POLICY GRADIENT

Peaking popularity

Diego Klabjan
Professor, Industrial Engineering and Management Sciences

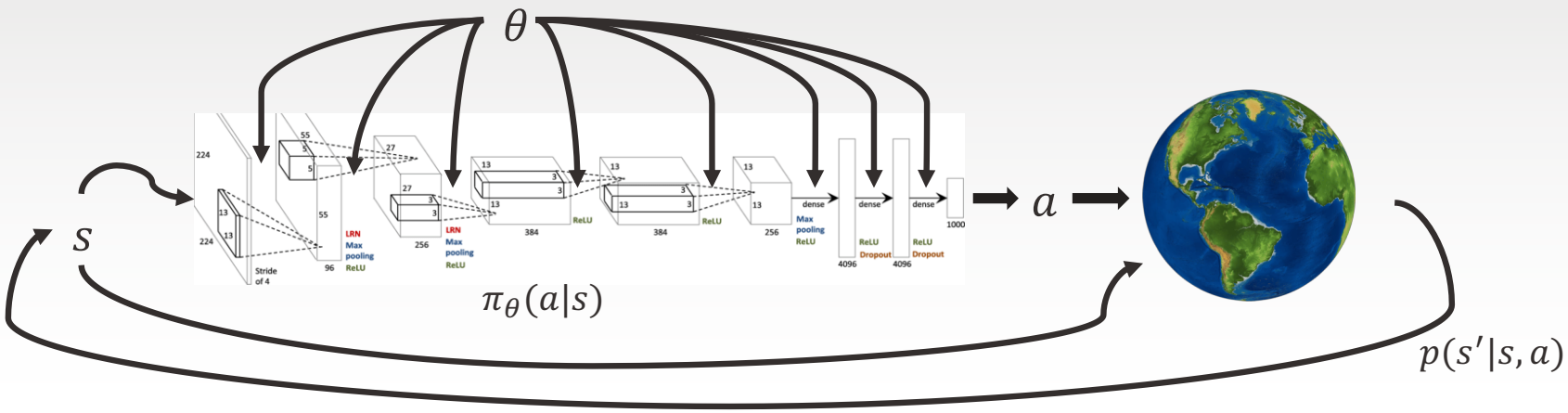
Northwestern | McCORMICK SCHOOL OF ENGINEERING

Outline

- Formulation
- Algorithm
- Issues
- Bias

FORMULATION

Episodes



$$\underbrace{p_\theta(s_1, a_1, \ldots, s_T, a_T)}_{\pi_\theta(\mathcal{T})} = p(s_1)\prod_{t=1}^{T}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

$$\theta^* = \arg\max_\theta E_{\mathcal{T}\sim\pi_\theta(\mathcal{T})}\left[\sum_t r(s_t, a_t)\right]$$

Episodes

$$\theta^* = \underset{\theta}{\text{argmax}}\, E_{\tau \sim \pi_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

$$\theta^* = \underset{\theta}{\text{argmax}}\, E_{(s,a) \sim p_\theta(s,a)} \left[\sum_t r(s, a) \right]$$

infinite horizon case

$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{t=1}^{T} E_{(s_t,a_t) \sim p_\theta(s_t,a_t)} [r(s_t, a_t)]$$

finite horizon case

Telescoping

$$\theta^* = \underset{\theta}{\text{argmax}} \underbrace{E_{\tau \sim \pi_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]}_{J(\theta)}$$

- Assuming everything is finite
  - Otherwise an approximation

$$J(\theta) = E_{\tau \sim \pi_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right] = \frac{1}{N} \sum_i \sum_t r(s_t^i, a_t^i)$$

sum over samples from π_θ

ALGORITHM

Gradient Optimization

- $J(\theta)$ not convex
- Gradient optimization still applicable

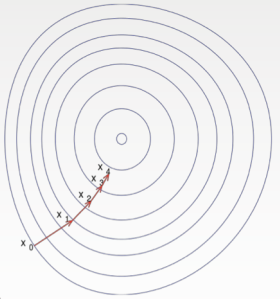
$$\theta = \theta + \alpha \nabla J(\theta)$$

  - α learning rate
- Challenge to find the gradient
- $g(\theta) = E_{z \sim Q} f(\theta, z)$
  - $\nabla g(\theta) = E_{z \sim Q} \nabla_\theta f(\theta, z)$
- What about $g(\theta) = E_{z \sim Q(\theta)} f(\theta, z)$?

Policy Gradient

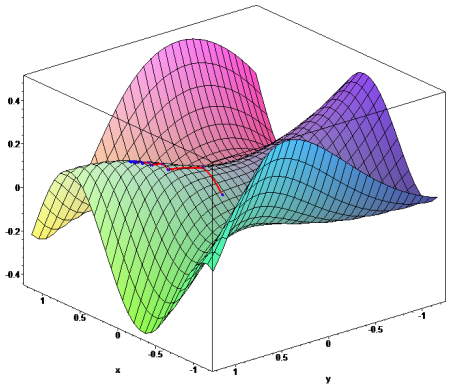
- Policy gradient algorithms search for a local maximum of $J(\theta)$
- Ascending the gradient of the policy with respect to parameters θ

$$\Delta\theta = \alpha \nabla_\theta J(\theta)$$



- $\nabla_\theta J(\theta)$ is the policy gradient

$$\nabla_\theta J(\theta) = \begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_n} \end{pmatrix}$$



- α learning rate parameter

Reparametrization Trick

- $g(\theta) = E_{z \sim N(\mu(\theta), \sigma(\theta))} f(\theta, z)$
- Rewrite as
  - $g(\theta) = E_{w \sim N(0,1)} f(\theta, \mu(\theta) + w \cdot \sigma(\theta))$
- Result
  - $\nabla g(\theta) = E_{w \sim N(0,1)} \nabla_\theta f(\theta, \mu(\theta) + w \cdot \sigma(\theta))$
- Can approximate by samples $w_1, w_2, \cdots, w_N$
  - $\nabla g(\theta) \approx \frac{1}{N} \sum_i \nabla_\theta f\left(\theta, \mu(\theta) + w_i \cdot \sigma(\theta)\right)$
- Does not work for any distribution

Reparametrization Trick

- $g(\theta) = E_{z \sim N(\mu(\theta), \sigma(\theta))} f(\theta, z)$
- $g(\theta) = \int e^{\frac{-(z-\mu(\theta))^2}{2\sigma(\theta)}} f(\theta, z) dz$
- $\nabla g(\theta) = \int \nabla_\theta [e^{\frac{-(z-\mu(\theta))^2}{2\sigma(\theta)}} f(\theta, z)] dz$
- Derivative can be messy
- Not clear how to sample after the derivative
  - Can use numerical integration
  - Sampling better – easier, cleaner

Direct Policy Differentiation

$$\theta^* = \underbrace{\arg\max_\theta E_{\tau\sim\pi_\theta(\tau)}\left[\sum_t r(s_t, a_t)\right]}_{J(\theta)}$$

a convenient identity

$$\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) = \pi_\theta(\tau)\frac{\nabla_\theta \pi_\theta(\tau)}{\pi_\theta(\tau)} = \nabla_\theta \pi_\theta(\tau)$$

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}\underbrace{[r(\tau)]}_{\sum_{t=1}^{T} r(s_t, a_t)} = \int \pi_\theta(\tau) r(\tau) d\tau$$

Unclear how to use reparametrization trick

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau)\, r(\tau) d\tau = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)\, r(\tau)]$$

Direct Policy Differentiation

$$\theta^* = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)\, r(\tau)]$$

$$\underbrace{p_\theta(s_1, a_1, \ldots, s_T, a_T)}_{\pi_\theta(\tau)} = p(s_1)\prod_{t=1}^{T}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

log of both sides

$$\log \pi_\theta(\tau) = \log p(s_1) + \sum_{t=1}^{T}\log \pi_\theta(a_t|s_t) + \log p(s_{t+1}|s_t, a_t)$$

$$\nabla_\theta \left[\cancel{\log p(s_1)} + \sum_{t=1}^{T}\log \pi_\theta(a_t|s_t) + \cancel{\log p(s_{t+1}|s_t, a_t)}\right]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}\left[\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t|s_t)\right)\left(\sum_{t=1}^{T} r(s_t, a_t)\right)\right]$$

Algorithm

$$J(\theta) = E_{\tau \sim p_\theta(\tau)}\left[\sum_t r(s_t, a_t)\right] \approx \frac{1}{N}\sum_i \sum_t r(s_t^i, a_t^i)$$

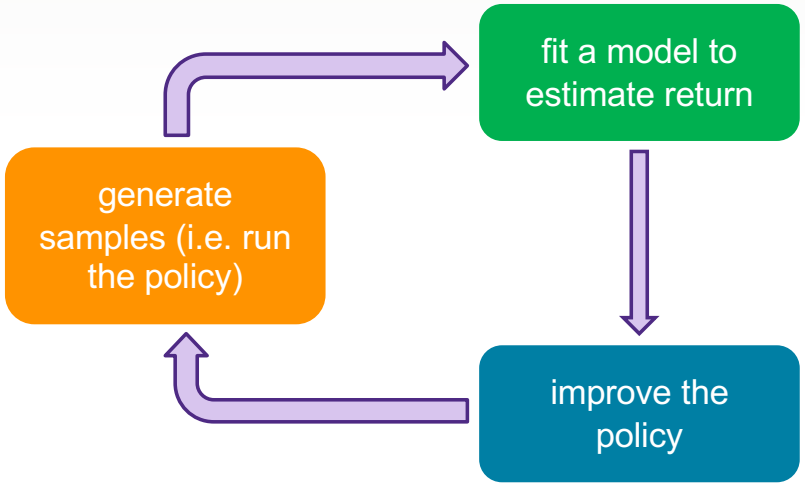
$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)}\left[\left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\right)\left(\sum_{t=1}^{T} r(s_t, a_t)\right)\right]$$

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)\left(\sum_{t=1}^{T} r(s_t^i, a_t^i)\right)$$

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

REINFORCE algorithm:

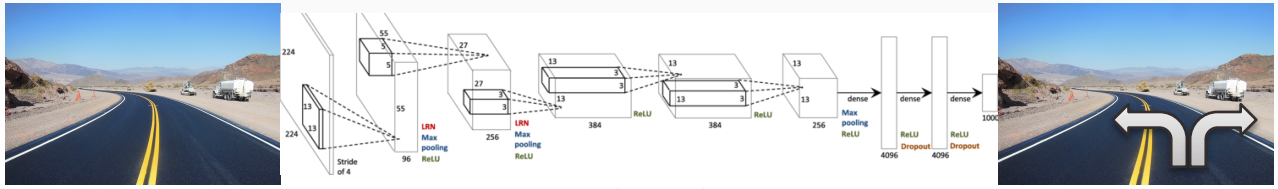
1. sample $\{\tau^i\}$ from $\pi_\theta(a_t|s_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta\left(a_t^i|s_t^i\right)\right)\left(\sum_t r\left(s_t^i|a_t^i\right)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$



Algorithm

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)\left(\sum_{t=1}^{T} r(s_t^i, a_t^i)\right)$$

what is this?



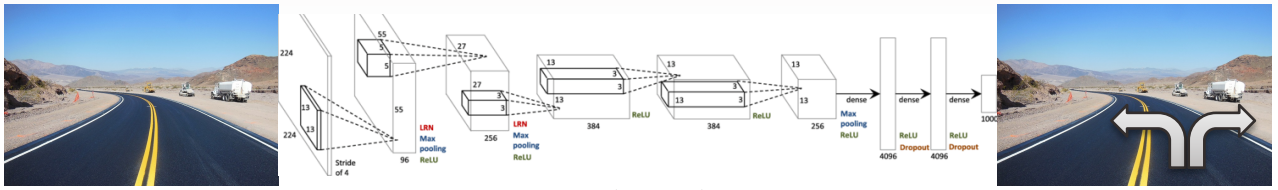
s_t $\pi_\theta(a_t|s_t)$ a_t

- Network produces softmax
- Take log as “loss” function
  - Standard back propagation

Comparison to Maximum Likelihood

policy gradient: $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)\left(\sum_{t=1}^{T} r(s_t^i, a_t^i)\right)$

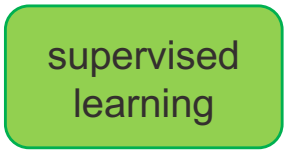
maximum likelihood: $\nabla_\theta J_{ML}(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)$



s_t $\pi_\theta(a_t|s_t)$ a_t

s_t

a_t



$\pi_\theta(a_t|s_t)$

Example: Gaussian Policies

- Network produces mean of Gaussian
- Assume covariance matrix is fixed

$$\pi_\theta(a_t|s_t) = \mathcal{N}(f_{\text{neural network}}(s_t); \Sigma)$$

$$\log \pi_\theta(a_t|s_t) = -\frac{1}{2}\|f(s_t) - a_t\|_\Sigma^2 + \text{const}$$

$$\nabla_\theta \log \pi_\theta(a_t|s_t) = -\frac{1}{2}\Sigma^{-1}(f(s_t) - a_t)\frac{df}{d\theta}$$

$-\frac{1}{2}\Sigma^{-1}(f(s_t) - a_t)(\sum_t r(s_t, a_t)) \cdot$ backpropagation

Softmax Policy

- Weight actions using linear combination of features $\phi(s,a)^T\theta$
- Probability of action is proportional to exponentiated weight

$$\pi_\theta(s,a) \propto e^{\phi(s,a)^T\theta}$$

- Score function

$$\nabla_\theta \log \pi_\theta(a|s) = \phi(s,a) - \mathbb{E}_{\pi_\theta}[\phi(s,\cdot)]$$

Where is this coming from?

One-Step MDP

- One-step MDPs
  - Starting in state $s \sim d(s)$
  - Terminating after one time-step with reward $r = R_{s,a}$
- Likelihood ratio to compute the policy gradient

$$
\begin{aligned}
J(\theta) &= \mathbb{E}_{\pi_\theta}[r] \\
&= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) R_{s,a} \\
\nabla_\theta J(\theta) &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s)\, R_{s,a} \\
&= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s)\, r]
\end{aligned}
$$

Multi-step MDP

- Replace instantaneous reward r with long-term value $Q^{\pi}(s, a)$
- For discounted objective
  - Any differentiable policy $\pi_{\theta}(a|s)$
  - Can be shown

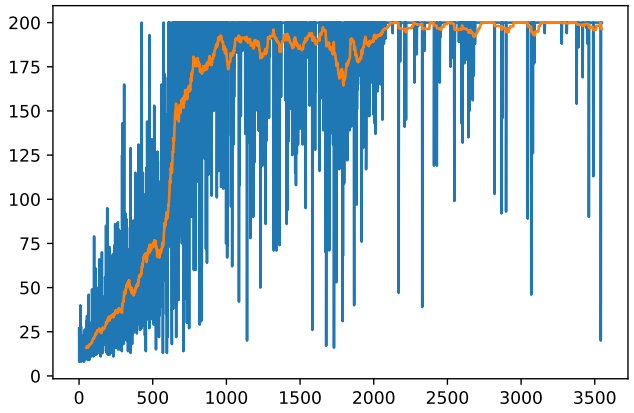
$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(a|s) Q^{\pi_{\theta}}(s, a)]$$

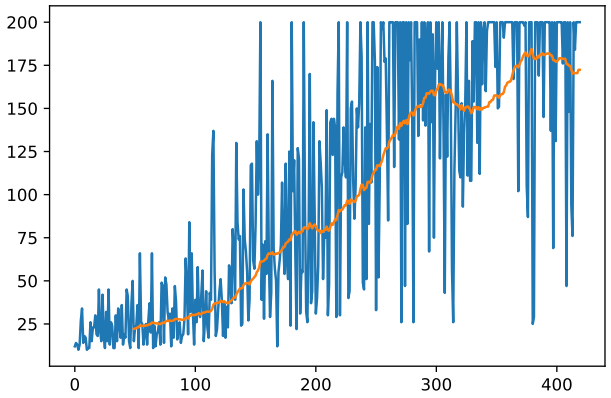
ENHANCEMENTS

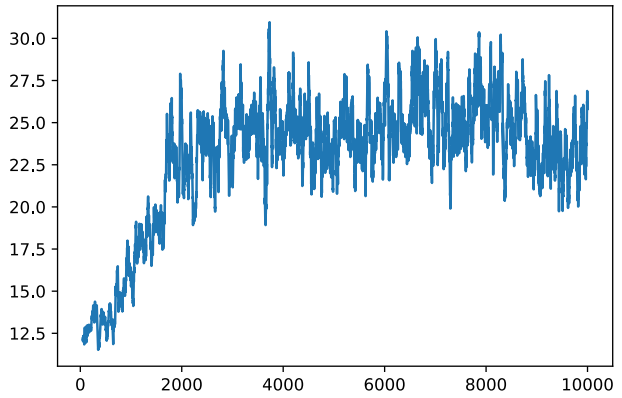
Variance of Policy-Gradient

- Known to have high variance
- Goals of behavior policy
  1. Exploration
  2. Reduce the variance of learned policy

Created by Ghani Ebrahimi

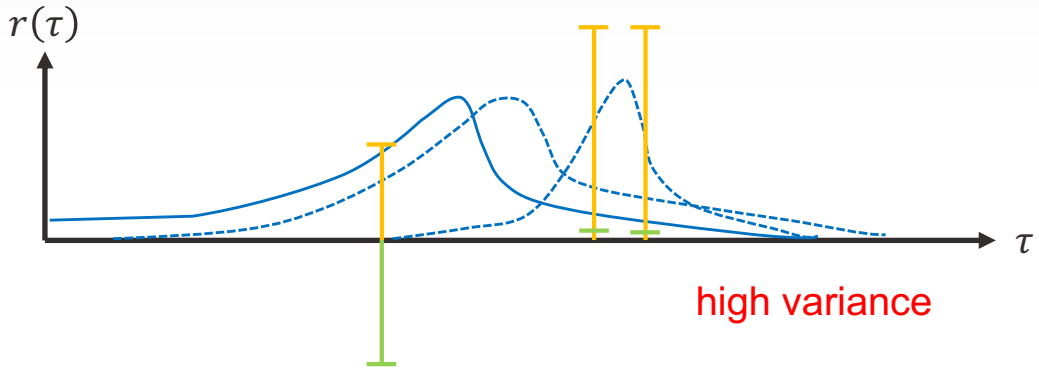






Issues

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(\tau)\, r(\tau)$$



A “good” sample can have $r(\tau) = 0$!!!

Baselines

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(\tau)[r(\tau) - b]$$

a convenient identity

$$\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) = \nabla_\theta \pi_\theta(\tau)$$

- Let *b* (baseline) be constant
  - Independent of θ

$$E[\nabla_\theta \log \pi_\theta(\tau) b] = \int \pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) b \, d\tau = \int \nabla_\theta \pi_\theta(\tau) b \, d\tau = b\nabla_\theta \int \pi_\theta(\tau) \, d\tau = b\nabla_\theta 1 = 0$$

  - Subtracting a baseline is unbiased in expectation
  - Candidate

$$b = \frac{1}{N}\sum_{i=1}^{N} r(\tau)$$

Reducing Variance

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)\left(\sum_{t=1}^{T} r(s_t^i, a_t^i)\right)$$

- Policy at time t' cannot affect reward at time t when $t < t'$

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\underbrace{\left(\sum_{t'=1}^{T} r\left(s_{t'}^i, a_{t'}^i\right)\right)}_{\text{“reward to go” } \hat{Q}_{i,0}}$$

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\left(\sum_{t'=t}^{T} r\left(s_{t'}^i, a_{t'}^i\right)\right)$$ REINFORCE algorithm

Off-policy PG and Importance Sampling

- Recall $\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q^{\pi_\theta}(s,a)]$
  - At time t
    - Q matters
    - Q captures reward only from t onwards
- Scientific justification for truncation

Analyzing Variance

$$\text{Var}[x] = E[x^2] - E[x]^2$$

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$$

$$\text{Var} = E_{\tau\sim\pi_\theta(\tau)}[(\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b))^2] - \underbrace{E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]^2}_{[E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]]^2}$$

$$\frac{d\text{Var}}{db} = \frac{d}{db} E[g(\tau)^2(r(\tau) - b)^2] = \frac{d}{db}(E[g(\tau)^2 r(\tau)^2] - 2E[g(\tau)^2 r(\tau) b] + b^2 E[g(\tau)^2])$$

$$= -2E[g(\tau)^2 r(\tau)] + 2bE[g(\tau)^2] = 0$$

$$b = \frac{E[g(\tau)^2 r(\tau)]}{E[g(\tau)^2]}$$

This is just expected reward, but weighted by gradient magnitudes!

Optimal constant baseline

Reducing Variance Using a Baseline

- Good baseline
  - State value function $B(s) = V^{\pi_\theta}(s)$
- Policy gradient

$$A^{\pi_\theta}(s,a) = Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\, A^{\pi_\theta}(s,a)]$$

- Advantage function $A^{\pi_\theta}(s,a)$
  - For optimal policy advantage function always non-positive

Estimating Advantage Function

- Two function approximators and two parameter vectors

$$V_v(s) \approx V^{\pi_\theta}(s)$$
$$Q_w(s,a) \approx Q^{\pi_\theta}(s,a)$$
$$A(s,a) = Q_w(s,a) - V_v(s)$$

- θ fixed; w, v learnable
  - Updating both value functions by TD learning

Policy Gradient Algorithms

- Many equivalent forms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\, r_t] \quad \text{REINFORCE}$$

$$= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\, Q^w(s,a)] \quad \text{Q Actor-Critic}$$

$$= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\, A^w(s,a)] \quad \text{Advantage Actor-Critic}$$

- Equality for expectation
- When sampling and approximating functions
  - Different algorithms

Exploration-Exploitation

- Exploration-Exploitation tradeoff
- Have visited part of the state space and found a reward of 100
  - Is this the best we can hope for?
  - Should we keep ‘pounding’ the visited states and figure out actions that lead to 100?
  - Perhaps there are other states that we have not yet visited that lead to even higher reward
- Exploitation
  - Should we stick with what we know
  - Find a good policy with respect to this knowledge
    - Risk of missing out on a better reward somewhere else
- Exploration
  - Should we look for states with more reward
    - At risk of wasting time and getting some negative reward

Exploration-Exploitation Epsilon-Greedy

- Exploitation
  - Follow your current best policy
  - Be greedy
- Exploration
  - Deviate to explore
  - Strategy
    - Greedy policy vector a
    - Perturb it by epsilon
      $a + \epsilon$
    - ϵ random vector; has to be probability vector

- Common exploration
  - Probability(*a)*
    - $1 - \epsilon$ if *a* = optimal (argmax)
    - $\epsilon/(|\mathcal{A}| - 1)$ otherwise

Exploration-Exploitation

- Sample from one policy
  - Exploration
  - Behavior policy
- Update and optimization different policy
  - Exploitation
  - Learning policy
- Same policies: on-policy learning
- Two different policies: off-policy learning

Off-policy vs On-policy

- On-policy algorithm
  - Learn the policy being executed by the agent
  - One policy
- Off-policy algorithm
  - Evaluate a policy from samples generated by a different policy
    - Target policy
  - Learn policy independent of policy taken by agent (behavior policy)
  - Two policies

Off-policy Framework

- $\pi_{\theta'}$ is target policy
  - Gradient with respect to this policy
  - Gradient at this point
- Loop
  - Sample episode based on policy π_θ
    - // Behavior policy
  - Perform gradient step at $\pi_{\theta'}$
    - // Adjusts $\pi_{\theta'}$
- Challenge
  - The two policies should be somehow related
    - Idea: gradient taken with respect to a function that involves π_θ

Policy Gradient and On-policy

$$\theta^* = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Same policy

- Neural networks change only slightly with each gradient step
- On-policy learning inefficient

Must have this

REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(a_t|s_t)$
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta\left(a_t^i|s_t^i\right)\right)\left(\sum_t r\left(s_t^i|a_t^i\right)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Off-policy PG and Importance Sampling

$$\theta^* = \arg\max_\theta J(\theta)$$

$$J(\theta') = E_{\tau \sim \pi_{\theta'}(\tau)}[r(\tau)]$$

We have samples from some $\bar{\pi}(\tau)$

$$J(\theta') = E_{\tau \sim \bar{\pi}(\tau)}\left[\frac{\pi_{\theta'}(\tau)}{\bar{\pi}(\tau)} r(\tau)\right]$$

$$\pi_\theta(\tau) = p(s_1)\prod_{t=1}^{T} \pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

$$\frac{\pi_{\theta'}(\tau)}{\bar{\pi}(\tau)} = \frac{p(s_1)\prod_{t=1}^{T} \pi_{\theta'}(a_t|s_t)p(s_{t+1}|s_t, a_t)}{p(s_1)\prod_{t=1}^{T} \bar{\pi}(a_t|s_t)p(s_{t+1}|s_t, a_t)} = \frac{\prod_{t=1}^{T} \pi_{\theta'}(a_t|s_t)}{\prod_{t=1}^{T} \bar{\pi}(a_t|s_t)}$$

Importance sampling

$$E_{x\sim p(x)}[f(x)] = \int p(x)f(x)dx$$

$$= \int \frac{q(x)}{q(x)} p(x)f(x)dx$$

$$= \int q(x)\frac{p(x)}{q(x)} f(x)dx$$

$$= E_{x\sim q(x)}\left[\frac{p(x)}{q(x)} f(x)\right]$$

Off-policy PG and Importance Sampling

$$\theta^* = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[r(\tau)]$$

Convenient identity

$$\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) = \nabla_\theta \pi_\theta(\tau)$$

Estimate the value of some *new* parameters θ':

$$J(\theta') = E_{\tau \sim \pi_\theta(\tau)}\left[\frac{\pi_{\theta'}(\tau)}{\pi_\theta(\tau)} r(\tau)\right]$$

Depends on θ'

$$\nabla_{\theta'} J(\theta') = E_{\tau \sim \pi_\theta(\tau)}\left[\frac{\nabla_{\theta'}\pi_{\theta'}(\tau)}{\pi_\theta(\tau)} r(\tau)\right] = E_{\tau \sim \pi_\theta(\tau)}\left[\frac{\pi_{\theta'}(\tau)}{\pi_\theta(\tau)} \nabla_{\theta'} \log \pi_{\theta'}(\tau) r(\tau)\right]$$

At $\theta = \theta'$

$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$ - Original policy gradient

Off-policy PG and Importance Sampling

$$\theta^* = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[r(\tau)]$$

$$\nabla_{\theta'} J(\theta') = E_{\tau\sim\pi_\theta(\tau)}\left[\frac{\pi_{\theta'}(\tau)}{\pi_\theta(\tau)}\nabla_{\theta'}\log\pi_{\theta'}(\tau)r(\tau)\right]$$

$$\frac{\pi_{\theta'}(\tau)}{\pi_\theta(\tau)} = \frac{\prod_{t=1}^{T}\pi_{\theta'}(a_t|s_t)}{\prod_{t=1}^{T}\pi_\theta(a_t|s_t)}$$

$$= E_{\tau\sim\pi_\theta(\tau)}\left[\left(\prod_{t=1}^{T}\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)}\right)\left(\sum_{t=1}^{T}\nabla_{\theta'}\log\pi_{\theta'}(a_t|s_t)\right)\left(\sum_{t=1}^{T}r(s_t,a_t)\right)\right]$$

$$= E_{\tau\sim\pi_\theta(\tau)}\left[\sum_{t=1}^{T}\nabla_{\theta'}\log\pi_{\theta'}(a_t|s_t)\left(\prod_{t'=1}^{t}\frac{\pi_{\theta'}(a_{t'}|s_{t'})}{\pi_\theta(a_{t'}|s_{t'})}\right)\left(\sum_{t'=t}^{T}r(s_{t'},a_{t'})\right)\right]$$

future has no effect on present

at t past reward does not matter (sunk cost)

Off-policy PG and Importance Sampling

$$\nabla_{\theta'} J(\theta') = E_{\tau \sim \pi_\theta(\tau)} \left[\sum_{t=1}^{T} \nabla_{\theta'} \log \pi_{\theta'}(a_t|s_t) \left(\prod_{t'=1}^{t} \frac{\pi_{\theta'}(a_{t'}|s_{t'})}{\pi_\theta(a_{t'}|s_{t'})} \right) \left(\sum_{t'=t}^{T} r(s_{t'}, a_{t'}) \right) \right]$$

exponential in T…

$$\theta^* = \arg\max_\theta \sum_{t=1}^{T} E_{(s_t,a_t) \sim p_\theta(s_t,a_t)}[r(s_t, a_t)]$$

$$J(\theta) = \sum_{t=1}^{T} E_{(s_t,a_t) \sim p_\theta(s_t,a_t)}[r(s_t, a_t)] = \sum_{t=1}^{T} E_{s_t \sim p_\theta(s_t)} \left[E_{a_t \sim \pi_\theta(a_t|s_t)}[r(s_t, a_t)] \right]$$

$$J(\theta') = \sum_{t=1}^{T} E_{s_t \sim p_\theta(s_t)} \left[\frac{p_{\theta'}(s_t)}{p_\theta(s_t)} E_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} r(s_t, a_t) \right] \right]$$

Ignore this part

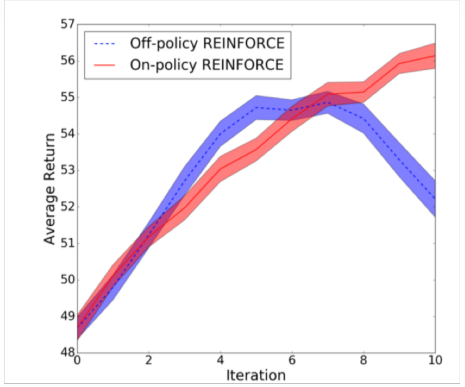
Perform importance sampling for states and actions

Off-policy PG Algorithm

Off-policy REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(a_t|s_t)$ // behavior policy
2. $\nabla_{\theta'} J(\theta') \approx \sum_i \left(\sum_t \nabla_{\theta'} \log \pi_{\theta'}(a_t^i|s_t^i)\right) \left(\sum_t r(s_t^i|a_t^i)/\pi_\theta(a_t^i|s_t^i)\right)$
3. $\theta' \leftarrow \theta' + \alpha \nabla_{\theta'} J(\theta')$ // target policy

- Choice of behavior policy
  - Epsilon greedy with respect to learned policy
  - Policy that minimizes the variance of learned policy
    - Details complicated



Hanna and Stone 2018: Towards a Data Efficient Off-Policy Policy Gradient

Policy Gradient in Practice

- Remember that the gradient has high variance
  - Not the same as supervised learning
    - Much more uncertainty
  - Gradients will be really noisy
- Consider using much larger batches
- Tweaking learning rates is very hard
  - Adaptive step size rules like ADAM help

Advantages of Policy-Based RL

- Advantages:
  - Good convergence properties
  - Effective in high-dimensional or continuous action spaces
  - Learns stochastic policies
- Disadvantages:
  - Typically converge to a local rather than global optimum
  - Evaluating a policy is typically inefficient and high variance