Dynamic Programming

Csaba Szepesvári

Monday 6th February, 2023

Dynamic programming is the key,
To solve problems, smart and neat,
It takes complex tasks with glee,
And breaks them down to bits to see.

With overlapping subproblems found,
Optimal solutions can be wound,
From bottom up or top to bottom,
A path to the answer, so simple and common.

From knapsack to sequence alignment,
Dynamic programming's got your back in,
Saving time and space, oh what a treat,
A problem solver, truly elite!
– ChatGPT

1 Dynamic Programming

Fix a finite MDP $M = (\mathcal{S}, \mathcal{A}, \mathcal{R}, p)$ and consider the continuing case with the discounted total expected reward criterion with discount factor $0 \le \gamma < 1$ ("finite, discounted MDP"). Recall that $r_{\max} = \max\{|r| : r \in \mathcal{R}\}$.

Dynamic programming is a set of tools and techniques to compute solutions to various problems that involves recursion and caching, or memoization. The classic example of dynamic programming is the computation of the Fibonacci sequence, where the idea is to store the previous values computed which dramatically reduces the computation time of the simple recursive approach.

In the context of MDPs, dynamic programming is a set of techniques, such as value and policy iteration, or linear programming, to compute a near optimal policy through calculating either the optimal value function, or a good approximation of it.

2 Value iteration

Recall that the optimal value function, v^*, is the fixed-point of the Bellman-optimality operator, T. That is, $v^* = Tv^*$. Further, by the contraction mapping theorem, for any $v_0 \in \mathbb{R}^{\mathcal{S}}$, the iteration

$$v_{n+1} = Tv_n, \quad n \geq 0 \tag{1}$$

leads to a sequence $(v_n)_{n\geq 0}$ that converges to v^* at a geometric rate:

$$\|v_n - v^*\| \leq \gamma^n \|v_0 - v^*\| \,, \tag{2}$$

where $\|\cdot\|$ is the maximum-norm: $\|v\| = \max_{s\in\mathcal{S}} |v(s)|$. Recalling the definition of T, the computation in Equation (1) takes the form

$$v_{n+1}(s) = \max_{a\in\mathcal{A}} \left\{ r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) v_n(s') \right\} \,, \qquad s \in \mathcal{S} \,.$$

The usual choice for v_0 is to choose $v_0(s) = 0$, $s \in \mathcal{S}$ (the constant zero function). With this choice, Equation (1) is called the vanilla (simple) version of *value iteration*. Fix this choice for now (that is, $v_0 = 0$). Because $\|v^*\| \leq r_{\max}/(1-\gamma)$ (why?), we then have

$$\|v_n - v^*\| \leq \frac{\gamma^n}{1-\gamma} r_{\max} \,.$$

We call the quantity $1/(1-\gamma)$ the *effective horizon* and one often uses the symbol $H = 1/(1-\gamma)$. With this, the above inequality takes the form

$$\|v_n - v^*\| \leq \gamma^n H r_{\max} \,. \tag{3}$$

The next question is when to stop this iteration and what to do with the computed approximation v_n of the optimal value function? The answer to the second question comes from the Fundamental Theorem, which stated that greedy policies with respect to v^* are optimal. The idea then is that if v_n is a good approximation to v^*, then a policy that is greedy with respect to v_n may be near-optimal. Indeed, the following result holds:

Proposition 2.1. *Let $v \in \mathbb{R}^{\mathcal{S}}$ and let π be greedy with respect to v. Then*

$$\|v_\pi - v^*\| \leq 2\gamma H \|v - v^*\| \,.$$

Proof. We have

$$\begin{aligned} v^* - v_\pi &= Tv^* - T_\pi v_\pi \\ &= Tv^* - Tv + Tv - T_\pi v_\pi \\ &= Tv^* - Tv + T_\pi v - T_\pi v_\pi \,. && \text{(because } \pi \text{ is greedy w.r.t. } v) \end{aligned}$$

Taking the norm of both sides, and using the triangle inequality,

$$\begin{aligned} \|v^* - v_\pi\| &\leq \|Tv^* - Tv\| + \|T_\pi v - T_\pi v_\pi\| \\ &\leq \gamma\|v^* - v\| + \gamma\|v - v_\pi\| \,. && (T,\, T_\pi \text{ are contractions}) \end{aligned}$$

By the triangle inequality, we have $\|v - v_\pi\| \le \|v - v^*\| + \|v^* - v_\pi\|$. Plugging this in and reordering,

$$(1-\gamma)\|v^* - v_\pi\| \le 2\gamma\|v^* - v\| \,.$$

Dividing both sides by $(1-\gamma)$ gives the desired result. □

Proposition 2.2 (A priori iteration bound for value iteration)**.** *Fix any $\varepsilon > 0$. Let $n \ge 0$ be large enough so that*

$$\gamma^{n+1} \le \frac{\varepsilon}{2H^2 r_{\max}} \,. \tag{4}$$

Let π be greedy w.r.t. v_n. Then π is ε-optimal: $v_\pi \ge v^* - \varepsilon \mathbf{1}$ where $\mathbf{1}$ is the all-one function.

Proof. Fix some $n \ge 0$ and let π be greedy w.r.t. v_n. Combining Equation (3) with Proposition 2.1 we have that

$$\|v_\pi - v^*\| \le 2\gamma^{n+1} H^2 r_{\max}$$

Then, if n satisfies the condition of the theorem, we have $\|v_\pi - v^*\| \le \varepsilon$. □

Taking the logarithm of both sides of Equation (4) and reordering we see that the condition on n is

$$n \ge \frac{\log\left(\frac{2H^2 r_{\max}}{\varepsilon}\right)}{\log(1/\gamma)} - 1 \,.$$

Thus if n is the *smallest* positive integer such that Equation (4) holds then

$$n \le \frac{\log\left(\frac{2H^2 r_{\max}}{\varepsilon}\right)}{\log(1/\gamma)} \,.$$

Note also that $1/\log(1/\gamma) \le H$. Indeed, $\log(1/\gamma) = -\log(\gamma) = -\log(1-(1-\gamma))$. Now, $\log(1+x) \le x$ holds for any $x > -1$. Hence, $\log(1/\gamma) \ge 1 - \gamma = 1/H$. Thus, it follows that

$$n \le H \log\left(\frac{2H^2 r_{\max}}{\varepsilon}\right) \,. \tag{5}$$

While the dependence on $1/\varepsilon$ is mild, we see that the effort required scales linearly with H.

We now formulate an alternate stopping criterion. To develop this recall that by a previous result (Prop 5.9), for any $n \ge 0$,

$$\|v_n - v^*\| \le H\|v_n - v_{n+1}\| \,. \tag{6}$$

Proposition 2.3 (Stopping just-in-time)**.** *Fix $\varepsilon > 0$. Let $n \ge 0$ be the smallest integer such that*

$$\|v_n - v_{n+1}\| \le \frac{\varepsilon}{2\gamma H^2} \,. \tag{7}$$

holds. Then, if π is greedy with respect to v_n then π is ε-optimal.

Note that choosing π to be greedy with respect to v_{n+1} also gives an ε-optimal policy (why?); the logic here is that if v_{n+1} is already computed, v_{n+1} is expected to be a better approximation to v^* then v_n then a greedy policy with respect to v_{n+1} is also expected to be better than a greedy policy with respect to v_n. Note that this does not prove that the policy that is greedy with respect to v_{n+1} is necessary better than that is greedy w.r.t. v_n.

Proof. Fix some $n \geq 0$ and let π be greedy w.r.t. v_n. Combining Equation (6) with Proposition 2.1 we have that

$$\|v_\pi - v^*\| \leq 2\gamma H^2 \|v_n - v_{n+1}\| \,.$$

Hence, if Equation (7) holds then π is indeed ε-optimal. □

It remains to see whether stopping based on Equation (7) is actually an improvement compared to stopping based on Equation (4). For this, we have the following result:

Proposition 2.4. *As before, let $v_0 = 0$. Let n be the smallest positive integer such that the inequality in* (7) *holds. Then,*

$$n \leq 1 + H \log\left(\frac{4\gamma H^3 r_{\max}}{\varepsilon}\right) . \tag{8}$$

Proof. Fix n as in the statement. For $k \geq 0$,

$$\|v_k - v_{k+1}\| \leq \|v_k - v^*\| + \gamma\|v_k - v^*\| \leq 2\gamma^k \|v_0 - v^*\| \leq 2\gamma^k H r_{\max} \,.$$

Now, if k is an integer such that

$$2\gamma^k H r_{\max} \leq \frac{\varepsilon}{2\gamma H^2} \tag{9}$$

then $n \leq k$. Solving for the smallest k such that Equation (9) holds, we find that

$$k \leq 1 + \frac{\log\left(\frac{4\gamma H^3 r_{\max}}{\varepsilon}\right)}{\log(1/\gamma)} \leq 1 + H \log\left(\frac{4\gamma H^3 r_{\max}}{\varepsilon}\right) .$$

The proof is finished by chaining this inequality with $n \leq k$. □

Comparing the right-hand side of Equation (8) with that of Equation (5) we see that the second bound is somewhat larger than the first one. However, they are fundamentally of the same order as $H \to \infty$ ($\gamma \to 1$), or $\varepsilon \to 0$. We note in passing that comparing upper bounds is not entirely satisfactory: It can be that any of these upper bounds is loose. Can we say something definite about whether the first or the second stopping rule is better? This remains to be seen.

3 Policy iteration

Policy iteration construct a sequence of policies $(\pi_n)_{n\ge 0}$ as follows:

1. Choose an arbitrary memoryless policy π_0;
2. Given π_n, let π_{n+1} be a greedy policy with respect to v_{π_n}, with ties broken in favor of actions used by π_n.

Note that if in step 2, $\pi_n = \pi_{n+1}$ then we also have $\pi_n = \pi_{n+k}$ for any $k \ge 0$ and we may even stop the procedure. Indeed, this is how typically policy iteration is presented (and implemented).

Definition 3.1 (Monotonicity)**.** The function $F : \mathbb{R}^{\mathcal{S}} \to \mathbb{R}^{\mathcal{S}}$ is called *monotonous* if for any $u \le v$, $F(u) \le F(v)$ also holds.

Proposition 3.2 (Bellman operators are monotonous)**.** *For any memoryless policy π, T_π is monotonous. The same holds for the Bellman optimality operator T.*

Proof. Left as an exercise. □

Proposition 3.3 (Policy iteration is not slower than value iteration)**.** *Let $(\pi_n)_{n\ge 0}$ be obtained in the process at the beginning of the section. Further let $v_0 = v_{\pi_0}$ and $v_{n+1} = Tv_n$. Then,*

$$\|v_{\pi_n} - v^*\| \le \|v_n - v^*\|$$

and, in particular, we also have

$$\|v_{\pi_n} - v^*\| \le \gamma^n \|v_{\pi_0} - v^*\| . \tag{10}$$

Proof. We claim that for any n,

$$v^* \ge v_{\pi_n} \ge v_n .$$

From this, the result follows using Equation (2).

That $v^* \ge v_{\pi_n}$ follows from the definition of v^*. We prove the rest by induction on $n \ge 0$. The result holds by definition for $n = 0$. Hence, let $n \ge 0$ and assume that

$$v_{\pi_n} \ge v_n \tag{11}$$

holds.

We have

$$T_{\pi_{n+1}} v_{\pi_n} = T v_{\pi_n} \ge T_{\pi_n} v_{\pi_n} = v_{\pi_n} .$$

Applying $T_{\pi_{n+1}}$ to both sides and using that by Proposition 3.2 $T_{\pi_{n+1}}$ is monotonous, we get that for any $k \ge 0$,

$$T^k_{\pi_{n+1}} v_{\pi_n} \ge T^{k-1}_{\pi_{n+1}} v_{\pi_n} \ge \cdots \ge T_{\pi_{n+1}} v_{\pi_n} = T v_{\pi_n} .$$

Taking $k \to \infty$, we get

$$v_{\pi_{n+1}} \geq T v_{\pi_n} \,.$$

Now, by Equation (11), $v_{\pi_n} \geq v_n$ holds. Using that by Proposition 3.2 T is monotonous, we get that

$$v_{\pi_{n+1}} \geq T v_{\pi_n} \geq T v_n = v_{n+1} \,,$$

which finishes the induction, and, thus, also the proof. □

Corollary 3.4 (A priori iteration bound for policy iteration)**.** *Fix $\varepsilon > 0$. Let $n \geq 0$ be large enough so that*

$$\gamma^{n+1} \leq \frac{\varepsilon}{2 H r_{\max}} \,. \tag{12}$$

Then, π_n is ε-optimal.

Proof. From Equation (10), $\|v_{\pi_n} - v^*\| \leq \gamma^n \|v_{\pi_0} - v^*\| \leq 2\gamma^n H r_{\max}$. Plugging in the definition of n gives the result. □

The bound that can be extracted from this result that is comparable to Equation (5) is

$$n \geq H \log\left(\frac{2 H r_{\max}}{\varepsilon}\right) - 1 \,. \tag{13}$$

Note that while this bound is smaller than that in Equation (5), the difference is that here we have H inside the logarithm, while in Equation (5) we had H^2. Since $\log(H^2) = 2\log(H)$, this is a factor of 2 difference in an additive (ε-free) term, which is not expected to make a large difference.

One can also develop stopping conditions for policy iteration, given some desired suboptimality level ε. The details are left as an exercise.

4 Some thoughts

Value iteration and policy iteration as written here, in their vanilla form, are naive computational methods in that they require "sweeps" through the state-action space. Further, they require access the the transition probabilities and the immediate rewards. As such, they can only be used for small problems and when the complete MDP model is available. Occasionally, this may be the case, but more often than not, either the state, or the action space, or both may be too large, or the required data about the MDP is not available. In these cases, alternate methods will be needed. Yet, value iteration and policy iteration serve as a good starting point in developing these alternate methods and the proof techniques ("contraction arguments") used to analyze their behavior are also generally useful even beyond the analysis done here.

Recent results show that policy iteration results in the *optimal* policy in $\mathrm{poly}(|\mathcal{S}|, |\mathcal{A}|, H)$ steps. Note that this assumes that all the calculations are done with exact arithmetic. Nevertheless, this is a highly intriguing result. A similar result does not hold for value iteration: That is, one can always find an MDP such that to get the optimal policy in the MDP, for any fixed polynomial function, poly, more than $\mathrm{poly}(|\mathcal{S}|, |\mathcal{A}|, H)$ computational steps are needed by value iteration to give an optimal policy. While this is curious, while exact optimal policies are obviously nice to have, more often than not, a policy that is nearly optimal will be perfectly satisfactory and when the goal is to calculate such policies, the advantage of policy iteration is less clear. In particular, policy iteration requires the exact evaluation of policies. This requires solving for the fixed point of T_π. Because of the special structure of T_π, this fixed point computation reduces to solving an $|\mathcal{S}| \times |\mathcal{S}|$ linear system of equations, which can be done in polynomial time in $|\mathcal{S}|$. However, the cost of solving this linear system can make policy iteration less desirable.

There are many modifications of these basic methods to speed them up. While interesting, ultimately, for us it will be more interesting to consider the case when the state space is so large that exact calculations are out of question, and one needs to use sampling and function approximation. Another more interesting problem will be how to get near optimal policies when the MDP is not known but can be experimented with. The significance of the basic results presented here is that they will form the basis of many of the algorithms developed for these more challenging scenarios.

5 Simultaneous policy improvement

It is interesting to note that one can simultaneously improve upon a set of policies in simple ways. We formalize these results in this section. We need some notation first. For two functions $f, g : \mathcal{X} \to \mathbb{R}$ we use $f \vee g$ to denote the function mapping $\mathcal{X}$ to the reals that is obtained as the pointwise maximum of f and g:

$$(f \vee g)(x) = \max(f(x), g(x)), \qquad x \in \mathcal{X} .$$

We also generalize this to more than two functions: For $f_1, \dots, f_k : \mathcal{X} \to \mathbb{R}$, we let $\vee_{i\in[k]} f_i$ be the function defined by

$$(\vee_{i\in[k]} f_i)(x) = \max(f_1(x), \dots, f_k(x)), \qquad x \in \mathcal{X} .$$

Let π be a memoryless policy. We also introduce $M_\pi, M : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}}$:

$$\begin{aligned} (Mq)(s) &= \max_{a\in\mathcal{A}} q(s,a) \\ (M_\pi q)(s) &= \sum_{a\in\mathcal{A}} \pi(a|s) q(s,a) , \end{aligned}$$

where $s \in \mathcal{S}$, $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and π is any memoryless policy. Recall that $v_\pi = M_\pi q_\pi$ and q_π is the unique fixed-point of the Bellman operator $\tilde{T}_\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ defined by

$$(\tilde{T}_\pi)(s,a) = r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a)((M_\pi q)(s')) .$$

Proposition 5.1. *Let $\pi_1, \dots, \pi_k$ be policies, $\bar{q} = \vee_{i\in[k]} q_{\pi_i}$. Let $\bar{\pi}$ be a policy that satisfies any of the following conditions:*

1. *For any $i \in [k]$, $M_{\bar{\pi}} \bar{q} \ge M_{\pi_i} q_{\pi_i}$;*

2. *$\bar{\pi}$ is greedy with respect to q: $M_{\bar{\pi}} \bar{q} = M \bar{q}$;*

3. *For any $s \in \mathcal{S}$, $\bar{\pi}(s) = \pi_i(s)$ for any index $i \in [k]$ such that $v_{\pi_i}(s) = \max_{j\in[k]} v_{\pi_j}(s)$.*

Then, it holds that $\bar{\pi}$ is at least as good as any of the policies π_i:

$$v_{\bar{\pi}} \geq v_{\pi_i}\,, \qquad i \in [k]\,. \tag{14}$$

Proof. We first prove Equation (14) under the first condition and then we show that if either the second, or the third condition holds then the first also holds.
Part 1: Fix $i \in [k]$. Assume that $M_{\bar{\pi}}\bar{q} \geq M_{\pi_i} q_{\pi_i}$. Our goal is to show that $v_{\bar{\pi}} \geq v_{\pi_i}$. For this, it suffices to show that

$$\tilde{T}_{\bar{\pi}}\bar{q} \geq \bar{q} \tag{15}$$

because then iterating $\tilde{T}_{\bar{\pi}}$, we get

$$\bar{q} \leq \tilde{T}_{\bar{\pi}}^{k}\bar{q} \to q_{\bar{\pi}} \quad \text{as } k \to \infty$$

and thus, applying $M_{\bar{\pi}}$ to both sides,

$$v_{\pi_i} = M_{\pi_i} q_{\pi_i} \leq M_{\bar{\pi}} q_{\pi_i} \leq M_{\bar{\pi}}\bar{q} \leq M_{\bar{\pi}} q_{\bar{\pi}} = v_{\bar{\pi}}\,,$$

finishing the proof. It remains to show Equation (15). Fix $i \in [k]$. By our assumption on $\bar{\pi}$, $M_{\bar{\pi}}\bar{q} \geq M_{\pi_i} q_{\pi_i}$. Hence, it follows that $\tilde{T}_{\bar{\pi}}\bar{q} \geq \tilde{T}_{\pi_i} q_{\pi_i} = q_{\pi_i}$. Since this holds for any $i \in [k]$, $\tilde{T}_{\bar{\pi}}\bar{q} \geq \vee_{i\in[k]} q_{\pi_i} = \bar{q}$.
Part 2: Assume that $M_{\bar{\pi}}\bar{q} = M\bar{q}$ holds. Fix $i \in [k]$. It suffices to show that $M_{\bar{\pi}}\bar{q} \geq M_{\pi_i} q_{\pi_i}$. Indeed,

$$M_{\bar{\pi}}\bar{q} = M\bar{q} \geq M_{\pi_i}\bar{q} \geq M_{\pi_i} q_{\pi_i}\,,$$

where the first inequality used the definition of M, the second used that $\bar{q} \geq q_{\pi_i}$, which holds by the construction of $\bar{q}$.
Part 3: Fix $s \in \mathcal{S}$. Assume that $i \in [k]$ is such that $\bar{\pi}(s) = \pi_i(s)$. Then, for any $j \in [k]$,

$$(M_{\bar{\pi}}\bar{q})(s) = \bar{q}(s, \pi_i(s)) = v_{\pi_i}(s) \geq v_{\pi_j}(s) = q_{\pi_j}(s, \pi_j(s)) = M_{\pi_j} q_{\pi_j}\,.$$

□