Chapter 4 - Dynamic Programming: Dec 26, 2021

Dynamic programming refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as an MDP.

- Classical DP models are limited because they assume a perfect model and are computationally expensive.
- We assume finite MDP's!
  - ↳ states, actions, rewards are finite (S, A, R)
  - ↳ dynamics are given by set of probabilities $p(s', r|s, a)$ for all $s \in S$, $a \in A(s)$, $r \in R$, $s' \in S^+$
    - S^+ is S plus the terminal state for episodic tasks

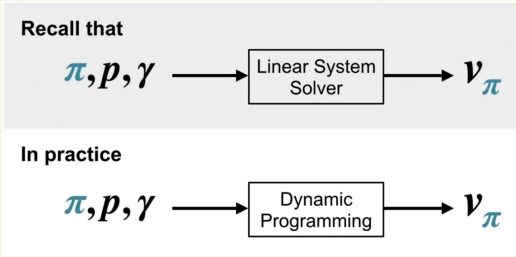
tl;dr: DP algorithms are obtained by turning Bellman equations into update rules for improving approximations of the desired value functions.

- ↳ Bellman equations define iterative algorithms for both policy evaluation and control.

4.1 - Policy Evaluation (The Prediction Problem):

We want to compute the state-value function v_π for an arbitrary policy π. This is called **policy evaluation** or **the prediction problem**.

$\pi \xrightarrow{\text{policy evaluation}} v_\pi$



Recall:

$$\begin{aligned} v_\pi(s) &\doteq E_\pi[G_t \mid S_t = s] \\ &= E_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \end{aligned}$$

※ E & U of v_π is guaranteed if $\gamma < 1$ or π guarantees that all states eventually terminate.

To find a system of $v_\pi(s)$ to solve:

1. Assign $v_0 =$ arbitrary real number
2. $v_{k+1}(s) \doteq E_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s]$ ※ Update rule

$$= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')]$$

※ This is the **iterative policy evaluation**.

Iterative Policy Evaluation:

✡ For estimating $V \approx v_\pi$

Input: π ✡ The policy to be evaluated.

Hyperparameter: θ ✡ Small threshold for determining accuracy of estimation, $\theta > 0$.

Initialize $V(s)$ for all $s \in S^+$ arbitrarily, except $V(\text{terminal}) = 0$.

Loop:

$\Delta \leftarrow 0$

For each $s \in S$:

$v \leftarrow V(s)$

$$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

break when $\Delta < \theta$

✡ In this case, $v \neq V(s)$ ever, but if $|v - V(s)| << \theta$, then good enough! $v \approx V(s) = v_\pi$

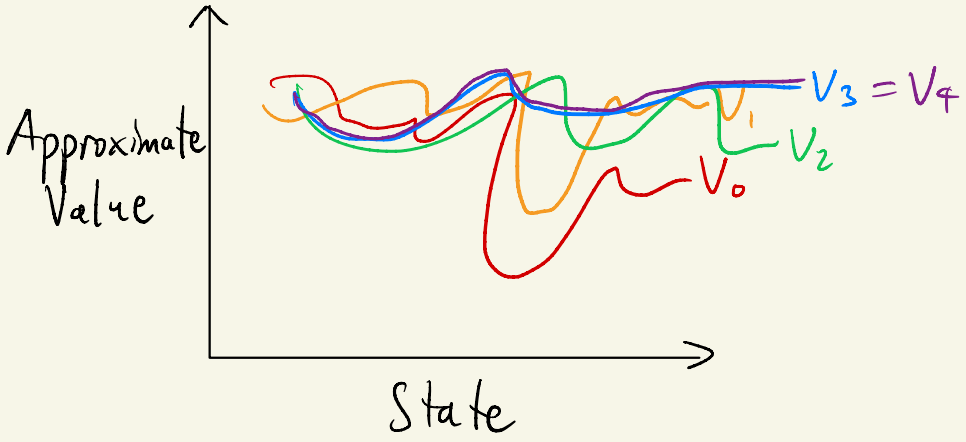
✡ Initialization and Policy Evaluation

Intuitively!

✡ V(s) update function

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma V_k(s')]$$

Start w/ random V_0 → we want $V(s) = V_\pi(s)$.



✡ Each iteration applies update function to every state $s \in S$. This is known as a **sweep**.

✡ If update leaves $V(s)$ unchanged (e.g. $V_3 = V_4$), then V_π is found.

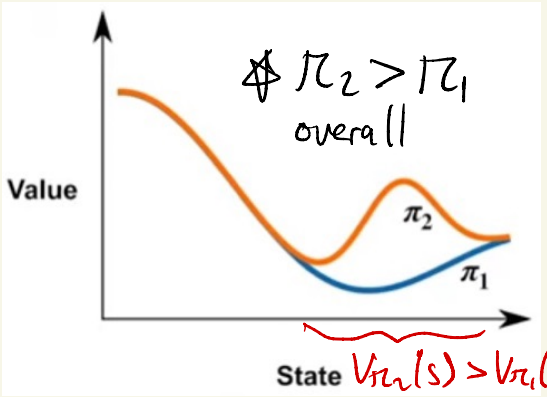
↳ This is because V_π obeys E&U, so it is unique.

✡ For any V_0, $V_k \to V_\pi$ as $k \to \infty$.

Control:

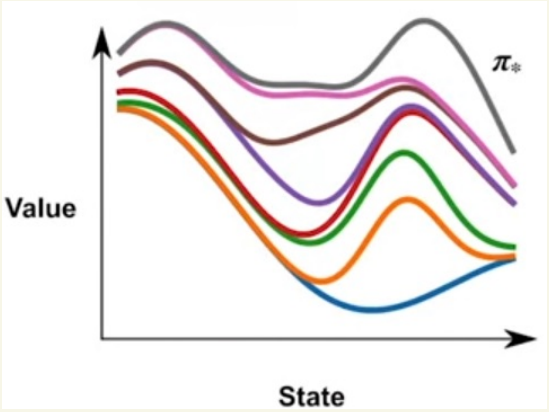
Control is the task of improving a policy.

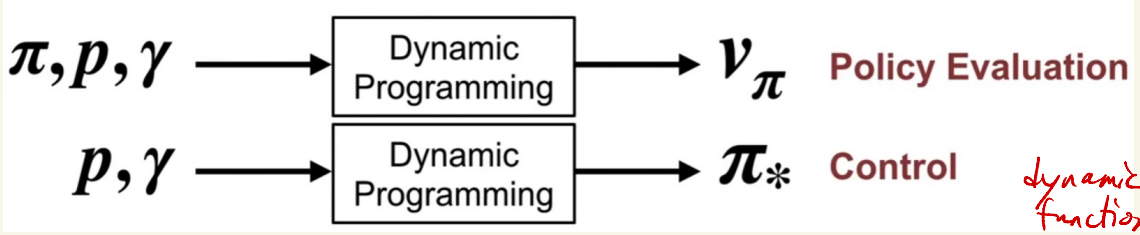
π_2 is strictly better than π_1 if $\pi_2 \geq \pi_1$ for all states and there is a section where $V_{\pi_2}(s) > V_{\pi_1}(s)$.



The purpose of a control task is to keep modifying the policy until it cannot be strictly better anymore.

When the policy reaches the optimal policy, the control task is complete.





DP can solve both tasks if we have access to p.

4.2 - Policy Improvement:

Recall:

$$q_\pi(s,a) \equiv \mathbb{E}[R_{t+1} + \gamma V_\pi(S_{t+1}) \mid S_t = s, A_t = a]$$
$$= \sum_{s',r} p(s', r \mid s, a)[r + \gamma V_\pi(s')]$$

Policy Improvement Theorem: ✡✡

Let π and π' be any pair of deterministic policies s.t. for all $s \in S$:

$$q_\pi(s, \pi'(s)) \geq V_\pi(s) \quad \text{✡4.6}$$

Then $\pi' \geq \pi$.

i.e. It must obtain greater or equal expected return from all states $s \in S \rightarrow V_{\pi'}(s) \geq V_\pi(s)$ ✡4.7

Proof:

$$\begin{aligned}
v_\pi(s) &\leq q_\pi(s, \pi'(s)) \\
&= \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = \pi'(s)] && \text{(by (4.6))} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \mid S_t = s] && \text{(by (4.7))} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_{t+1}, A_{t+1} = \pi'(S_{t+1})] \mid S_t = s] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \mid S_t = s] \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(S_{t+3}) \mid S_t = s] \\
&\vdots \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \mid S_t = s] \\
&= v_{\pi'}(s).
\end{aligned}$$

Policy Improvement:

Policy improvement is the process of making a new policy that improves on an original one by making it **greedy** w.r.t. the value function of the original policy.

eg. Greedy policy π' ⇔ Policy Improvement

$$\begin{aligned}\pi'(s) &\doteq \arg\max_a q_\pi(s,a) \\ &= \arg\max_a \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \arg\max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_\pi(s')]\end{aligned}$$

If $\pi' = \pi$:

Then $v_\pi = v_{\pi'}$ for all $s \in S$, and it follows that:

$$v_{\pi'}(s) = \max_a \mathbb{E}[R_{t+1} + \gamma v_{\pi'}(S_{t+1}) \mid S_t = s, A_t = a]$$

$$\max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_{\pi'}(s')]$$

which is the same as the Bellman Optimality eqn:

$$v_*(s) = \max_a \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma v_*(s')]$$

$\therefore$ Both π and π' are optimal policies.

So how do we actually use DP to find π_*?

↳ We have iterative policy evaluation for v_π, but how do we incorporate changing π?

4.3 - Policy Iteration:

$$\pi_0 \xrightarrow{E} v_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} v_{\pi_1} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} v_*$$

E: policy evaluation

I: policy improvement

Since a finite MDP has a finite number of policies, this process must converge to an optimal policy in a finite number of iterations.

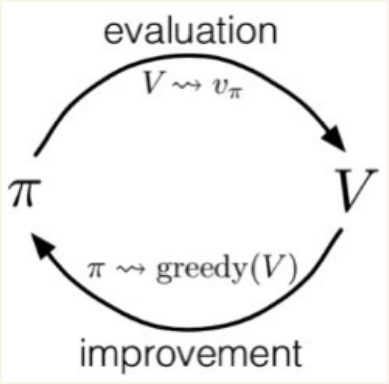
Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

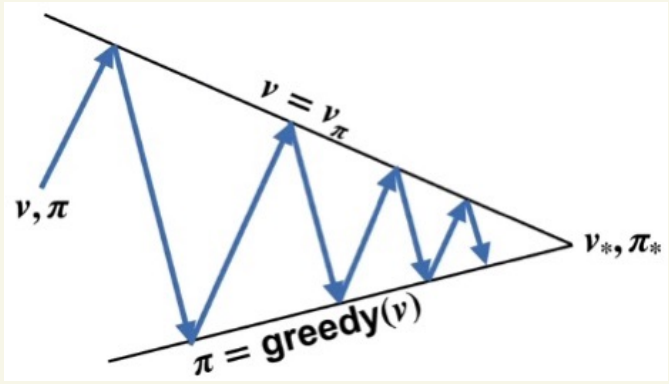
1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

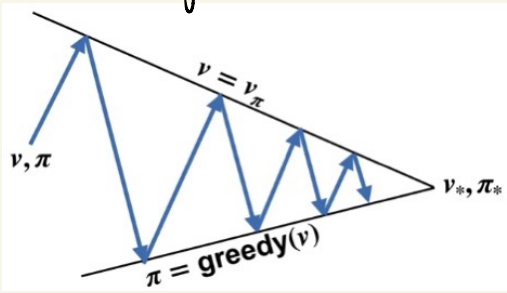
Intuitively!

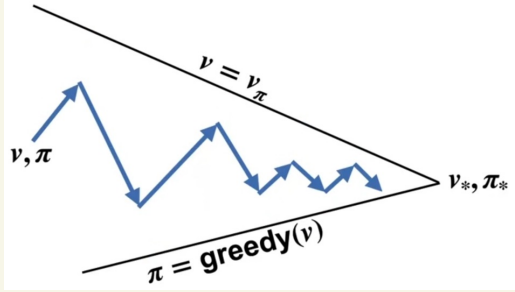




Flexibility of the Policy Iteration Framework!

Imagine this becomes this





- Each policy improvement step makes policy a little more greedy, but not totally greedy.
  - ↳ We should still get π_*!

Generalized policy iteration refers to all the different combinations of policy evaluation and improvement.

↳ One algorithm/combination is value iteration.

4.4 - Value Iteration:

Policy evaluation is stopped after one sweep. $\{V_k\}$ is still shown to converge to V_* under the same conditions.

$$V_{k+1}(s) \doteq \max_a E_\pi[R_{t+1} + \gamma V_k(S_{t+1}) \mid S_t = s]$$
$$= \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V_k(s')]$$

Policy Evaluation:

$$V_{k+1}(s) \doteq E_\pi[R_{t+1} + \gamma V_k(S_{t+1}) \mid S_t = s]$$
$$= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)[r + \gamma V_k(s')]$$

for value iteration

Value Iteration, for estimating $\pi \approx \pi_*$

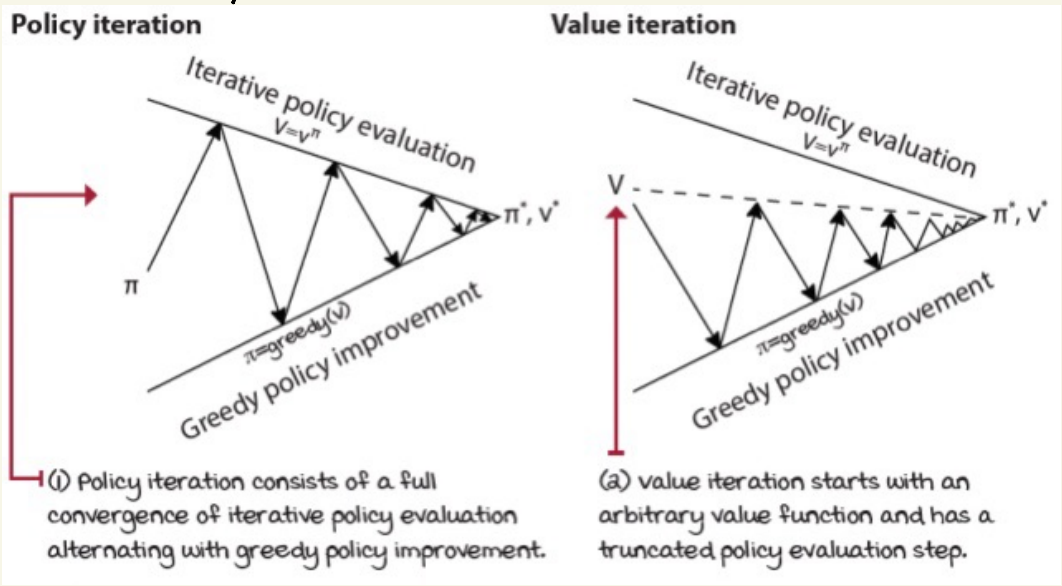
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $\quad v \leftarrow V(s)$
| $\quad V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$
| $\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Update using action that maximizes current value estimate

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \arg\max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$

Intuitively!



(i) Policy iteration consists of a full convergence of iterative policy evaluation alternating with greedy policy improvement.

(a) value iteration starts with an arbitrary value function and has a truncated policy evaluation step.

Summary:

Policy evaluation is the task of determining the state-value function v_π for a particular policy π.

Iterative Policy Evaluation:

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\left[r + \gamma v_k(s')\right]$$

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Control refers to the task of improving a policy.

Policy Improvement Theorem:

$$\pi'(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

$\Rightarrow \pi' > \pi$ unless $\pi = \pi_*$ (optimal)

Policy Iteration:

Used to find π_* (and v_*).

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Generalized Policy Iteration:

Where evaluation/improvement steps need not run to completion.

↳ Examples are policy iteration, value iteration.