Course Notes On Dynamic Optimization (Fall 2023)
Lecture 10: Policy iteration methods

Instructor: Daniel Russo

Email: djr2174@gsb.columbia.edu

Graduate Instructor: David Cheikhi
Email: d.cheikhi@columbia.edu

These notes are partly based of scribed notes from a previous edition of the class. I have done some follow up light editing, but there may be typos or errors.

1 Problem setup

Throughout these notes we continue to study indefinite horizon problems under the assumption that all policies eventually reach the terminal state.

We continue to assume that all policies are assured to reach the terminal state and

$$\mathbb{E}^{\pi}[\tau \mid x_0 = x] \leqslant M$$

for any x and any (possibly non-stationary) policy π.

1.1 Policy evaluation and matrix notation

Recall that

$$J_\mu(x) = g(x, \mu(x)) + \sum_{x' \in \mathcal{X}} p(x'|x, \mu(x)) J_\mu(x') \qquad \text{for all } x \in \mathcal{X}.$$

This can be written in matrix/vector notation as

$$J_\mu = g_\mu + P_\mu J_\mu. \tag{1}$$

We refer to the task of forming J_μ for a given μ as *policy evaluation*.

1.2 State occupancy notation

We will often work with the matrix

$$(I - P_\mu)^{-1} = \sum_{n=0}^{\infty} P_\mu^n.$$

It's a nice, convenient expression, but it can mask the intuition about what is going on. So let's parse what this expression means. Recall that the n step transition matrix has elements

$$(P_\mu^k)_{x,x'} = \mathbb{P}^\mu(x_k = x' | x_0 = x),$$

which encode the probability of transition from x to x' over k steps. As a result

$$\begin{aligned}
\lim_{N\to\infty} \left(\sum_{n=0}^N P_\mu^n \right)_{x,x'} = \lim_{N\to\infty} \mathbb{E}\left[\sum_{n=0}^N \mathbb{1}(x_n = x') \mid x_0 = x \right] &= \mathbb{E}\left[\sum_{n=0}^N \mathbb{1}(x_n = x') \mid x_0 = x \right] \\
&= \mathbb{E}\left[\sum_{n=0}^{N\wedge(\tau-1)} \mathbb{1}(x_n = x') \mid x_0 = x \right] \\
&= \mathbb{E}\left[\sum_{n=0}^{\tau-1} \mathbb{1}(X_n = x') \mid x_0 = x \right] \\
&\leqslant M
\end{aligned}$$

where the penultimate equality can be justified using the monotone convergence theorem. This reveals two things: (1) $(I - P_\mu)^{-1}$ exists due to our assumption that all states transition to the terminal state eventually and (2) $(I - P_\mu)^{-1}$ is a matrix that encodes the expected number of visits *to* any state x' *from* any initial state x.

2 Policy iteration

Algorithm 1 Policy iteration

```
Require: initial policy $\mu_0$
 1: for Episode $n = 0, 1, 2, \ldots$ do
 2:     Form $J_{\mu_n} \in \mathbb{R}^{\mathcal{X}}$ by solving the linear system (1)        ▷ Policy evaluation
 3:     for $x \in \mathcal{X}$ do
 4:         $\mu_{n+1}(x) \leftarrow \arg\min_{u \in U(x)} g(x,u) + \sum_{x' \in \mathcal{X}} p(x'|x,u) J_{\pi_n}(x')$.     ▷ Policy improvement
 5:     end for
 6:     if $\mu_{n+1} = \mu_n$ then
 7:         break
 8:     end if
 9: end for
10: return $\mu_n$
```

Written abstractly, each iteration of PI performs these steps:

1. Form one-step lookahead policy: $\mu_{n+1} \in G(J_{\mu_n})$,
2. Evaluate the new policy's performance over an infinite horizon: $J_{\mu_{n+1}} = (I - P_{\mu_n})^{-1} g_{\mu_n}$.

In the first step, we optimize *as if* μ_{n+1} is to be used for just one period. In the second step, we change our mind and decide to deploy μ_n in all periods, evaluating its performance over an infinite horizon. The magic that makes this work is monotonicity: applying μ_{n+1} in all periods is even better than applying it for just one period and then returning to μ_n.

2.1 Convergence of policy iteration

I believe this result dates back to about 1950. Back then, it was very exciting to prove that an algorithm terminated (with a certificate) in a finite number of iterations. Today, it is more fashionable to ask "what is the worst-case running time to get an ϵ–optimal policy?" One can prove that kind of result as well.

Proposition 1 (Classical guarantee for policy iteration)**.** *Suppose $\mathcal{X}$ and U are finite. Then, policy iteration terminates after finite number of iterations and returns the optimal policy.*

Beyond the result itself, the (in)equalities in the proof are important in their own right. It's worth making sure you can reprove this result.

Proof. Since $\mathcal{X}$ and U are finite, the set of possible (Deterministic, Markov) policies is finite.

Since

$$T_{\mu_{n+1}} J_{\mu_n} = T J_{\mu_n} \preceq T_{\mu_n} J_{\mu_n} = J_{\mu_n}.$$

The first equality used that μ_{n+1} is greedy with respect to J_{μ_n}. The inequality used the fact that $TJ \preceq T_\mu J$ for all μ, by definition of T. The final equality used that J_{μ_n} is a fixed point of T_{μ_n}..

We've concluded that $T_{\mu_{n+1}} J_{\mu_n} \preceq J_{\mu_n}$. Applying $T_{\mu_{n+1}}$ to each side and appealing to the monotonicity of the Bellman operator yields

$$J_{\mu_n} \succeq T_{\mu_{n+1}} J_{\mu_n} \succeq T^2_{\mu_{n+1}} J_{\mu_n} \succeq \ldots \succeq T^k_{\mu_{n+1}} J_{\mu_n} \succeq \ldots \succeq J_{\mu_{n+1}}.$$

We have two cases

1. $J_{\mu_{n+1}} = J_{\mu_n}$, in which case $TJ_{\mu_n} = J_{\mu_n}$ and we know that J_{μ_n} is optimal. Policy iteration terminates and returns an optimal policy.
2. $J_{\mu_{n+1}} \preceq J_{\mu_n}$ and for at least one x, $J_{\mu_{n+1}}(x) \prec J_{\mu_n}(x)$. In this case, we've moved from μ_n to a different and superior policy.

Since the set of policies is finite, case (2) can only happen a finite number of times. □

3 How policy iteration ideas are used (Mostly discussed in person)

We've presented policy iteration as nn algorithm for exactly computing the optimal policy for a small MDPs. With that view, the main question is whether it's faster than value iteration.

I (Dan) am not especially interested in rapidly solving MDPs with 1000 states. Nevertheless, I find policy iteration to be extremely important. Here are three ways it's important:

1. Given a base policy, you can implement a policy iteration update *in real time* by always performing lookahead from your current state to select an action and then rolling out the base policy. This technique was central to alpha-go. (There, "Monte-Carlo-Tree Search" is a heuristic way to do selective lookaheads that are longer.)
2. You consult for some organization that wants to optimize a long-term performance metric. (E.g. Ron Howard, inventor of policy iteration, worked on catalogue mailing polices for Sears in the 1950s.) That organization has lots of data on an incumbent policy μ, allowing you to

estimate J_μ in a reasonable way. If you have enough data to reasonable estimate the impact of one-step deviations for μ, you can perform a policy iteration update to find an improved policy.

3. Policy gradient methods, covered in the next class, apply (stochastic) gradient descent to minimize total expected long-run costs over a parameterized policy class. These align very well with modern nueral networks and compute infrastructure; armed with a differentiable simulator you can use GPUs and pytorch/tensorflor/jax to directly optimize what you care about. The theory we'll review in the next class clarifies that these methods, and their convergence properties, are intimately linked with policy iteration.

Notice that (2) and (3) seem to implicitly assume that one policy iteration update is enough to have a big impact.

4 Gap in the thoery

Practical experience suggests that policy iteration may often terminate after very few iterations (e.g. 7) and that a couple of iterations have a big impact. In fact, both motivations (1) and (2) only make sense if a single policy improvement step is already quite impactful.

The rest of these notes tries to offer some insight into the convergence rate of PI. I'll first review some textbook results and then offer some of my own efforts to make sense of when PI is slow and when it is guaranteed to be very very fast.

5 Convergence rate: "textbook results"

5.1 PI is at least as fast as value iteration

Lemma 1. *Under PI, for any n*

$$J_{\pi_n} \preceq T^n J_{\mu_0}.$$

Proof. We begin with the base case. As in our earlier analysis, we have

$$J_{\mu_0} \succeq T_{\mu_1} J_{\mu_0} \succeq T^2_{\mu_1} J_{\mu_0} \succeq \ldots \succeq T^k_{\mu_1} J_{\mu_0} \succeq \ldots \succeq J_{\mu_1}.$$

By the definition of μ_1, we have $T_{\mu_1} J_{\mu_0} = T J_{\mu_0}$, so $T J_{\mu_0} \succeq J_{\mu_1}$.

Assume now that $T^k J_{\mu_0} \succeq J_{\mu_k}$. An identical argument shows $T J_{\mu_k} \succeq J_{\mu_{k+1}}$. By monotonicity, $T J_{\mu_k} \succeq T^{k+1} J_{\mu_0}$, so we have $T^{k+1} J_{\mu_0} \succeq J_{\mu_{k+1}}$, completing the induction step. □

5.2 PI is Newton's method

The next result identifies a surprising alternative interpretation of policy iteration: it is exactly Newton's method (for root finding) applied to solving the equation $J - TJ = 0$.

This is our first suggestion that PI potentially must faster than VI. Since it, J_{μ_n} converges quadratically to J^* once it is in the neighborhood of J^*. Practical experience suggests (e.g. fitting nonlinear regression models) suggests that Newton's method often requires very few iterations to converge — far fewer than worst-case theory (at least the part I know) predicts. Like with PI, the issue is that each iteration requires solving a big linear system.

Proposition 2. *Define $B(J) = J - TJ$, then under PI, if $G(J_{\mu_n})$ is a singleton, then*

$$J_{\mu_{n+1}} = J_{\mu_n} - [\nabla B(J_{\mu_n})]^{-1} B(J_{\mu_n}).$$

This is Newton's method applied to solving $B(J) = 0$. Newton's method converges quadratically. That is, for J_{μ_n} sufficiently close to J^*, we should have

$$\|J_{\mu_{n+1}} - J^*\| \leqslant C\|J_{\mu_n} - J^*\|^2.$$

Toward proving this result, we recall a result from an earlier class. Given our interpretation of $(I - P_\mu)^{-1}$, this lemma is means

$$J(x) - J_\mu(x) = \mathbb{E}^\mu \left[\sum_{k=0}^{\tau-1} (J - T_\mu J)(x_k) \mid x_0 = x \right].$$

The expected gap between J and J_μ is the expected total Bellman error across states visited under μ.

Lemma 2 (Variational form of Bellman's equation)**.** *For any $J \in \mathbb{R}^{\mathcal{X}}$ and policy μ,*

$$J - J_\mu = (I - P_\mu)^{-1}(J - T_\mu J).$$

Proof.

$$\begin{aligned} J - J_\mu &= (J - T_\mu J) + (T_\mu J - J_\mu) \\ &= (J - T_\mu J) + (T_\mu J - T_\mu J_\mu) \\ &= (J - T_\mu J) + P_\mu(J - J_\mu) \end{aligned}$$

Subracting $P_\mu(J - J_\mu)$ from both sides yields the result. □

Now we prove that policy iteration is Newton's method. Viewed a bit more broadly, we show that a Newton update to J is the cost-to-go function J_μ for μ derived by one-step lookahead on J.

Proof of proposition. Fix J *(you can think of this as* J_{μ_n}) and suppose there is a unique greedy policy $G(J) = \{\mu\}$. (*you can think of this as* μ_{n+1}).

For some sufficiently small ϵ

$$\begin{aligned} \|J' - J\| \leqslant \epsilon &\implies G(J') = \{\mu\} \\ &\implies B(J') = J' - T_\mu J' = (I - P_\mu)J' - g_\mu. \end{aligned}$$

We find

$$\nabla B(J) = \left(I - \alpha P_\mu\right)$$

A Newton step to J produces J_μ:

$$\begin{aligned} J - [\nabla B(J)]^{-1} B(J) &= J - \left(I - \alpha P_\mu\right)^{-1} \left(J - T_\mu J\right) \\ &= J - \left(J - J_\mu\right) \\ &= J_\mu \end{aligned}$$

□

6 When is PI slow? When is it fast? A tale of distribution shift.

I do not fully understand when PI is slow and when it is fast. I suspect no one does. Despite this, I can share some partial understanding. Some of the problems where PI is slow resemble problems where exploration is very challenging. In problems with "low distribution shift", it is possible to prove that even a single iteration of PI makes enormous progress.

6.1 A case when it's slow: river swim revisited

Recall this problem used to illustrate the challenges of exploration.

We consider the following MDP describing the decision process of one person swimming across the river from land (state 1) to the island (state n) in Figure 1.

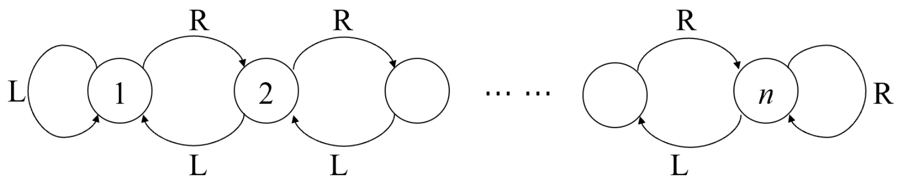

Figure 1: State-action Space of River Swim Problem

Here $\mathcal{X} = \{1, 2, \ldots, n\}$ and $U(x) = \{L, R\}, \forall x \in \mathcal{X}$. The cost function is given by:

$$
\begin{aligned}
g(s, L) &= 0, \\
g(s, R) &= \begin{cases} \varepsilon, & \text{if } s \leqslant n-1 \\ -1, & \text{if } s = n \end{cases}
\end{aligned}
$$

And the transition is deterministic, i.e. $p(s-1|s, L) = 1, s \geqslant 2; p(s+1|s, R) = 1, s \leqslant n-1$. And $p(1|1, L) = p(n|n, R) = 1$.

Consider a discount variation of the problem, where the discount factor is α is close to 1. As long as ε is small relative to $1-\alpha$, then the optimal policy is $\mu^*(x) = R, \forall x$. The optimal policy swims from the mainland to the island and then stays there perpetually.

The next

Lemma 3 (informal)**.** *If μ_0 is a policy that moves left from all states, then $\mu_1 = G(J_{\mu_0})$ is a policy that moves left from all states except state n. The policy $\mu_2 \in G(J_{\mu_1})$ is a policy that moves left from all states except states n and $n-2$, and so on.*

A policy that outperforms μ_0 when employed from the leftmost state ($x = 1$) is attained only after n policy iteration updates.

proof idea. When you perform a policy iteration update to μ_1, you calculate the lookahead optimal action

$$\arg\min_{u \in U(x)} g(x, u) + \sum_{x' \in \mathcal{X}} p(x'|x, u) J_{\mu_1}(x').$$

You're asking "if I could deviate from μ_1 for one time period, what should I do?" Well the policy μ_1 always moves left. Given that you'll always swim directly back to the mainland, and swimming sucks, there is no reason to ever start swimming to the island. The only exception is once you're already on the island. Then, you might as well stay there for one period. □

6.2 A case when PI is fast: low distribution shift

This section argues the following: *A policy iteration update to some policy makes enormous progress if that policy visits states with frequency similar to an optimal policy.*

Let w be some initial distribution over states, viewed as a row vector. Define the discounted state occupancy measure

$$d_\mu = w\left(I - P_\mu\right)^{-1}$$

An alternative way of expressing the same thing is

$$d_\mu(x) = \mathbb{E}^\mu\left[\sum_{n=0}^{\tau-1} \mathbb{1}(x_k = x)\right] \qquad x_0 \sim w.$$

We'll say distribution shift is low if $d_{\mu^*}(x)/d_\mu(x)$ is never too large, for any x. The next result shows that $d_{\mu^*}(x)/d_\mu(x) \leqslant 2$ implies a single policy iteration update to μ removes 50% of the policy's optimality gap. This guarantee is completely vacuous in river swim, where $d_\mu(x) = 0$ for some states that the optimal policy visits.

Proposition 3.

$$\underbrace{\mathbb{E}_{x_0\sim w}[J_{\mu_{n+1}}(x_0) - J_{\mu^*}(x_0)]}_{\text{Avg optimality gap at } n+1} \leqslant \beta_n \underbrace{\mathbb{E}_{x_0\sim w}[J_{\mu_n}(x_0) - J_{\mu^*}(x_0)]}_{\text{Avg optimality gap at } n}.$$

where β_n is the distribution shift coefficient

$$\beta_n = \max_{x\in\mathcal{X}} \frac{d_{\mu^*}(x) - d_{\mu_n}(x)}{d_{\mu^*}(x)}$$

On first reading, you should understand μ^* to be the optimal policy, so $J_{\mu^*} = J^*$. But that fact is never used in the proof. So in general, we'd say that a single policy iteration update to μ_n closes much of the optimality gap with any policy whose occupancy measure is relatively close to that of μ_n.

The next lemma gives more a precise quantification of policy improvement.

Lemma 4. $J_{\mu_n} - J_{\mu_{n+1}} = (I - P_{\mu_{n+1}})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right)$

Proof. Apply the variational Bellman eq w/ $J \equiv J_{\mu_n}$ and $\mu \equiv \mu_{n+1}$:

$$\begin{aligned} J_{\mu_n} - J_{\mu_{n+1}} &= (I - P_{\mu_{n+1}})^{-1}\left(J_{\mu_n} - T_{\mu_{n+1}}J_{\mu_n}\right) \\ &= (I - P_{\mu_{n+1}})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right) \end{aligned}$$

□

Heuristically at least, this is suggestive of much faster convergence than value iteration, since $(I - P_{\mu_n})^{-1}(x)$ is very big.

The next lemma complements that with a (similar looking) bound on the optimal gap of a policy.

Lemma 5. $J_{\mu_n} - J_{\mu^*} \preceq (I - P_{\mu^*})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right)$

Proof.

$$\begin{aligned} J_{\mu_n} - J_{\mu^*} &= (I - P_{\mu^*})^{-1}\left(J_{\mu_n} - T_{\mu^*}J_{\mu_n}\right) \\ &\preceq (I - P_{\mu^*})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right). \end{aligned}$$

□

We now complete the proof by relating those two expressions.

Proof. Observe that

$$\beta_n = 1 - \frac{1}{c_n} \quad \text{where} \quad c_n = \max_x \frac{d_{\mu^*}(x)}{d_{\mu_{n+1}}(x)}.$$

Now, the definition of c_n gives $c_n d_{\mu_{n+1}} \succeq d_{\mu^*}$, or

$$c_n w(I - P_{\mu_{n+1}})^{-1} \succeq w(I - P_{\mu^*})^{-1}.$$

Recall that w is a row vector. We have,

$$\begin{aligned} \mathbb{E}_{x_0\sim w}[J_{\mu_{n+1}}(x_0) - J^*(x_0)] &= w\left(J_{\mu_{n+1}} - J^*\right) \\ &= w\left[J_{\mu_n} - J^* - (I - P_{\mu_{n+1}})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right)\right] \\ &\preceq w\left[J_{\mu_n} - J^*\right] - c_n^{-1}w\left[(I - P_{\mu^*})^{-1}\left(J_{\mu_n} - TJ_{\mu_n}\right)\right] \\ &\preceq \left(1 - c_n^{-1}\right) w\left(J_{\mu_n} - J_{\mu^*}\right) \\ &= \left(1 - c_n^{-1}\right)\mathbb{E}_{x_0\sim w}[J_{\mu_n}(x_0) - J^*(x_0)] \\ &= \beta_n \mathbb{E}_{x_0\sim w}[J_{\mu_n}(x_0) - J^*(x_0)]. \end{aligned}$$

□

6.3 On the blessings of uncontrollable systems

Something curious emerges from this:

Problems in which the decision maker has limited influence on state are, in a sense, easier.

This statement depends on how you think about performance. If you want to attain low costs, then you'll hope to have perfect control of states. If you want to have low sub-optimality gap, then this may be easier when your limited influence on states handicaps both your controller and the optimal one. (At least, we have a guarantee that policy iteration is very very fast.)

My rough intuition is that many problems that involve repeated interactions with users/customers etc are problems with limited control. Users tend to do what they want, and small changes to the system don't make users 50X as likely to behave one way or another. An effect size of 1% in an A/B test might be considered large.