Policy Gradient Methods

- brand new
- learn parametrized policy, select actions without a value function
+ value function still useful for learning the policy, but unnecessary for action selection
x $\theta \in \mathbb{R}^{d'}$ for policy's parameter vector
x

$$\pi(a|s,\theta) = Pr\{A_t = a \mid S_t = s, \theta_t = \theta\}$$

- assign numerical preference, highest probability is selected
+

$$\pi(a|s,\theta) \doteq \frac{e^{h(s,a,\theta)}}{\sum_b e^{h(s,b,\theta)}}$$

e- softmax

+ $h(s,a,\theta)$ can be parametrized in any way
x Deep ANN (AlphaGo)
x if linear function $h(s,a,\theta) = \theta^T x(s,a)$
- parameterized policies can become deterministic, unlike ε-greedy
+ or, can stay in arbitrary preference, which is necessary in imperfect info.
+ sometimes, just easier
+ also, easier to inject a-priori info

The Policy Gradient Theorem

- theoretically stronger
+ continuous policy parametrization, action probs change smoothly as a function of learned param
+ ε-greedy, changes are abrupt, especially when new max is involved
+ slower, smoother change = greater convergence
x continuity also helps approximate gradient descent
- episodic
+ performance defined as value of start state

$$J(\theta) \doteq v_{\pi_\theta}(s_0)$$

v_{π_θ} is the true value function for π_θ, policy determined by θ

x assume no discounting
- func approx, how to guarantee improvement
+ action selection and state distribution are both impacted by policy param
x impact on action and reward is easy to compute
x on state dist, usually dep on env, unknown
- policy gradient theorem expresses gradient of performance w.r.t. policy parameter (for gradient ascent) w/o state distribution

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla\pi(s,a|\theta)$$

↳ proportional to avg length of ep

REINFORCE: MC Policy Gradient

- stochastic gradient ascent relies on samples whose overall expectation of sample gradient ∝ true gradient

$$= \mathbb{E}_\pi\left[\sum_a q_\pi(S_t,a) \nabla v_\pi(a|S_t,\theta)\right]$$

- could just become

$$\theta_{t+1} \doteq \theta_t + \alpha \sum_a \hat{q}(S_t,a,w) \nabla\pi(a|S_t,\theta)$$

$\hat{q}$ learned approx of q_π

+ technically called an all-actions method
x REINFORCE uses just one action A_t
- same principle as before: if using A_t, not a, must scale expectation of sample

$$\nabla J(\theta) \propto \mathbb{E}_\pi\left[\sum_a \pi(a|S_t,\theta) q_\pi(S_t,a) \cdot \frac{\nabla\pi(a|S_t,\theta)}{\pi(a|S_t,\theta)}\right]$$

$$= \mathbb{E}_\pi\left[q_\pi(S_t,A_t) \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}\right]$$

$$= \mathbb{E}_\pi\left[G_t \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t,S_t,\theta)}\right]$$

- REINFORCE update

$$\theta_{t+1} \doteq \theta_t + \alpha G_t \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta_t)}$$

- intuitive!
+ increment is proportional to return times gradient of probability of taking that action divided by probability of that action
x vector is direction in param space most increases probability of repeating action A_t on future visits of S_t
x update increases parameter in direction proportional to return, inverse to probability
* reinforce direction, good
* decrease advantage of frequently selected actions otherwise would overpower even with lower reward
- REINFORCE uses G_t, so full return, thus MC
- code uses $\nabla \ln \pi(A_t|S_t,\theta)$
+ $\nabla \ln x = \frac{\nabla x}{x}$, called eligibility vector
- good convergence with small enough α

REINFORCE with Baseline

- generalize to compare to a baseline
- $\nabla J(\theta) \propto \sum_s \mu(s) \sum_a (q_\pi(s,a) - b(s)) \cdot \nabla\pi(a|s,\theta)$
+ baseline $b(s)$
x any function, should not vary with a

$$\theta_{t+1} = \theta_t + \alpha(G_t - b(S_t)) \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$$

* can significantly reduce variance, & help with action differentiation in high value scenarios
* akin to gradient bandit using avg reward to compare
- good baseline: $\hat{v}(S_t,w)$ with $w \in \mathbb{R}^d$
+ just use an MC state-value method to compare to.

Actor-Critic Methods

- REINFORCE with baseline uses only initial state of transition
+ can't evaluate action
- actor-critic uses second state in transition, essentially capturing $G_{t:t+1}$
+ essentially evaluating actions
- TD one step return better for variance, though it adds bias
- use state-value becomes critic
- overall policy-gradient method called actor-critic

- one-step actor-critic
+ remember, always online, incremental

$$\theta_{t+1} \doteq \theta_t + \alpha\left(G_{t:t+1} - \hat{v}(S_t,w)\right) \frac{\nabla\pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$$

$$\doteq \theta_t + \alpha\left(R_{t+1} + \gamma\hat{v}(S_{t+1},w) - \hat{v}(S_t,w)\right) \frac{\nabla\pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$$

$$= \theta_t + \alpha\delta_t \frac{\nabla\pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$$

- natural, easy state value func for pairing is semi-gradient TD(0)
- generalize to n-step or λ-return are easy
+ $G_{t:t+1} \rightarrow G_{t:t+n}$ or G_t^λ
+ backward view for eligibility trace also simple, keeping trace for actor & critic

Policy Gradient for Continuing Problems

- continuing case, performance is now average rate of reward per time step

$$J(\theta) \doteq r(\pi) \doteq \lim_{h\to\infty} \frac{1}{h} \sum_{t=1} \mathbb{E}[R_t \mid A_{0:t-1} \sim \pi]$$

$$= \lim_{t\to\infty} \mathbb{E}[R_t \mid S_0, A_{0:t-1} \sim \pi]$$

$$= \sum_s \mu(s) \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) r$$

- $\mu(s)$ is steady state dist under π
+ $\mu(s) \doteq \lim_{t\to\infty} Pr\{S_t = s \mid A_{0:t} \sim \pi\}$
+ assumed to exist & be independent of S_0 (ergodicity assumption)
x $\sum_s \mu(s) \sum_a \pi(a|s,\theta) p(s'|s,a) = \mu(s')$
- $v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s]$

$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$

$G_t \doteq R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r(\pi)$

- given all these alternate definitions, policy gradient theorem still holds

Policy Parametrization for Continuous Actions

- policy good for practical ways of handling large action spaces / continuous action spaces
+ $A \in \mathbb{R}$, normally distributed, e.g.
+ don't compute $pr(a)$, compute statistics of probability distribution / function
+ value $p(x)$ is density of probability of x, not probability of x
- policy defined as normal probable density over a real-valued scalar action, with mean & standard deviation by func approximators that depend on state

$$\pi(a|s,\theta) \doteq \frac{1}{\sigma(s,\theta)\sqrt{2\pi}} \exp\left(-\frac{(a-\mu(s,\theta))^2}{2\sigma(s,\theta)^2}\right)$$

$$\mu : \mathcal{S} \times \mathbb{R}^{d'} \to \mathbb{R} \quad \sigma : \mathcal{S} \times \mathbb{R}^{d'} \to \mathbb{R}^+$$

- how to form approximators
+ $\theta = [\theta_\mu, \theta_\sigma]^T$
+ mean can be linearly approximated
+ sd must be positive, better to use exponential of linear

$$\mu(s,\theta) \doteq \theta_\mu^T x_\mu(s)$$

$$\sigma(s,\theta) \doteq \exp(\theta_\sigma^T x_\sigma(s))$$

$x_\mu(s)$, $x_\sigma(s)$ state feature vectors constructed with some kind of coding

Summary

- prior, all about action value methods
- now, parametrized policy for action decisions
+ policy gradient methods: update policy parameter in direction of estimate of gradient on performance
- policy gradient theorem shows performance formula without being affected by state distribution
- REINFORCE is simplest impl
+ with baseline reduces variance without bias
+ uses state-value func for baseline = policy → actor, state-value func → critic
- different strengths & weaknesses