Value Function Methods Notes

1 Prelude

$A^\pi(s_t, a_t)$: how much better is a_t than the average action according to π.
$\arg\max_{a_t} A^\pi(s_t, a_t)$: best action from s_t, if we then follow π, and is at least as good as any $a_t \sim \pi(a_t|s_t)$ (since in the worst case, every action selected according to the policy will have equal chance), we can then construct a new policy π' so that:

$$\pi'(a_t|s_t) = \begin{cases} 1, & \text{if } a_t = \arg\max_{a_t} A^\pi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

2 Policy Iteration

Algorithm 1 Policy Iteration Algorithm

evaluate $A^\pi(s, a)$

construct policy $\pi'(a_t|s_t) = \begin{cases} 1, & \text{if } a_t = \arg\max_{a_t} A^\pi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases}$

set $\pi \leftarrow \pi'$

As before, we can evaluate $A^\pi(s, a) = r(s, a) + \gamma E[V^\pi(s')] - V^\pi(s)$.

2.1 Dynamic programming

Assume $p(s'|s, a)$ and s and a are both discrete (and small). We can store the value function in a table. We can also bootstrap the update:

$$V^\pi(s) \leftarrow E_{a\sim\pi(a|s)}[r(s, a) + \gamma E_{s'\sim p(s'|s,a)}[V^\pi(s')]] \tag{2}$$

Because the policy is deterministic, we can plug it back into the equation:

$$V^\pi(s) \leftarrow [r(s, \pi(s)) + \gamma E_{s'\sim p(s'|s,\pi(s))}[V^\pi(s')]] \tag{3}$$

2.2 Even simpler dynamic programming

$$\pi'(a_t|s_t) = \begin{cases} 1, & \text{if } a_t = \arg\max_{a_t} A^\pi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

thus we have:

$$A^\pi(s, a) = r(s, a) + \gamma E[V^\pi(s')] \tag{5}$$

and $\arg\max_{a_t} A^\pi(s_t, a_t) = \arg\max_{a_t} Q^\pi(s_t, a_t)$ by removing the last term since it does not depend on both s_t and a_t. We can skip the policy and compute values directly:

Algorithm 2 Value Iteration Algorithm

set $Q(s,a) \leftarrow r(s,a) + \gamma E[V(s')]$
set $V(s) \leftarrow \max_a Q(s,a)$

3 Fitted Value Iteration

Algorithm 3 Value Iteration Algorithm

set $y_i \leftarrow \max_{a_i}(r(s_i,a_i) + \gamma E[V_\phi(s_i')])$
set $\phi \leftarrow \arg\min_\phi \frac{1}{2}\sum_i ||V_\phi(s_i) - y_i||^2$

This requires us to know the outcomes for different actions. We can approximate $E[V(s')] \approx \max_{a'} Q_\phi(s_i', a_i')$, and we will get:

Algorithm 4 Full Fitted Q-Iteration

collect dataset $\{(s_i, a_i, s', r_i)\}$ using some policy (parameters: size N, collection policy)
set $y_i \leftarrow r(s_i,a_i) + \gamma \max_{a_i'} Q_\phi(s_i', a_i')$
set $\phi \leftarrow \arg\min_\phi \frac{1}{2}\sum_i ||Q_\phi(s_i,a_i) - y_i||^2$ (parameters: gradient steps S)

in which steps 2 and 3 can be repeated K times before collecting new data.
In tabular case, the max term improves the policy. Most guarantees are lost when we leave the tabular case (e.g. use neural networks)

3.1 Online Q iteration algorithm

Algorithm 5 Online Q-Iteration

take some action a_i and observe $\{(s_i, a_i, s', r_i)\}$
set $y_i \leftarrow r(s_i,a_i) + \gamma \max_{a_i'} Q_\phi(s_i', a_i')$
set $\phi \leftarrow \phi - \alpha \frac{dQ_\phi}{d\phi}(s_i,a_i)(Q_\phi(s_i,a_i) - y_i)$

which is off-policy. Always taking the greedy policy might be bad especially at the start, we can use ϵ-greedy:

$$\pi(a_t|s_t) = \begin{cases} 1-\epsilon & \text{if } a_t = \arg\max_{a_t} Q_\phi(s_t, a_t) \\ \epsilon/(|\mathcal{A}|-1) & \text{otherwise} \end{cases} \tag{6}$$

We can also use Boltzmann-exploration $\pi(a_t|s_t) \propto \exp(Q_\phi(s_t, a_t))$

4 Value function learning theory

Algorithm 6 Value Iteration Algorithm

set $Q(s,a) \leftarrow r(s,a) + \gamma E[V(s')]$
set $V(s) \leftarrow \max_a Q(s,a)$

Define an operator $\mathcal{B} : \mathcal{B}V = \max_a r_a + \gamma \mathcal{T}_a V$, in which r_a is the stacked vector of rewards at all states for action a, and $\mathcal{T}$ denotes the matrix of transitions for action a such that $\mathcal{T}_{a,i,j} = p(s' = i|s = j, a)$. V^* is a fixed point of $\mathcal{B}$:

$$V^*(s) = \max_a r(s,a) + \gamma E[V^*(s')] \tag{7}$$

so $V^* = \mathcal{B}V^*$, which always exists, is unique and is optimal policy. We can prove that value iteration reaches V^* because $\mathcal{B}$ is a contraction: for any V and $\bar{V}$, we have $||\mathcal{B}V - \mathcal{B}\bar{V}||_\infty \leq \gamma||V - \bar{V}||_\infty$.

4.1 Non-tabular value function learning

Algorithm 7 Fitted Value Iteration Algorithm

set $y_i \leftarrow \max_{a_i}(r(s_i, a_i) + \gamma E[V_\phi(s_i')])$
set $\phi \leftarrow \arg\min_\phi \frac{1}{2}\sum_i ||V_\phi(s_i) - y_i||^2$

we construct a hypothesis set of all networks Ω in which:

$$V' \leftarrow \arg\min_{V' \in \Omega} \frac{1}{2} \sum ||V'(s) - (\mathcal{B}V)(s)||^2 \tag{8}$$

Define new operator Π : $\Pi V = \arg\min_{V' \in \Omega} \frac{1}{2} \sum ||V'(s) - V(s)||^2$, and we can rewrite the expression of V' as $V \leftarrow \Pi\mathcal{B}V$. Π is a projection onto Ω (in terms of l_2 norm).
$\mathcal{B}$ is a contraction w.r.t. ∞-norm, and we have $||\mathcal{B}V - \bar{\mathcal{B}}V||_\infty \leq \gamma||V - \bar{V}||_\infty$
Π is a contraction w.r.t. l_2-norm (Euclidean distance), and we have $||\Pi V - \bar{V}||^2 \leq \gamma||V - \bar{V}||^2$
but $\Pi\mathcal{B}$ is not a contraction of any kind. Therefore, fitted value iteration does not converge in general. Similar thing applied to fitted Q-iteration.