Lecture Note 12

1 Value Function Approximation and Policy Performance

Recall that two tasks must be accomplished in order to for a value function approximation scheme to be successful:

1. We must pick a "good" representation $\tilde{J}$, such that $J^*(\cdot) \approx \tilde{J}(\cdot, r)$ for at least some parameter r;
2. We must pick a "good" parameter $\tilde{r}$, such that $J^*(x) \approx \tilde{J}(x, \tilde{r})$.

Consider approximating J^* with a linear architecture, i.e., let

$$\tilde{J}(x, r) = \sum_{t=1}^{p} \phi_i(x) r_i,$$

for some functions $\phi_i, i = 1, \ldots, p$. We can define a matrix $\Phi \in \Re^{|\mathcal{S}| \times p}$ given by

$$\Phi = \begin{bmatrix} | & & | \\ \phi_1 & \ldots & \phi_p \\ | & & | \end{bmatrix}.$$

With this notation, we can represent each function $\tilde{J}(\cdot, r)$ as

$$\tilde{J} = \Phi r.$$

Fig. 1 gives a geometric interpretation of value function approximation. We may think of J^* as a vector in $\Re^{|\mathcal{S}|}$; by considering approximations of the form $\tilde{J} = \Phi r$, we restrict attention to the hyperplane $J = \Phi r$ in the same space. Given a norm $\|\cdot\|$ (e.g., the Euclidean norm), an ideal value function approximation algorithm would choose r minimizing $\|J^* - \Phi r\|$; in other words, it would find the projection Φr^* of J^* onto the hyperplane. Note that $\|J^* - \Phi r^*\|$ is a natural measure for the quality of the approximation architecture, since it is the best approximation error that can be attained by any algorithm given the choice of Φ.

Algorithms for value function approximation found in the literature do not compute the projection Φr^*, since this is an intractable problem. Building on the knowledge that J^* satisfies Bellman's equation, value function approximation typically involves adaptation of exact dynamic programming algorithms. For instance, drawing inspiration from value iteration, one might consider the following approximate value iteration algorithm:

$$\Phi r_{k+1} = \Pi T \Phi r_k,$$

where Π is a projection operator which maps $T\Phi r_k$ back onto the hyperplane Φr.

Faced with the impossibility of computing the "best approximation" Φr^*, a relevant question for any value function approximation algorithm A generating an approximation Φr_A is how large $\|J^* - \Phi r_A\|$ is in comparison with $\|J^* - \Phi r^*\|$. In particular, it would be desirable that, if the approximation architecture

is capable of producing a good approximation to J^*, then the approximation algorithm should be able to produce a relatively good approximation.

Another important question concerns the choice of a norm $\|\cdot\|$ used to measure approximation errors. Recall that, ultimately, we are not interested in finding an approximation to J^*, but rather in finding a good policy for the original decision problem. Therefore we would like to choose $\|\cdot\|$ to reflect the performance associated with approximations to J^*.

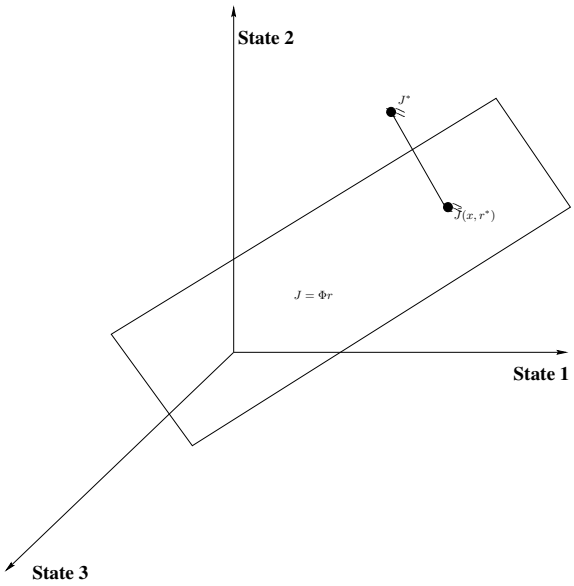

Figure 1: Value Function Approximation

2 Performance Bounds

We are interested in the following question. Let u_J be the greedy policy associated with an arbitrary function J, and J_{u_J} be the cost-to-go function associated with that policy. Can we relate the increase in cost $\|J_{u_J} - J^*\|$ to the approximation error $\|J^* - J\|$?

Recall the following theorem, from Lecture Note 3:

Theorem 1 *Let J be arbitrary, u_J be a greedy policy with respect to J.*[1] *Let J_{u_J} be the cost-to-go function for policy u_J. Then*

$$\|J_{u_J} - J^*\|_\infty \leq \frac{2}{1-\alpha}\|J - J^*\|_\infty.$$

The previous theorem suggests that choosing an approximation $\tilde{J}$ that minimizes $\|J^* - \tilde{J}\|_\infty$ may be an appropriate choice. Indeed, if $\|J^* - \tilde{J}\|_\infty$ is small, then we have a guarantee that the cost increase incurred by using the (possibly sub-optimal) policy u_J is also relatively small. However, what about the reverse? If $\|J^* - \tilde{J}\|_\infty$ is large, do we necessarily have a bad policy? Note that, for problems of practical scale, it

[1] That is, $T_{u_J}J = TJ$, and $u_J(x) = \arg\min_a \left\{ g_a(x) + \alpha \sum_y P_a(x,y)J(y) \right\}$.

is unrealistic to expect that we could approximate J^* uniformly well over all states (which is required by Theorem 1) or that we could find a policy u_J that yields a cost-to-go uniformly close to J^* over all states.

The following example illustrates the notion that having a large error $\|J^* - \tilde{J}\|_\infty$ does not necessarily lead to a bad policy. Moreover, minimizing $\|J^* - \tilde{J}\|_\infty$ may also lead to undesirable results.

Example 1 *Consider a single queue with controlled service rate. Let x denote the queue length, B denote the buffer size, and*

$$\begin{aligned} P_a(x, x+1) &= \lambda, \quad \forall\, a, x = 0, 1, \ldots, B-1 \\ P_a(x, x-1) &= \mu(a), \quad \forall\, a, x = 1, 2, \ldots, B, \\ P_a(B, B+1) &= 0, \quad \forall\, a \\ g_a(x) &= x + q(a) \end{aligned}$$

Suppose that we are interested in minimizing the average cost in this problem. Then we would like to find an approximation to the differential cost function h^*. Suppose that we consider only linear approximations: $\tilde{h}(x, r) = r_1 + r_2 x$. At the top of Figure 1, we represent h^* and two possible approximations, h_1 and h_2. h_1 is chosen so as to minimize $\|h^* - \tilde{h}\|$. Which one is a better approximation? Note that h_1 yields smaller approximation errors than h_2 over large states, but yields large approximation errors over the whole state space. In particular, as we increase the buffer size B, it should lead to worse and worse approximation errors in almost all states. h_2, on the other hand, has an interesting property, which we now describe. At the bottom of Figure 1, we represent the stationary state distribution $\pi(x)$ encountered under the optimal policy. Note that it decays exponentially with x, and large states are rarely visited. This suggests that, for practical purposes, h_2 may lead to a better policy, since it approximates h^* better than h_1 over the set of states that are visited almost all of the time.

What the previous example hints at is that, in the case of a large state space, it may be important to consider errors $\|J^* - \tilde{J}\|$ that differentiate between more or less important states. In the next section, we will introduce the notion of weighted norms and present performance bounds that take state relevance into account.

2.1 Performance Bounds with State-Relevance Weights

We first introduce the following weighted norms:

$$\begin{aligned} ||J^* - \tilde{J}(\cdot, r)||_\infty &= \max_{x \in \mathcal{S}} |J^*(x) - \tilde{J}(x, r)| \\ \|J^* - \tilde{J}(\cdot, r)\|_{\infty,\nu} &= \max_{x \in \mathcal{S}} \nu(x) |J^*(x) - \tilde{J}(x, r)| \\ \|J^* - \tilde{J}(\cdot, r)\|_{1,\nu} &= \sum_{x \in \mathcal{S}} \nu(x) |J^*(x) - \tilde{J}(x, r)| \quad (\nu > 0) \\ &= \mathrm{E}\left[|J^*(x) - \tilde{J}(x, r)| : x \sim \nu \right] \end{aligned}$$

With this notation, we can prove the following theorem.

Theorem 2 *Let J be such that $J \leq TJ$. Let J_{u_J} be the cost-to-go of the greedy policy u_J. Then, for all $c > 0$,*

$$\|J_{u_J} - J^*\|_{1,c} \leq \frac{1}{1-\alpha} \|J - J^*\|_{1,\mu_{c,J}}$$

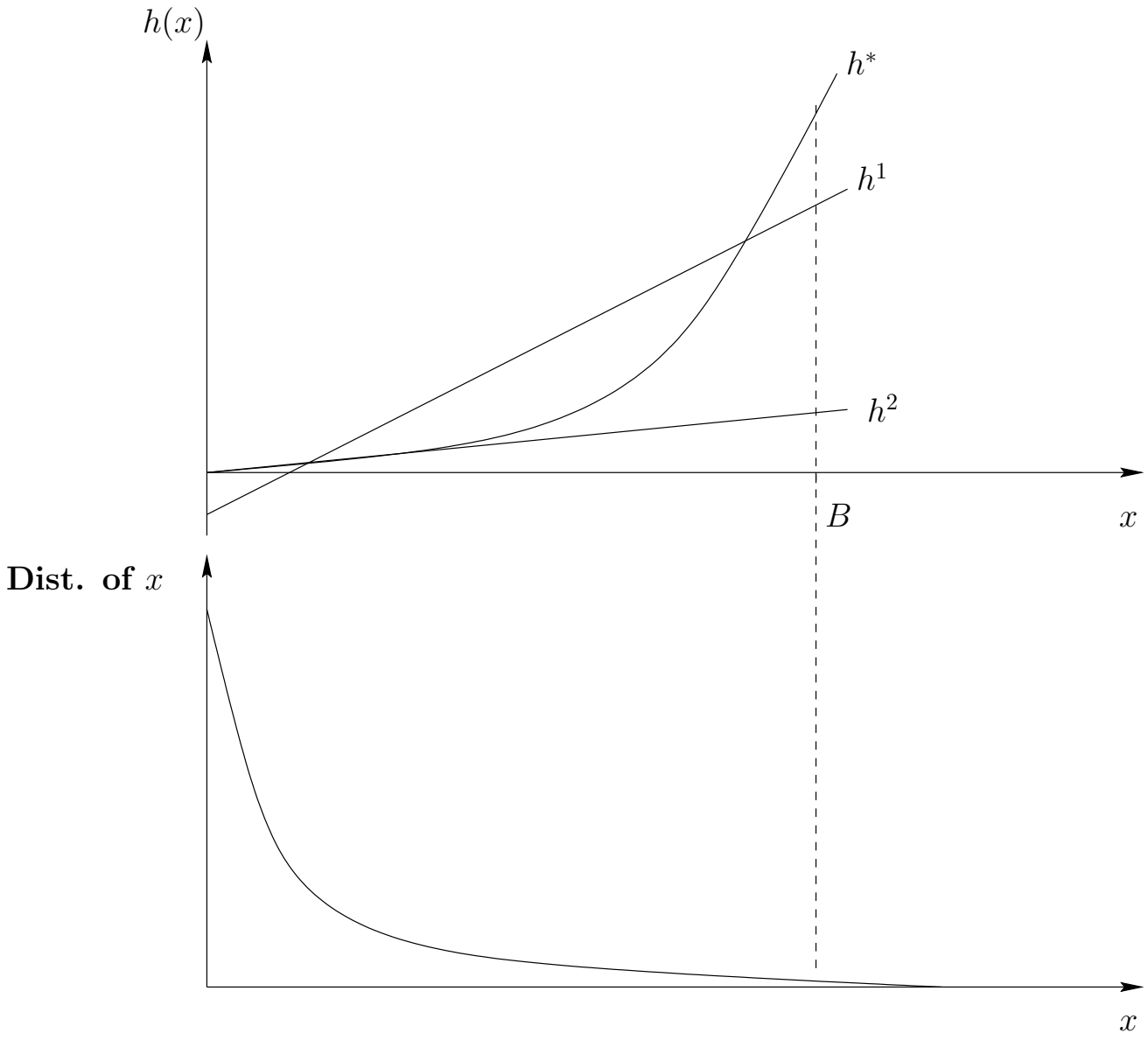

Figure 2: Illustration of Performance Bounds for Example 1

where

$$\mu_{c_J}^T = (1-\alpha)c^T(I-\alpha P_{u_J})^{-1} = (1-\alpha)c^T\sum_{t=0}^{\infty}\alpha^t P_{u_J}^t$$

or equivalently

$$\mu_{c,J}(x) = (1-\alpha)\sum_y c(y)\sum_{t=0}^{\infty}\alpha^t P_{u_J}^t(y,x), \quad \forall\, x \in \mathcal{S}.$$

Proof: We have $J \leq TJ \leq J^* \leq J_{u_J}$. Then

$$\begin{aligned}
\|J_{u_J} - J^*\|_{1,c} &= c^T(J_{u_J} - J^*) \\
&\leq c^T(J_{u_J} - J) \\
&= c^T\left[(I-\alpha P_{u_J})^{-1}g_{u_J} - J)\right] \\
&= c^T(I-\alpha P_{u_J})^{-1}(g_{u_J} + \alpha P_{u_J}J - J) \\
&= c^T(I-\alpha P_{u_J})^{-1}(TJ - J) \\
&\leq c^T(I-\alpha P_{u_J})^{-1}(J^* - J) \\
&= \frac{1}{1-\alpha}\|J^* - J\|_{1,\mu_{c,J}}
\end{aligned}$$

□

Comparing Theorems 1 and 2, we have

$$\begin{aligned}
\|J_{u_J} - J^*\|_\infty &\leq \frac{2}{1-\alpha}\|J - J^*\|_\infty \\
\|J_{u_J} - J^*\|_{1,c} &\leq \frac{1}{1-\alpha}\|J - J^*\|_{1,\mu_{c,J}}.
\end{aligned}$$

There are two important differences between these bounds:

1. The first bound relates performance to the worst possible approximation error over all states, whereas the second involves only a weighted average of errors. Therefore we expect the second bound to exhibit better scaling properties.

2. The first bound presents a worst-case guarantee on performance: the cost-to-go starting from any initial state x cannot be greater than the stated bound. The second bound presents a guarantee on the expected cost-to-go, given that the initial state is distributed according to distribution c. Although this is a weaker guarantee, it represents a more realistic requirement for large-scale systems, and raises the possibility of exploiting information about how important each different state is in the overall decision problem.

3 From Value to Actions

The analysis presented in the previous sections is based on the greedy policy $u_{\tilde{J}}$ associated with approximation $\tilde{J}$. In order to use this policy, we need to compute

$$u(x) = \arg\min_{a\in\mathcal{A}}\left\{g_a(x) + \alpha\sum_y P_a(x,y)\tilde{J}(y,r)\right\}. \tag{1}$$

This step is typically done in real-time, as the system is being controlled. If the set of available actions $\mathcal{A}$ is relatively small and the summation $\sum_y P_a(x,y)\tilde{J}(y,r)$ can be computed relatively fast, then evaluating (1) directly when an action at state x is needed is a feasible approach. However, if this is not the case, alternative solutions must be considered. We describe a few:

- If the action set is relatively small but there are many $y's$ to sum over, we can estimate

$$\sum_y P_a(x,y)\tilde{J}(y,r) \tag{2}$$

by sampling:

$$\frac{1}{N}\sum_{i=1}^{N} J(y_i,r). \tag{3}$$

- In some cases, a very large number of samples may be required for the empirical average (3) to be a reasonable estimate of (2). In these cases, the computation of (2) or (3) could be done offline, and stored via Q-factors:

$$Q_{\tilde{J}}(x,a) = g_a(x) + \alpha\sum_y P_a(x,y)\tilde{J}(y,r)$$

Clearly, $Q_J(x,a)$ requires space proportional to the size of the state space, and cannot be stored exactly. In the same spirit as value function approximation, we can approximate it with a parametric representation:

$$Q(x,a,s) \approx Q_{\tilde{J}}(x,a)$$

We may find an approximate parameter s based on the approximation $\tilde{J}$ by solving

$$\min_s \frac{1}{N}\sum_{i=1}^{N}\left(Q(x,a,s) - Q_{\tilde{J}}(x,a)\right)^2 \quad \text{(offline)}$$

or, alternatively, we could do value function approximation to approximate the Q-factor directly.

- Finally, if the action space is too large, computing $\min_{a\in\mathcal{A}} Q(x,a,s)$ may be prohibitively expensive as well. As an alternative, we may consider also a parametric representation for policies:

$$u(x) \approx u(x,t)$$

We may find an appropriate parameter t by fitting $u(x,t)$ to the greedy policy $u(x)$, computed offline:

$$\min_t \frac{1}{N}\sum_{i=1}^{N}\left(u(x_i) - u(x_i,t)\right)^2$$

or consider algorithms which mix together value function approximation and policy approximation.