

# Reinforcement Learning in Finance

Haotian Deng

Shanghai University of Finance and Economics

April 30, 2024



# Contents

- ① Basic Concept of RL
- ② Application in Finance
- ③ RL Algorithm Introduction

- 1 Basic Concept of RL
- 2 Application in Finance
- 3 RL Algorithm Introduction

# Basic Concepts

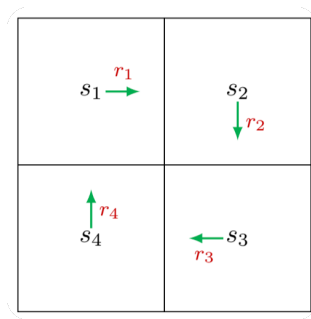


Figure 1: Grid World.

- state-action-reward

$$S_t \xrightarrow{A_t} R_{t+1}, S_{t+1} \xrightarrow{A_{t+1}} \dots$$

- discounted return

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \dots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

- state value function

$$\begin{aligned} v_\pi(s) &= \mathbb{E}[G_t \mid S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \underbrace{\mathbb{E}[R_{t+1} \mid S_t = s]}_{\text{immediate rewards}} \\ &\quad + \underbrace{\gamma \mathbb{E}[G_{t+1} \mid S_t = s]}_{\text{discounted future rewards}} \end{aligned}$$

# Bellman Equation

Bootstrapping: the returns rely on each other.

$$v_1 = r_1 + \gamma r_2 + \gamma^2 r_3 + \dots = r_1 + \gamma (r_2 + \gamma r_3 + \dots) = r_1 + \gamma v_2$$

$$v_2 = r_2 + \gamma r_3 + \gamma^2 r_4 + \dots = r_2 + \gamma (r_3 + \gamma r_4 + \dots) = r_2 + \gamma v_3$$

$$v_3 = r_3 + \gamma r_4 + \gamma^2 r_1 + \dots = r_3 + \gamma (r_4 + \gamma r_1 + \dots) = r_3 + \gamma v_4$$

$$v_4 = r_4 + \gamma r_1 + \gamma^2 r_2 + \dots = r_4 + \gamma (r_1 + \gamma r_2 + \dots) = r_4 + \gamma v_1$$

$$\underbrace{\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}}_{\mathbf{v}} = \underbrace{\begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix}}_{\mathbf{r}} + \underbrace{\begin{bmatrix} \gamma v_2 \\ \gamma v_3 \\ \gamma v_4 \\ \gamma v_1 \end{bmatrix}}_{\gamma \mathbf{P} \mathbf{v}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{P}} \underbrace{\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}}_{\mathbf{v}}$$

$$\text{Bellman Equation: } \mathbf{v} = \mathbf{r} + \gamma \mathbf{P} \mathbf{v} \Rightarrow \mathbf{v} = (\mathbf{I} - \gamma \mathbf{P})^{-1} \mathbf{r}$$

# Bellman Equation (element-wise form)

$$\begin{aligned}v_{\pi}(s) &= \mathbb{E}[R_{t+1} \mid \mathbf{S}_t = s] + \gamma \mathbb{E}[G_{t+1} \mid \mathbf{S}_t = s] \\&= \underbrace{\sum_{a \in \mathcal{A}} \pi(a \mid s) \sum_{r \in \mathcal{R}} p(r \mid s, a) r}_{\text{mean of immediate rewards}} \\&\quad + \underbrace{\sum_{a \in \mathcal{A}} \pi(a \mid s) \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s, a) v_{\pi}(s')}_{\text{mean of discounted future rewards}} \\&= \sum_{a \in \mathcal{A}} \pi(a \mid s) \left[ \sum_{r \in \mathcal{R}} p(r \mid s, a) r + \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s, a) v_{\pi}(s') \right] \\v_{\pi}(s) &= r_{\pi}(s) + \gamma \sum_{s' \in \mathcal{S}} p_{\pi}(s' \mid s) v_{\pi}(s')\end{aligned}$$

# Bellman Optimality Equation

- state-action value

$$\begin{aligned} q_{\pi}(s, a) &\doteq \mathbb{E}\left[G_t \mid S_t = s, A_t = a\right] \\ &= \sum_{r \in \mathcal{R}} p(r \mid s, a) r + \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s, a) v_{\pi}(s') \end{aligned}$$

- Bellman optimality equation

$$\begin{aligned} v(s) &= \max_{\pi(s)} \sum_{a \in \mathcal{A}} \pi(a \mid s) \left[ \sum_{r \in \mathcal{R}} p(r \mid s, a) r + \gamma \sum_{s' \in \mathcal{S}} p(s' \mid s, a) v(s') \right] \\ &= \max_{\pi(s)} \sum_{a \in \mathcal{A}} \pi(a \mid s) q(s, a) = \max_{a \in \mathcal{A}} q(s, a) \end{aligned}$$

- (deterministic) greedy optimal policy

$$\begin{aligned} \pi(a \mid s) &= \begin{cases} 1 & a = a^* \\ 0 & a \neq a^* \end{cases}, \quad \text{where } a^* = \arg \max_{a \in \mathcal{A}} q(s, a) \\ v^* &= \max_{\pi} (r_{\pi} + \gamma P_{\pi} v^*) = r_{\pi^*} + \gamma P_{\pi^*} v^* = v_{\pi^*} \geq v_{\pi} \end{aligned}$$

- 1 Basic Concept of RL
- 2 Application in Finance
- 3 RL Algorithm Introduction



# Optimization Problem for a Social Planner in discrete-time

$$\begin{aligned} \max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t u(c_t) \\ \text{s.t. } c_t + k_{t+1} = f(k_t) \end{aligned} \Rightarrow -u'[c_t] + \beta u'[c_{t+1}] f'(k_{t+1}) = 0$$

- Traditional method:

$$\begin{aligned} V^*[k_0] &= \max_{\{k_{t+1}\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t u[f(k_t) - k_{t+1}] \\ &= \max_{\{k_{t+1}\}_{t=0}^{\infty}} \left\{ u[f(k_0) - k_1] + \beta [u(f(k_1) - k_2) + \dots] \right\} \\ &= \max_{k_1} \left\{ u[f(k_0) - k_1] + \beta V^*[k_1] \right\} \\ &\Downarrow \text{Bellman Equation} \end{aligned}$$

$$V^*[k_t] = \max_{k_{t+1}} \left\{ u[f(k_t) - k_{t+1}] + \beta V^*[k_{t+1}] \right\}$$

- RL method:  $v^*(k_t) = \max_{c(t)} \left\{ u[c(t)] + \beta v^*(k_{t+1}) \right\}$

# Merton's Portfolio Problem in continuous-time

- The state and action at time  $t$  is  $(t, W_t)$  and  $(\pi_t, c_t)$ .

$$dW_t = \left[ (r + \pi_t \cdot (\mu - r)) \cdot W_t - c_t \right] \cdot dt + \pi_t \cdot \sigma \cdot W_t \cdot dz_t$$

- The reward per unit time at time  $t$  is

$$\begin{cases} U(c_t) = \frac{c_t^{1-\gamma}}{1-\gamma}, & t < T \\ B(T) \cdot U(W_t) = \varepsilon^\gamma \cdot \frac{W_t^{1-\gamma}}{1-\gamma}, & t = T \end{cases}$$

- The return at time  $t$  is the accumulated discounted reward

$$\int_t^T e^{-\rho(s-t)} \cdot \frac{c_s^{1-\gamma}}{1-\gamma} \cdot ds + \frac{e^{-\rho(T-t)} \cdot \varepsilon^\gamma \cdot W_T^{1-\gamma}}{1-\gamma}$$

- Our goal is to determine optimal allocation  $\pi(t, W_t)$  and consumption  $c(t, W_t)$  at any time  $t$  to maximize

$$\mathbb{E} \left[ \int_t^T \frac{e^{-\rho(s-t)} \cdot c_s^{1-\gamma}}{1-\gamma} \cdot ds + \frac{e^{-\rho(T-t)} \cdot B(T) \cdot W_T^{1-\gamma}}{1-\gamma} \middle| W_t \right]$$

# HJB as a Continuous-Time Version of BOE

- BOE in discrete-time:

$$V^*(s) = \max_{a \in \mathcal{A}_t} \left\{ \mathcal{R}(s, a) + \gamma \cdot \sum_{s' \in \mathcal{N}} \mathcal{P}(s, a, s') \cdot V^*(s') \right\}$$

- BOE in continuous-time:

$$V^*(t, s_t) = \max_{a_t \in \mathcal{A}_t} \left\{ \mathcal{R}(t, s_t, a_t) \cdot dt + \mathbb{E}_{(t, s_t, a_t)} \left[ e^{-\rho \cdot dt} \cdot V^*(t + dt, s_{t+dt}) \right] \right\}$$

- Multiplying throughout by  $e^{-\rho t}$  and re-arranging, we have

$$\begin{aligned} \frac{\mathcal{R}(t, s_t, a_t)}{e^{\rho t}} \cdot dt + \mathbb{E}_{(t, s_t, a_t)} \left[ \frac{V^*(t + dt, s_{t+dt})}{e^{\rho(t+dt)}} - \frac{V^*(t, s_t)}{e^{\rho t}} \right] \\ \frac{V^*(t + dt, s_{t+dt})}{e^{\rho(t+dt)}} - \frac{V^*(t, s_t)}{e^{\rho t}} &= d[e^{-\rho t} \cdot V^*(t, s_t)] \\ &= e^{-\rho t} \cdot [dV^*(t, s_t) - \rho \cdot V^*(t, s_t) \cdot dt] \end{aligned}$$

# HJB as a Continuous-Time Version of BOE

- Thus we have

$$\max_{a_t \in \mathcal{A}_t} \left\{ \frac{\mathcal{R}(t, \mathbf{s}_t, a_t)}{e^{\rho t}} \cdot dt + \mathbb{E}_{(t, \mathbf{s}_t, a_t)} \left[ \frac{dV^*(t, \mathbf{s}_t) - \rho \cdot V^*(t, \mathbf{s}_t) \cdot dt}{e^{\rho t}} \right] \right\} = 0$$

- Multiplying throughout by  $e^{\rho t}$  and re-arranging, we have

$$\rho \cdot V^*(t, \mathbf{s}_t) \cdot dt = \max_{a_t \in \mathcal{A}_t} \left\{ \mathbb{E}_{(t, \mathbf{s}_t, a_t)} [dV^*(t, \mathbf{s}_t)] + \mathcal{R}(t, \mathbf{s}_t, a_t) \cdot dt \right\}$$

- Assume that the transitions for  $\mathbf{s}$  are given by an Ito process:

$$d\mathbf{s}_t = \boldsymbol{\mu}(t, \mathbf{s}_t, a_t) \cdot dt + \boldsymbol{\sigma}(t, \mathbf{s}_t, a_t) \cdot d\mathbf{z}_t$$

- Apply multivariate Ito's Lemma for  $V^*$  as a function of  $t$  and  $\mathbf{s}_t$

$$dV^*(t, \mathbf{s}_t) = \left( \frac{\partial V^*}{\partial t} + (\nabla_{\mathbf{s}} V^*)^T \cdot \boldsymbol{\mu}_t + \frac{1}{2} \text{Tr} [\boldsymbol{\sigma}_t^T \cdot (\Delta_{\mathbf{s}} V^*) \cdot \boldsymbol{\sigma}_t] \right) \cdot dt + (\nabla_{\mathbf{s}} V^*)^T \cdot \boldsymbol{\sigma}_t \cdot d\mathbf{z}_t$$

# HJB as a Continuous-Time Version of BOE

- Substituting the expression for  $dV^*(t, \mathbf{s}_t)$ , noting that

$$\mathbb{E}_{(t, \mathbf{s}_t, a_t)} \left[ (\nabla_{\mathbf{s}} V^*)^T \cdot \boldsymbol{\sigma}_t \cdot d\mathbf{z}_t \right] = 0$$

- Dividing throughout by  $dt$ , we finally get to the **HJB equation**

$$\begin{aligned} & \rho \cdot V^*(t, \mathbf{s}_t) \\ &= \max_{a_t \in \mathcal{A}_t} \left\{ \frac{\partial V^*}{\partial t} + (\nabla_{\mathbf{s}} V^*)^T \cdot \boldsymbol{\mu}_t + \frac{1}{2} \text{Tr} [\boldsymbol{\sigma}_t^T \cdot (\Delta_{\mathbf{s}} V^*) \cdot \boldsymbol{\sigma}_t] + \mathcal{R}(t, \mathbf{s}_t, a_t) \right\} \end{aligned}$$

- For a finite-horizon problem terminating at time  $T$ , the above equation is subject to **terminal condition**:

$$V^*(T, \mathbf{s}_T) = \mathcal{T}(\mathbf{s}_T)$$

# Merton's Portfolio Problem in continuous-time

- Back to our asset-allocation and consumption problem, the general **HJB equation** specializes here to the following:

$$\max_{\pi_t, c_t} \left\{ \mathbb{E}_t \left[ dV^*(t, W_t) + \frac{c_t^{1-\gamma}}{1-\gamma} \cdot dt \right] \right\} = \rho \cdot V^*(t, W_t) \cdot dt$$

- Similarly, use Ito's Lemma on  $dV^*$ , we have

$$\begin{aligned} \rho \cdot V^*(t, W_t) &= \max_{\pi_t, c_t} \{ \mathcal{Q}^*(t, W_t, \pi_t, c_t) \} \\ \mathcal{Q}^*(t, W_t, \pi_t, c_t) &= \frac{\partial V^*}{\partial t} + \frac{\partial V^*}{\partial W_t} \cdot \left[ (\pi_t(\mu - r) + r) W_t - c_t \right] \\ &\quad + \frac{\partial^2 V^*}{\partial W_t^2} \cdot \frac{\pi_t^2 \cdot \sigma^2 \cdot W_t^2}{2} + \frac{c_t^{1-\gamma}}{1-\gamma} \end{aligned}$$

- This HJB equation is subject to the **terminal condition**:

$$V^*(T, W_T) = \epsilon^\gamma \cdot \frac{W_T^{1-\gamma}}{1-\gamma}$$

- 1 Basic Concept of RL
- 2 Application in Finance
- 3 RL Algorithm Introduction

# Overview

- How to solve the Bellman optimality equation?

$$v = f(v) = \max_{\pi} (r_{\pi} + \gamma P_{\pi} v)$$

- We need algorithms to learn optimal policies!

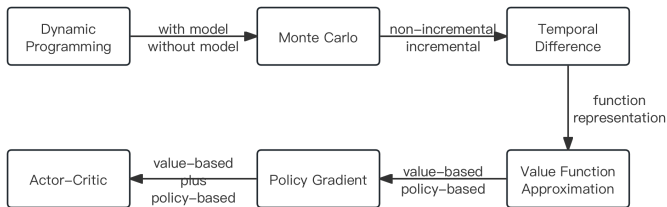


Figure 2: The map of RL algorithms



# Dynamic Programming

- Value Iteration

- ① Policy Update:  $\pi_{k+1} = \arg \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_k)$

- ② Value Update:  $v_{k+1} = r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_k$

- Policy Iteration

- ① Policy Evaluation:  $v_{\pi_k} = r_{\pi_k} + \gamma P_{\pi_k} v_{\pi_k}$

- ② Policy Improvement:  $\pi_{k+1} = \arg \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_{\pi_k})$

$$v_k(s) \rightarrow q_k(s, a) \xrightarrow{\text{policy update}} \pi_{k+1}(s) \xrightarrow{\text{value update}} \underbrace{v_{k+1}(s) = \max_a q_k(s, a)}_{\text{new value (not state value)}}$$

Policy Iteration:  $\pi_0 \xrightarrow{\text{PE}} v_{\pi_0} \xrightarrow{\text{PI}} \pi_1 \xrightarrow{\text{PE}} v_{\pi_1} \xrightarrow{\text{PI}} \pi_2 \xrightarrow{\text{PE}} v_{\pi_2} \xrightarrow{\text{PI}} \dots$

Value Iteration:  $v_0 \xrightarrow{\text{PU}} \pi'_1 \xrightarrow{\text{VU}} v_1 \xrightarrow{\text{PU}} \pi'_2 \xrightarrow{\text{VU}} v_2 \xrightarrow{\text{PU}} \dots$

# Monte Carlo

What is the key for the policy improvement step?

$$\begin{aligned}\pi_{k+1}(s) &= \arg \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_{\pi_k}) \\ &= \arg \max_{\pi} \sum_a \pi(a | s) \left[ \sum_r p(r | s, a) r + \gamma \sum_{s'} p(s' | s, a) v_{\pi_k}(s') \right] \\ &= \arg \max_{\pi} \sum_a \pi(a | s) q_{\pi_k}(s, a), \quad s \in \mathcal{S}\end{aligned}$$

$$q_{\pi_k}(s, a) = \sum_r p(r | s, a) r + \gamma \sum_{s'} p(s' | s, a) v_{\pi_k}(s') \leftarrow \text{model-based}$$

$\Downarrow$

$$\begin{aligned}q_{\pi_k}(s, a) &= \mathbb{E}[G_t | S_t = s, A_t = a] \\ &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s, A_t = a]\end{aligned}$$

$$q_{\pi_k}(s, a) = \mathbb{E}[G_t | S_t = s, A_t = a] \approx \frac{1}{n} \sum_{i=1}^n g_{\pi_k}^{(i)}(s, a) \leftarrow \text{model-free}$$

# Temporal-Difference: Robbins-Monro Algorithm

- Bellman expectation equation: state value function

$$\begin{aligned} v_{\pi}(s) &= \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s] \\ v_{\pi}(s_t) &= \mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s_t] \end{aligned}$$

- RM algorithm to solve bellman expectation equation

$$\begin{aligned} g(v_{\pi}(s_t)) &\doteq v_{\pi}(s_t) - \mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s_t] = 0 \\ \tilde{g}(v_{\pi}(s_t)) &= v_{\pi}(s_t) - [r_{t+1} + \gamma v_{\pi}(s_{t+1})] \\ &= \underbrace{(v_{\pi}(s_t) - \mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s_t])}_{g(v_{\pi}(s_t))} \\ &\quad + \underbrace{(\mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s_t] - [r_{t+1} + \gamma v_{\pi}(s_{t+1})])}_{\eta} \\ v_{t+1}(s_t) &= v_t(s_t) - \alpha_t(s_t) \tilde{g}(v_t(s_t)) \\ &= v_t(s_t) - \alpha_t(s_t) (v_t(s_t) - [r_{t+1} + \gamma v_{\pi}(s_{t+1})]) \end{aligned}$$

# Temporal-Difference: TD Target and TD Error

$$\underbrace{v_{t+1}(s_t)}_{\text{new estimate}} = \underbrace{v_t(s_t)}_{\text{current estimate}} - \alpha_t(s_t) \left\{ \overbrace{v_t(s_t) - [r_{t+1} + \gamma v_t(s_{t+1})]}^{\text{TD error } \delta_t} \right\}$$

TD target  $\bar{v}_t$

- TD target:  $v(s_t) \rightarrow \bar{v}_t$

$$\begin{aligned}
 & v_{t+1}(s_t) = v_t(s_t) - \alpha_t(s_t) [v_t(s_t) - \bar{v}_t] \\
 \implies & v_{t+1}(s_t) - \bar{v}_t = v_t(s_t) - \bar{v}_t - \alpha_t(s_t) [v_t(s_t) - \bar{v}_t] \\
 \implies & v_{t+1}(s_t) - \bar{v}_t = [1 - \alpha_t(s_t)] [v_t(s_t) - \bar{v}_t] \\
 \implies & |v_{t+1}(s_t) - \bar{v}_t| = |1 - \alpha_t(s_t)| |v_t(s_t) - \bar{v}_t| \\
 \implies & |v_{t+1}(s_t) - \bar{v}_t| \leq |v_t(s_t) - \bar{v}_t|
 \end{aligned}$$

- TD error:  $\delta_t = v_t(s_t) - \underbrace{[r_{t+1} + \gamma v_t(s_{t+1})]}_{\text{from innovation } (s_t, r_{t+1}, s_{t+1})}$

$$\begin{aligned}
 \mathbb{E}[\delta_t \mid \mathbf{S}_t = \mathbf{s}_t] &= \mathbb{E}[v_\pi(\mathbf{S}_t) - (R_{t+1} + \gamma v_\pi(\mathbf{S}_{t+1})) \mid \mathbf{S}_t = \mathbf{s}_t] \\
 &= v_\pi(\mathbf{s}_t) - \mathbb{E}[R_{t+1} + \gamma v_\pi(\mathbf{S}_{t+1}) \mid \mathbf{S}_t = \mathbf{s}_t] = 0
 \end{aligned}$$

# Temporal-Difference: Sarsa

## Sarsa

- Bellman expectation equation for state-action value

$$q_{\pi}(s, a) = \mathbb{E}\left[R + \gamma q_{\pi}(S', A') \mid s, a\right], \quad \text{for all } (s, a)$$

- some experience (online):  $\{(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})\}_t$

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t) \left\{ q_t(s_t, a_t) - [r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1})] \right\}$$

## n-step Sarsa

$$\{s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}\} \rightarrow \{s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}, \dots, r_{t+n}, s_{t+n}, a_{t+n}\}$$

$$\text{Sarsa} \leftarrow G_t^{(1)} = R_{t+1} + \gamma q_{\pi}(s_{t+1}, a_{t+1})$$

$$\vdots$$

$$\text{n-step Sarsa} \leftarrow G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \dots + \gamma^n q_{\pi}(s_{t+n}, a_{t+n})$$

$$\vdots$$

$$\text{MC} \leftarrow G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \dots$$

# Temporal-Difference: Q-learning

- Bellman optimality equation: directly find the maximize state-action value  $q(s, a)$

$$q(s, a) = \mathbb{E} \left[ R_{t+1} + \gamma \max_a q(S_{t+1}, a) \mid S_t = s, A_t = a \right], \quad \forall s, a$$

- Q-learning method

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t)$$

$$- \alpha_t(s_t, a_t) \left\{ q_t(s_t, a_t) - \left[ r_{t+1} + \gamma \max_{a \in \mathcal{A}(s_{t+1})} q_t(s_{t+1}, a) \right] \right\}$$

- On-policy and off-policy: behavior policy = target policy ?
  - behavior policy: generate experience samples
  - target policy: constantly updated toward an optimal policy
- What exactly is the TD algorithm? A stochastic approximation algorithm for solving BE or BOE  $q(s, a) = \mathbb{E}[\bar{q}_t \mid s, a]$ .

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) - \alpha_t(s_t, a_t) \left[ q_t(s_t, a_t) - \underbrace{\bar{q}_t}_{\text{TD target}} \right]$$

# Value Function Approximation

- Our goal is to find the best  $w$  that can minimize  $J(w)$ .

$$\min_w J(w) = \mathbb{E} \left[ (v_\pi(S) - \hat{v}(S, w))^2 \right]$$

- To minimize the objective function  $J(w)$ , we can use gradient-descent algorithm:  $w_{k+1} = w_k - \alpha_k \nabla_w J(w_k)$

$$\begin{aligned} \nabla_w J(w_k) &= \nabla_w \mathbb{E} \left[ (v_\pi(S) - \hat{v}(S, w_k))^2 \right] = \mathbb{E} \left[ \nabla_w (v_\pi(S) - \hat{v}(S, w_k))^2 \right] \\ &= 2\mathbb{E} [(v_\pi(S) - \hat{v}(S, w_k)) (-\nabla_w \hat{v}(S, w_k))] \\ &= -2\mathbb{E} [(v_\pi(S) - \hat{v}(S, w_k)) \nabla_w \hat{v}(S, w_k)] \end{aligned}$$

- Use the stochastic gradient to replace the true gradient:

$$\text{SGD: } w_{t+1} = w_t + \alpha_t [\mathbf{v}_\pi(\mathbf{s}_t) - \hat{v}(\mathbf{s}_t, w_t)] \nabla_w \hat{v}(\mathbf{s}_t, w_t)$$

- By the spirit of TD learning,  $r_{t+1} + \gamma \hat{v}(\mathbf{s}_{t+1}, w_t)$  can be viewed as an approximation of  $v_\pi(\mathbf{s}_t)$ . Then, the algorithm becomes

$$\text{TD: } w_{t+1} = w_t + \alpha_t [\mathbf{r}_{t+1} + \gamma \hat{v}(\mathbf{s}_{t+1}, w_t) - \hat{v}(\mathbf{s}_t, w_t)] \nabla_w \hat{v}(\mathbf{s}_t, w_t)$$

# Value Function Approximation: Deep Q-learning

- Deep Q-learning aims to minimize the Bellman optimality error

$$\min J = \mathbb{E} \left[ \left( R + \gamma \max_{a \in \mathcal{A}(S')} \hat{q}(S', a, w) - \hat{q}(S, A, w) \right)^2 \right]$$

- The parameter  $w$  appears in two  $\hat{q}(S', a, w)$ .

$$J = \mathbb{E} \left[ \left( R + \gamma \max_{a \in \mathcal{A}(S')} \hat{q}(S', a, w) - \hat{q}(S, A, w) \right)^2 \right]$$

- Assume that  $w$  in  $y(w) = R + \gamma \max_{a \in \mathcal{A}(S')} \hat{q}(S', a, w)$  is fixed (at least for a while) when we calculate the gradient.

$$\nabla_w J = \mathbb{E} \left[ \left( R + \gamma \max_{a \in \mathcal{A}(S')} \underbrace{\hat{q}(S', a, w_T)}_{\text{target network}} - \underbrace{\hat{q}(S, A, w)}_{\text{main network}} \right) \nabla_w \hat{q}(S, A, w) \right]$$



# Policy Gradient

- Policies can be represented by functions:  $\pi(a|s, \theta)$
- Use gradient-based optimization algorithms to search for optimal policies:

$$\theta_{t+1} = \theta_t + \alpha \nabla_{\theta} J(\theta_t)$$

$$J(\theta) = \lim_{n \rightarrow \infty} \mathbb{E} \left[ \sum_{t=0}^n \gamma^t R_{t+1} \right] = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \sum_{s \in \mathcal{S}} \eta(s) \sum_{a \in \mathcal{A}} \nabla_{\theta} \pi(a | s, \theta) q_{\pi}(s, a) \\ &= \mathbb{E}_{S \sim \eta, A \sim \pi(S, \theta)} [\nabla_{\theta} \ln \pi(A | S, \theta) q_{\pi}(S, A)] \end{aligned}$$

# Monte Carlo Policy Gradient (REINFORCE)

$$\theta_{t+1} = \theta_t + \alpha \nabla_{\theta} J(\theta_t)$$

$$= \theta_t + \alpha \mathbb{E} [\nabla_{\theta} \ln \pi(A | S, \theta_t) q_{\pi}(S, A)]$$

⇓ replace the true gradient with a stochastic gradient

$$\theta_{t+1} = \theta_t + \alpha \underbrace{\nabla_{\theta} \ln \pi(a_t | s_t, \theta_t)}_{\frac{\nabla_{\theta} \pi(a_t | s_t, \theta_t)}{\pi(a_t | s_t, \theta_t)}} \underbrace{q_t(s_t, a_t)}_{\text{MC estimation}}$$

⇓

$$\theta_{t+1} = \theta_t + \alpha \underbrace{\left( \frac{q_t(s_t, a_t)}{\pi(a_t | s_t, \theta_t)} \right)}_{\beta_t} \nabla_{\theta} \ln \pi(a_t | s_t, \theta_t)$$

⇓

$$\theta_{t+1} = \theta_t + \alpha \beta_t \nabla_{\theta} \ln \pi(a_t | s_t, \theta_t)$$

# Actor-Critic

- Actor: Policy Update Algorithm

$$\theta_{t+1} = \theta_t + \alpha \nabla_{\theta} \ln \pi(a_t | s_t, \theta_t) q_t(s_t, a_t)$$

- Critic: Sarsa + Value Function Approximation

$$\begin{aligned} w_{t+1} = & w_t \\ & + \alpha_w [r_{t+1} + \gamma q(s_{t+1}, a_{t+1}, w_t) - q(s_t, a_t, w_t)] \nabla_w q(s_t, a_t, w_t) \end{aligned}$$

- A framework which can be extended to many other algorithms
  - Deterministic AC (DPG)
  - DDPG
  - A2C, A3C
  - SAC
  - PPO
  - TD3

***Thanks!***