Reinforcement Learning Formula Sheet

Eddie Guo

Multi-Armed Bandit Problem

Expected reward of action a: $q_*(a) \equiv \mathbb{E}[R_t \mid A_t = a]$

Estimate of $q_*(a)$ at time t: $Q_t(a) \equiv \dfrac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$

Optimization: $Q_{n+1} = Q_n + \dfrac{1}{n}(R_n - Q_n)$

$\lim_{t\to\infty} Q_t(a) = q_*(a)$ by LLN

Greedy action selection: $A_t = \operatorname{argmax}_a Q_t(a)$

ϵ-greedy selection: greedy most of time but selects random action w/ small probability ϵ

Nonstationary problems: constant step-size parameter

$$Q_{n+1} \equiv Q_n + \alpha(R_n - Q_n), \quad \alpha \in [0, 1)$$

$$Q_{n+1} = (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i$$

Notice exponentially decaying past rewards.

1: A simple bandit algorithm

Initialize, for $a = 1$ to k:
- $Q(a) \leftarrow 0$
- $N(a) \leftarrow 0$

Loop:

$$A \leftarrow \begin{cases} \operatorname{argmax}_a Q(a), & \text{with probability } 1 - \epsilon \\ \text{random action}, & \text{with probability } \epsilon \end{cases}$$

$R \leftarrow \mathit{bandit}(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \dfrac{1}{N(A)}[R - Q(A)]$

Upper Confidence Bound (UCB) Action Selection

"Optimism in the face of uncertainty"

Same as greedy except initialize $Q_t(a)$ to a high value, select value that optimizes an action A_t, and updates the upper bound to $Q_t(a)$.

$$A_t \equiv \operatorname{argmax} \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$

Finite Markov Decision Processes

State: $S_t \in \mathcal{S}$, Action: $A_t \in \mathcal{A}(s)$, Reward: $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$

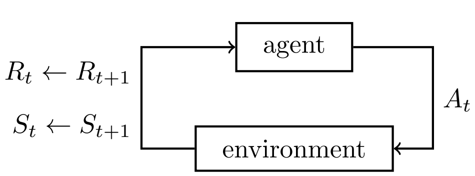
Transition dynamics fn (joint PMF):

Joint prob of next state s' and reward r given state s and action a.

$$p(s', r \mid s, a) \equiv \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

$$p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$$

$$\sum_{s'\in\mathcal{S}} \sum_{r\in\mathcal{R}} p(s', r \mid s, a) = 1, \quad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}(s)$$



State-Transition Probabilities (Alternative Forms)

$$p(s', r \mid s, a) \equiv \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

$$p(s' \mid s, a) = \Pr\{S_t = s' \mid S_{t-1} = s, A_{t-1} = a\} = \sum_{r\in\mathcal{R}} p(s', r \mid s, a)$$

$$r(s, a) \equiv \mathbb{E}[R_t \mid S_{t-1} = s, A_{t-1} = a] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s', r \mid s, a)$$

$$r(s, a, s') \equiv \mathbb{E}[R_t \mid S_{t-1} = s, A_{t-1} = a, S_t = s'] = \sum_{r\in\mathcal{R}} r \frac{p(s', r \mid s, a)}{p(s' \mid s, a)}$$

Markov property: future states of Markov process depend only on present state and not on past events.

Agent-envir interactions: episode $\to$ terminal state $\to$ reset

Goal of agent: maximize expected return, G_t

Episodic tasks: $G_t \equiv R_{t+1} + R_{t+2} + \cdots + R_T$

Continuing Tasks (no terminal state)

$$G_t \equiv R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{k-1} R_{t+k} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$G_t = R_{t+1} + \gamma G_{t+1}, \qquad \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}, \qquad \gamma \in [0, 1) \text{ is discount rate}$$

$$G_t \equiv \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \qquad T = \infty \text{ or } \gamma = 1 \text{ (but not both)}$$

Notice that future rewards are discounted more.

$\gamma = 0$: agent only cares about immediate reward (greedy).

$\gamma \to 1$: future rewards contribute more.

Policies

Law of total expectation: $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X \mid Y]]$

Partition formula: $\mathbb{E}[X] = \sum_i \mathbb{E}[X \mid A_i] P(A_i)$

Policy: mapping from states to probs of selecting each possible action.

$$\pi(a|s) = p(a \mid s) = \Pr\{A_t = a \mid S_t = s\}$$

Expectation of R_{t+1} in terms of π and p:

$$\mathbb{E}[R_{t+1} \mid S_t = s] = \sum_a \pi(a \mid S_t) \sum_{s',r} p(s', r \mid s, a) r$$

Value Functions

Value fns give expected return G_t when starting in state s and following policy π thereafter.

State-value fn: $v_\pi(s) \equiv \mathbb{E}_\pi[G_t \mid S_t = s]$ $\qquad G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

Value of terminal state is always 0.

Action-value fn: $q_\pi(s, a) \equiv \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$

v_π in terms of q_π and π: $v_\pi(s) = \sum_a \pi(a \mid S_t) q_\pi(s, a)$

q_π in terms of v_π and p: $q_\pi(s, a) = \sum_{r,s'} p(s', r \mid s, a)[r + \gamma v_\pi(s')]$

Bellman Equations

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

$$q_\pi(s,a) = \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma \sum_{a'} \pi(a' \mid s') q(s', a') \right]$$

Optimal value fns: $\pi_1 \geq \pi_2 \iff v_{\pi_1}(s) \geq v_{\pi_2}(s), \quad \forall s \in \mathcal{S}$

$$v_*(s) = \max_\pi v_\pi(s) = \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_*(s')], \quad \forall s \in \mathcal{S}$$

$$q_*(s,a) = \max_\pi q_\pi(s,a) = \sum_{s',r} p(s', r \mid s, a)[r + \gamma \max_{a'} q_*(s', a')]$$

$$\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$$

Policy Evaluation

$$\pi_* = \operatorname*{argmax}_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_*(s')]$$

2: Iterative Policy Evaluation

Input π, the policy to be evaluated
$\vec{V} \leftarrow \vec{0}, \vec{V}' \leftarrow \vec{0}$
loop:
  $\Delta \leftarrow 0$
  loop for each $s \in \mathcal{S}$:
    $V'(s) \leftarrow \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$
    $\Delta \leftarrow \max(\Delta, |V'(s) - V(s)|)$
  $V \leftarrow V'$
until $\Delta < \theta$ (small positive number)
return $V \approx v_\pi$

Policy improvement thm: $q_\pi(s, \pi'(s)) \geq v_\pi(s), \quad \forall s \in \mathcal{S}$

$$\pi'(s) \equiv \operatorname*{argmax}_a q_\pi(s,a) = \operatorname*{argmax}_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_\pi(s')]$$

3: Policy Iteration

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

2. Policy Evaluation
Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s))[r + \gamma V(s')]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

3. Policy Improvement
policy-stable $\leftarrow$ true
For each $s \in \mathcal{S}$:
  old-action $\leftarrow \pi(s)$
  $\pi(s) \leftarrow \operatorname*{argmax}_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$
  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.

Monte Carlo Methods

- Reqs only sample sequences of states, actions, rewards from interactions w/ envir. Works in RL by averaging sample returns.
- MC only for episodic tasks b/c only upon completion of episode are value estimates and policies changed.

4: First-visit MC prediction for estimating $V \approx v_\pi$

Input: policy π to be evaluated

Initialize:
  $V(s) \in \mathbb{R}$, arbitrarily $\forall s \in \mathcal{S}$
  Returns$(s) \leftarrow$ empty list $\forall s \in \mathcal{S}$

Loop (for each episode):
  Generate episode following π
  $G \leftarrow 0$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:
      Append G to *Returns*(S_t)
      $V(S_t) \leftarrow$ average(*Returns*(S_t))

MC Estimation of Action Values

$$\pi(s) \equiv \operatorname*{argmax}_a q(s,a), \qquad q_{\pi_k}(s, \pi_{k+1}(s)) \geq q_{\pi_k}(s, \pi_k(s)) \geq v_{\pi_k}(s)$$

5: First-visit MC prediction for estimating $V \approx v_\pi$

Input: policy π to be evaluated

Initialize:
  $\pi(s) \in \mathcal{A}(s)$, arbitrarily $\forall s \in \mathcal{S}$
  $Q(s,a) \in \mathbb{R}$, arbitrarily $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$
  Returns$(s) \leftarrow$ empty list $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$

Loop (for each episode):
  Choose $S_0 \in \mathcal{S}, A_0 \in \mathcal{A}(S_0)$ randomly st all pairs have probabilities greater than 0
  Generate episode from S_0, A_0 following π
  $G \leftarrow 0$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    Unless S_t appears in $S_0, A_0, \ldots, S_{t-1}, A_{t-1}$:
      Append G to *Returns*(S_t)
      $Q(S_t, A_t) \leftarrow$ average(*Returns*(S_t, A_t))
      $\pi(S_t) \leftarrow \operatorname*{argmax}_a Q(S_t, a)$

Note the last three lines can be made more efficient:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{n}(G - Q(S_t, A_t))$$

$$\pi(S_t) \leftarrow \operatorname*{argmax}_a Q(S_t, a))$$

MC Control w/o Exploring Starts

On-policy: tries to evaluate or improve policy used to make decisions.

Off-policy: same as on-policy but policy is different from that used to generate data: target policy + behaviour policy.

ϵ-soft policy: all nongreedy actions given minimal probability of selection $\epsilon/|\mathcal{A}(s)|$ whereas greedy action given probability $1 - \epsilon + \epsilon/|\mathcal{A}(s)|$.

6: On-policy first-visit MC control (for ϵ-soft policies, $\pi \approx \pi_*$)

Algorithm parameter: small $\epsilon > 0$

Initialize:
  $\pi \leftarrow$ arbitrary ϵ-soft policy
  $Q(s,a) \in \mathbb{R}$, arbitrarily $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$
  $Returns(s) \leftarrow$ empty list $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$

Loop (for each episode):
  Generate episode from S_0, A_0 following π
  $G \leftarrow 0$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    Unless pair S_t, A_t appears in $S_0, A_0, \ldots, S_{t-1}, A_{t-1}$:
      Append G to $Returns(S_t)$
      $Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
      $A^* \leftarrow \operatorname{argmax}_a Q(S_t, a)$
      $\forall a \in \mathcal{A}(S_t)$:

$$\pi(a \mid S_t) \leftarrow \begin{cases} 1 - \epsilon + \epsilon/|A(S_t)| & \text{if } a = A^* \\ \epsilon/|A(S_t)| & \text{if } a \neq A^* \end{cases}$$

Off-Policy Prediction via Importance Sampling

$\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} = \prod_{k=t}^{T-1} \pi(A_k \mid S_k) p(S_{k+1} \mid S_k, A_k)$

$\mathbb{E}[G_t \mid S_t = s] = v_b(s)$ $\mathbb{E}[\rho_{t:T-1} G_t \mid S_t = s] = v_\pi(s)$

$$\rho_{t:T-1} = \prod_{k=t}^{T-1} \frac{\pi(A_k \mid S_k)}{b(A_k \mid S_k)}$$

Ordinary importance sampling: $V(s) \equiv \dfrac{\sum_{t \in \mathcal{T}(s)} \rho G_t}{|\mathcal{T}(s)|}$

Weighted importance sampling: $V(s) \equiv \dfrac{\sum_{t \in \mathcal{T}(s)} \rho G_t}{\sum_{t \in \mathcal{T}(s)} \rho}$

Ordinary unbiased w/ high variance; weighted is biased w/ lower variance (preferred method).

Incremental Implementation

Suppose we have seq of returns $G_1, G_2, \ldots, G_{n-1}$ all starting from same state with random weight W_i. We wish to estimate

$$V_n \equiv \frac{\sum_{k=1}^{n-1}}{W}_k G_k \sum_{k=1}^{n-1} W_k, \quad n \geq 2$$

We can use the following equation:

$$V_{n+1} \equiv V_n + \frac{W_n}{C_n}(G_n - V_n), \quad n \geq 1$$

where $C_{n+1} \equiv C_n + W_{n+1}$ and $C_0 = 0$ (C_n is sum of weights).

7: Off-policy MC prediction (policy evaluation) $Q \approx Q_\pi$)

Input: an arbitrary target policy π

Initialize, $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$:
  $Q(s,a) \in \mathbb{R}$ (arbitrarily)
  $C(s,a) \leftarrow 0$

Loop (for each episode):
  $b \leftarrow$ any policy w/ coverage of π
  Generate an episode following $b : S_0, A_0, R_1, \ldots$
  $G \leftarrow 0$
  $W \leftarrow 1$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$ while $W \neq 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \dfrac{W}{C(S_t, A_t)}[G - Q(S_t, A_t)]$
    $W \leftarrow W \dfrac{\pi(A_t \mid S_t)}{b(A_t \mid S_t)}$

8: Off-policy MC control $\pi \approx \pi_*$

Initialize, $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$:
  $Q(s,a) \in \mathbb{R}$ (arbitrarily)
  $C(s,a) \leftarrow 0$
  $\pi(s) \leftarrow \operatorname{argmax}_a Q(s,a)$
Loop (for each episode):
  $b \leftarrow$ any policy w/ coverage of π
  Generate an episode following $b : S_0, A_0, R_1, \ldots$
  $G \leftarrow 0$
  $W \leftarrow 1$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \dfrac{W}{C(S_t, A_t)}[G - Q(S_t, A_t)]$
    $\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$
    If $A_t \neq \pi(S_t)$ then exit inner Loop
    $W \leftarrow W \dfrac{1}{b(A_t \mid S_t)}$

Temporal-Difference Learning

TD(0) update: $V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$

TD error: $\delta_t \equiv R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$

9: Tabular TD(0) for estimating v_π

Input: policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, $\forall s \in \mathcal{S}$ arbitrarily except $V(\text{terminal}) = 0$

Loop (for each episode):
  Initialize S
  Loop for each step of episode:
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(s) \leftarrow V(s) + \alpha[R + \gamma V(s') - V(s)]$
    $S \leftarrow S'$
  until S is terminal

MC error: $G_t - V(S_t) = \sum_{k=t}^{T-1} \gamma^{k-t} S_t$

Sarsa: On-Policy TD Control

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$

10: Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameter: step size $\alpha \in (0, 1]$, small $\epsilon > 0$
Initialize $Q(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ arbitrarily except $Q(\text{terminal}, \cdot) = 0$

Loop (for each episode):
  Initialize S
  Choose A from S using policy derived from Q:
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$
    $S \leftarrow S'; A \leftarrow A'$
  until S is terminal

Q-Learning: Off-Policy TD Control

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$

11: Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameter: step size $\alpha \in (0, 1]$, small $\epsilon > 0$
Initialize $Q(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ arbitrarily except $Q(\text{terminal}, \cdot) = 0$

Loop (for each episode):
  Initialize S
  Choose A from S using policy derived from Q:
  Loop for each step of episode:
    Choose A from S using policy derived from Q
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \max_a \gamma Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  until S is terminal

Double Q-learning addresses maximization bias problem. Instead of single $Q(S, A)$ update, with 0.5 probability, choose one of:

$$\begin{cases} Q_1(S, A) + \alpha \left(R + \gamma Q_2(S', \operatorname*{argmax}_a Q_1(S', a)) - Q_1(S, A) \right) \\ Q_2(S, A) + \alpha \left(R + \gamma Q_1(S', \operatorname*{argmax}_a Q_2(S', a)) - Q_2(S, A) \right) \end{cases}$$

Can adapt this for Sarsa and expected Sarsa updates.

Expected Sarsa

$$\begin{aligned} &Q(S_t, A_t) \\ &\leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \mathbb{E}_\pi[Q(S_{t+1}, A_{t+1}) \mid S_{t+1}] - Q(S_t, A_t)] \\ &\leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \sum_a \pi(a \mid S_{t+1}) Q(S_{t+1}, a) - Q(S_t, A_t)] \end{aligned}$$

Eliminates variance due to random selection of A_{t+1} from Sarsa.

Models and Planning

model $\xrightarrow{\text{planning}}$ policy

12: Random-sample one-step tabular Q-planning

Loop:
1. Select a state, $S \in \mathcal{S}$ and action, $A \in \mathcal{A}(S)$, at random
2. Send S, A to a sample model and obtain sample next reward R and sample next state S'
3. Apply one-step tabular Q-learning to S, A, R, S': $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

Dyna-Q

13: Tabular Dyna-Q

Initialize $Q(s, a)$ and $Model(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$
Loop:
1. $S \leftarrow$ current (nonterminal) state
2. $A \leftarrow \epsilon$-greedy(S, Q)
3. Take action A; observe resultant reward R and state S'
4. $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
5. $Model(S, A) \leftarrow R, S'$ (assuming deterministic envir)
6. Loop repeat n times:
    $S \leftarrow$ random previously observed state
    $A \leftarrow$ random action previously taken in S
    $R, S' \leftarrow Model(S, A)$
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

Dyna-Q+ has reward of $r + \kappa\sqrt{\tau}$, where κ is a constant and τ is the number of time steps since a given transition.

n-Step Bootstrapping

14: n-step TD for estimating $V \approx v_\pi$

Input: policy π to be evaluated
Algorithm parameters: step size $\alpha \in (0, 1]$, positive integer n
Initialize $V(s)$, arbitrarily $\forall s \in \mathcal{S}$
All store and access operations (for S_t and R_t) can take their index mod $(n + 1)$

Loop (for each episode):
  Initialize and store $S_0 \neq$ terminal
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
    If $t < T$, then:
      Take an action according to $\pi(\cdot \mid S_t)$
      Observe and store R_{t+1}, S_{t+1}
      If S_{t+1} terminal, then $T \leftarrow t + 1$
    $\tau \leftarrow t - n + 1$
    If $\tau \geq 0$:
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n, T)} \gamma^{i-\tau-1} R_i$
      If $\tau + n < T$, then: $G \leftarrow G + \gamma^n V(S_{\tau+n})$
      $V(S_\tau) \leftarrow V(S_\tau) + \alpha[G - V(S_\tau)]$
  Until $\tau = T - 1$

Prediction Objective

$$\overline{VE} \equiv \sum_{s\in\mathcal{S}} \mu(s)[v_\pi(s) - \hat{v}(s,\mathbf{w})]^2 \qquad \mu(s) \ge 0, \quad \sum_s \mu(s) = 1$$

Goal: $\overline{VE}(\mathbf{w}^*) \le \overline{VE}(\mathbf{w}),\ \forall \mathbf{w}$

$\mu(s)$ is often the fraction of time spent in state s

Stochastic-Gradient and Semi-Gradient Methods

$$\mathbf{w}_{t+1} \equiv \mathbf{w}_t - \frac{1}{2}\alpha\nabla[v_\pi(S_t) - \hat{v}(S_t,\mathbf{w}_t)]^2$$

$$= \mathbf{w}_t + \alpha[v_\pi(S_t) - \hat{v}(S_t,\mathbf{w}_t)]\nabla\hat{v}(S_t,\mathbf{w}_t) \qquad \nabla \text{ wrt } \mathbf{w}$$

Oftentimes $v_\pi(S_t)$ noise-corrupted, so we denote target as U_t:

$$w_{t+1} = \mathbf{w}_t + \alpha[U_t - \hat{v}(S_t,\mathbf{w}_t)]\nabla\hat{v}(S_t,\mathbf{w}_t)$$

15: Gradient MC algorithm for estimating $\hat{v} \approx v_\pi$

Input: policy π to be evaluated
Input: a differentiable fn $\hat{v} : \mathcal{S} \times \mathbb{R}^d \mapsto \mathbb{R}$
Algorithm parameter: step size $\alpha > 0$
Initialize value-fn weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Loop (for each episode):
  Generate episode $S_0, A_0, R_1, \ldots, R_T, S_T$ using π
  Loop for each step of episode, $t = 0, 1, \ldots, T-1$:
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha[G_t - \hat{v}(S_t,\mathbf{w})]\nabla\hat{v}(S_t,\mathbf{w})$

Semi-gradient TD(0) uses $U_t \equiv R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t)$

16: Semi-gradient TD(0) for estimating $\hat{v} \approx v_\pi$

Input: policy π to be evaluated
Input: a differentiable fn $\hat{v} : \mathcal{S} \times \mathbb{R}^d \mapsto \mathbb{R}$ st $\hat{v}(\text{terminal}, \cdot) = 0$
Algorithm parameter: step size $\alpha > 0$
Initialize value-fn weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Loop (for each episode):
  Initialize $\mathcal{S}$
  Loop for each step of episode:
    Choose $A \sim \pi(\cdot \mid s)$
    Take action A, observe R, S'
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha[R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})]\nabla\hat{v}(S,\mathbf{w})$
    $S \leftarrow S'$
  Until S is terminal

Linear Methods

Let $\hat{v}(\cdot,\mathbf{w})$ be linear and for each state s, $\mathbf{x}(s) \equiv [x_1(s), \ldots, x_d(s)]^\top$.

$$\hat{v}(s,\mathbf{w}) \equiv \langle \mathbf{w}, \mathbf{x}(s)\rangle = \mathbf{w}^\top\mathbf{x}(s) = \sum_{i=1}^d w_i x_i(s), \qquad x_i : S \mapsto \mathbb{R}$$

$$\nabla v(s,\mathbf{w}) = \mathbf{x}(s) \implies \mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[U_t - \hat{v}(S_t,\mathbf{w}_t)]\mathbf{x}(S_t)$$

$$\mathbf{w}_{\text{TD}} = A^{-1}\mathbf{b}$$

$$A \equiv \mathbb{E}[\mathbf{x}_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})]^\top \in \mathbb{R}^d \times \mathbb{R}^d \qquad \mathbf{b} \equiv \mathbb{E}[R_{t+1}\mathbf{x}_t] \in \mathbb{R}^d$$

$$\overline{VE}(\mathbf{w}_{\text{TD}}) = \frac{1}{1-\gamma}\min_{\mathbf{w}}\overline{VE}(\mathbf{w})$$

17: n-step semi-gradient TD for estimating $V \approx v_\pi$

Input: policy π to be evaluated
Input: a differentiable fn $\hat{v} : \mathcal{S} \times \mathbb{R}^d \mapsto \mathbb{R}$ st $\hat{v}(\text{terminal}, \cdot) = 0$
Algorithm parameters: step size $\alpha \in (0, 1]$, positive integer n
Initialize value-fn weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
All store and access operations (for S_t and R_t) can take their index mod $(n+1)$

Loop (for each episode):
  Initialize and store $S_0 \neq$ terminal
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
    If $t < T$, then:
      Take an action according to $\pi(\cdot \mid S_t)$
      Observe and store R_{t+1}, S_{t+1}
      If S_{t+1} terminal, then $T \leftarrow t+1$
    $\tau \leftarrow t - n + 1$
    If $\tau \ge 0$:
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1}R_i$
      If $\tau + n < T$, then: $G \leftarrow G + \gamma^n\hat{v}(S_{\tau+n},\mathbf{w})$
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha[G - \hat{v}(S_\tau,\mathbf{w})]\nabla\hat{v}(S_\tau,\mathbf{w})$
  Until $\tau = T - 1$

Parameterized Policies

$$\pi(a \mid s, \boldsymbol{\theta}) \ge 0, \quad \forall a \in \mathcal{A}, s \in \mathcal{S} \qquad \sum_{a\in\mathcal{A}} \pi(a \mid s, \boldsymbol{\theta}) = 1, \quad \forall s \in \mathcal{S}$$

Softmax policy: $\pi(a \mid s, \boldsymbol{\theta}) \equiv \dfrac{e^{h(s,a,\theta)}}{\sum_{b\in\mathcal{A}} e^{h(s,b,\boldsymbol{\theta})}}$

Action preference: $e^{h(s,a,\theta)}$

Avg reward formulation: $G_t = \sum_{t=0}^{\infty} R_t - r(\pi)$

Avg reward objective:

$$\mathfrak{r}(\pi) = E_\pi[R_t] = \sum_s \mu(s)\sum_a \pi(a \mid s, \boldsymbol{\theta})\sum_{s',r} p(s', r \mid s, a)r$$

$$\mathbb{E}_\pi[R_t \mid S_t = s, A_t = a] = \sum_{s',r} p(s', r \mid s, a)r$$

$$\mathbb{E}_\pi[R_t \mid S_t = s] = \sum_a \pi(a \mid s, \boldsymbol{\theta})\sum_{s',r} p(s', r \mid s, a)r$$

Policy Gradient

Policy gradient thm: $\nabla r(\pi) = \sum_s \mu(s)\sum_a \nabla\pi(a \mid s, \boldsymbol{\theta})q_\pi(s,a)$

Expected return: $J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[r(\tau)]$

Gradient ascent: $\boldsymbol{\theta}_{t+1} \equiv \boldsymbol{\theta}_t + \alpha \underbrace{\nabla_\theta J(\pi_\theta)}_{\text{policy grad}} |_{\theta_k}$

Probability of trajectory $\tau = (s_0, a_0, \ldots, s_{T+1}) \sim \pi_\theta$:

$$P(\tau \mid \theta) = \rho_0(s_0)\prod_{t=0}^{T} P(s_{t+1} \mid s_t, a_t)\pi_\theta(a_t \mid s_t)$$

Log-probability of trajectory:

$$\log P(\tau \mid \theta) = \log\rho_0(s_0) + \sum_{t=0}^{T}\left(\log P(s_{t+1} \mid s_t, a_t) + \log\pi_\theta(a_t \mid s_t)\right)$$

Grad-log-prob of trajectory: $\nabla_\theta \log P(\tau \mid \theta) = \sum_{t=0}^{T} \nabla_\theta \log\pi_\theta(a_t \mid s_t)$

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \mathbb{E}_{\tau\sim\pi_\theta}[r(\tau)] = \int_\tau \nabla_\theta P(\tau \mid \theta) r(\tau)\ d\tau$$
$$= \int_\tau P(\tau \mid \theta)\nabla_\theta \log P(\tau \mid \theta) r(\tau)\ d\tau$$
$$= \mathbb{E}_{\tau\sim\pi_\theta}[\nabla_\theta \log P(\tau \mid \theta) r(\tau)]$$
$$= \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) r(\tau)\right]$$

EGLP Lemma: $\mathbb{E}_{x\sim P_\theta}[\nabla_\theta \log P_\theta(x)] = 0$

$$\int_x P_\theta(x)\ dx = 1 \implies \nabla_\theta \int_x P_\theta(x)\ dx = \nabla_\theta 1 = 0$$
$$0 = \nabla_\theta \int_x P_\theta(x)\ dx = \int_x \nabla_\theta P_\theta(x)\ dx = \int_x P_\theta(x) \nabla_\theta \log P_\theta(x)\ dx$$
$$\implies 0 = \mathbb{E}_{x\sim P_\theta}[\nabla_\theta \log P_\theta(x)]$$

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1})\right]$$
$$= \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \left(\sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1}) - b(s_t)\right)\right]$$

Due to Markov property of MDPs and EGLP lemma

$b(s_t)$ is a baseline (common to choose $b(s_t) = V^\pi(s_t)$)

MSE objective: $\phi_k = \underset{\phi}{\operatorname{argmin}}\, \mathbb{E}_{s_t, \hat{R}_t\sim\pi_k}\left[\left(V_\pi(s_t) - \hat{R}_t\right)^2\right]$

Alternative Forms of Policy Gradient

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\Phi_t\right]$$

We can choose Φ_t to be any of $\Phi_t = R(\tau)$, $\Phi_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1})$, or $\Phi_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1}) - b(s_t)$.

All choices of Φ_t have the same expectation, just diff variances.

On-policy action-value fn: $\Phi_t = Q^{\pi_\theta}(s_t, a_t)$

Advantage fn: $\Phi_t = A^{\pi_\theta}(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$

Vanilla Policy Gradient

18: Vanilla Policy Gradient Algorithm

Input: initial policy params θ_0, intitial value fn params ϕ_0
for $k = 0, 1, 2, \ldots$ do
  Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ following $\pi_k = \pi(\theta_k)$
  Compute rewards $\hat{R}_t$
  Compute $\hat{A}_t$
  Estimate policy gradient:
$$\hat{g}_k = \frac{1}{|\mathcal{D}_k|}\sum_{\tau\in\mathcal{D}_k}\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)|_{\theta_k} \hat{A}_t$$
  Compute policy update using SGD: $\theta_{k+1} = \theta_k + \alpha_k \hat{g}_k$
  Fit value fn by regression on MSE
$$\phi_{k+1} = \operatorname{argmin} \frac{1}{|\mathcal{D}_k| T}\sum_{\tau\in\mathcal{D}_k}\sum_{t=0}^{T}\left(V_\phi(s_t) - \hat{R}_t\right)^2$$

Actor-Critic Algorithms

19: Q Actor-Critic

Initialize parameters, s, θ, w and learning rates α_θ, α_w; sample $a \sim \pi_\theta(a \mid s)$
for $t = 1 \ldots T$ do
  Sample reward $r_t \sim R(s, a)$ and next state $s' \sim P(s' \mid s, a)$
  Sample next action $a' \sim \pi_\theta(a' \mid s')$
  Update policy params: $\theta \leftarrow \theta + \alpha_\theta Q_w(s, a) \nabla_\theta \log \pi_\theta(a \mid s)$
  Compute TD error: $\delta_t = r_t + \gamma Q_w(s', a') - Q_w(s, a)$
  Update params of Q function: $w \leftarrow w + \alpha_w \delta_t \nabla_w Q_w(s, a)$
  $a \leftarrow a', \quad s \leftarrow s'$
end for

Deep Deterministic Policy Gradient (DDPG)

Mean-squared Bellman error (MSBE):

$$L(\phi, \mathcal{D}) = \mathbb{E}_{(s,a,r,s',d)\sim\mathcal{D}}\left[\left(Q_\phi(s,a) - (r + \gamma(1-d)\max_{a'} Q_\phi(s', a'))\right)^2\right]$$

$\mathcal{D}$ is set of transitions (s, a, r, s', d), d tells us if transition is terminal

Target: $r + \gamma(1-d)\max_{a'} Q_\phi(s', a')$

Target network update: $\phi_{\text{targ}} = \rho\phi_{\text{targ}} + (1-\rho)\phi$

Policy learning: gradient ascent wrt $\max_\theta \mathbb{E}_{s\sim\mathcal{D}}[Q_\phi(s, \mu_\theta(s))]$

$\mu_\theta(s)$ is deterministic policy

20: DDPG

Input: initial policy params θ, Q-fn params ϕ, empty replay buffer $\mathcal{D}$
Set target params equal to main params: $\theta_{\text{targ}} \leftarrow \theta, \phi_{\text{targ}} \leftarrow \phi$
Loop (until convergence):
  Observe s and select $a = \text{clip}(\mu_\theta(s) + \epsilon, a_{\text{lo}}, a_{\text{hi}}), \quad \epsilon \sim \mathcal{N}$
  Execute a and observe s', r, d
  Store (s, a, r, s', d) in replay buffer $\mathcal{D}$
  If s' terminal, reset environment state
  If time to update then:
    For however many updates do:
      Sample batch $B = \{(s, a, r, s', d)\} \in \mathcal{D}$
      Compute targets:
$$y(r, s', d) = r + \gamma(1-d) Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))$$
      Update Q-function by gradient descent
$$\nabla_\phi \frac{1}{|B|}\sum_{(s,a,r,s',d)\in B}(Q_\phi(s,a) - y(r, s', d))^2$$
      Update policy by gradient descent
$$\nabla_\theta \frac{1}{|B|}\sum_{s\in B} Q_\phi(s, \mu_\theta(s))$$
      Update target networks with
$$\phi_{\text{targ}} \leftarrow \rho\phi_{\text{targ}} + (1-\rho)\phi, \quad \theta_{\text{targ}} \leftarrow \rho\theta_{\text{targ}} + (1-\rho)\theta$$