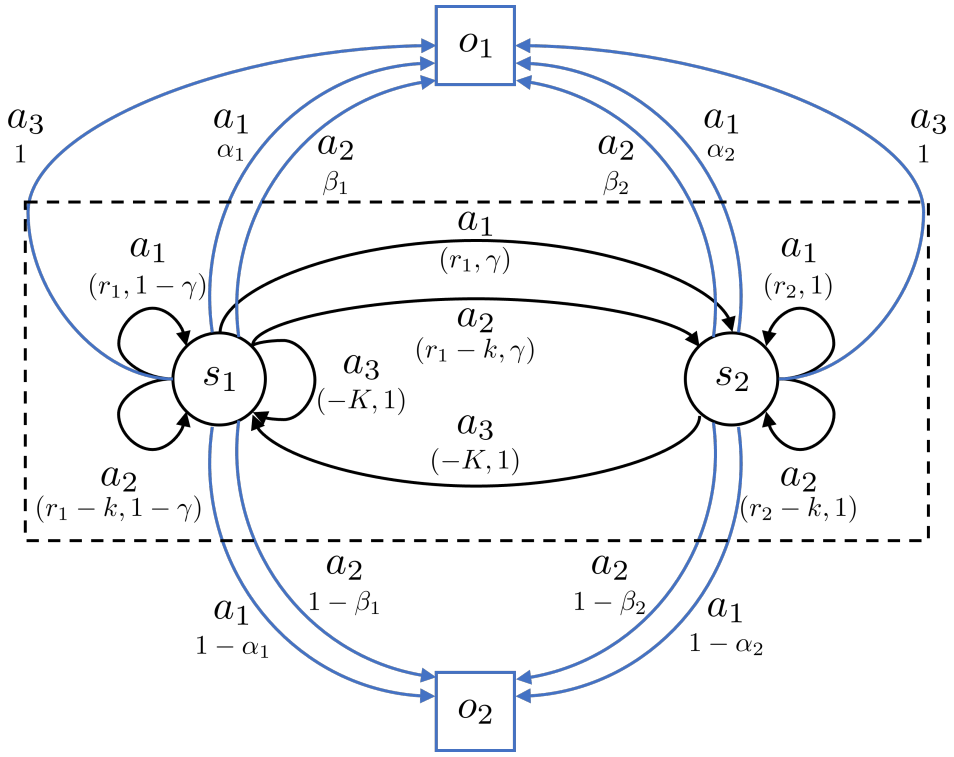


Figure 8.9: Graphical representation of the preventive maintenance POMDP model. The area in the dashed box represents the hidden (unobservable) Markov decision process. Labels (r, p) on the unobservable transitions refer to rewards and probabilities, respectively. The squared boxes indicate the observable signals, with probabilities of observation on the arcs from the hidden states.

Rewards: Let $\mathbf{b} = (q_1, q_2)$. From equation (8.16), with a scalar action,

$$\tilde{r}(\mathbf{b}, a) = \sum_{s \in S} r(s, a) b(s) = \begin{cases} r_1 q_1 + r_2 q_2 & a = a_1 \\ r_1 q_1 + r_2 q_2 - k & a = a_2 \\ -K & a = a_3. \end{cases}$$

Transition Probabilities: Let $\mathbf{b} = (q_1, q_2)$ and $\mathbf{b}' = (q_1', q_2')$. From equation (8.18),

$$\begin{aligned} \eta(o_1 | \mathbf{b}, a) &= \sum_{s' \in S} u(o_1 | s', a) \sum_{s \in S} p(s' | s, a) b(s) \\ &= \begin{cases} \alpha_1(1-\gamma)q_1 + \alpha_2(\gamma q_1 + q_2) & a = a_1 \\ \beta_1(1-\gamma)q_1 + \beta_2(\gamma q_1 + q_2) & a = a_2 \\ 1 & a = a_3 \end{cases} \end{aligned}$$

and

$$\begin{aligned}\eta(o_2|\mathbf{b},a) &= \sum_{s'\in S} u(o_2|s',a)\sum_{s\in S} p(s'|s,a)b(s)\\ &= \begin{cases}(1-\alpha_1)(1-\gamma)q_1+(1-\alpha_2)(\gamma q_1+q_2) & a=a_1\\ (1-\beta_1)(1-\gamma)q_1+(1-\beta_2)(\gamma q_1+q_2) & a=a_2\\ 0 & a=a_3.\end{cases}\end{aligned}$$

From equation (8.19),

$$\boldsymbol{\tau}(o_1,a_1,\mathbf{b}) = \left(\frac{\alpha_1(1-\gamma)q_1}{\alpha_1(1-\gamma)q_1+\alpha_2(\gamma q_1+q_2)}, \frac{\alpha_2(\gamma q_1+q_2)}{\alpha_1(1-\gamma)q_1+\alpha_2(\gamma q_1+q_2)}\right) \tag{8.105}$$

$$\boldsymbol{\tau}(o_1,a_2,\mathbf{b}) = \left(\frac{\beta_1(1-\gamma)q_1}{\beta_1(1-\gamma)q_1+\beta_2(\gamma q_1+q_2)}, \frac{\beta_2(\gamma q_1+q_2)}{\beta_1(1-\gamma)q_1+\beta_2(\gamma q_1+q_2)}\right) \tag{8.106}$$

$$\boldsymbol{\tau}(o_1,a_3,\mathbf{b}) = (1,0) \tag{8.107}$$

$$\boldsymbol{\tau}(o_2,a_1,\mathbf{b}) = \left(\frac{(1-\alpha_1)(1-\gamma)q_1}{(1-\alpha_1)(1-\gamma)q_1+(1-\alpha_2)(\gamma q_1+q_2)}, \frac{(1-\alpha_2)(\gamma q_1+q_2)}{(1-\alpha_1)(1-\gamma)q_1+(1-\alpha_2)(\gamma q_1+q_2)}\right) \tag{8.108}$$

$$\boldsymbol{\tau}(o_2,a_2,\mathbf{b}) = \left(\frac{(1-\beta_1)(1-\gamma)q_1}{(1-\beta_1)(1-\gamma)q_1+(1-\beta_2)(\gamma q_1+q_2)}, \frac{(1-\beta_2)(\gamma q_1+q_2)}{(1-\beta_1)(1-\gamma)q_1+(1-\beta_2)(\gamma q_1+q_2)}\right) \tag{8.109}$$

Note that $\boldsymbol{\tau}(o_2,a_3,\mathbf{b})$ is not well-defined, since $\eta(o_2|\mathbf{b},a_3)=0$. Under a_3, with probability 1 the system transitions to s_1 and observation o_1 is generated. This is an example where there are fewer updated belief states than there are observations, for a given current belief state and action.

Finally,

$$\tilde{p}(\mathbf{b}'|\mathbf{b},a) = \begin{cases}\eta(o_1|\mathbf{b},a_1) & \mathbf{b}'=\boldsymbol{\tau}(o_1,a_1,\mathbf{b})\\ \eta(o_1|\mathbf{b},a_2) & \mathbf{b}'=\boldsymbol{\tau}(o_1,a_2,\mathbf{b})\\ \eta(o_1|\mathbf{b},a_3) & \mathbf{b}'=\boldsymbol{\tau}(o_1,a_3,\mathbf{b})\\ \eta(o_2|\mathbf{b},a_1) & \mathbf{b}'=\boldsymbol{\tau}(o_2,a_1,\mathbf{b})\\ \eta(o_2|\mathbf{b},a_2) & \mathbf{b}'=\boldsymbol{\tau}(o_2,a_2,\mathbf{b})\\ 0 & \text{otherwise.}\end{cases}$$

8.6.2 Models with unknown parameters: Bayesian decision problems

Models with unknown parameters represent a particularly useful class of POMDPs. In such models the decision maker knows the form of the distribution of a random variable that affects rewards and transition probabilities up to the unknown parameter values. For example, in a production process the defect rate may be binomial with unknown defect probability. The decision maker acquires information about the parameter through a sequence of actions that affect both the information received and the reward earned. Formulations of such models are illustrated through examples below. The key feature is that the state represents the unknown parameters and remains unchanged over time.

Newsvendor with unknown demand

As an example, consider a variant of the newsvendor problem introduced in Section 3.1.2, in which the demand has a discrete distribution with unknown parameter μ. Denote its probability mass function by $f(d|\mu)$. Assume that μ can assume a finite number of values $\{\mu_1, \mu_2 \ldots, \mu_M\}$ and the newsvendor has a prior distribution on μ, denoted $w_1(\mu)$.

In such a model, inventory is perishable and lasts only one period. Each period, the decision maker purchases units of a product from a supplier at a cost c and sells them at price $g > c$ throughout the period. If any items remain unsold at the end of the period, they are scrapped at a reduced price $h < c$. Although the model with a known demand distribution is on a one-period model, this model has multiple decision epochs that account for updates on the knowledge regarding the value of the unknown parameter.

Decision epochs: The model may be either finite or infinite horizon.

$$T = \{1, 2, \ldots, N\}, \quad N \leq \infty.$$

Hidden states: States represent possible values for the unknown parameter. Under the above assumption,

$$S = \{\mu_1, \mu_2 \ldots, \mu_M\}.$$

In greater generality, S could be an interval (or the positive real line).

Actions: Actions represent the number of units to order at each decision epoch,

$$A = \{1, \ldots, a_{\max}\},$$

where $a_{\max}$ is some large finite value representing the maximum amount that can be ordered in a period.

Observations: The newsvendor observes only the number of units sold. If external demand is less than the quantity ordered, the newsvendor observes demand, but when demand exceeds inventory, demand is *censored* by the order quantity. That is, the decision maker is only guaranteed to observe a lower bound on the true demand. Thus,

$$O = \{0, 1, \ldots, a_{\max}\}.$$

It is important to note that action choice determines the values of o that have positive probability. The observation distribution can be written as

$$u(o|\mu, a) = \begin{cases} f(o|\mu) & o < a, \mu \in S \\ \sum_{i=a}^{\infty} f(i|\mu) & o = a, \mu \in S \\ 0 & o > a, \end{cases}$$

where the state is written as μ to emphasize its meaning.

Rewards: The newsvendor receives rewards that depend on the values of a and μ. The following expression for the *expected* reward accounts for the effect of the unknown parameter:

$$r(\mu, a) = \sum_{d=0}^{a-1} (Gd - L(a-d)) f(d|\mu) + Ga \sum_{d=a}^{\infty} f(d|\mu). \tag{8.110}$$

When the order quantity a exceeds demand d, the first expression on the right hand side of (8.110) combines the gain from sales at a profit $G = g - c$ with the loss of $L = h - c$ for unsold items when the order quantity a exceeds demand d. The second expression corresponds to the demand being greater than or equal to the order quantity. In this case, only the number of ordered units can be sold.

Transition probabilities: Since the parameter value remains constant over the planning horizon, for all $a \in A$

$$p(j|s, a) = \begin{cases} 1 & j = s \\ 0 & j \neq s. \end{cases}$$

Searching for an object

A searcher seeks to find an object in a region divided into an M-cell grid. A prior distribution on the object's location initiates the search. At each decision epoch the searcher decides which cell to search. Search is not perfect: if the searched cell contains the object, the searcher finds the item with probability $p_f > 0$. Upon finding the object, the searcher receives a reward of R and terminates the search. Model variants include multiple within-cell search routines with different costs, search costs that depend on the cell, moving objects, and a maximum search budget.

The POMDP formulation below describes particular model features. It includes some of the same features as the optimal stopping problem in Section 3.8 and may be classified as an *episodic* model.

Decision epochs: Since the object is not found with certainty, the number of decision epochs to find it is unbounded. Therefore

$$T = \{1, 2, \ldots\}.$$

If search were perfect, at most M epochs would be required.

Hidden states: States represent the cell number that contains the object and a stopped state Δ corresponding to the search being completed after finding the object.

$$S = \{1, 2, \ldots, M, \Delta\}.$$

Note that this model combines unobservable states corresponding to the location of the hidden object and an observable state resulting from the object being found.

Actions: Actions represent the cell to search at each decision epoch. Once the object is found the decision maker may only choose the zero-cost action ϕ leaving it in the stopped state. So

$$A_s = \begin{cases} \{1, \ldots, M\} & s \neq \Delta \\ \{\phi\} & s = \Delta \end{cases}$$

Observations: Set $O = \{F, NF, \Delta_o\}$ where F indicates "object found", NF indicates "object not found" and Δ_o corresponds to being in the stopped state. The searcher observes F only when successfully searching the cell that contains the object. Observation probabilities follow. When the search is ongoing, that is when $s \neq \Delta$,

$$u(o|s, a) = \begin{cases} p_f & o = F,\ a = s \\ 1 - p_f & o = NF,\ a = s \\ 1 & o = NF,\ a \neq s \end{cases}$$

and when the search has stopped, $u(\Delta_o|\Delta, \phi) = 1$.

Rewards: It costs one time unit to search a cell. Rewards in the stopped state are distinguished from those in the non-stopped state. When $s \neq \Delta$

$$r(s, a, j) = \begin{cases} R & a = s,\ j = \Delta \\ -1 & a = s,\ j = s \\ -1 & a \neq s. \end{cases}$$

The first case corresponds to finding the object, which results in a transition to the stopped state Δ. The second case corresponds to searching the correct cell, but not finding it. The third case corresponds to searching an incorrect cell. When the search has stopped, $r(\Delta, \phi, \Delta) = 0$.

Transition probabilities: When $s \neq \Delta$

$$p(j|s,a) = \begin{cases} 1-p_f & a = s,\ j = s \\ p_f & a = s,\ j = \Delta \\ 0 & a = s,\ j \neq s \\ 1 & a \neq s,\ j = a \\ 0 & a \neq s,\ j \neq a \end{cases}$$

and when $s = \Delta$

$$p(j|s,a) = \begin{cases} 1 & a = \phi,\ j = \Delta \\ 0 & a = \phi,\ j \neq \Delta. \end{cases}$$

The reasoning behind these transition probabilities is as follows. The hidden state s represents the true but unknown location of the object. The system state only changes state when the searcher finds the object, in which case the system moves to the stopped state Δ. This occurs with probability p_f when the state containing the object is searched. No other transitions are possible corresponding to the 0 probabilities above. If the searcher searches a state that does not contain the object, that is $a \neq s$, the state does not change. Once in the stopped state the system remains there forever.

Minimally invasive surgery

An interesting example of an optimal search problem arises in minimally invasive (robotic) surgery. A surgeon seeks to maneuver instruments from an initial incision on the surface of the body to a target organ while avoiding damage to critical structures such as arteries along the way. Pre-operative images provide some indication of the location of these structures but their precise location might change during the surgery. The surgeon has two search options: quick cuts through overlying tissue or slowly peeling away layers of tissue to safely learn what lies behind the tissue.

In practice there is a bound on the time to reach the organ and search is limited to contiguous regions. That means that the choice of the location for the next action must be contiguous to the site of the previous action.

8.6.3 Multi-armed bandits

Bandit models have received a great deal of attention in the literature; they are simple to describe, widely applicable and have produced some elegant mathematical results. The expression "bandit" refers to a slot machine, a common feature in a gambling casino. In this context, a player selects a machine, inserts a coin or token, pulls its "arm", and receives an uncertain payoff. The reward received from playing a particular machine does not depend on or provide information about any of the other machines available to the player. Questions of interest are which machine to play, when to switch to other machines, and when to stop playing.

Bandit models abstract the key features of this process. At each decision epoch, the decision maker selects one of K bandits to play or equivalently "arms to pull". Bandit models may be classified on the basis of the reward generating mechanism as follows:

1. In a *classical statistical bandit* "pulling an arm" results in receiving a reward from a probability distribution with unknown parameters corresponding to that arm.

2. In a *restless Markov bandit*, "pulling an arm" results in receiving a reward that depends on the state of an unobservable Markov chain corresponding to that arm.

Models may include a cost for switching between arms.

Applications

Selecting an arm to pull involves a trade-off between *exploitation* and *exploration*. The decision maker may either exploit the arm that is thought to have the highest expected payoff or explore other arms (pull other arms) to find one with a higher expected payoff. Specific examples include:

1. **Optimal Foraging:** To maximize survival, an animal may either continue to forage in the current area or move to another area (patch) with the hope of acquiring more nutrition while expending less energy. The bandit model applies where an "arm" corresponds to which patch to explore. This application requires including energy expenditure associated with changing patches.

2. **Project Management:** Faced with K projects in progress, at each decision epoch, the decision maker decides which project to work on.

3. **Sequential Clinical Trials:** Randomized controlled clinical trials have become the gold standard for establishing the effectiveness of new drugs, therapies and medical procedures. In a multi-armed clinical trial, patients are randomly assigned to a control or to one of several treatments under study. The goal is to determine which treatment is most effective and which have a larger effect than placebo. Random assignment of patients to a treatment or control arm accounts for the effect of potentially confounding factors.

   When patients are assigned to a treatment or control sequentially[20], the challenge is to develop a mechanism to assign of a patient to a treatment "arm" based on previously acquired data. The use of a multi-armed bandit framework has been shown to efficiently determine the best treatment and as well assign the most patients to the best treatment during the trial.

[20] This contrasts with the standard approach of randomizing all participants at the start of the trial.

4. **A/B Testing:** Similarly to a controlled clinical trial, A/B testing refers to a controlled statistical experiment with two "treatments" referred to as Treatment A and Treatment B. While deeply rooted in the statistical literature, it has become a widely used approach for choosing between alternative web site designs and online marketing campaigns such as the form and content of banner ads on web pages. When used sequentially, this may be viewed as a two-armed bandit. In the case of online advertising, when a user clicks a website link, the decision maker (i.e., website) chooses whether to display ad design A or ad design B, seeking to determine which design generates the most clicks or the most revenue. Subsequently, new designs are introduced and compared to the previous best design. Multi-armed bandits apply when there are more than two designs to compare.

The next two sections formulate the above bandit models as POMDPs.

POMDP formulation of classical statistical bandit

In the classical statistical bandit model, pulling arm k, $k = 1, \ldots, K$, results in an observation drawn from a discrete probability distribution $w(\cdot|\mu_k)$ with unknown mean μ_k on a discrete set $O = \{o_1, o_2, \ldots, o_L\}$, where L may be finite or infinite. When $O = \{0, 1\}$ the model is referred to as a Bernoulli bandit and applies to settings when the outcome may be viewed as "success" or "failure" such as in A/B testing or clinical trials. Assume that for $k = 1, \ldots, K$, μ_k assumes values in the finite set $M = \{m_1, \ldots, m_J\}$. This means that the unknown mean for each arm may take on any value in M.

Continuous variants of this model have been widely studied but are outside of the scope of this book. In such models, O is a continuum with probability density functions replacing probability mass functions.

Decision epochs: The model may be either finite or infinite horizon

$$T = \{1, 2, \ldots, N\}, \quad N \leq \infty.$$

Hidden states: Hidden states represent the possible values for the *unknown* mean for each of the K arms. Thus

$$S = M \times M \times \cdots \times M = M^K = \{(s_1, \ldots, s_K) \mid s_k \in M, k = 1, \ldots, K\}.$$

Actions: Actions represent which arm to play

$$A = \{1, \ldots, K\}.$$

Observations: When choosing the k-th arm, the decision maker observes the value $o \in O$ with probability $w(o|s_k)$ so that

$$u(o|(s_1, \ldots, s_K), k) = w(o|s_k).$$

This specification uses the fact that since $s_k \in M$, it equals one of $m_1, m_2, ..., m_J$.

Reward: Assuming no cost for switching arms,

$$r((s_1, \ldots, s_K), k) = \sum_{o \in O} f(o) w(o|s_k),$$

where $f(\cdot)$ is a known real-valued function on O. Usually the reward and observation are the same so $f(o) = o$ in which case $r((s_1, \ldots, s_K), k) = s_k$.

Transition probabilities: Regardless of the action chosen, the state of the arm does not change, so that for each $a \in A$

$$p((s'_1, s'_2, \ldots, s'_K)|(s_1, s_2, \ldots, s_K), a) = \begin{cases} 1 & \text{for } s'_k = s_k,\ k = 1, 2, \ldots, K \\ 0 & \text{otherwise.} \end{cases}$$

Implementation of this model requires an initial belief state or prior distribution on M for each arm. When the decision maker plays arm k, only that distribution is updated. When the state space for each Markov chain is M-dimensional, the POMDP formulation of the model has M^K states, so the model might appear intractable. However, it has been shown that the model decomposes into K independent problems that can be analyzed independently.

A Bernoulli bandit example

This example shows how this formulation applies to a three-armed Bernoulli bandit and provides the equivalent Markov decision process formulation. In this example $K = 3$, $O = \{0, 1\}$ and $M = \{m_1, m_2\}$ with $m_2 > m_1$. The state of the system, (s_1, s_2, s_3) indicates possible values for the mean of each arm, that is $s_k = m_1$ or $s_k = m_2$ for each $k = 1, 2, 3$. The Bernoulli assumption means that if arm k is in state s_k,

$$1 - w(0|s_k) = w(1|s_k) = s_k.$$

This means that s_k is the *probability* of observing a success, equivalently a 1, if arm k is selected. Hence, the observation probabilities are given by

$$u(0|(s_1, s_2, s_3), k) = 1 - s_k \quad \text{and} \quad u(1|(s_1, s_2, s_3), k) = s_k$$

Assume further that $f(o) = o$ so that the decision maker receives reward 1 if the arm pull generates a success and 0 if it generates a failure. Thus in this example rewards and observations are not distinguished.

The equivalent MDP provides additional insight into this example and the interpretation of quantities in this transformation. Let q_k denote the decision maker's assessment that $\mu_k = m_1$ so that $1 - q_k$ denotes the assessment that $\mu_k = m_2$. There are eight possible values for (s_1, s_2, s_3) so that a belief state provides the joint probability of each. For example, belief states are of the form, assuming independent arms:

$$b_1(m_2, m_2, m_1) = P\big[X_1 = (m_2, m_2, m_1)\big] = (1 - q_1)(1 - q_2)q_3.$$

Choosing action k results[21] in an (expected) reward

$$\tilde{r}(\mathbf{b}, k) = m_1 q_k + m_2(1 - q_k)$$

and observation probabilities

$$\eta(0|\mathbf{b}, k) = (1 - m_1)q_k + (1 - m_2)(1 - q_k) \quad \text{and} \quad \eta(1|\mathbf{b}, k) = m_1 q_k + m_2(1 - q_k).$$

Since the underlying state does not change, choosing action k only impacts q_k. Let $q'_k(o)$ denote the posterior probability of the state of arm k after observing signal $o = 0$ or $o = 1$. When $o = 0$,

$$\begin{aligned} q'_k(0) &= P[\mu_k = m_1 | o = 0] \\ &= \frac{P[o = 0|\mu_k = m_1]P[\mu_k = m_1]}{P[o = 0|\mu_k = m_1]P[\mu_k = m_1] + P[o = 0|\mu_k = m_2]P[\mu_k = m_2]} \\ &= \frac{(1 - m_1)q_k}{(1 - m_1)q_k + (1 - m_2)(1 - q_k)} \end{aligned}$$

and when $o = 1$,

$$q'_k(1) = \frac{m_1 q_k}{m_1 q_k + m_2(1 - q_k)}.$$

Then $\boldsymbol{\tau}(0, k, \mathbf{b})$ and $\boldsymbol{\tau}(1, k, \mathbf{b})$ are each 8-dimensional vectors of joint probabilities with $q'_k(0)$ and $q'_k(1)$ replacing q_k in the joint probability distributions.

To gain more insight into this formulation, assume $m_1 = 0.5$ and $m_2 = 0.6$ and suppose the prior probability that $s_1 = m_1$ is $q_1 = 0.7$, the prior probability that $s_2 = m_1$ is $q_2 = 0.5$ and the prior probability that $s_3 = m_1$ is $q_3 = 0.2$. Then the components of $\mathbf{b}_1$ are given by

$$b_1(s_1, s_2, s_3) = \begin{cases} 0.7 \cdot 0.5 \cdot 0.2 & \text{if } (s_1, s_2, s_3) = (m_1, m_1, m_1) \\ 0.7 \cdot 0.5 \cdot 0.8 & \text{if } (s_1, s_2, s_3) = (m_1, m_1, m_2) \\ \quad \cdots & \\ 0.3 \cdot 0.5 \cdot 0.8 & \text{if } (s_1, s_2, s_3) = (m_2, m_2, m_2). \end{cases}$$

[21] This calculation is actually more complex and must account for all eight possible values of (s_1, s_2, s_3).

Suppose arm 3 is selected and a 1 is observed. then

$$q_3'(1) = \frac{0.5 \cdot 0.2}{0.5 \cdot 0.2 + 0.6 \cdot 0.8} = 0.172.$$

In this case, this means that the probability that $s_3 = m_2$ has increased from 0.8 to 0.828 and in $\mathbf{b}_2$, 0.172 replaces 0.2 and 0.828 replaces 0.8 in the above expressions for the components of $\mathbf{b}_1$.

POMDP formulation of the restless Markov bandit

The unobservable state of arm k, $k = 1, \ldots, K$ changes according to a Markov chain with state space S_k and transition probability $p_k(j|s)$. Assume the following timing of events. After selecting arm k the decision maker receives a reward $r_k(s)$ and an observation o with probability $u_k(o|s)$ if the hidden state of arm k equals s. Subsequently the arm changes state to state j with probability $p_k(j|s)$.

Note that to ensure the decision maker receives no information from the reward, it can be assumed that each $r_k(\cdot)$ has the same range for each k, that is $r_k(\cdot) = r(\cdot)$ for $k = 1, \ldots, K$. Note that this model may be viewed as choosing between K independent unobservable Markov reward processes.

Decision epochs: The model may be either finite or infinite horizon

$$T = \{1, 2, \ldots, N\}, \quad N \leq \infty.$$

Hidden states:

$$S = S_1 \times S_2 \ldots \times S_K = \{(s_1, \ldots, s_K) \mid s_k \in S_k, k = 1, \ldots, K\}.$$

Actions: Actions represent which arm to play

$$A = \{1, \ldots, K\}.$$

Observations: When the decision maker chooses action $k \in A$ and Markov chain k is in state s_k the decision maker observes $o \in O$ with probability $u_k(o|s)$. Therefore

$$u(o|(s_1, \ldots, s_K), k) = u_k(o|s_k).$$

This means that the observation depends on the state of arm k.

Rewards: When the decision maker chooses action $k \in A$ and Markov chain k is in state s, the decision maker receives reward

$$r((s_1, \ldots, s_K), k) = r_k(s_k).$$

Transition probabilities: For $k \in A$,

$$p((s'_1, \ldots, s'_K)|(s_1, \ldots, s_K), k) = \begin{cases} p_k(s'_k|s_k) & \text{for } s'_i = s_i, i \neq k \\ 0 & \text{otherwise} \end{cases}$$

8.6.4 Breast cancer screening

The American Cancer Society [2024] estimates that approximately one in eight women will be diagnosed with breast cancer in their lifetime. Using mammography to screen for breast cancer has been shown to reduce breast cancer mortality through early detection. Fundamental issues are at what age to start screening and how frequently to screen. Since women have different risk factors for breast cancer, screening policies may differ between individuals of the same age. The particular model developed below is an example of one in which some of the transition probabilities depend on the observation.

Consider the situation where a woman's true health state, that is, whether or not she has breast cancer and possibly what type, is unknown. Mammography provides a noisy preliminary indication of breast cancer. The model recommends periodically whether a woman should or should not have a mammogram. If the "no mammogram" option is chosen, the patient is encouraged to use self-examination as a rough indicator of the presence of cancer.

Action choice involves trade-offs. The downsides of too frequent mammograms are that they are painful, expose a patient to radiation, and false positives may heighten the patient's anxiety and tax health system resources. On the other hand, not having them frequently enough may miss the presence of disease. The goal is to develop a personalized screening policy so that diagnosis and subsequent timely treatment occurs as early as possible. The model is limited to screening. It assumes that after a positive mammogram the patient undergoes a biopsy, which is a more invasive and a highly reliable test. A positive biopsy results in the start of treatment; a negative biopsy returns the patient to their screening regimen. The model can be generalized to include follow-up decisions and treatment policies.

Consider the following sequence of events.

1. Immediately prior to the current decision epoch, the decision maker updates the belief state.

2. The decision maker chooses either mammography or self-examination in the current period.

3. The result of the mammogram, if performed, is found to be positive or negative.

4. A positive mammogram results in a biopsy. If the biopsy is also positive, treatment starts prior to the next period. A negative biopsy indicates the patient does not have breast cancer.

5. If the decision maker chose self-examination, the result is revealed prior to the next decision epoch.

6. The health state (cancer free, cancer present, or death) is updated.

Decision epochs: Decisions are made relative to a woman's age. Assume screening starts at age T_0 and decisions are made periodically up to age T_F[22]. Hence

$$T = \{T_0, T_1, \ldots, T_F\}.$$

Hidden states: States represent a woman's health condition at each decision epoch. The set of states is given by

$$S = \{CF, BC, TR, DE\}$$

where CF represents cancer free, BC represents breast cancer present, TR represents a patient under treatment and DE represents death. Assuming treatment begins once cancer is diagnosed, states TR and DE are *observable* and absorbing, while CF and BC are *hidden*. Thus this model combines both observable and unobservable states. Cancer states can be further distinguished by the stage of breast cancer, or whether the cancer is localized versus invasive, if needed.

Actions: In hidden states CF and BC, the decision maker chooses from MA (mammography) and SE (self-examination). In observable states TR and DE, the decision maker may only choose an artificial action ϕ, which leaves the process in the same state. Therefore:

$$A_s = \begin{cases} \{SE, MA\} & \text{for } s \in \{CF, BC\} \\ \{\phi\} & \text{for } s \in \{TR, DE\} \end{cases}$$

Observations: Set $O = \{+, -, \Delta\}$ where $+$ denotes a positive mammogram or self-exam, $-$ denotes a negative test and Δ denotes the screening process has terminated.

[22] Epochs could be every six months, or more frequently for self-examination even if the mammogram option is not available. The final epoch T_F should be chosen to be sufficiently high, e.g., 100. The reality that a woman may die before age 100 is accounted for by adding an absorbing "death" state to the hidden state space.

The probabilities of these observations are given for $t = T_0, T_1, \ldots, T_F$ by

$$u_t(o|s,a) = \begin{cases} \alpha_{t,SE} & \text{for } o = -, s = CF, a = SE \\ 1 - \alpha_{t,SE} & \text{for } o = +, s = CF, a = SE \\ 1 - \beta_{t,SE} & \text{for } o = -, s = BC, a = SE \\ \beta_{t,SE} & \text{for } o = +, s = BC, a = SE \\ \alpha_{t,MA} & \text{for } o = -, s = CF, a = MA \\ 1 - \alpha_{t,MA} & \text{for } o = +, s = CF, a = MA \\ \beta_{t,MA} & \text{for } o = +, s = BC, a = MA \\ 1 - \beta_{t,MA} & \text{for } o = -, s = BC, a = MA \\ 1 & \text{for } o = \Delta, s \in \{TR, DE\}, a = \phi \\ 0 & \text{otherwise.} \end{cases}$$

Observation probabilities are based on the medical testing concepts of *specificity* and *sensitivity*. In this case, the specificity of a test is the probability of a negative test $(\alpha_{t,SE}, \alpha_{t,MA})$ in a cancer-free woman while the sensitivity is the probability of a positive test $(\beta_{t,SE}, \beta_{t,MA})$ in a woman with cancer. These probabilities are obtainable from the medical literature. Moreover, sensitivity and specificity have been shown to be age related so that the observation probabilities vary with decision epoch. In general, for mammography to be a useful screening test, its sensitivity and specificity must exceed that of self-examination.

Rewards: The units for rewards are *Quality Adjusted Life Years* (QALYs). Note that the reward depends on the observation. Then

$$r_t(s,a,o) = \begin{cases} L_t & \text{for } s \in \{CF, BC\}, a = SE, o \in \{+, -\} \\ L_t - k & \text{for } s \in \{CF, BC\}, a = MA, o = - \\ L_t - K & \text{for } s = CF, a = MA, o = + \\ q_t & \text{for } s = BC, a = MA, o = + \\ 0 & \text{for } s = \{TR, DE\}, a = \phi, 0 = \Delta \end{cases}$$

with a terminal reward $r_{T_F}(s) = q_{T_F}(s)$ for $s \in S$. In the above, L_t denotes the expected QALYs in the period following decision epoch t, k the disutility of having a mammogram, K the disutility of a false positive mammogram, and q_t the estimated QALYs for an age t woman beginning cancer treatment (which implicitly includes the disutility of a positive mammogram).

Transition probabilities: For $t = T_0, T_1, \ldots, T_F$,

$$p_t(s'|s,a,o) = \begin{cases} (1-\lambda_t)\delta_t & \text{for } s' = BC, s = CF, a = SE, o \in \{+,-\} \\ (1-\lambda_t)(1-\delta_t) & \text{for } s' = CF, s = CF, a = SE, o \in \{+,-\} \\ 1 & \text{for } s' = TR, s = BC, a = MA, o = + \\ (1-\lambda_t) & \text{for } s' = BC, s = BC, a = MA, o = - \\ (1-\lambda_t)\delta_t & \text{for } s' = BC, s = CF, a = MA, o = + \\ (1-\lambda_t)\delta_t & \text{for } s' = CF, s = CF, a = SE, o = + \\ \lambda_t & \text{for } s' = DE, s \in \{CF, BC\}, a = SE, o \in \{+,-\} \\ 1 & \text{for } s' = s, s = \{TR, DE\}, a = \phi, o = \Delta \\ 0 & \text{otherwise} \end{cases}$$

where λ_t denotes the probability a woman dies in the period following decision epoch t and δ_t denotes the probability an age t cancer-free woman develops cancer in the period following decision epoch t. A transition from BC to TR can only occur if a woman with cancer has a positive mammogram.

To apply this model requires obtaining patient-specific data from the medical literature, modifying belief state updating and optimality equations to account for rewards and transition probabilities that depend on the observation, and developing a computational algorithm. A policy will partition the first two components of the belief state (probability of being cancer free and probability of breast cancer) into regions where the action choice is mammography versus self examination. Presumably, the model will recommend a mammogram when the risk of cancer is high and self-examination when the probability is low.

8.6.5 Controlling autonomous robots with noisy sensors

An autonomous controlled robot (ACR) possesses the following components:

1. **Sensors** to detect features of the robot's surroundings. Sensors may use light (cameras or lasers) or sound (sonar) to acquire information about the local environment.

2. **Controllers** to process the sensor information and determine appropriate actions.

3. **Actuators** to execute the action specified by the controller.

In addition, the robot may incorporate a low-level object avoidance system that works independently of the controller.

Robotic navigation competitions have demonstrated that ACRs are prone to *localization errors*. Robots often lose track of their position, hindering their ability to reach

their targets. This observation underscores the need for intelligent control algorithms that avoid localization errors and enable the robot to reach its target.

POMDP models provide a framework for robotic control. The following stylized example is intended to illustrate some of the modeling issues. In this model, the ACR strives to complete a task of moving through a series of hallways or a maze to reach a target location. Assume the robot has a stored map of its environment and knows the location of the target. Randomness enters through imprecise movements and noisy sensor readings.

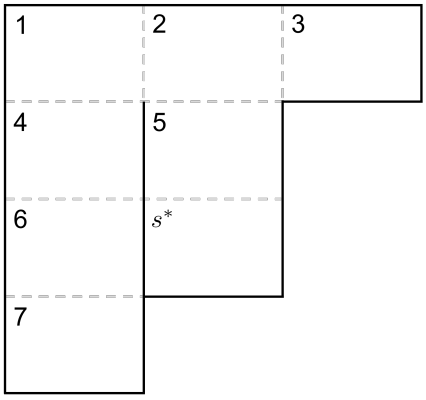


Figure 8.10: Sample grid for robot navigation POMDP model. Thick lines indicate walls.

The following combines features of many models in the literature. The environment (see Figure 8.10 for an example) is divided into discrete cells, which may be enclosed by up to three walls. This might be regarded as movement through the corridors of an office environment or through a maze. Given a map of the area to be navigated, the ACR can either move or activate its sensors. An onboard compass orients the robot. Since the robot does not know its precise location it uses sensor readings to update its belief distribution.

Decision epochs: Decision epochs correspond to the instance right before action choice. The number of periods to achieve the target is not known a priori. Good policies should achieve the target in finitely many periods, but cycling is possible so an infinite horizon is required to formulate the model:

$$T = \{1, 2, \ldots\}.$$

Hidden states: In an environment with K states plus a target state s^*,

$$S = \{1, \ldots, K, s^*\},$$

where the state denotes the robot's actual location.

Actions: Actions model the robot's movement and sensing capabilities. The action set is given by:

$$A = \{N, S, E, W, SNS, SEW\}$$

where N, S, E, W represent an attempt to move one cell in the north, south, east or west directions. It is important to emphasize that these actions represent the robot's *intended* movement. Transition probabilities account for the impact of the environment on the robots intended move.

Actions SNS and SEW refer to the robot performing a sensor sweep to detect walls immediately adjacent to its position in the north-south or east-west directions, respectively. When executing these actions, the robot does not move.

These actions represent both active and passive information acquisition.

Observations: When actions N, S, E and W are chosen, the robot gains no additional information about its state. Let ϕ denote this "none-observation" corresponding to choosing these actions. Also assume the robot knows when it reaches the target state, denoting this information by γ.

When either sensing action is chosen, the robot's sensors provide noisy information about the number of walls (0, 1 or 2) in the directions searched. Thus

$$O = \{0, 1, 2, \phi, \gamma\}.$$

Denote by $\omega(o|i)$ the probability the robot interprets the sensor information as there being o walls when in fact there are i walls. For example, when there are two walls in the directions searched, $\omega(o|2)$ may be given by

$$\omega(o|2) = \begin{cases} 0.05 & o = 0, \\ 0.15 & o = 1, \\ 0.8 & o = 2. \end{cases}$$

For sensors to be effective, one would expect that $\omega(i|i)$ for $i = 0, 1, 2$ are the largest probabilities. Thus,

$$u(o|s, a) = \begin{cases} 1 & s \neq s^*, a \in \{N, S, E, W\} \text{ and } o = \phi \\ 1 & s = s^* \text{ and } o = \gamma \\ \omega(o|W_s) & s \neq s^*, a \in \{SNS, SEW\}, W_s \in \{0, 1, 2\}, o \in \{0, 1, 2\}, \end{cases}$$

where W_s denotes the (unknown) number of walls in the specified directions in cell s.

Rewards: The following reward functions captures most of the intended features of the model:

$$r(s, a, j) = \begin{cases} R & s \neq s^*, j = s^* \\ -c & s \neq s^*, j \neq s^* \\ 0 & s = s^*, j = s^*, \end{cases}$$

where R denotes the reward for reaching the target state and c denotes the cost per action taken. Alternatively there could be different costs associated with sensing and movement actions. Assuming the process terminates when the target state is reached, the robot remains there and receives no subsequent reward.

Transition probabilities: Transition probabilities depend on the precise orientation and configuration of cells in the grid. When sensing, the robot does not move so that

$$p(j|s,a) = \begin{cases} 1 & j = s, a \in \{SNS, SEW\} \\ 0 & j \neq s, a \in \{SNS, SEW\} \end{cases}$$

Assume that the robot's movement is subject to randomness. When it tries to move in a direction it succeeds with probability 0.7 and moves in each other direction with probability 0.1. If it bumps into a wall it does not move. For the environment in Figure 8.10, some selected probabilities are given below:

$$p(j|6,N) = \begin{cases} 0.7 & j = 4 \\ 0.1 & j = 7 \\ 0.2 & j = 6 \end{cases}, \quad p(j|2,S) = \begin{cases} 0.7 & j = 5 \\ 0.1 & j \in \{1,2,3\} \end{cases}, \quad p(j|4,E) = \begin{cases} 0.8 & j = 4 \\ 0.1 & j \in \{1,6\} \end{cases}$$

For all $a \in A$

$$p(j|s^*,a) = \begin{cases} 1 & j = s^* \\ 0 & j \neq s^* \end{cases}$$

The derived MDP of this POMDP can be formulated with actions that are actions in the underlying MDP. That is, assume that every action is possible in every state, $\tilde{A} = A$ for all $s \in S$, since infeasible actions (asking the robot to walk through a wall) lead the robot to stay in the same cell. This logic is encoded in the transition probabilities as described above.

As an alternative, consider the situation where a robot is not able to detect an infeasible action before implementing it, and thus would attempt to walk through a wall if asked to do so, which is undesirable. In this case, the derived MDP needs to be formulated with actions that are decision rules in the underlying MDP, $\tilde{A} \subset \Pi_{s \in S} A_s$, with A_s including only feasible actions in hidden state s.

Belief updating

The following belief states updates using (8.14) distinguishes the effect of the two types of actions. Note that the results simplify because under a move action, the only signal is ϕ and under a sensor action, the system state is not altered.

Suppose a movement action is selected. Then applying (8.19) gives the updated belief state

$$b'(s') = P[X' = s'|Y = a, Z = \phi, \mathbf{b}] = \sum_{s \in S} p(s'|s,a) b(s),$$

where the expression simplifies because $u(\phi|s', a) = 1$ for $s' \neq s^*$ and $a \in \{N, S, E, W\}$.

For example, under the action N, cell 1 can only be reached from cells 1, 2 and 4 so that if $b(s) = 1/7$ for $s \neq s^*$,

$$b'(1) = p(1|1, N)b(1) + p(1|2, N)b(2) + p(1|4, N)b(4) = \frac{0.8 + 0.1 + 0.7}{7} = 0.229$$

Suppose a sensor action a is chosen and *oo* walls are observed, then

$$b'(s') = \frac{u(o|s', a)b(s')}{\sum_{s \in S} u(o|s, a)b(s)}$$

where the expression simplifies because choosing a sensor action does not result in robot movement, that is $p(j|s, a) = 1$ for $j = s$ and 0 otherwise.

For example suppose action SNS is chosen and 1 wall is observed and $b(s)$ is as above. Then assuming $\omega(1|0) = 0.1, \omega(1|1) = 0.8$ and $\omega(1|2) = 0.15$, and noting that $b(s)$ is constant,

$$\begin{aligned} b'(1) =& \frac{u(1|1, SNS)b(1)}{\sum_{s=1}^{7} u(1|s, SNS)b(s)} = \frac{u(1|1, SNS)}{\sum_{s=1}^{7} u(1|s, SNS)} \\ =& \frac{0.8}{0.8 + 0.8 + 0.15 + 0.1 + 0.1 + 0.1 + 0.8} = 0.280. \end{aligned}$$

Note that the probability of receiving this signal is 0.407. Thus in this sample calculation, sensing in the north-south direction increases the probability of being in cell 1 more than trying to move north.

Note that more complex (and perhaps realistic) models combine sensor readings and movement in single actions.

Bibliographic remarks

Our development in this chapter is in the spirit of Krishnamurthy [2016] and draws heavily on the classical operations research literature. The finite horizon and infinite horizon models were presented and analyzed in Smallwood and Sondik [1973] and Sondik [1978]. The survey by Monahan [1982] includes the algorithm that we present in this chapter for solving the finite horizon model. The linear support model by Cheng [1988] builds up the set Γ instead of pruning in Monahan's algorithm and is the basis for several subsequent algorithms. The approximation method presented in this chapter is due to Lovejoy [1991a]. The subsequent survey by Lovejoy [1991b] provides a clear exposition of this and several algorithms for solving POMDPs. Subsequent advances in approximate algorithms include the witness algorithm of Cassandra et al. [1997] and the point-based value iteration method described in Pineau et al. [2003] and Spaan and Vlassis [2005].

The preventive maintenance and inspection example is adapted from Monahan [1982]. The searching for an object example comes from Shechter et al. [2015] and was

motivated by a challenge in minimally invasive surgery. Kohavi and Thomke [2017] describes the impact that A/B testing has had on major technology companies. An optimal policy for the multi-armed bandit problem is characterized by the well-known Gittins Index [Gittins and Jones, 1974], which allows the problem to be solved by analyzing each arm separately. The Markov chain bandit model follows Krishnamurthy [2016]. The restless bandit model was proposed by Whittle [1988]. The model for breast cancer screening was developed by Ayer et al. [2012]. The robotic control example is based on the discussion in Simmons and Koenig [1995].

The website Cassandra [2025] contains an insightful verbal tutorial, links to some key references and code for some popular algorithms. We strongly encourage you to consult it.

Exercises

1. Consider the two-state model and calculations in Section 8.1.2.
   (a) Calculate the expected total reward when $q > 0.5$ with terminal reward 0 and when $x = 3$ and $y = 4$.
   (b) Let q^* denote the point at which the value of the decision rules (8.2) and (8.3) with terminal reward 0 are equal. Plot $v_d(q)$ as a function of q for $q \in [0, 1]$ using the decision rule in (8.2) for $q \leq q^*$ and the decision rule in (8.3) for $q > q^*$. What do you observe about the shape of $v_d(q)$?
   (c) Compute $P[X_2 = s_1|Z_2 = o_1, Y_1 = g(X_1), X_1]$ and $P[X_2 = s_1|Z_2 = o_2, Y_1 = f(X_1), X_1]$ using the approach in Section 8.1.2.
2. Suppose in the two-state model that the policy can choose between **any** of the four deterministic Markovian decision rules in the underlying Markov decision process.
   (a) Generate the analog of Figure 8.3 for several choices of x and y.
   (b) Specify the optimal policy as a function of q for each choice of x and y.
   (c) Compute the reward and transition probabilities for the two remaining actions in $\tilde{A}$ in a similar manner to (8.25) and (8.30).
   (d) Compute and interpret transition probabilities and rewards when $q_1 = 0.4$.
3. **The tiger problem** [Cassandra et al., 1997]. You are faced with the opportunity of opening one of two doors, one to the left and one to the right. Behind one door is a tiger and behind the other door is a valuable prize. Unfortunately, you do not know which object is behind which door. At each instance, you can open the left door, open the right door or listen. If you decide to listen, you will hear the tiger behind the correct door with probability 0.85 and behind the incorrect door with probability 0.15.

Assume listening costs 1 unit, opening the door with the tiger results in a penalty of 100 units and opening the door with the prize results in a reward of 10 units. The objective is to determine how long to listen before deciding to open a door. Clearly this depends on your belief regarding which door the tiger is behind.

(a) Formulate this as a POMDP and provide the equivalent MDP.

(b) Solve it for $N = 5, 10$ with Algorithm 8.2. Plot the optimal value function and represent the optimal policy.

(c) Use Algorithms 8.3 and 8.4 to find bounds and the lower bound approximate policy for a range of belief states, some that are in H and some that are not, for $N = 5, 10, 20$. Compare the quality of the approximations and the lower bound approximate policy to the optimal policy.

(d) Reformulate the model with three doors, two prizes and one tiger. How does this change analysis?

4. **Machine maintenance** [Smallwood and Sondik, 1973]. A machine has two identical components, both of which must operate on a product before it is completed. Refer to the time required to produce an item as a *production cycle*. The components fail, that is become inoperative, independently. The probability each component fails during a production cycle is 0.1. Assume for simplicity that if components fail, they do so at the start of the production cycle.

   The number of operative components affects the quality of the product as follows. If both components are operative, the product will be non-defective. If a single component is inoperative, the product will be defective with probability 0.5 and if two components are inoperative, the probability the product is defective is 0.75. Assume that a non-defective product generates 1 unit of revenue and a defective product generates no revenue.

   Assume that four control options are available at the start of each production cycle:

   - Continue as is: Do not inspect the product.
   - Inspect output: Inspect the finished product to determine if it is defective at a cost of 0.25 units. Assume this action can be completed in the current production cycle.
   - Condition-based maintenance: Inspect and repair components if necessary. The cost of replacing a failed component is 1 unit in addition to a cost of 0.5 units for inspection. This requires an entire production cycle so a product is not produced.
   - Preventive maintenance: Replace both internal components without inspection. This costs 2 units but foregoes the inspection cost. This also requires an entire production cycle so a product is not produced.

The objective is to maximize expected total revenue over a finite horizon.

(a) Draw a timeline of events within a period.

(b) Formulate this model as a POMDP.

(c) Express the model entities in matrix form.

(d) Solve the problem using Algorithm 8.2 for $N = 3, 4, 7, 11$. (These are the problem length in the paper.) Assume a terminal cost of 0 if both components are operative, 1 if a single component is operative, and 2 if both components are inoperative at termination.

(e) Represent the optimal policy by partitioning $\tilde{S}$ into regions where each action is optimal.

(f) Use Algorithms 8.3 and 8.4 to find bounds and the lower bound approximate policy for a range of belief states some that are in H and some that are not, for $N = 3, 4, 7, 11$.

5. Consider the calculations in the examples in Sections 8.4.4 and 8.4.5.

(a) Apply Algorithm 8.2 to find optimal policies and optimal value functions in models with $N = 4$ and $N = 5$.

(b) Apply Algorithms 8.3 and 8.4 to find the lower bound approximate policy and bounds for models with $N = 4$ and $N = 5$.

(c) Compare the policies and assess the quality of the bounds.

(d) Suppose the exact solution for $N = 5$ is regarded as an approximation to the infinite horizon version of the problem. Apply the bounds in (8.103) to assess the quality of the approximation.

6. Verify that the quantities in the matrix formulation of the derived MDP (8.21) -(8.24) are equivalent to the corresponding quantities in the component formulation of the derived MDP.

7. **Some formulation exercises:**

(a) Formulate a POMDP model of the classical statistical bandit model in which there is a fixed cost C associated with changing arms.

(b) Formulate the derived MDP for the newsvendor problem from Section 8.6.2 when $M = 3$.

(c) Formulate the derived MDP for the search problem from Section 8.6.2. This is challenging because of the presence of the stopped state.

(d) Reformulate the search problem when the searcher finds the object with certainty.

(e) Reformulate the search model when the object can change cells between decision epochs. Assume the object follows a random walk where p_L denotes the probability the object moves from cell i to cell $i-1$ and p_H denotes the probability the object moves from cell i to cell $i+1$.

(f) Formulate the two derived MDPs of the robot control problem described at the end of Section 8.6.5.

8. Prove that a function written in the form (8.45) is convex and establish the five properties of piecewise linear convex functions listed immediately below (8.45).

9. Derive all probabilities for the robot control model in Section 8.6.5.

10. Formulate the following model as a POMDP. Consider the case of a male patient who may have prostate cancer. In this setting, observation becomes "active surveillance", inspection corresponds to having a PSA test, and advance testing may correspond to a biopsy or imaging. This setting becomes more complex in that as a result of a positive biopsy, a patient may decide to receive hormone therapy, have surgery or continue active surveillance. Moreover more than two states may be necessary to represent different disease stages.

Part III - Reinforcement Learning

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk[1].

John von Neumann, Hungarian-American mathematician, physicist, computer scientist, game theorist and more, 1903-1957.

This part of the book contains three chapters:

- Chapter 9: Value function approximation
- Chapter 10: Simulation in tabular models
- Chapter 11: Simulation with function approximation

The material in Chapter 9 is often referred to as *approximate dynamic programming (ADP)* and that in Chapters 10 and 11 as *reinforcement learning*.

While von Neumann's quote was relevant for statistical models in the 1950s, things have changed. Many modern reinforcement learning methods, particularly those based on neural networks, use models with millions of parameters and still generalize well to new settings. This progress comes from advances in neural network techniques, access to massive datasets, and powerful optimization algorithms. As a result, the link between model complexity and performance is far more subtle than in von Neumann's time.

Reinforcement learning can be viewed from two perspectives:

- The *artificial intelligence (AI)* perspective is that reinforcement learning methods seek to develop an "agent which can interpret any environment, and learn a task to superhuman ability, all with minimal user interaction[2]."

[1] Dyson [2004].

[2] McKenzie and McDonnell [2022]

- The *operations research (OR)* perspective is that in a model-based environment in which rewards and transition probabilities are known, reinforcement learning provides a toolbox of methods that can be used to find effective policies.

Although these appear to be distinct, the boundaries are far from clear cut in practice. Table 8.2 sheds light on how these perspectives have evolved in practice.

Aspect	OR perspective	AI perspective
Horizon	Infinite or finite and fixed	Finite and random
Model	Specified	Not specified
Rewards	At each decision epoch	At termination
Possible actions	Pre-specified	Pre-specified
Transition probabilities	Known	Not known
Algorithms	Offline	Online

Table 8.2: Distinguishing features of the OR and AI perspectives on reinforcement learning. These are not intended to be exhaustive or mutually exclusive, the boundary between them is often blurred.

Overview

Chapters 4 - 8 provided computational algorithms (e.g., value iteration, policy iteration, linear programming) for finding optimal policies in a fully or partially observable Markov decision process that can be stored on a computer. Since these methods are computationally prohibitive in models with extremely large (or continuous) state spaces, this section of the book and most current research develops and investigates methods to do so.

Motivated by the need to reduce the dimension of the state space, Chapter 9 provides generalizations of value iteration, policy iteration and linear programming for analyzing Markov decision processes in which the underlying value function is approximated by a low-dimensional linear or nonlinear function of features of the state space or state-action space. Such algorithms are applied directly to the Markov decision process model. No simulation is involved.

The penultimate chapter, Chapter 10, represents a major departure from previous chapters. In it, simulation (or real-time experimentation) replaces exact computation. It focuses on models that can be represented in *tabular* form. That is, models that are sufficiently small so that value functions and state-action value functions can be easily stored in memory. In such models, approximating value and state-action value functions is unnecessary. The reason for focusing on tabular models is to compare the performance of simulation to classical methods. Approaches include Monte Carlo methods, temporal differencing and Q-learning. This chapter also distinguishes *model-based* analyses, in which components of the underlying Markov decision process are used in algorithms, and *model-free* analyses, which are based only on the generated or

observed sequence of states, actions and rewards. The underlying mathematical tools are stochastic approximation and stochastic gradient descent.

The concluding chapter, Chapter 11, combines the concepts from the previous two chapters, namely value function or state-action value function approximation and simulation and introduces policy-based approaches as well. It is motivated by applications in which the state space (and/or action space) is so large (or continuous) that tabular representations and exact computation are impractical. The two distinct approaches are:

1. **Value-based methods** use the reward to update the stored state-action value function or an approximation to it. When rewards corresponding to an action are greater than expected, the state-action value function for the specified state-action pair is increased, making the chosen action in that state more attractive. Conversely, when the reward is less than expected, the state-action value function is decreased making the action less attractive. Thus, the rewards lead these algorithms to learn a good approximations to the "true" underlying state-action value function from which actions can be selected greedily in subsequent implementations.

2. **Policy-based methods** use the reward to directly update a randomized policy by gradient ascent. When an action chosen in a state produces a larger than expected reward, the probability of choosing that action in the state is increased. Conversely if the reward is less than expected, the probability of choosing that action is decreased. Repeating this process produces effective policies.

Objectives

This part of the book offers an accessible introduction to reinforcement learning concepts and algorithms, rather than a comprehensive (and soon outdated) overview. By mastering these fundamental methods, readers will establish a solid base from which to delve more deeply into rapidly evolving reinforcement learning research and applications.

So, how should one learn this material? Following the reinforcement learning philosophy, you must learn by doing. This means:

1. Carefully formulating your applications as Markov decision processes.

2. Developing your own code for the algorithms[3] in these chapters.

Often your codes will not replicate published results. Unlike the methods in Chapters 4 – 8, reinforcement learning methods offer few practical guarantees on convergence and have many parameters to tune. Trial-and-error is a natural part of the process, and a necessity for developing the insights needed to create robust and effective algorithms.

[3]This is not recommended for linear programming and regression, where excellent commercial codes are available.

Figure III.1: Image a Brio Labyrinth owned by one of the authors of this book.

An example: The Brio labyrinth

To provide a flavor of the type of problem that the reinforcement learning community seeks to solve, consider the challenge of optimally guiding a ball through the Brio Labyrinth shown in Figure III.1. Bi and D'Andrea [2023] use reinforcement learning to find and implement a good strategy for playing this game. Details and methods are beyond the scope of this book but hopefully will be understandable after reading these chapters.

In this game, a player places a metal ball at a location designated as the start at the top center of the maze. Using the two knobs on the sides of the labyrinth, the player dynamically tilts the plane of the surface so as to guide the ball to a destination at the right center without falling through any of the 39 numbered holes[4].

To "solve" this problem using reinforcement learning requires complex engineering, considerable ingenuity and the use of state-of-the-art algorithms. The researchers developed a robotic system that used computer-controlled motors attached to the rotors to adjust the tilt of the surface. A camera mounted above the surface provided real-time information about the ball position, surface angles and the geometry around the ball.

An MDP model

The system was modeled by a discrete-time undiscounted infinite horizon (episodic) MDP. States represent information about the ball, the surface, and the local geometry (position of walls and holes) around the ball. The ball position is given by its (x, y)-

[4]Holes may be numbered twice when they can interact with the ball at two different segments of its path.

coordinates. The surface is summarized in terms of the angle of tilt of the surface in the x and y directions. Rates of change of these quantities are estimated using sequences of images. The local geometry around the ball is captured by 6 cm $\times$ 6 cm images centered at the ball, taken at discrete time steps and encoded in a 64×64 RGB image. Actions represent the angular velocity of the two motors attached to the knobs controlling the surface tilt.

Camera images also provided a basis for a reward structure that measures progress (in cms) along the directional path represented by the black line from the origin to the destination in Figure III.1. Note in episodic models such as this, it is customary to only receive a reward when achieving a goal or incur a penalty when terminating in an undesirable state. Such a reward structure presents challenges for reinforcement learning methods so the introduction of these intermediate rewards facilitates learning.

Algorithms and results

Good policies are found using a model-based actor-critic algorithm (see Section 11.7.1) called DreamerV3 [Hafner et al., 2025]. Transition probabilities (obtained from "a world model"), policies, and values are approximated by neural networks. Underlying symmetries in the board structure improve algorithmic efficiency.

The system learned to "solve" the game in 5 hours (1 million time steps) using real-world experiences. Running the resulting policy over 50 replicates resulted in a success rate (reaching the target location) of 76% and a completion time of 15.73 ± 0.36 seconds. To put this in perspective, the human record is 15.95 seconds.