RECALL: REINFORCE WITH BASELINE.

Want: X, Y positively correlated;

FOR REINFORCE:

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{L-1} \gamma^t \Big(Q^{\pi_\theta}(s_t, a_t) - \underbrace{f(s_t)}_{\text{Baseline}} \Big) \nabla \log \pi_\theta (a_t | s_t)$$

Here: $f : S \to \mathbb{R}$ is a function indep of τ a traj

For any function $g(s)$,

$$\mathbb{E}\Big[\nabla \log \pi(a|s)\, g(s) \Big]$$

$$= \sum_a \pi(a|s)\, \nabla \log(\pi(a|s))\, g(s)$$

$$= \sum_a \frac{\cancel{\pi(a|s)}}{\cancel{\pi(a|s)}} \nabla \pi(a|s)\, g(s)$$

$$= g(s) \sum_a \nabla \pi(a|s) = g(s)\, \nabla \sum_a \pi(a|s)$$

$$= g(s)\, \nabla(1) = \underline{\underline{0}}.$$

Recall - Reinforce algorithm.

Algorithm: Stochastic Gradient Ascent on J.

REINFORCE -

- Initialize θ arbitrarily.
- for each episode do

Generate $S_0\, A_0\, R_0\, S_1 A_1 \cdots S_{L-1} A_{L-1} R_{L-1}$

using θ.

For each t compute $G_t = \overbrace{q^{\pi_\theta}(s_t, a_t)}$

$$\widehat{\nabla J(\theta)} = \sum_{t=0}^{L-1} \gamma^t G_t \frac{\partial \ln \pi(S_t | A_t ; \theta)}{\partial \theta}$$

$$\theta = \theta + \alpha \widehat{\nabla J(\theta)}.$$

end

The modification with baseline:

$$\theta_{t+1} \doteq \theta_t + \alpha\left(G_t - b(S_t)\right)\frac{\nabla\pi(A_t|S_t;\theta)}{\pi(A_t|S_t;\theta)}$$

A natural choice of the baseline is an estimate of the state value $\hat{v}(S_t, w)$, where $w \in \mathbb{R}^d$ is a wt vector learned by a method to be described!

Softmax policies: 1) $\pi_\theta(a|s) = \dfrac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$ | $\Theta = \mathbb{R}^{|S||A|}$

2) Linear softmax policies:

$$\pi_\theta(a|s) = \frac{\exp(\theta \cdot \phi_{s,a})}{\sum_{a'} \exp(\theta \cdot \phi_{s,a'})}$$

For each s,a we have a feature map $\phi_{s,a} \in \mathbb{R}^d$; then $\theta \in \mathbb{R}^d$.

3) Neural softmax policies:

$$\pi_\theta(a|s) = \frac{\exp(f_\theta(s,a))}{\sum_{a'} \exp(f_\theta(s,a'))}$$

$f_\theta(s,a)$ parametrized by a neural network with $\theta \in \mathbb{R}^d$

Recall: in policy gradient, given a trajectory τ, we defined $R(\tau) = (1-\gamma) \sum_{t=0} \gamma^t r(s_t, a_t)$

discounted total reward

and $V^{\pi_\theta}(\mu) = \mathbb{E}_{\tau \sim \Pr_\mu^{\pi_\theta}} [R(\tau)]$

and we were optimizing $V^{\pi_\theta}(\mu)$ (i.e.) maximizing expected discounted total reward.

Using softmax policies,

$$\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d^{\pi_\theta}(s) \pi_\theta(a|s) A^{\pi_\theta}(s,a)$$

check: easy calculation

- However the gradients with respect to $\theta_{s,a}$ may be small even if $A(s,a)$ is large because $\pi_\theta(s|a)$ is small;

- To prevent probabilities from getting too small add a regularizer and optimize

$$L_\lambda(\theta) := V^{\pi_\theta}(\mu) - \lambda \mathbb{E}_{s \sim \mathrm{Unif}_S}\left[KL(\mathrm{Unif}_A, \pi_\theta(\cdot|s)) \right]$$

$$KL(p,q) := \mathbb{E}_{x\sim p} -\log\left(\frac{q(x)}{p(x)}\right)$$

Policy gradient for $L_\lambda(\theta)$:

$$\theta^{t+1} \leftarrow \theta^t + \eta \nabla_\theta L_\lambda(\theta^t).$$

THM: Starting with any initial θ^0 and using

$$\lambda = \frac{\varepsilon(1-\gamma)}{2\left\|\frac{d_\rho^{\pi^*}}{\mu}\right\|_\infty} \quad \text{and} \quad \eta = 1/\beta_\lambda, \quad \beta_\lambda := \frac{8\gamma}{(1-\gamma)^2} + \frac{2\lambda}{|S|}$$

we have, for all starting distributions ρ,

$$\min_{t<T} \left\{ V^*(s) - V^t(s) \right\} \le \varepsilon, \text{ for } T \ge O\left(\frac{|S|^2|A|^2}{(1-\gamma)^6\varepsilon^2} \left\|\frac{d_s^{\pi^*}}{\mu}\right\|_\infty^2\right)$$

Natural gradients:

Idea: $$f(x+y) = f(x) + \nabla f(x)^T y + \frac{1}{2} y^T H(f)_x y.$$

Symm matrix.

A stationary pt in the nbd satisfies:

$$\nabla f(x) + H(f)_x y = 0.$$

$$\boxed{\therefore\ y = H(f)_x^{-1} \nabla f(x).}$$

- The metric used to measure distances in the nbd is the one induced by the Hessian of f at x.

- Amari argued that for policy gradient algorithms, we shouldn't use the standard Euclidean metric;

If the function being optimized is the loss function of a parametrized distribution $f(\pi_\theta)$, f: loss, π_θ the distribution, Amari suggested using FISCHER INFORMATION MATRIX, as the choice for the symm matrix giving us the bilinear form;

- for each $s \in S$, $\pi_\theta(\cdot \mid s)$ is a distribution.

$$\mathcal{F}_\mu^\theta \triangleq \mathbb{E}_{s \sim d_\mu^{\pi_\theta}} \mathbb{E}_{a \sim \pi_\theta(\cdot \mid s)} \left[\nabla \log \pi_\theta(a \mid s) \nabla \log\left(\pi_\theta(a \mid s)\right)^T \right]$$

- Natural policy gradient: NPG

$$\boxed{\theta \leftarrow \theta + \eta \left(F_\mu^\theta\right)^\dagger \nabla V^{\pi_\theta}(\mu).}$$

- Motivation for using that?

For a distribution ν over $S \times A$, and $w \in \mathbb{R}^d$ ($d =$ the # parameters), define the compatible function approximator error L_ν and minimal compatible function approximation error L_ν^* as:

$$L_\nu(\omega; \theta) = \mathbb{E}_{s,a \sim \nu}\left[\left(A^{\pi_\theta}(s,a) - \omega^T \nabla_\theta \log \pi_\theta(a|s)\right)^2\right]$$

$$\text{eg. } \underbrace{\nu(s,a) = d_\mu^{\pi_\theta} \pi_\theta(a|s)}$$

$$\left(\underbrace{A^{\pi_\theta}(s,a) - \omega \nabla_\theta \log \pi_\theta}(a|s)\right)^2$$

$$= \left(A^{\pi_\theta}(s,a) - \omega^T \nabla_\theta \log \pi_\theta(a|s)\right)\left(A^{\pi_\theta}(s,a) - \nabla_\theta \log \pi_\theta(a|s)^T \cdot \omega\right)$$

$$\nabla_\omega : -2 \nabla_\theta \log(\pi_\theta(a|s)) \cdot A^{\pi_\theta}(s,a)$$

$$+ 2 \nabla_\theta \log \pi_\theta(a|s) \cdot (\nabla_\theta \log \pi_\theta(a|s))^T \omega$$

For a stationary point:

∇_ω :

$$\mathbb{E}_{s\sim d^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[-A^{\pi_\theta}(s,a)\, \nabla_\theta \log \pi_\theta(a|s) + \nabla_\theta \log \pi_\theta(a|s)\, \nabla_\theta \log \pi_\theta(a|s)^T \omega \right]$$

$= 0$ iff.

$$\underbrace{\mathbb{E}_{s\sim d^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} A^{\pi_\theta}(s,a)\, \nabla_\theta \log \pi_\theta(a|s)}_{\nabla V^{\pi_\theta}(\mu)} = \left[\mathbb{E}_{s\sim d^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \nabla_\theta \log \pi_\theta(a|s)\, \nabla_\theta \log \pi_\theta(a|s)^T \right] \omega$$

$$\boxed{\therefore\ \omega = \left(\mathcal{F}_\theta^\mu\right)^\dagger \nabla V^{\pi_\theta}(\mu).}$$

Letting $\theta \in \mathbb{R}^{S\times A}$ and using softmax policy and NPG

$$\left(\mathcal{F}_\theta^\mu\right)^\dagger \nabla V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma} A^{\pi_\theta}(s,a) - \underbrace{c(s)}_{\text{fn of state}}.$$

A simple update

$$\pi^{(t+1)}(a|s) = \frac{\pi^t(a|s)\exp\left(\eta A^t(s,a)/(1-\gamma)\right)}{Z_t(s)}$$

$$= \sum_{a \in A} \pi^t(a|s)\exp\left(\eta \frac{A^t(s,a)}{(1-\gamma)}\right)$$

<u>Thm:</u> For softmax policy class, perform the update $\theta \Leftarrow \theta + \eta (F_\mu^\theta)^\dagger \nabla V^{\pi_\theta}(\mu)$

using $\mu \in \Delta(S)$

$$V^T(\mu) \geq V^*(\mu) - \frac{\log(A)}{\eta T} - \frac{1}{(1-\gamma)^2 T}.$$

Set $\eta = (1-\gamma)^2 \log|A|$, then we get to within ε in $\frac{2}{(1-\gamma)^2\varepsilon}$ steps

(Eq 1)

$$\frac{1}{1-\gamma}\sum_s d^{\pi}(s)\sum_a \frac{\partial \pi_\theta(s,a)}{\partial\theta}\cdot Q^{\pi_\theta}(s,a)$$

$$\shortparallel$$

- Recall

$$\nabla V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}\ \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[Q^{\pi_\theta}(s,a)\ \nabla\log\pi_\theta(a|s)\right]$$

or $\nabla\log\pi_\theta(s,a)$

Suppose we have $f_w : S\times A \to \mathbb{R}$ is an approximator to Q^{π}, with parameter w.

Natural: learn f_w, by following π & updating w by

$$\Delta w_t \propto \frac{\partial}{\partial w}\left[\hat{Q}^{\pi}(s_t,a_t) - f_w(s_t,a_t)\right]^2$$

$$\propto \left[\hat{Q}^{\pi}(s_t,a_t) - f_w(s_t,a_t)\right]\frac{\partial f_w(s_t,a_t)}{\partial w}.$$

some unbiased estimator

So when we converge to a local minima,

$$\sum_s d^{\pi}(s)\sum_a \pi(s,a)\left[Q^{\pi}(s,a) - f_w(s,a)\right]\frac{\partial f_w(s,a)}{\partial w} = 0. \quad **$$

Suppose we find $f_w(s,a)$ s.t.:

$$\nabla V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma}\sum_s d^{\pi}(s)\sum_a \frac{\partial\pi_\theta(s,a)}{\partial\theta} f_w(s,a) \quad *$$

Its clear that choosing f_w s.t

$$\frac{\partial f_w(s,a)}{\partial w} = \frac{1}{\pi(s,a)} \frac{\partial \pi(a,s)}{\partial \theta},$$ we can satisfy

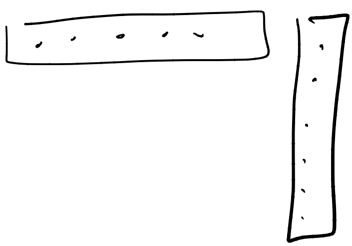
* assuming **;

<u>Proof:</u> From ** and $\frac{\partial f_w(s,a)}{\partial w} = \frac{1}{\pi(s,a)} \frac{\partial \pi(a,s)}{\partial \theta}$

we get

$$\sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} \left[Q^\pi(s,a) - f_w(s,a) \right] = 0$$

Eq (2)



error is $\perp$ gradient of policy;

<u>Eq(1) - Eq(2)</u>

$$= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} \left[Q^\pi(s,a) - Q^\pi(s,a) + f_w(s,a) \right]$$

$$= \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} f_w(s,a)$$

· The gradient has the form as in the red expression.

So a good approximator is

$$f_w(s,a) = \underbrace{w^T \nabla_\theta \ln \pi_\theta(s,a)}$$

then $\dfrac{\partial f_w(s,a)}{\partial w} = \dfrac{1}{\pi_\theta(s,a)} \nabla_\theta \pi_\theta(s,a)$

· Is the motivation for compatible function approximator definition:

Thm: If the Q value function approximator is compatible to the policy (π) $\nabla_w f_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$

and the parameters w minimize

$$\mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - f_w(s,a)\right)^2\right]$$ then

policy gradient is exact

$$\nabla_\theta V^\theta(\rho) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\, f_w(s,a)\right]$$

- Recall the general value function update

$$v(S_t) \leftarrow v(S_t) + \alpha\left[G_t - V(S_t)\right]$$

<u>Comes from:</u> Maximizing

$$\mathbb{E}\left[\frac{1}{2}\left(v^{\pi}(s) - v(s)\right)^2\right]$$

$$v \leftarrow v + \alpha\,\mathbb{E}\left[\left(v^{\pi}(s) - v(s)\right)\frac{\partial v(s)}{\partial v}\right]$$

- Do a stochastic gradient update at time t since we don't know $v^{\pi}(s)$;

Get an unbiased estimate - sample s & use G_t instead of $v^{\pi}(S_t)$

$$\therefore \boxed{v \leftarrow v + \alpha\left(G_t - V(S_t)\right)\frac{\partial v(S_t)}{\partial v}}$$

If w parametrizes values of states,

$\underbrace{v_w : S \to \mathbb{R}}_{\text{function approximator.}}$,

get update: $\boxed{w \leftarrow w + \alpha \left(G_t - v_w(S_t)\right) \frac{\partial v_w(S_t)}{\partial w}}$

- TD Learning: Bootstrapping method. bases its update on existing estimate.

An evaluation alg, estimates v^π

TD update: $v(s) \leftarrow \underbrace{v(s) + \alpha\left(r + \gamma v(s') - v(s)\right)}$

(i.e.) $v(S_t) \leftarrow v(S_t) + \alpha\left[R_t + \gamma V(S_{t+1}) - V(S_t)\right]$

- Instead of using G_t - use $R_t + \gamma V(S_{t+1})$

$\underbrace{\delta_t = R_t + \gamma v(S_{t+1}) - v(S_t)}$. TD error.

- Both TD & MC use estimates: $V(S_{t+1})$ in TD
G_t in MC

. Note if we used v^{π}, $\mathbb{E}[\delta_t \mid S_t = s] = 0$.

TD update with function approximator:

$$w \leftarrow w + \alpha \left(R_{t+1} + \gamma v_w(s_{t+1}) - v_w(s_t) \right) \frac{\partial v_w(s_t)}{\partial w}$$

. TD is not really a gradient update algorithm

. Has good convergence properties; → SHOWN TO BE CONVERGENT TO V_{π} ON AN average.

. Converges when using linear function approximation; say $V_w(s) = w^T \underbrace{\phi(s)}_{\text{features extracted from } s}$;

– Comparison with Monte Carlo estimation of $\Pr[s, a, s']$ & $R[s, a]$?

TD converges to the same as $\hat{V}^{\pi}$ if every state is seen at least once;

- TD - only $|S|$ values;
- Dynamic / Tabular $|S|^2|A|$ values;

- SARSA - TD for control:

$$q(s,a) \leftarrow q(s,a) + \alpha\left[r + \gamma q(s',a') - q(s,a)\right]$$

stochastic update;

Using function approximator:

$$w \leftarrow w + \alpha\left(r + \gamma q_w(s',a') - q_w(s,a)\right)\frac{\partial q_w}{\partial w}\Big|_{s,a}$$

Gne:

- Initialize $q(s,a)$;

for each episode do (ε greedy / softmax)

  $\pi \leftarrow$ Policy(q)

  $s \sim d_0$

  Choose a

  for each time step until s is absorbing do

    Take action a, get r & get s'

    Choose a' from s' using π

    $q(s,a) \leftarrow q(s,a) + \alpha\left[r + q(s',a') - q(s,a)\right]$

    $s \leftarrow s'$; $a \leftarrow a'$

- Similarly SARSA with function approximator; REINFORCE with baseline for estimating $\pi_\theta \simeq \pi_*$.

Input: A differentiable policy parametrization $\pi(a|s,\underline{\theta})$ & state value parametrization $\hat{v}(s,w)$

Parameters: $\alpha^\theta > 0$, $\alpha^w > 0$

Initialize: θ & w

Loop (for each episode):

$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$ following $\pi(\cdot|\cdot,\theta)$

for each step $t = 0, \ldots, T-1$

$$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$$

$$\delta \leftarrow G - \hat{v}(S_t, w)$$

$$w \leftarrow w + \alpha^w \delta \nabla_w \hat{v}(S_t, w)$$

$$\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \nabla \ln \pi(A_t|S_t,\theta).$$

end.

- Q learning : Off policy TD - control.

- gets an estimate of q^*;

$$q(s,a) \leftarrow q(s,a) + \alpha \left[r + \gamma \max_{a'} q(s',a') - q(s,a) \right]$$

Off policy - update done to get to q^* and really independent of π used to generate the data;

- Algorithm as before;

$\boxed{\text{Next TD}(\lambda).}$:

Combine the goodness of MC & TD(0)

updates after the entire episode is run; immediate update

n-step bootstrapping.

- Has the benefits of MC & one-step,
  Actions cause updates instantly in TD(0),
  but variance is large because of this;

- MC- variance is less ∵ updates are done
  at the end.

- In MC updates are performed for each state
  based on the entire sequence of rewards
  from that state to the end of an episode.

- <u>Natural idea:</u> Perform an update based on
  an intermediate number of rewards.

Ex: A two step update, would look at first
two rewards and estimated value of the
state two steps later.

- $$G_{t,t+2} := R_{t+1} + \gamma R_{t+2} + \gamma^2 \underbrace{V_{t+1}(S_{t+2})}$$

the estimate for S_{t+2} takes place of

$\gamma^2 R_{t+3} + \gamma^4 R_{t+4} + \cdots + \gamma^{T-t-1} R_T$;

- target for n-step update is the n-step return

$$G_{t:t+n} := R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \cdots + \gamma^{n-1} R_{t+n-1} + \gamma^n V_{t+n-1}(S_{t+n}).$$

= (if $t+n \geq T$, missing terms are treated as zero)

- Involves future rewards; to use it we need to R_{t+n} and need to compute V_{t+n-1}

- $$\boxed{V_{t+n}(S_t) \doteq V_{t+n-1}(S_t) + \alpha\left[G_{t:t+n} - V_{t+n-1}(S_t)\right] \quad 0 \leq t \leq T}$$

- $V_{t+n}(s) = V_{t+n-1}(s) \quad \forall \; s \neq S_t$;

- No changes during first $n-1$ steps of an episode

TD [n]:

Input: π

Parameters: α, n

Initialize $V(s)$ arbitrarily $\forall \; s \in S$.

Loop for each episode:

Initialize & store $S_0 \neq$ terminal

$T \leftarrow \infty$

Loop for $t = 0, 1, \ldots$

If $t < T$

Take action as per $\pi(\cdot \mid S_t)$

Observe & Store reward as R_{t+1}, next state S_{t+1}

If S_{t+1} is terminal $T \leftarrow t+1$

$\tau \leftarrow t - n + 1$

If $\tau \geq 0$, $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n, T)} \gamma^{i-\tau-1} R_i$

If $\tau + n < T$, $G \leftarrow G + \gamma^n V(S_{\tau+n})$

$V(S_\tau) \leftarrow V(S_\tau) + \alpha [G - V(S_\tau)]$

until $\tau = T - 1$

$$\max_s^{\forall n \geq 1} \left[\mathbb{E}_\pi \left[G_{t:t+n} \mid S_t = s \right] - v_\pi(s) \right] \leq \gamma^n \max_s \left| V_{t+n-1}(s) - v_\pi(s) \right|$$