# Reinforcement Learning Pseudocode Collection

September 20, 2025

## Global Notation

States $s \in \mathcal{S}$, actions $a \in \mathcal{A}(s)$, rewards $r \in \mathbb{R}$. Discount $\gamma \in [0, 1)$, step-size $\alpha \in (0, 1]$, threshold $\theta > 0$. Value functions: $V(s)$, $Q(s, a)$. Policies: target $\pi(a|s)$ and behavior $b(a|s)$, where $b$ has coverage of $\pi$. Episodes are denoted $S_0, A_0, R_1, S_1, A_1, \dots, S_T$ with terminal at time $T$.

## 1 Bandits

**Algorithm 1** $\varepsilon$-Greedy $k$-Armed Bandit (Incremental Averages)

**Input:** $\varepsilon \in [0, 1]$

```
Initialize Q(a) ← 0, N(a) ← 0 for all arms a = 1, ..., k
for t = 1, 2, ... do
    With prob. ε: pick A ∼ Uniform({1, ..., k}); else A ← arg max_a Q(a)
    Pull arm A and observe reward R
    N(A) ← N(A) + 1
    Q(A) ← Q(A) + (1/N(A)) (R − Q(A))                    ▷ incremental mean
```

**Algorithm 2** Incremental Mean Update (Reference)

**Input:** Current mean $Q_n$, new sample $R_n$, count $n$

$Q_{n+1} \leftarrow Q_n + \frac{1}{n}\big(R_n - Q_n\big)$

## 2 Markov Decision Processes

We assume a known or unknown transition *kernel* $p(s', r \mid s, a)$ depending on the algorithm. In model-based Dynamic Programming (DP), $p$ is known; in model-free methods (MC/TD), we learn from experience.

# 3 Dynamic Programming (Model-Based)

**Algorithm 3** Iterative Policy Evaluation (Prediction)

**Input:** Policy $\pi$, model $p(s', r \mid s, a)$, discount $\gamma$, threshold $\theta$

Initialize $V(s) \leftarrow 0$ for all $s$
**repeat**
    $\Delta \leftarrow 0$
    **for** each state $s$ **do**
        $v \leftarrow V(s)$
        $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
**until** $\Delta < \theta$
**return** $V \approx v_\pi$

**Algorithm 4** Policy Iteration

**Input:** Model $p(s', r \mid s, a)$, discount $\gamma$, threshold $\theta$

Initialize $V(s)$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s$
**repeat** ▷ Policy Evaluation
    **repeat**
        $\Delta \leftarrow 0$
        **for** each state $s$ **do**
            $v \leftarrow V(s)$
            $a \leftarrow \pi(s)$
            $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
            $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
    **until** $\Delta < \theta$ ▷ Policy Improvement
    policy_stable $\leftarrow$ true
    **for** each state $s$ **do**
        old $\leftarrow \pi(s)$
        $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
        **if** $\pi(s) \neq$ old **then**
            policy_stable $\leftarrow$ false
**until** policy_stable
**return** $(V, \pi) \approx (v^*, \pi^*)$

**Algorithm 5** Value Iteration

**Input:** Model $p(s', r \mid s, a)$, discount $\gamma$, threshold $\theta$

```
Initialize V(s) ← 0 for all s
repeat
    Δ ← 0
    for each state s do
        v ← V(s)
        V(s) ← max_a Σ_{s',r} p(s', r | s, a)[r + γV(s')]
        Δ ← max(Δ, |v − V(s)|)
until Δ < θ
Derive π(s) ← arg max_a Σ_{s',r} p(s', r | s, a)[r + γV(s')]
return (V, π)
```

# 4 Monte Carlo Methods (Model-Free)

**Algorithm 6** First-Visit Monte Carlo Prediction (State Values)

**Input:** Policy $\pi$, discount $\gamma$

```
Initialize V(s) arbitrarily; Returns(s) ← empty list for all s
for all episodes do
    Generate S_0, A_0, R_1, ..., S_T using π
    G ← 0
    for t ← T − 1 down to 0 do
        G ← γG + R_{t+1}
        if S_t is first visit to its state in episode then
            append G to Returns(S_t)
            V(S_t) ← average(Returns(S_t))
```

**Algorithm 7** On-Policy First-Visit MC Control ( $\varepsilon$-Soft )

**Input:** Small $\varepsilon > 0$, discount $\gamma$

```
Initialize π as any ε-soft policy; Q(s, a) arbitrarily; Returns(s, a) ← empty lists
for all episodes do
    Generate S_0, A_0, R_1, ..., S_T using π
    G ← 0
    for t ← T − 1 down to 0 do
        G ← γG + R_{t+1}
        if (S_t, A_t) is first visit then
            append G to Returns(S_t, A_t)
            Q(S_t, A_t) ← average(Returns(S_t, A_t))
            A* ← arg max_a Q(S_t, a)
            for all a ∈ 𝒜(S_t) do
                if a = A* then
                    π(a|S_t) ← 1 − ε + ε/|𝒜(S_t)|
                else
                    π(a|S_t) ← ε/|𝒜(S_t)|
```

**Algorithm 8** MC Control with Exploring Starts (MCES)

**Input:** Discount $\gamma$; assume exploring starts

Initialize $Q(s, a)$ arbitrarily; $\pi(s)$ arbitrary
**for all** episodes **do**
    Choose $(S_0, A_0)$ with nonzero prob. for all state–action pairs
    Generate episode following $\pi$: $S_0, A_0, R_1, \ldots, S_T$
    $G \leftarrow 0$
    **for** $t \leftarrow T - 1$ down to 0 **do**
        $G \leftarrow \gamma G + R_{t+1}$
        **if** $(S_t, A_t)$ first visit **then**
            Update $Q(S_t, A_t) \leftarrow$ average of returns for $(S_t, A_t)$
            Improve $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

**Algorithm 9** Off-Policy MC Prediction (Weighted Importance Sampling)

**Input:** Target policy $\pi$, behavior policy $b$ (with coverage), discount $\gamma$

Initialize $Q(s, a)$ arbitrarily; $C(s, a) \leftarrow 0$ for all $(s, a)$
**for all** episodes **do**
    Generate episode using $b$: $S_0, A_0, R_1, \ldots, S_T$
    $G \leftarrow 0$, $W \leftarrow 1$
    **for** $t \leftarrow T - 1$ down to 0 **while** $W \neq 0$ **do**
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)}\big(G - Q(S_t, A_t)\big)$
        $W \leftarrow W \cdot \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$

**Algorithm 10** Off-Policy MC Control (Weighted IS)

**Input:** Behavior $b$ (soft, coverage), discount $\gamma$

Initialize $Q(s, a)$ arbitrarily; $C(s, a) \leftarrow 0$; $\pi(s) \leftarrow \arg\max_a Q(s, a)$
**for all** episodes **do**
    Generate episode using $b$: $S_0, A_0, R_1, \ldots, S_T$
    $G \leftarrow 0$, $W \leftarrow 1$
    **for** $t \leftarrow T - 1$ down to 0 **while** $W \neq 0$ **do**
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)}\big(G - Q(S_t, A_t)\big)$
        $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$
        **if** $A_t \neq \pi(S_t)$ **then**
            **break**
        $W \leftarrow W \cdot \frac{1}{b(A_t|S_t)}$

**Algorithm 11** Off-Policy Incremental Update (Scalar Form)

**Input:** Stream of returns $\{G_n\}$ with weights $\{W_n\}$

Initialize $v \leftarrow 0$, $c \leftarrow 0$
**for** $n = 1, 2, \dots$ **do**
  $c \leftarrow c + W_n$
  $v \leftarrow v + \frac{W_n}{c}\big(G_n - v\big)$

# 5 Temporal-Difference Methods (Model-Free)

**Algorithm 12** TD(0) Prediction

**Input:** Policy $\pi$, step-size $\alpha$, discount $\gamma$

Initialize $V(s)$ arbitrarily with $V(\text{terminal}) = 0$
**for all** episodes **do**
  Initialize $S$
  **while** $S$ not terminal **do**
    Choose $A \sim \pi(\cdot|S)$; take $A$, observe $R, S'$
    $V(S) \leftarrow V(S) + \alpha\big(R + \gamma V(S') - V(S)\big)$
    $S \leftarrow S'$

**Algorithm 13** SARSA (On-Policy TD Control)

**Input:** Step-size $\alpha$, discount $\gamma$; behavior=target policy (e.g., $\varepsilon$-greedy w.r.t. $Q$)

Initialize $Q(s, a)$ arbitrarily with $Q(\text{terminal}, \cdot) = 0$
**for all** episodes **do**
  Initialize $S$; choose $A$ from $S$ using policy derived from $Q$
  **while** $S$ not terminal **do**
    Take $A$, observe $R, S'$
    Choose $A'$ from $S'$ using policy derived from $Q$
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big(R + \gamma Q(S', A') - Q(S, A)\big)$
    $S \leftarrow S'$; $A \leftarrow A'$

**Algorithm 14** Q-Learning (Off-Policy TD Control)

**Input:** Step-size $\alpha$, discount $\gamma$; behavior policy (e.g., $\varepsilon$-greedy w.r.t. $Q$)

Initialize $Q(s, a)$ arbitrarily with $Q(\text{terminal}, \cdot) = 0$
**for all** episodes **do**
  Initialize $S$
  **while** $S$ not terminal **do**
    Choose $A$ using behavior policy; take $A$, observe $R, S'$
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big(R + \gamma \max_a Q(S', a) - Q(S, A)\big)$
    $S \leftarrow S'$

**Algorithm 15** $n$-Step TD Prediction

**Input:** Policy $\pi$, step-size $\alpha$, discount $\gamma$, integer $n \geq 1$

Initialize $V(s)$ arbitrarily; $T \leftarrow \infty$
**for all** episodes **do**
  Initialize and store $S_0 \neq$ terminal
  **for** $t = 0, 1, 2, \ldots$ **do**
    **if** $t < T$ **then**
      Take $A_t \sim \pi(\cdot|S_t)$; observe $R_{t+1}, S_{t+1}$
      **if** $S_{t+1}$ terminal **then**
        $T \leftarrow t + 1$
    $\tau \leftarrow t - n + 1$ ▷ time whose estimate to update
    **if** $\tau \geq 0$ **then**
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      **if** $\tau + n < T$ **then**
        $G \leftarrow G + \gamma^n V(S_{\tau+n})$
      $V(S_\tau) \leftarrow V(S_\tau) + \alpha\big(G - V(S_\tau)\big)$
    **if** $\tau = T - 1$ **then**
      **break**

**Algorithm 16** $n$-Step SARSA (On-Policy)

**Input:** Step-size $\alpha$, discount $\gamma$, integer $n \geq 1$, $\varepsilon$ for $\varepsilon$-greedy

Initialize $Q(s, a)$ arbitrarily; define $\pi$ as $\varepsilon$-greedy w.r.t. $Q$; $T \leftarrow \infty$
**for all** episodes **do**
  Initialize $S_0$; choose $A_0 \sim \pi(\cdot|S_0)$
  **for** $t = 0, 1, 2, \ldots$ **do**
    **if** $t < T$ **then**
      Take $A_t$, observe $R_{t+1}, S_{t+1}$
      **if** $S_{t+1}$ terminal **then**
        $T \leftarrow t + 1$
      **else**
        Choose $A_{t+1} \sim \pi(\cdot|S_{t+1})$
    $\tau \leftarrow t - n + 1$
    **if** $\tau \geq 0$ **then**
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      **if** $\tau + n < T$ **then**
        $G \leftarrow G + \gamma^n Q(S_{\tau+n}, A_{\tau+n})$
      $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\big(G - Q(S_\tau, A_\tau)\big)$
      Update $\pi(\cdot|S_\tau)$ to be $\varepsilon$-greedy w.r.t. $Q(S_\tau, \cdot)$
    **if** $\tau = T - 1$ **then**
      **break**

**Algorithm 17** $n$-Step Off-Policy SARSA (Importance Sampling)

**Input:** Step-size $\alpha$, discount $\gamma$, integer $n \geq 1$; target policy $\pi$; behavior policy $b$

```
Initialize Q(s,a) arbitrarily; T ← ∞
for all episodes do
    Initialize S_0; choose A_0 ~ b(·|S_0)
    for t = 0, 1, 2, ... do
        if t < T then
            Take A_t, observe R_{t+1}, S_{t+1}
            if S_{t+1} terminal then
                T ← t + 1
            else
                Choose A_{t+1} ~ b(·|S_{t+1})
        τ ← t − n + 1
        if τ ≥ 0 then
            ρ ← ∏_{k=τ+1}^{min(τ+n−1, T−1)} π(A_k|S_k) / b(A_k|S_k)
            G ← ∑_{i=τ+1}^{min(τ+n,T)} γ^{i−τ−1} R_i
            if τ + n < T then
                G ← G + γ^n Q(S_{τ+n}, A_{τ+n})
            Q(S_τ, A_τ) ← Q(S_τ, A_τ) + α ρ (G − Q(S_τ, A_τ))
        if τ = T − 1 then
            break
```

# 6 Greedy Policy Improvement (Helper)

Given action-values $Q(s, \cdot)$, a deterministic improvement is $\pi(s) \leftarrow \arg\max_a Q(s, a)$. An $\varepsilon$-soft version at state $s$ sets the greedy action prob. to $1 - \varepsilon + \varepsilon/|\mathcal{A}(s)|$ and all others to $\varepsilon/|\mathcal{A}(s)|$.