Dynamic Programming & Tabular Methods of RL

Self-Adaptive & Learning Algorithms

Milan R. Rapaić

Chair of Automatic Control
Computing and Control Department
Faculty of Technical Sciences
University of Novi Sad
Novi Sad • Serbia

November 15, 2023

Outline

Bellman Equation & Dynamic Programming

What is it? What problem does it solve?

Bellman Equation

Bellman Equation (in its various forms) is the equation which, if solved, would specify optimal action of every possible state. Any policy satisfying the Bellman Equation is an optimal policy.

Dynamic Programming

The term dynamic programming is often used in somewhat different contexts, however in the present context it will be used solely to indicate a group of techniques for finding an optimal policy by solving Bellman Equation assuming that the model of the underlying Markov Decision Process (MDP) is completely known.

Markov Decission Processes

Just to repeat a few points...

MDP is a model of the environment (process). It tells us ...

- ... how the state of the process is changing in reaction to the applied (control) actions
- ... what observations the agent (controller) may receive

Deterministic MDP

$$s^+ = f(s, a)$$

$$r = h(s, a)$$

Stochastic MDP

$$p(s^+, r|s, a) = \mathbb{P}\left\{S^+ = s^+, R = r \middle| S = s, A = a\right\}$$

Bellman Equation

... and its various beautiful forms :)

Bellman Equation for State Values

$$v^*(s) = \max_{a \in \mathcal{A}} \{h(s,a) + \gamma v^*(f(s,a))\}$$

$$v^*(s) = \max_{a \in \mathcal{A}} \sum_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r|s,a) \left[r + \gamma v^*(s^+)\right]$$

Bellman Equation for State-Action Values

$$q^*(s,a) = h(s,a) + \gamma \max_{a^+ \in \mathcal{A}} q^*(f(s,a), a^+)$$

$$q^*(s,a) = \sum_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r|s,a) \left[r + \gamma \max_{a^+ \in \mathcal{A}} q^*(s^+, a^+)\right]$$

Due to the appearance of the `max` operator, the Bellman equation is a set of coupled nonlinear equations. Direct, closed-form solution is no longer possible, not even in principle.

Introduction to Tabular Methods

Tabular methods of reinforcement learning work well for problems with finite and relatively small state-action spaces. In these cases, it is reasonable to represent state value function v as a vector $\mathbf{v}$, and the state-action function q as a matrix $\mathbf{Q}$.

Value Iteration

Introduction to Value Iteration methods

Successive approximations of the value function...

Value iteration algorithms exploit the fact that the Bellman Equation has a form of a fixed-point formula

$$\mathbf{v} = \mathcal{T}_v \{\mathbf{v}\}$$

$$\mathbf{Q} = \mathcal{T}_Q \{\mathbf{Q}\}$$

These methods start from an initial guess ($\mathbf{v}^0$ or $\mathbf{Q}^0$) and proceed to construct better-and-better approximations by transforming Bellman identity into an iterative procedure

$$\mathbf{v}^{k+1} \leftarrow \mathcal{T}_v \left\{\mathbf{v}^k\right\}$$

$$\mathbf{Q}^{k+1} \leftarrow \mathcal{T}_Q \left\{\mathbf{Q}^k\right\}$$

The procedure stops as soon as two successive approximations become close enough (i.e. if $\|\mathbf{v}^{k+1} - \mathbf{v}^k\| \leq \varepsilon$ or $\|\mathbf{Q}^{k+1} - \mathbf{Q}^k\| \leq \varepsilon$, for some sufficiently small $\varepsilon > 0$.)

Finally, the greedy policy with respect to the found value function is considered as a sufficiently good approximation to the optimal policy.

Algorithm of Successive Approximations

What is it and what problem does it solve?

Fixed point of a function

Consider a function F mapping a set $\mathbb{X}$ into itself: $F : \mathbb{X} \to \mathbb{X}$. A fixed point of F is any $x \in \mathbb{X}$ such that

$$x = F(x) \ .$$

Algorithm of Successive Approximations

Given an arbitrary $x^0 \in \mathbb{X}$, the sequence of points $x^{k+1} = F(x^k)$ converges to a fixed point of F (under mild conditions).

Value Iteration for the State Value Function

... in the deterministic case

Algorithm 1 v-iteration algorithm

Require: the model (f, h), γ, $\varepsilon > 0$

Require: $\mathbf{v}^0$

  $\mathbf{v} \leftarrow \mathbf{v}^0$

  repeat

    for $s \in \mathcal{S}$ **do**

      $\mathbf{v}(s)^+ \leftarrow \max_{a \in \mathcal{A}} [h(s,a) + \gamma \mathbf{v}(f(s,a))]$

    end for

    $\delta \leftarrow \|\mathbf{v}^+ - \mathbf{v}\|_\infty$

    $\mathbf{v} \leftarrow \mathbf{v}^+$

  until $\delta > \varepsilon$

  return $\mathbf{v}$, $\pi = \pi_{\mathbf{v}}^{\text{greedy}}$

Value Iteration for the State Value Function

... in the stochastic case

Algorithm 2 v-iteration algorithm

Require: the model (p), γ, $\varepsilon > 0$

Require: $\mathbf{v}^0$

$\mathbf{v} \leftarrow \mathbf{v}^0$

repeat

for $s \in \mathcal{S}$ **do**

$$\mathbf{v}(s)^+ \leftarrow \max_{a \in \mathcal{A}} \sum_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r | s, a) \left[r + \gamma \mathbf{v}(s^+) \right]$$

end for

$\delta \leftarrow \| \mathbf{v}^+ - \mathbf{v} \|_\infty$

$\mathbf{v} \leftarrow \mathbf{v}^+$

until $\delta > \varepsilon$

return $\mathbf{v}$, $\pi = \pi_{\mathbf{v}}^{\text{greedy}}$

Value Iteration for the State-Action Value Function

... in the deterministic case

Algorithm 3 Q-iteration algorithm

Require: the model $(f,\, h)$, γ, $\varepsilon > 0$

Require: $\mathbf{Q}^0$

  $\mathbf{Q} \leftarrow \mathbf{Q}^0$

  repeat

    for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**

      $\mathbf{Q}(s,a)^+ \leftarrow h(s,a) + \gamma \max_{a^+ \in \mathcal{A}} \mathbf{Q}(f(s,a), a^+)$

    end for

    $\delta \leftarrow \|\mathbf{Q}^+ - \mathbf{Q}\|_\infty$

    $\mathbf{Q} \leftarrow \mathbf{Q}^+$

  until $\delta > \varepsilon$

  return $\mathbf{Q}$, $\pi = \pi^{\text{greedy}}_{\mathbf{Q}}$

Value Iteration for the State-Action Value Function

... in the stochastic case

Algorithm 4 Q-iteration algorithm

Require: the model (p), γ, $\varepsilon > 0$

Require: $\mathbf{Q}^0$

  $\mathbf{Q} \leftarrow \mathbf{Q}^0$

  repeat

    for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**

      $\mathbf{Q}(s,a)^+ \leftarrow \sum_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r|s,a) \left[r + \gamma \max_{a^+ \in \mathcal{A}} \mathbf{Q}(s^+, a^+) \right]$

    end for

    $\delta \leftarrow \|\mathbf{Q}^+ - \mathbf{Q}\|_\infty$

    $\mathbf{Q} \leftarrow \mathbf{Q}^+$

  until $\delta > \varepsilon$

  return $\mathbf{Q}$, $\pi = \pi^{\text{greedy}}_{\mathbf{Q}}$

Policy Iteration

Introduction to Policy Iteration methods

Policy iteration methods work directly on policies: they start from an initial policy π^0 and proceed to construct better-and-better policies.

In each iteration of the policy iteration procedure, a value function is constructed based on the given policy (this operation is relatively cheap). Then, the greedy policy with respect to that value function is used as the policy in the subsequent iteration.

$$\pi^0 \xrightarrow{evaluate} \mathbf{Q}^0 \xrightarrow{greedy} \pi^1 \xrightarrow{evaluate} \mathbf{Q}^1 \xrightarrow{greedy} \dots$$

The procedure stops when two identical consecutive policies are encountered. Analogous procedure can be used to iterate over policies using V-functions.

Policy Iteration

Policy Iteration using V-functions

Iterative evaluation of State Value Function

... in the deterministic case

Algorithm 5 Evaluation of the state value function for a given policy

Require: the model (f, h), γ, $\varepsilon > 0$, π

Require: $\mathbf{v}_\pi^0$

$\mathbf{v}_\pi \leftarrow \mathbf{v}_\pi^0$

repeat

for $s \in \mathcal{S}$ **do**

$\mathbf{v}_\pi(s)^+ \leftarrow h(s, \pi(s)) + \gamma \mathbf{v}_\pi(f(s, \pi(s)))$

end for

$\delta \leftarrow \|\mathbf{v}_\pi^+ - \mathbf{v}_\pi\|_\infty$

$\mathbf{v}_\pi \leftarrow \mathbf{v}_\pi^+$

until $\delta > \varepsilon$

return $\mathbf{v}_\pi$

Iterative evaluation of the State Value Function

... in the stochastic case

Algorithm 6 Evaluation of the state value function for a given policy

Require: the model (p), γ, $\varepsilon > 0$, π

Require: $\mathbf{v}_\pi^0$

$\mathbf{v}_\pi \leftarrow \mathbf{v}_\pi^0$

repeat

for $s \in \mathcal{S}$ **do**

$$\mathbf{v}_\pi(s)^+ \leftarrow \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{\substack{s^+\in\mathcal{S} \\ r\in\mathcal{R}}} p(s^+, r|s, a) \left[r + \gamma \mathbf{v}(s^+)\right]$$

end for

$\delta \leftarrow \|\mathbf{v}_\pi^+ - \mathbf{v}_\pi\|_\infty$

$\mathbf{v}_\pi \leftarrow \mathbf{v}_\pi^+$

until $\delta > \varepsilon$

return $\mathbf{v}_\pi$

Obtaining Greedy Policy for the given V-Function

Algorithm 7 Evaluation of the Greedy Policy (Deterministic Case)

Require: the model $(f,\, h)$, γ, $\mathbf{v}$

  for $s \in \mathcal{S}$ **do**

    $\pi(s) \leftarrow \operatorname{argmax}_{a \in \mathcal{A}} \left[h(s,a) + \gamma \mathbf{v}(f(s,a)) \right]$

  end for

  return π

Algorithm 8 Evaluation of the Greedy Policy (Stochastic Case)

Require: the model (p), γ, $\mathbf{v}$

  for $s \in \mathcal{S}$ **do**

    $\pi(s) \leftarrow \operatorname{argmax}_{a \in \mathcal{A}} \sum_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r | s, a) \left[r + \gamma \mathbf{v}(s^+) \right]$

  end for

  return π

Notes on the Greedy Policy Evaluation Procedure

Greedy policy is not unique!

The `argmax` operation is multivalued. We are free to arbitrarily choose one (any one) of the maximum-achieving actions. We are also free to choose a stochastic policy in which one chooses all maximum-achieving actions with a certain probability (or just a subset of them). All such policies are greedy with respect to the given value function.

Greedy policy can always be selected so that it is deterministic!

Since in all cases we may select only one of all maximum-achieving actions in the greedy policy, such policy can always be selected as a deterministic policy.

Policy Iteration for the State Value Function

Algorithm 9 Policy iteration algorithm using $\mathbf{v}$

Require: `the model`, γ

Require: π^0

$\pi \leftarrow \pi^0$

repeat

Evaluate $\mathbf{v}_\pi$ using `the model`.

$\pi^+ \leftarrow \pi^{\text{greedy}}_{\mathbf{v}_\pi}$

TERMINATE $\leftarrow \pi^+ = \pi$

$\pi \leftarrow \pi^+$

until TERMINATE

return π

Policy Iteration

Policy Iteration using Q-Function

Evaluation of the State-Action Value Function

... in the deterministic case

Algorithm 10 Evaluation of the state-action value function for a given π

Require: the model $(f,\, g)$, γ, $\varepsilon > 0$

Require: $\mathbf{Q}_\pi^0$

$\mathbf{Q}_\pi \leftarrow \mathbf{Q}_\pi^0$

repeat

for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**

$\mathbf{Q}_\pi(s,a)^+ \leftarrow h(s,a) + \gamma \mathbf{Q}_\pi(f(s,a), \pi(f(s,a)))$

end for

$\delta \leftarrow \|\mathbf{Q}_\pi^+ - \mathbf{Q}_\pi\|_\infty$

$\mathbf{Q}_\pi \leftarrow \mathbf{Q}_\pi^+$

until $\delta > \varepsilon$

return $\mathbf{Q}_\pi$, $\pi = \pi^{\text{greedy}}_{\mathbf{Q}_\pi}$

Evaluation of the State-Action Value Function

... in the stochastic case

Algorithm 11 Evaluation of the state-action value function for a given π

Require: the model (p), γ, $\varepsilon > 0$

Require: $\mathbf{Q}_\pi^0$

$\mathbf{Q}_\pi \leftarrow \mathbf{Q}_\pi^0$

repeat

for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**

$$\mathbf{Q}_\pi(s,a)^+ \leftarrow \sum\nolimits_{\substack{s^+ \in \mathcal{S} \\ r \in \mathcal{R}}} p(s^+, r|s,a)\left[r + \gamma \mathbf{Q}_\pi(s^+, \pi(s^+))\right]$$

end for

$\delta \leftarrow \|\mathbf{Q}_\pi^+ - \mathbf{Q}_\pi\|_\infty$

$\mathbf{Q}_\pi \leftarrow \mathbf{Q}_\pi^+$

until $\delta > \varepsilon$

return $\mathbf{Q}_\pi$, $\pi = \pi^{\text{greedy}}_{\mathbf{Q}_\pi}$

Obtaining Greedy Policy for the given Q-Function

... identical in both deterministic and stochastic case!

Algorithm 12 Evaluation of the Greedy Policy (Deterministic Case)

Require: $\mathbf{Q}$

  for $s \in \mathcal{S}$ **do**

    $\pi(s) \leftarrow \operatorname{argmax}_{a \in \mathcal{A}} \mathbf{Q}(s, a)$

  end for

  return π

Notes on the Greedy Policy Evaluation Procedure

Greedy policy is not unique and may be always selected as deterministic!

We do not actually need the model, once Q is known!

When evaluating greedy policies based on known Q function, MDP model is not needed.

Policy Iteration for the State-Action Value Function

Algorithm 13 Policy iteration algorithm using $\mathbf{Q}$

Require: `the model`, γ

Require: π^0

$\pi \leftarrow \pi^0$

repeat

Evaluate $\mathbf{Q}_\pi$ using `the model`

$\pi^+ \leftarrow \pi^{\text{greedy}}_{\mathbf{Q}_\pi}$

TERMINATE $\leftarrow \pi^+ = \pi$

$\pi \leftarrow \pi^+$

until TERMINATE

return π

Further Reading

For further reading please consult [Sutton and Barto, 2018], **Chapter 4**.

References I


Sutton, R. S. and Barto, A. G. (2018).
Reinforcement Learning: An Introduction.
The MIT Press, second edition.