Dynamic Programming Algorithms

Uses the Bellman equations to define iterative algorithms for both policy evaluation and control

- Policy evaluation defines state value function v_π for a particular policy π
- Policy evaluation improves a particular policy by modifying it to make a better policy. We can continuously improve the policy until it is no longer possible to improve, which means the last policy must be equal to the optimal policy

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

Input π, the policy to be evaluated

$V \leftarrow \vec{0}, V' \leftarrow \vec{0}$

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in \mathcal{S}$:

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

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

$V \leftarrow V'$

Until $\Delta \leftarrow \theta$

Output $V \approx v_\pi$

3. **Policy Improvement**

$policy - stable \leftarrow true$

For each $s \in \mathcal{S}$:

$old - action \leftarrow \pi(s)$

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

If $old - action \neq \pi(s)$, then $policy - stable = false$

If $policy - stable$, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Generalized Policy Iteration

1. **Value iteration**

   Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of the estimation.

   Initialize $V(s)$, for all $s \in S^+$, arbitrarily except that $V(terminal) = 0$

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in \mathcal{S}$:

$v \leftarrow V(s)$

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

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

$V \ \leftarrow V'$

Until $\Delta \leftarrow \theta$

Output a deterministic policy $\pi \approx \pi_*$, such that:

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

2. Alternatives

- Monte Carlo method: averaging the value on a set of results from the policy π
- Bootstrapping: using the previous value estimate to improve the current value estimate
- Brute-force search: evaluating every possible deterministic policy one at the time and selecting the one with the highest value