Value-based methods: overview

The simplest model-based approach are *Value-based* methods.

They revolve around the idea of

- assigning a *State Value function $\statevalfun_\pi(\state)$ to each state $\state \in \States$
$$\statevalfun_\pi : \States \to \Reals$$
- $\statevalfun_\pi(\state)$ is an approximation of
$$\E_\pi(G_| \stateseq_= \state)$$
the expected return achievable from state $\state$

Given $\statevalfun_\pi$, the optimal deterministic policy is

$$\pi^*(\state) = \argmax_{\act} \sum_{\transp(\state', \rew|\state, \act) \neq 0} \statevalfun_\pi(\state')$$

- From state $\state$
- Choose the action $\act$
- that results in next state $\state'$
- with maximal $\statevalfun_\pi(\state')$

Note: the $\argmax$ results in a *deterministic* policy.

Using a value-based method is practical only if we can discover the state value function $\backslash\mathrm{statevalfun}$, which is initially unknown.

Most methods are iterative in nature and create a sequence of increasingly accurate approximations of $\backslash\mathrm{statevalfun}$

$$\backslash\mathrm{statevalfun}_{\pi,0} \ldots \backslash\mathrm{statevalfun}_{\pi,k} \ldots$$

In the limit

$$\lim_{k\to\infty} \backslash\mathrm{statevalfun}_{\pi,k} = \backslash\mathrm{statevalfun}_{\pi}$$

As we obtain an improved approximation $\backslash\mathrm{statevalfun}_{\pi,k+1}$

- we may reflect this improved knowledge by updating the policy

Thus, we periodically improve the policy based on improved approximations of $\backslash\mathrm{statevalfun}_{\pi}$

This results in a sequence of increasingly accurate approximations of the policy π

$$\pi_0, \ldots, \pi_p, \ldots$$

which hopefully converges to π^*.

We will show two broad classes of Value-based methods

- Model-based
- Model-free

```
Value-Based Methods
├── Model-Based
│   └── DP Methods
│       ├── Value Iteration
│       └── Policy Iteration
│
└── Model-Free
    ├── Monte Carlo (MC)
    ├── Temporal Difference (TD)
    └── Q-Learning
```

A common characteristic of both Model-based and Model-free Value methods

- use of *Dynamic Programming* type solutions.

In Dynamic Programming

- there is a *Bellman update* equation
- which relates the value of a state $\backslash\mathrm{state}$
- to the values of each of the potential successor states $\backslash\mathrm{state}'$
    - which the environment may return in response to the Learner taking an action in state $\backslash\mathrm{state}$

Information flowing from successor to predecessor state is called a *backup*

The Bellman update equation asserts that

$$\statevalfun_\pi(\state)$$

can be derived from

- the immediate reward $\rew(\state, \act, \state')$
  - received by taking action $\act$ in state $\state$ and transitioning to state $\state'$
- and the discounted (by γ) value of the successor state $\state'$

$$\statevalfun_\pi(\state')$$

The main difference between Model-based and Model-free methods

- a Model-based method does *not need to actively gather* experience
    - if the model is completely known
    - all necessary information is available without actively interacting with the Environment
        - $\backslash\mathrm{transp}(\backslash\mathrm{state}' \mid \backslash\mathrm{state}, \backslash\mathrm{act})$ is known $\forall \backslash\mathrm{state}, \backslash\mathrm{state}' \in \backslash\mathrm{States}, \backslash\mathrm{act} \in \backslash\mathrm{Actions}$
- a Model-free method *must* actively interact with the Environment
    - gathers a *sample* from $\backslash\mathrm{transp}(\backslash\mathrm{state}' \mid \backslash\mathrm{state}, \backslash\mathrm{act})$ for each $\backslash\mathrm{state}, \backslash\mathrm{act}$ pair in the episode
    - through multiple episodes
        - the *effect* of the true $\backslash\mathrm{transp}(\backslash\mathrm{state}' \mid \backslash\mathrm{state}, \backslash\mathrm{act})$ is derived

Method	Category	Model-based?	Update Equation	Key Characteristics	On-/Off-policy	Convergence Target	
Value Iteration	Value-based	Yes	(V(s) \leftarrow \max*a* *sum*{s'} P(s'	s,a)[R(s,a,s') + \gamma V(s')])	Uses full model; synchronous updates	N/A	(V^), (*pi*^)
Policy Iteration	Policy-based	Yes	Eval: (V^\pi(s) \leftarrow \sum_{s'} P(s'	s,\pi(s))[R + \gamma V^\pi(s')]) Improve: (\pi(s) \leftarrow \arg\max_a Q^\pi(s,a))	Alternates evaluation/policy update	N/A	(V^), (*pi*^)

Method	Category	Model-based?	Update Equation	Key Characteristics	On-/Off-policy	Convergence Target
TD(0)	Value-based	No	(V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)])	Incremental via samples, bootstraps	On-policy	(V^\pi)
Q-learning	Value-based	No	(Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)])	Off-policy, learns optimal Q	Off-policy	(Q^), (\pi^)
Monte Carlo	Value-based	No	(V(s) \leftarrow V(s) + \alpha[G - V(s)]), where (G) = episode return	No bootstrapping; full episodes	On-policy	(V^\pi)

In [2]:

```
print("Done")
```

```
Done
```