ML & SC

① terms of RL & MDP

② DL for SC & PDEs

① Foundation of ML & SC.

① Basics of MDP $(X, A, P, r=(f,g))$

① X: State ; A: action ; $x_{t+1} \sim P_t(x_t, a_t)$;
transition probability (t, x, a)

Reward: ① running reward $f(x,a)$ ② terminal reward $g(x)$ ③ Discount factor $\beta \in [0,1]$

② Basics of RL

① policy: π (sequence π_0) ⇒ ① deterministic ② stochastic
$\pi_0: X \to A$ $\pi_0: X \to P(A)$
control: $a \sim \pi$
policy: $\pi_0 = a \sim \pi$

② State value function: $V_t^\pi(x) = E_\pi\left[\sum_{s=t}^{T-1} f(x_s, a_s) + g(x_T) \mid x_t = x\right]$, $t \in \{0, \dots, T\}$, $x \in X$
$V_t^\pi(x) = E[Q_t^\pi(x,a)]$

③ Q-function of π: $Q_t^\pi(x,a) = E_\pi\left[\sum_{s=t}^{T-1} f(x_s, a_s) + g(x_T) \mid x_t = x, a_t = a\right]$, $t \in \{0, \dots, T\}$, $(x,a) \in X \times A$.

④ goal: find an optimal policy π^* maximize V^π $V := \sup_\pi V^\pi$, $Q := \sup_\pi Q^\pi$

③ DP: MDP is global dynamic optimisation ⇒ reduce the optimisation problem to local optimisation problem.

① Bellman Equation $V_T^\pi(x) = g(x)$; $V_t^\pi(x) = E[f(x,a) + E[V_{t+1}^\pi(x')]]$
$Q_t^\pi(x,a)$
immediate reward
在状态 x 执行动作 a 后
expected future reward $x' \sim P_t(x,a)$

④ Classes of RL: ① value-based methods: learn value & Q ⇒ policy derived

② policy-based methods: learn policy ⇒ value function estimated

⑤ TD-learning: $V_t^\pi(x) = E[f(x,a) + V_{t+1}^\pi(x')]$

(TD(1) & TD(0))

$\hat{V}_t^\pi(x) \leftarrow (1-\eta)\hat{V}_t^\pi(x) + \eta[f_t + \hat{V}_{t+1}^\pi(x')] = \hat{V}_t^\pi(x) + \eta[f_t + \hat{V}_{t+1}^\pi(x') - \hat{V}_t^\pi(x)]$

prediction
$\eta \in (0,1)$ learning rate.
TD error δ_t

⑥ TD-learning & Q-learning

① Bellman Equation $V_t^\pi(x) = E[f(x,a) + V_{t+1}^\pi(x')]$ → ① initialize ② $\pi \sim (t, x, a, f_t, x')$

TD(0)

② $\hat{V}_t^\pi(x) \leftarrow \hat{V}_t^\pi(x) + \eta[f_t + \hat{V}_{t+1}^\pi(x') - \hat{V}_t^\pi(x)]$

TD $\delta_t = f_t + \hat{V}_{t+1}^\pi(x') - \hat{V}_t^\pi(x)$

TD(1) - $\lambda = 1$ 用整条轨迹的回报更新. $\lambda \in [0,1]$.

② Q-learning $Q_t(x,a) = f(x,a) + E[\sup Q_{t+1}(x', a')]$

Robbins-Monro condition: $\sum \eta_t = \infty$, $\sum \eta_t^2 < \infty$ $\Rightarrow \hat{Q}_t(x,a) \xrightarrow{a.s.} Q_t(x,a)$

① SARSA: on/one: 同时更新 policy & action
off line: 全收集后, 再更新 policy & action.

① PE: $Q_t^{\pi_t^k}(x,a) = f(x,a) + E[Q_{t+1}^{\pi_t^k}(x',a')]$ ② PI: $\pi_t^{k+1}(x) \in \underset{a\in A}{\operatorname{argmax}}\, Q_t^{\pi_t^k}(x,a)$

(SARSA): $\delta_t = f_t + \hat{Q}_t(x',a') - \hat{Q}_t(x,a)$; (Q-learning) $\delta_t = f_t + \sup_a \hat{Q}_t(x',a') - \hat{Q}_t(x,a)$

② policy:

① ε-greedy policy: $a_t \sim \begin{cases} \underset{a\in A}{\operatorname{argmax}}\, Q_t(x,a) & \text{with } 1-\varepsilon \quad \text{- exploitation} \\ U(A) & \text{with } \varepsilon \quad \text{- exploration.} \end{cases}$ (uniform distribution)

② softmax policy: $a_t \sim \pi_t(x)$ $\pi_\theta(x,a) = \dfrac{\exp\left(\frac{Q_t(x,a)}{\lambda}\right)}{\sum_{a\in A}\exp(Q_t(x,a')/\lambda)}$ $a\in A$

$\lambda \to \infty$ constant explore; $\lambda \to 0^+$ extreme greedy; temperature parameter.

② parameterized randomized policy: Score.

discrete ① Softmax: $\pi(t,x,a) = \dfrac{\exp(\psi_\theta(t,x,a))}{\sum \exp(\psi_\theta(t,x,a'))} \Rightarrow \nabla \ln \pi_\theta(t,x,a) = \nabla \psi_\theta(t,x,a) - E[\nabla \psi_\theta(t,x,\cdot)]$

continuous ② Gaussian: $a \mapsto \pi(t,x,a) \sim N(\psi(t,x),\theta^2) \Rightarrow \nabla \ln \pi_\theta(t,x,a) = \dfrac{(a-\psi_\theta(t,x))\nabla\psi_\theta(t,x)}{\theta^2}$

$J(\theta) = E_{\pi_\theta}\left[\sum_{t=0}^{T-1} f(x_t,a_t) + g(x_T)\right]$ $R(s) = \sum_{t=0}^{T-1} f(x_t,a_t) + g(x_T)$ $J(\theta) = E[R(s)]$

$\nabla J(\theta) = E_{\pi^\theta}\left[R(s)\sum_{t=0}^{T-1}\nabla_\theta \ln \pi^\theta(t,x_t,a_t)\right]$

$$\nabla_\theta J(\theta) = \int R(s)\nabla_\theta p_\theta(s) Q(\mathrm{ds}) = \int R(s)\left[\nabla_\theta \ln p_\theta(s)\right] p_\theta(s) Q(\mathrm{ds})$$
$$= \mathbb{E}_{\pi_\theta}\left[R(S)\nabla_\theta \ln p_\theta(S)\right] = \mathbb{E}_{\pi_\theta}\left[R(S)\sum_{t=0}^{T-1}\nabla_\theta \ln \pi_\theta(t,x_t,a_t)\right].$$

③ Reinforce Algorithms:

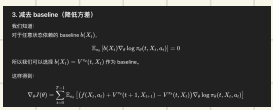
① Sampling trajectory: $S^k = (x_0^k, a_0^k, x_1^k, \dots, x_T^k)$ $a_t^k \sim \pi_\theta(t,x_t^k,\cdot)$ $x_{t+1}^k \sim P_t(x_t^k,a_t^k,\cdot)$ $f_t^k = f(x_t^k,a_t^k)$

$R(S^k) = \sum_{t=0}^{T-1} f_t^k + f_T^k$

② $\theta \leftarrow \theta + \eta\left[\frac{1}{K}\sum_{k=1}^{K} R(S^k)\sum_{t=0}^{T-1}\nabla_\theta \ln \pi_\theta(t,x_t^k,a_t^k)\right]$

④ Actor-Critic network approach: value function (one critic); policy (actor) by NN.

by PE — by policy gradient in direction.



⑤ Basics of SC: DS dynamical system: $x(t_0); t$; $\frac{dx}{dt} = f(x,t)$, $x_{t+1} = F(x_t,t)$

optimization of DS in a random environ.

① By SDEs: $dX_t = b(X_t,a_t)\,dt + \sigma(X_t,a_t)\,dW_t$ → SDS

State dynamics — drift term: 平均变化率; diffusion term: 不确定性强度; W_t Brownian motion 随机噪声; $dW_t \sim N(0,dt)$

② objective: $J(\alpha) = E\left[\int_0^T f(x_t,a_t)\,dt + g(x_T)\right]$ max/min over $\alpha = (a_t)_t$

E: average; running cost summup; terminal reward

② DL for SC & PDEs

① For MDP

Algo NNContP : control learning by policy iteration.

Algo Actor / Critic

$$\mathbb{E}_M|V_t(X_t) - \hat{V}_t^M(X_t)| = \mathcal{O}_{\mathbb{P}}\left(\underbrace{\left(\gamma^4 K \frac{\ln M}{M}\right)^{\frac{1}{2(T-t)}}}_{\text{statistical error}} + \underbrace{\varepsilon_t^{NN}(\pi^*)}_{\text{policy approximation error}} + \underbrace{\varepsilon_t^{NN}(V)}_{\text{value approximation error}}\right)$$

DNN+ SGD Backward induction

$\hat{\pi}_t \in \arg\max_{\pi} \mathbb{E}[f(X_t, \pi(X_t)) + \sum_{s=t+1}^{T-1} f(\hat{X}_s, \hat{\pi}_s(\hat{X}_s)) + g(\hat{X}_T)]$

$\hat{\pi}_t \in \arg\max_{\pi} \mathbb{E}[f(X_t, \pi(X_t)) + \hat{V}_{t+1}(X_{t+1})]$

$$\mathbb{E}_M|V_t(X_t) - \hat{V}_t^M(X_t)| = \mathcal{O}_{\mathbb{P}}\left(\underbrace{\gamma^{T-t}\sqrt{\frac{\ln M}{M}}}_{\text{statistical error}} + \underbrace{\sup_{s=t,\dots,T-1} \inf_{p\in\mathcal{A}_K^\gamma} \|p(X_s) - \pi_s^*(X_s)\|_{L^1}}_{\varepsilon_t^{NN}(\pi^*)\text{ approximation error}}\right)$$

② For PDE

$\partial_t v(t,x) = \mathcal{M}(v)(t,x) = H(t,x,v,\nabla_x v, \dots)$, (+ boundary conditions)

dif wrt t — PDE — 初值条件+边界条件.

① PDE $[0,T] \times \mathbb{R}^d$; objective function $v(t,x)$

DNN approximates $v(t,x)$ / $\nabla_x v$

① loss function → ② SGD on X (domain) ③ Tensorflow automatic differentiation

② $\begin{cases} \partial_t v(t,x) = \mathcal{M}(v)(t,x) & (t,x) \in [0,T] \times \mathbb{R}^d \\ v(T,x) = g(x) & x \in \mathbb{R}^d \end{cases}$

→ NN ⇒ PDE

loss function: $\ell(\theta,t,x) = |u_\theta(T,x) - g(x)|^2 + |\partial_t u_\theta(t,x) - \mathcal{M}[u_\theta](t,x)|^2$

① why PDE:
independent variables: PDEs - t & x + $\partial_t V$; $\nabla_x V$

PDE + terminal condition / boundary condition ⇒ unique solution

② NN approximation:
NN $u_\theta(t,x) \to V(t,x)$

input: t, x → output $V(t,x)$

autodiff: time derivative $\partial_t u_\theta$; space derivative $\nabla_x u_\theta$

2nd derivative $\nabla_x^2 u_\theta$

③ loss function:
NN satisfy: ① PDE ② boundary / terminal condition.

(PINN loss) $L(\theta) = E[|u_\theta(T,x) - g(x)|^2 + |\partial_t u_\theta - \mathcal{M}[u_\theta]|^2]$

terminal condition error ($t \to T \Rightarrow g(x)$) — PDE error (其它部分 NN error.)

(DGM loss) 核和loss一致 + Monte Carlo Sampling

④ Optimisation:
domain $[0,T] \times \mathbb{R}^d \to (t,x)$ Sampling randomly

Calculate PDE error + terminal condition error ⇒ $\theta_{k+1} = \theta_k - \eta_k \cdot$ (loss gradient)

5. DGM vs PINN

	DGM	PINN
采样方式	Monte Carlo 随机采样	固定网格或采样点
优点	高维问题不需要网格，灵活	容易融合物理定律约束
典型应用	金融、控制	物理模拟、工程 PDE

⑤ With control problems:

① In stochastic control, optimal function V satisfy PDE (HJB function)

② traditional measure: PDE ⇒ curse of dimensionality

③ DGM / PINN NN approximates V via sampling

③ RL methods in continuous times.

① Problem Setting:

① continuous MDP: $dX_t = b(X_t, a_t)dt + \sigma(X_t, a_t)dW_t$

- X_t: state, a_t: action/control (continuous), b, σ

② objective: $J(\alpha) = \mathbb{E}\left[\int_0^T f(X_t, a_t)dt + g(X_T)\right]$

② policy:

① Deterministic policy: $a_t = \pi_t(X_t)$ ② randomized policy: $a_t \sim \pi_t(x)$

③ Value function & HJB PDE:

$V^{\pi}(t,x) = \mathbb{E}^{\pi}\left[\int_t^T f(X_s, a_s)ds + g(X_T) \mid X_t = x\right]$

HJB PDE: $\partial_t V + \sup_a \left[b(x,a)\cdot \nabla_x V + \frac{1}{2}\mathrm{Tr}(\sigma\sigma^T \nabla_x^2 V) + f(x,a)\right] = 0 \quad V(T,x) = g(x)$

④ Bellman Principle in Continuous Time

任意时刻的最优策略 = 从当前状态往后看仍然是整体最优策略 ⇒ HJB form

⑤ RL approaches in continuous time.

① Policy Gradient Method: $\theta \leftarrow \theta + \eta \nabla_\theta J(\pi_\theta)$ ⇒ 学习 Ito's formula + Stochastic Calculus

② Actor-Critic Method: Actor: policy + Critic: Value Function ⇒ on-policy

③ Q-learning in Continuous time: ① Define $Q(t,x,a)$ ⇒ Bellman Equation → ② Sampling + Approximate NN

⑥ Function Approximation

① NN approximate $V(t,x)$ / $Q(t,x,a)$

HJB PDEs:

System: $dX_t = b(X_t, a_t)dt + \sigma(X_t, a_t)dW_t$

Bellman: 现在 optimal + 未来 optimal = global optimal. $V(t,x) = \sup_a \mathbb{E}\left[\int_t^T f(X_s, a_s)ds + g(X_T) \mid X_t = x\right]$

HJB PDE 就是一个方程，它的解。$V(t,x)$ 告诉你在任意时间、任意状态下能得到的最优总收益；解这个 PDE，就等于解决了整个最优控制问题。