Dynamic Programming & Optimal Control

Lecture 5
Solving the Bellman Equation

Fall 2023

Prof. Raffaello D'Andrea

ETH Zurich

Learning Objectives

Topic: Solving the Bellman Equation

Objectives

- You understand the *Value iteration* and *Policy Iteration* algorithms and are able to apply them.
- You know the difference between *Value* and *Policy Iteration*.

Outline

Value Iteration (VI)

We will look at several methods to solve the Bellman Equation for the SSP problem: Value Iteration, Policy Iteration and Linear Programming.

Value iteration applies the DP recursion introduced in Theorem 4.1:

$$V_{l+1}(i) = \min_{\mathrm{u} \in \mathcal{U}(i)} \left(q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V_l(j) \right), \quad \forall i \in \mathcal{S}^+,$$

with arbitrary initialization $V_0(i)$ for all $i \in \mathcal{S}^+$, until it converges. Remember the definition $\mathcal{S}^+ := \mathcal{S} \backslash \{0\}$.

Generally, value iteration requires an *infinite* number of iterations to converge to the optimum $J^*(\cdot)$. In practice, we define a threshold for $\|V_{l+1}(i) - V_l(i)\|$, $\forall i \in \mathcal{S}^+$, at which the algorithm terminates.

Outline

Policy Iteration (PI)

Policy iteration iterates over policies instead of values (costs). Consider the following theorem:

Theorem 5.1: Policy Evaluation

Under Assumption 4.1 (cost-free termination state), for *any* proper policy $\mu \in \Pi$, the associated cost vector $J_\mu := (J_\mu(1)\,, ..., J_\mu(n))$ is the unique solution of

$$J_\mu(i) = q(i, \mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) J_\mu(j)\,, \quad \forall i \in \mathcal{S}^+.$$

Given any initial conditions V_0, the sequence V_l generated by the recursion:

$$V_{l+1}(i) = q(i, \mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) V_l(j)\,, \quad \forall i \in \mathcal{S}^+$$

converges to J_μ.

Proof of Theorem 5.1

Note that this theorem is a special case of Theorem 4.1: consider a modified problem where the only allowable control at state i is $\mu(i)$.

Thus $\Pi = \{\mu\}$, containing only the proper policy under consideration, thereby satisfying Assumption 4.2 (existence of at least one proper policy).

Then invoke Theorem 4.1 to yield Theorem 5.1.

Policy Iteration (PI)

- **Initialization**: Initialize with a proper policy $\mu^0 \in \Pi$.
- **Stage 1** (*Policy Evaluation*): Given a policy μ^h, solve for the corresponding cost J_{μ^h} by solving the linear system of equations

$$J_{\mu^h}(i) = q\big(i, \mu^h(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^h(i)) J_{\mu^h}(j)\,,\ \forall i \in \mathcal{S}^+.$$

- **Stage 2** (*Policy Improvement*): Obtain a new stationary policy μ^{h+1} as follows

$$\mu^{h+1}(i) = \operatorname*{arg\,min}_{\mathrm{u} \in \mathcal{U}(i)} \left(q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) J_{\mu^h}(j) \right),\ \forall i \in \mathcal{S}^+.$$

Iterate between Stage 1 and 2 until $J_{\mu^{h+1}}(i) = J_{\mu^h}(i)$ for all $i \in \mathcal{S}^+$.

Policy Iteration (PI)

Theorem 5.2: Policy Iteration

Under Assumptions 4.1 (cost-free termination state) and 4.2 (existence of proper policy), policy iteration converges to an optimal policy after a *finite* number of steps.

Proof of Theorem 5.2 (1/5)

For a fixed h and a proper policy μ^h, consider the following auxiliary recursion in l:

$$V_0(i) = J_{\mu^h}(i),$$

$$V_{l+1}(i) = q\big(i, \mu^{h+1}(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_l(j)\,,\ \forall i \in \mathcal{S}^+.$$

Note that if we can show that μ^{h+1} is a proper policy, then $V_l(i)$ converges to $J_{\mu^{h+1}}(i)$ by Theorem 5.1.

By Theorem 5.1 (policy evaluation) applied to policy μ^h:

$$V_0(i) = J_{\mu^h}(i) = q\big(i, \mu^h(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^h(i)) V_0(j)\,.$$

Proof of Theorem 5.2 (2/5)

From Stage 2 of PI (policy improvement) it follows that:

$$\begin{aligned} V_0(i) &= q\big(i, \mu^h(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^h(i)) V_0(j) \\ &\geq q\big(i, \mu^{h+1}(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_0(j) \\ &= V_1(i)\,. \end{aligned}$$

Since $V_0(i) \geq V_1(i)$ for all $i \in \mathcal{S}^+$ (from above):

$$\begin{aligned} V_1(i) &= q\big(i, \mu^{h+1}(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_0(j) \\ &\geq q\big(i, \mu^{h+1}(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_1(j) \\ &= V_2(i)\,. \end{aligned}$$

Proof of Theorem 5.2 (3/5)

By continuing the above argument we find that

$$V_0\left(i\right) \geq V_1\left(i\right) \geq V_2\left(i\right) \geq ... \geq V_l\left(i\right), \quad \forall i \in \mathcal{S}^+.$$

Note that the policy μ^{h+1} is also proper: by contradiction, assume μ^{h+1} is improper. Thus by Assumption 4.2 (existence of proper policy), $J_{\mu^{h+1}}(i)$ is infinity for at least one state $i \in \mathcal{S}$.

Proof of Theorem 5.2 (4/5)

Furthermore, note that the auxiliary recursion

$$V_0(i) = J_{\mu^h}(i),$$
$$V_{l+1}(i) = q\big(i, \mu^{h+1}(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_l(j)\,,$$

is the DP recursion after the index substitution $l := N - k$ for the following expected cost function:

$$\begin{aligned}\tilde{J}^N_{\mu^{h+1}}(i) &= \underset{(X_1, W_0 | x_0 = i)}{\mathrm{E}} \left[\sum_{k=0}^{N-1} g\big(x_k, \mu^{h+1}(x_k), w_k\big) + J_{\mu^h}(x_N) \right] \\ &= J^N_{\mu^{h+1}}(i) + \underset{(X_1, W_0 | x_0 = i)}{\mathrm{E}} \big[J_{\mu^h}(x_N) \big]\,,\end{aligned}$$

with the control space constrained to $\mathcal{U}(i) = \{\mu^{h+1}(i)\}$. Thus, $V_N\,(i) = \tilde{J}^N_{\mu^{h+1}}(i)$ for all $i \in \mathcal{S}$.

Since $J^N_{\mu^{h+1}}(i)$ goes to infinity for some state i as N goes to infinity, the finite expected value of the terminal costs $J_{\mu^h}(x_N)$ become inconsequential for that state.

Proof of Theorem 5.2 (5/5)

Thus $V_l(i)$ will approach $J_{\mu^{h+1}}(i)$ in the limit and is infinite, which contradicts

$$V_0(i) \geq V_1(i) \geq V_2(i) \geq ... \geq V_l(i), \quad \forall i \in \mathcal{S}^+.$$

Therefore, μ^{h+1} is proper. As we initialize the PI with a proper policy μ^0, by induction, all policies μ^h in the PI are proper. Thus, by Theorem 5.1, Stage 1 of PI is guaranteed to have a unique solution.

As $l \to \infty$, $V_l \to J_{\mu^{h+1}}$ (since μ^{h+1} is proper) and therefore $J_{\mu^h}(i) \geq J_{\mu^{h+1}}(i)$ for all $i \in \mathcal{S}^+$. Since the number of stationary policies is finite, we will eventually have $J_{\mu^h} = J_{\mu^{h+1}}$ for some finite h.

Once we have converged, it follows from **Stage 2** that

$$J_{\mu^{h+1}}(i) = J_{\mu^h}(i) = \min_{\mathrm{u} \in \mathcal{U}(i)} \left(q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) J_{\mu^h}(j) \right), \; i \in \mathcal{S}^+.$$

This is the BE and thus we have converged to an optimal policy $\mu^* = \mu^h$ and the optimal cost is $J^* = J_{\mu^h}$.

Outline

Comparison between VI and PI

At first glance, PI looks very different from VI, but they actually have a lot in common. Let us rewrite the iteration in VI:

- **Stage 1** (*Value Update*):

$$V_{l+1}(i) = q\big(i, \mu^l(i)\big) + \sum_{j=1}^{n} P_{ij}(\mu^l(i)) V_l(j)\,, \quad \forall i \in \mathcal{S}^+.$$

- **Stage 2** (*Policy Improvement*):

$$\mu^{l+1}(i) = \underset{\mathrm{u} \in \mathcal{U}(i)}{\arg\min} \left(q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V_{l+1}(j) \right), \quad \forall i \in \mathcal{S}^+.$$

PI and VI differ in their respective Stage 1: PI performs a policy evaluation, which solves a system of linear equations and is equivalent to running the value update of VI an infinite number of times (Theorem 5.1).

Computational Complexity

Let p denote the maximum size of the control space $\mathcal{U}(i)$ for all $i \in \mathcal{S}^+$.

- Complexity of PI:
  - Stage 1 of PI involves solving a system of n linear equations in n unknowns, which has a computational complexity of $\mathcal{O}(n^3)$.
  - Stage 2 involves n minimizations over p possible control inputs, and evaluating the sum takes n steps. Thus, the complexity is $\mathcal{O}(n^2 p)$.

  $\Rightarrow$ Total complexity is $\mathcal{O}(n^2(n+p))$ *at each iteration*.
- Complexity of VI: involves n minimizations over p possible control inputs, and evaluating the sum takes n steps. Thus, the complexity is $\mathcal{O}(n^2 p)$ *at each iteration*.
- At each iteration, PI is more computationally expensive than VI. But remember: theoretically it takes an infinite number of iterations for VI to converge, whereas with PI, in the worst case the number of iterations is p^n.

Outline

Gauss-Seidel Update

In practice, VI would be implemented as follows:

```
For $i = 1$ to $n$:
    $\bar{V}(i) \leftarrow \min_{\mathrm{u} \in \mathcal{U}(i)} \left( q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V(j) \right),$
For $i = 1$ to $n$:
    $V(i) \leftarrow \bar{V}(i).$
```

We can use a **Gauss-Seidel Update** which updates the V in place as follows:

```
For $i = 1$ to $n$:
    $V(i) \leftarrow \min_{\mathrm{u} \in \mathcal{U}(i)} \left( q(i, \mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V(j) \right).$
```

Here we iterate one state at a time while incorporating into the computation the interim results.

Asynchronous Policy Iteration

With some assumptions, all combinations of the following steps will converge:

- Any number of value updates in between policy updates;
- Any number of states updated at each value update;
- Any number of states updated at each policy update.

Outline

Solving the Bellman Equation

- Value Iteration (VI)
- Policy Iteration (PI)
- Comparison between VI and PI
- Variants of VI and PI
- Connections to Linear Algebra
- Additional reading material

Connections to Linear Algebra

When performing policy evaluation (e.g. Stage 1 of PI), we solve a linear system of equations that has the following form:

$$J = G + PJ,$$

with $J := J_\mu \in \mathbb{R}^n$ whose i-th entry represents the cost-to-go at the corresponding state $i \in \mathcal{S}^+$, $G := (q(1, \mu(1)), \ldots, q(n, \mu(n))) \in \mathbb{R}^n$ is the stage cost vector, and $P \in \mathbb{R}^{n \times n}$ is the probability transition matrix under the proper policy μ, whose $(i, j)^{\text{th}}$ entry is $P_{ij}(\mu(i))$.

We can rewrite this as:

$$(I - P)J = G.$$

It is easy to see that there exists a unique solution for J if and only if $(I - P)$ is invertible.

Connections to Linear Algebra

We will now show that $(I - P)$ is guaranteed to be invertible when the policy is proper.

Note that $(I - P)$ is invertible if and only if P does not have eigenvalue(s) at 1. Furthermore, the (i, j)-th entry of P^2 is:

$$
\begin{aligned}
&\sum_{\xi \in \mathcal{S}^+} P_{i\xi}(\mu(i)) P_{\xi j}(\mu(\xi)) \\
&= \sum_{\xi \in \mathcal{S}^+} \Pr(x_2 = j | x_1 = \xi) \Pr(x_1 = \xi | x_0 = i) \\
&= \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j | x_1 = \xi) \Pr(x_1 = \xi | x_0 = i) \\
&= \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j | x_1 = \xi, x_0 = i) \Pr(x_1 = \xi | x_0 = i) \\
&= \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j, x_1 = \xi | x_0 = i) \\
&= \Pr(x_2 = j | x_0 = i) .
\end{aligned}
$$

Connections to Linear Algebra

If we continue on, the (i, j)-th entry of P^N is $\Pr(x_N = j | x_0 = i)$.

When the policy μ is proper, $\Pr(x_N = j | x_0 = i)$ goes to 0 as N goes to infinity for all $i, j \in \mathcal{S}^+$. Thus, the entire matrix P^N vanishes as N approaches infinity, which means that all eigenvalues of P must have modulus less than 1.

Therefore, there is no eigenvalue of P that is 1, and so the inverse of $(I - P)$ exists if the policy is proper.

Furthermore, note that:

$$\begin{aligned}
&(I - P)(I + P + P^2 + P^3 + ...) \\
&= (I + P + P^2 + P^3 + ...) - (P + P^2 + P^3 + P^4 + ...) \\
&= I,
\end{aligned}$$

since $P^N \to 0$ as $N \to \infty$ for proper policies. Thus:

$$(I - P)^{-1} = \sum_{k=0}^{\infty} P^k.$$

Connections to Linear Algebra

$$(I - P)^{-1} = \sum_{k=0}^{\infty} P^k$$

From the iteration in Theorem 5.1 (policy evaluation), we have the sequence:

$$\begin{aligned}
J_1 &= G + PJ_0 \\
J_2 &= G + PJ_1 = G + PG + P^2 J_0 \\
&\vdots \\
J_N &= (I + P + P^2 + P^3 + ... + P^{N-1})G + P^N J_0,
\end{aligned}$$

which gives $J_N \to (I - P)^{-1}G$ as $N \to \infty$, and is thus consistent with the claim of Theorem 5.1.

Outline

Solving the Bellman Equation

- Value Iteration (VI)
- Policy Iteration (PI)
- Comparison between VI and PI
- Variants of VI and PI
- Connections to Linear Algebra
- Additional reading material

Additional reading material

Another way to prove the convergence of the algorithms presented today is to use the theory of *contractive* operators. The key ingredients are:

1. An *operator*. Given a function $h(\cdot)$, an operator T maps $h(\cdot)$ to a new function $T[h](\cdot)$.
2. A *metric* $d(\cdot,\cdot)$ (see `https://en.wikipedia.org/wiki/Distance`) to compare how "close" the functions are. Specifically, we need a Hilbert space of functions, see `https://en.wikipedia.org/wiki/Hilbert_space`.
3. We say that an operator is *contractive* if there exists $\alpha \in [0,1)$ such that given any two functions $a(\cdot), b(\cdot)$, we have that $d(T[a](\cdot), T[b](\cdot)) \leq \alpha d(a(\cdot), b(\cdot))$.
4. The Banach Fixed Point Theorem, see `https://en.wikipedia.org/wiki/Banach_fixed-point_theorem`.

Additional reading material

These tools turn out to be very powerful to prove the convergence of various algorithms, beyond the ones described today. If you are curious to learn more, here are some additional references:

- Lecture Notes on Linear System Theory. John Lygeros & Federico A. Ramponi (class at ETH Zürich).
- Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Heinz H. Bauschke & Patrick L. Combettes.
- Abstract Dynamic Programming. Dimitri P. Bertsekas.