Function Approximation in Partially Observable Markov Decision Processes

Rohan Shiloh Shah

April 30, 2003

Contents

1 Introduction **2**

2 Framework for POMDP's **2**

2.1 Complete Information States 2

2.2 Belief States 3

3 Policy Trees **5**

4 Function Approximation **6**

4.1 Assuming Full Observability 6

4.2 Fast Informed Bound Method 7

4.3 Assuming No Observability 8

4.4 Comparison of Results 8

4.5 Grid Based Approach with value Interpolation-Extrapolation . 9

4.5.1 Convex Rules 10

4.6 Least-Squares Fit 10

1 Introduction

The assumption that the state space is not perfectly observable is what differentiates a POMDP from a MDP. This implies that we cannot decide with certainty the state of a system at any given time. As a result of this difference we define a new action, in addition to the control actions defined in MDP's, the purpose of which is to gather more information (or observations) in order to make the state space more resolute. So at each stage of the planning problem the agent is faced with the usual task of choosing the next *optimal* action: an agent in a POMDP environment bases this decision on (i) the new information that an action may provide, (ii) the expected reward of the action, and (iii) the changes to the system that are foreseen as a result of the execution of the action. As an example consider an agent whose task is to find the Twin Towers in New York starting from outside Manhattan: if the agent were to choose its action based solely on maximised expected reward it would never find the Twin Towers since they no longer exist. If it were to ask someone however it could terminate the task without wasting more time but runs the risk of being shot at by whoever it asks, thereby reducing the expected reward. So there is a balance that the agent must find between each of (i), (ii) and (iii).

So we see there are two sources of stochasticity in a POMDP: (i) we define $T : S \times S \times A \to [0,1]$: transition probabilities between a state, an action and the state that results from executing that action, and (ii) the imperfect observability of the state space.

2 Framework for POMDP's

We define a tuple $(S, A, \Theta, T, O, R, Q)$ where S is the set of states, A thet set of actions, Θ is a set of observations, O is a set of observation probabilities, T is a set of transition probabilities, R is a reward function, and Q is a prior distribution over the intial states S. Our objective is to maximise the discounted expected reward.

2.1 *Complete Information States*

Consider the task of navigating through a maze: we cannot make a decision based on our current observations since our current view of the maze is not neccesarily unique, but instead we must look at a history of all actions and observations we have made since we started navigating the maze. So we define a *complete information state* I_t as consisting of (i) $Q(S_0)$ where S_0 is the start state, and (ii) a set $\{o_0, a_0, o_1.a_1, \cdots, o_{t-1}, a_{t-1}, o_t\}$. So given a new observation o_{t+1} and the last action a_t it is trivial to form information state I_{t+1} from I_t. We can now abstract away from states completely and consider only information states and rewrite the Bellman Equation as:

$$V^*(I) = \max_{a \in A} \left[\overbrace{\sum_{s \in S} \rho(s,a) P(s|I)}^{\text{expected 1-step reward of executing a in I}} + \overbrace{\gamma \sum_{I'} P(I'|I,a) V^*(I')}^{\text{discounted expected value of } I'} \right]$$
$$\text{where } \rho(s,a) = \sum_{s'} T(s,a,s') \times R(s,a,s') \tag{1}$$

Since we know that consequent information states I_{t+1} are functions of the previous information state I_t and the new observation o_{t+1} as well as the last action a_t (i.e. $I_{t+1} = \tau(I_t, a_t, o_{t+1})$) we can rewrite the Bellman equation as:

$$V^*(I) = \max_{a \in A} \left[\sum_{s \in S} \rho(s,a) P(s|I) + \gamma \sum_{o \in \Theta} P(o|I,a) V^*\{\tau(I,a,o)\} \right] \tag{2}$$

Using this value function an appropriate optimal control policy can be extracted.

2.2 *Belief States*

Equations (1) and (2) are not useful computationally since keeping complete information states implies the usage of an infinite amount of memory in the infinite-horizon case. Since the environment is assumed to be Markov, we now look at a *sufficient statistic* that summarizes the full information content and is therefore more compact: we define a *belief state* θ to be a discrete probability distribution over the states conditioned on past actions and observations. We can compute belief states incrementally in the following way:

$$\begin{aligned}\theta_0(s') &= Q(s')\\ \theta_{t+1}(s') &= P(s'|o_{t+1}, a_t, \cdots, a_0, o_0)\\ &= \alpha \cdot P(o_{t+1}|s', a_t, o_t \cdots, a_o, o_0) \cdot P(s'|a_t, o_t, \cdots, a_0, o_0)\\ &= \alpha \cdot P(o_{t+1}|s', a_t) \sum_{s \in S} \left[P(s'|a_t, o_t, \cdots, a_0, o_0, s) \cdot P(s|a_t, o_t, \cdots, a_o, o_0)\right]\\ &= \alpha \cdot P(o_{t+1}|s', a_t) \sum_{s \in S} \left[\underbrace{P(s'|a_t, s)}_{\text{transition probability}} \times \overbrace{\theta_t(s)}^{\text{belief state at time t}} \right]\end{aligned} \tag{3}$$

So given the sufficient statistic θ_t, computing a new *belief state* distribution over the states θ_{t+1} (using the new observation o_{t+1} and the last action executed a_t) is possible and can be done independantly of the *complete information history.* Furthermore, the distributions derived using either *belief states* or *complete information states* are the same:

$$\theta_t = P(s_t|I_t) \tag{4}$$

We now extend the Bellman Equations to belief states in order to derive a policy mapping belief states to actions:

$$\begin{aligned}V(b) &= \max_{a \in A} \left[\rho(b, a) + \gamma \sum_{b'} P(b'|b, a) \cdot V(b')\right]\\ &= \max_{a \in A} \left[\sum_{s \in S} b(s) \cdot \rho(s, a) + \gamma \sum_{o \in \Theta} P(b'|a, b, o) \cdot P(o|a, b) \cdot V(\tau(b, o, a))\right]\\ &= \max_{a \in A} \left[\sum_{s \in S} b(s) \cdot \rho(s, a) + \gamma \sum_{o \in \Theta} I_{b'=\tau(b,o,a)} \left(\sum_{s \in S} P(o|s, a) \cdot b(s)\right) V(\tau(b, o, a))\right]\end{aligned} \tag{5}$$

where $I_{b'=\tau(b,o,a)}$ is an indicator function that is true when the incrementally computed new *belief state* distribution is equal to b'. In the policy improvement stage, we need to extract a policy $\mu : \mathcal{I} \rightarrow A$ where $\mathcal{I}$ is the space of all belief states using the value function we have just computed V:

$$\begin{aligned}\mu(b) &= arg\max_{a\in A}\left[\sum_{s\in S}\rho(s,a)b(s) + \overbrace{\gamma\sum_{o\in\Theta}P(o|b,a)V(\tau(b,o,a))}^{\text{discounted 1-step lookahead for all } b'}\right]\\ &= arg\max_{a\in A}\left[\sum_{s\in S}\rho(s,a)b(s) + \gamma\sum_{o\in\Theta}\left(\sum_{s\in S}P(o|s,a)\cdot b(s)\right)V(\tau(b,o,a))\right]\end{aligned} \tag{6}$$

Our aim is to compute $\mu^* : \mathcal{I} \to A$, the optimal action set, but since it is defined over the space of all belief distributions (a very high dimensional space) it is almost impossible to compute. So we resort to computing an approximation to $V^* : \mathcal{I} \to \Re$ which in turn will let us extract an approximately optimal action set.

3 Policy Trees

A *policy tree* is a t-step set of actions and the observations that can result from the execution of those action.

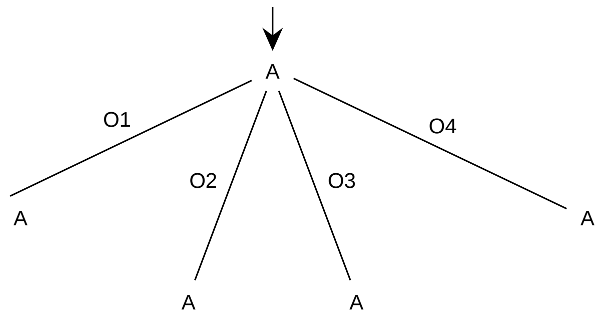

Figure 1: A 1-step Policy Tree where the observations are marked on the edges.

For the 1-step policy tree; the value of any state is simply the reward achieved by executing the action specified at the root of the policy tree. For a t-step policy tree p we have the following:

$$V_p(s) = \rho(s,a_p) + \gamma\sum_{s'\in S}P(s'|s,a_p)\sum_{o\in\Theta}P(o|s',a_p)V_{p(o)}(s') \tag{7}$$

where a_p is the action specified at the root of policy tree p and $V_{p(o)}(s')$ is the value of s' using the policy tree that results from deleting the root node in p and making observation o. Given that the state space is not perfectly observable, we must define the value function in terms of *belief states* like we did previously. In this case, $V_p(b)$ is simply the expected value of executing p over all states. If we define α_p to be a vector of the values over all states (generated using policy tree p and Equation 7):

$$\begin{aligned} V_p(b) &= \alpha_p \cdot b \\ V_t(b) &= \max_{p \in \mathcal{P}} b \cdot \alpha_p \end{aligned} \tag{8}$$

So we see that the that $V_p(b)$ is a linear function of the belief distribution. The application of the *max* operator indicates that $V_t(b)$ is the upper surface of all the linear surfaces generated by all possible policy trees: so $V_t(b)$ is piece-wise linear and convex. We could reformulate the update rule as:

$$\begin{aligned} V_p(b) &= \alpha_p \cdot b = \sum_{s \in S} b(s) \cdot V_p(s) \\ &= \sum_{s \in S} \rho(s,a) b(s) + \gamma \sum_{s' \in S} \left(\sum_{s \in S} P(s'|s,a_p) b(s) \right) \sum_{o \in \Theta} P(o|s',a_p) V_{p(o)}(s') \\ V_t(b) &= \max_{p \in \mathcal{P}} b \cdot \alpha_p \\ &= \max_{p \in \mathcal{P}} \left[\sum_{s \in S} \rho(s,a) b(s) + \gamma \sum_{s' \in S} \left(\sum_{s \in S} P(s'|s,a_p) b(s) \right) \sum_{o \in \Theta} P(o|s',a_p) V_{p(o)}(s') \right] \\ &= \max_{a \in A} \left[\sum_{s \in S} \rho(s,a) b(s) + \gamma \sum_{o \in \Theta} \max_{p \in P} \sum_{s' \in S} \left[\sum_{s \in S} P(s',o|s,a) \right] V_{p(o)}(s') \right] \end{aligned} \tag{9}$$

Although we have been able to provide a geometric interpretation for the value function and redefine it in terms of policy trees, the optimal value function is still defined over the space of all belief states. We must resort to approximate methods in order to get computable solutions.

4 Function Approximation

4.1 *Assuming Full Observability*

The easiet way to approximate the is to assume that the environment is fully obervable. Under this assumption we can easily solve for an optimal value function as follows:

$$V_{t+1}(s) = \max_{a \in A} \left[\rho(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_t(s') \right] \tag{10}$$

We know that an optimal solution V^* can be found in a finite amount of time. We can now use this optimal solution coupled with a *belief distribution* as follows:

$$\begin{aligned} V(b) &= \sum_{s \in S} b(s) V^*(s) \\ &= \sum_{s \in S} b(s) \max_{a \in A} \left[\rho(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_t^*(s') \right] \end{aligned} \tag{11}$$

In terms of policy trees this would give:

$$V(b) = \sum_{s \in S} b(s) \max_{a \in A} \left[\rho(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \max_{p \in \mathcal{P}} V_p(s') \right] \tag{12}$$

Notice that the difference between Equations 11 and 12 is that V^* is dropped to instead maximise V over all policy trees. The reason we rewrite the update rule in terms of policy trees is because we can make a comparison to the exact update rule derived in the section on policy trees (see section Comparison of Results).

4.2 *Fast Informed Bound Method*

Assuming partial observability, we can get a better approximation (tighter upper bound) to the exact POMDP solution than the *assuming full observability* approach using the following observation:

$$\underbrace{\sum_{o \in \Theta} \max_{p \in P} \sum_{s' \in S} \sum_{s \in S} P(s', o|s,a) V_{p(o)}(s')}_{\text{from Exact Policy Tree update rule}} \leq \underbrace{\sum_{o \in \Theta} \sum_{s \in S} \max_{p \in P} \sum_{s' \in S} P(s', o|s,a) V_{p(o)}(s')}_{\text{from Fast Informed Bound}} \tag{13}$$

The exact policy tree update rule performs a maximum over all policy trees after the summation over both s and s' is done. In contrast the Fast Informed Bound Method performs the same maximum after only the summation over all s' has been performed making it an upper bound approximation to the exact policy tree update rule. The Fast Informed Bound Method is as follows:

$$\begin{aligned} V(b) &= \max_{a\in A}\left[\sum_{s\in S}\rho(s,a)b(s)+\gamma\sum_{o\in\Theta}\sum_{s\in S}\max_{p\in P}\sum_{s'\in S}P(s',o|s,a)b(s)V_{p(o)}(s')\right] \\ &= \max_{a\in A}\sum_{s\in S}b(s)\left[\rho(s,a)+\gamma\sum_{o\in\Theta}\max_{p\in P}\sum_{s'\in S}P(s',o|s,a)V_{p(o)}(s')\right] \end{aligned} \tag{14}$$

Intuitively, this rearrangement of the exact policy tree update rule can be seen as selecting the best linear function for each observation and each current state s as compared to selecting the best linear function for each observation and every combination of both the current state and the next state.

4.3 *Assuming No Observability*

The other extreme of the fully observable assumption is to assume that there are no observations that can be made, which results in the following update rule:

$$V(b)=\max_{a\in A}\left[\sum_{s\in S}\rho(s,a)b(s)+\gamma\max_{p\in P}\sum_{s\in S}\sum_{s\in S'}P(s'|s,a)b(s)V_{p(o)}(s')\right] \tag{15}$$

The update rule is identical to the *exact update rule* except that the summation over all observations has been dropped. We are still maximising over all policy trees and so in a sense we are still considering observations; but only to direct our recurrence and never in a computation.

4.4 *Comparison of Results*

We can now compare the solutions that have been defined in terms of policy trees with the exact update rule. We already know that the Fast Informed Bound Method is an upper bound to the exact update rule, but how tight is this bound? We have that:

$$\begin{aligned}
V(s)_{exact} &= \max_{a\in A}\left[\sum_{s\in S}\rho(s,a)b(s) + \gamma\sum_{o\in\Theta}\max_{p\in P}\sum_{s'\in S}\left[\sum_{s\in S}P(s',o|s,a)\right]V_{p(o)}(s')\right] \\
&\leq \max_{a\in A}\sum_{s\in S}b(s)\left[\rho(s,a) + \gamma\sum_{o\in\Theta}\max_{p\in P}\sum_{s'\in S}P(s',o|s,a)V_{p(o)}(s')\right] \\
&\leq \sum_{s\in S}b(s)\max_{a\in A}\left[\rho(s,a) + \gamma\sum_{s'\in S}P(s'|s,a)\max_{p\in\mathcal{P}}V_p(s')\right]
\end{aligned} \tag{16}$$

We have already discussed why the first inequality holds (see section on Fast Informed Bound Method). The second inequality holds because the max and sum over all s operators have been switched: so in the gully observable method we are maximising many more times (i.e. for each $s \in S$) and them summing which has the effect that it upperbounds both the exact and the fast informed bounds methods. So we see that the Fast Informed Bound Method is a better approximation that the assuming full observability method since it has a tighter upper bound on the exact update rule.

4.5 *Grid Based Approach with value Interpolation-Extrapolation*

We can divide the belief space into a predefined set of grids and abstract out of the belief space and into the grid-belief space, learning a value function over the grids instead. Since the belief space may be very large, the grid based approach reduces the dimensionality of the problem considered. With this reduction however comes a loss in accuracy: the larger the grids, the more inacurate the resulting interpolated value function.

In order to generate the training set for the grid based approach, we select a set of grid points $G = \{b_1, b_2, \cdots, b_k\}$ and then use the exact update rule for a POMDP to generate the corresponding function values $\psi = \{V(b_1), V(b_2), \cdots, V(b_k)\}$. Any method that uses only this training data to compute an approximate value for any point in the belief space is called an *interpolation-extrapolation* rule. As new points in the belief space are assigned value approximations, new training data is generated which changes the value function and therefore the value of all old belief points, i.e. the approximation of $\overline{V}(b_{k+1})$ affects $\{V(b_1), V(b_2), \cdots, V(b_k)\}$ [1]. This is clearly

[1]Note that in some cases, applying this interpolcation-extrapolation rule every time new data arrives is unneccesary.

a disadvantage. However, the advantage of such an approach is that the expensive exact update rule is used to compute only a finite number of training data points, whereas solving the POMDP (until convergence) using this same exact update rule is infeasible since the belief space is infinite.

4.5.1 Convex Rules

There are a number of *rules* that an interpolation-extrapolation method might use in order to estimate or approximate the value function: if it satisfies the following properties:

$$\begin{aligned} \overline{V}(b) &= \sum_{j=1}^{k} \lambda_j^b V(B_j) \\ \sum_{j=1}^{k} \lambda_j^b &= 1 \\ 0 \leq \lambda_j^b &\leq 1, \quad \forall j \end{aligned} \tag{17}$$

then we say the rule is from the convex family.[2] Given a point in belief space b, we consider two examples: (i) the *Nearest-Neighbor* approach assigns a parameter setting $\lambda = 1$ for the nearest (nearnes defined as some distance metric in the belief space) to the point b. All other points are assigned $\lambda = 0$. (ii) *Kernel-Regression* considers more than a single point (some set of points that meets some criteria based on b) and assigns them each a normalised lambda score which is weighted by their distance to b.

4.6 *Least-Squares Fit*

We assume the value function has some predefined parametric distribution; our goal is to learn the appropriate parameters by minimising the least-squares error. This implies we are supplied with a training set of belief-value pairs. The grid based approach differs in that this training set must explicitly be stored at all times whereas we only store the parameters of the distribution in the least squares method.

Given the training set of points in the belief space, we compute their corresponding values using the exact value update. We now find appropriate parameters for a distribution that has a minimised square error. The distribution could be linear, but more appropriately a quadratic distribution is used. Alternatively, any other convex rules are also possible. Like the Grid

[2]Note that λ changes with each belief state.

approach only a finite number of value updates need to be calculated, however divergence has been shown to occur in some cases. If the data is only availible in an incremental fashion, then an online method might be most appropriate where as each data point becomes availible, its value update is first calculated and then the weights of the parametric distributed are updated using a gradient descent rule, i.e. use the new training example to take a step in the direction of the 'correct' parameters to the distribution.

References

[1] Value-Function Approximation for Partially Observable Markov Decision Processes, Milos Hauskrecht, 2000

[2] Planning and Acting in Partially Observable Stochastic Domains, Leslie Pack Kaelbling, Michael L. Littman, Anthony R. Cassandra, 1997