# Policy Gradient Methods

February 13, 2017

# Policy Optimization Problems

$$\underset{\pi}{\text{maximize}}\, \mathbb{E}_{\pi}\left[\mathit{expression}\right]$$

- Fixed-horizon episodic: $\sum_{t=0}^{T-1} r_t$
- Average-cost: $\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_t$
- Infinite-horizon discounted: $\sum_{t=0}^{\infty} \gamma^t r_t$
- Variable-length undiscounted: $\sum_{t=0}^{T_{\text{terminal}}-1} r_t$
- Infinite-horizon undiscounted: $\sum_{t=0}^{\infty} r_t$

# Episodic Setting

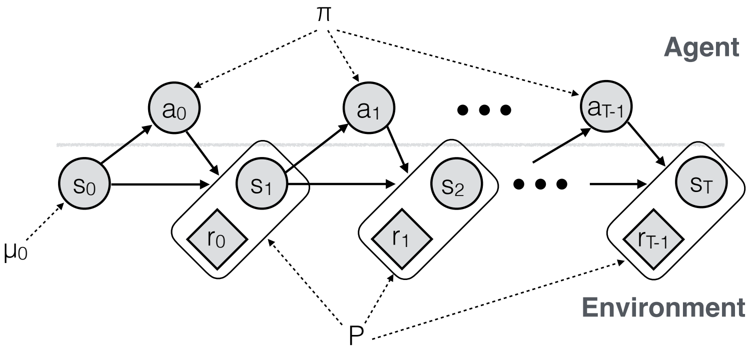
$$
\begin{aligned}
s_0 &\sim \mu(s_0) \\
a_0 &\sim \pi(a_0 \mid s_0) \\
s_1, r_0 &\sim P(s_1, r_0 \mid s_0, a_0) \\
a_1 &\sim \pi(a_1 \mid s_1) \\
s_2, r_1 &\sim P(s_2, r_1 \mid s_1, a_1) \\
&\dots \\
a_{T-1} &\sim \pi(a_{T-1} \mid s_{T-1}) \\
s_T, r_{T-1} &\sim P(s_T \mid s_{T-1}, a_{T-1})
\end{aligned}
$$

Objective:

$$
\text{maximize}\, \eta(\pi), \text{ where}
$$

$$
\eta(\pi) = E[r_0 + r_1 + \cdots + r_{T-1} \mid \pi]
$$

# Episodic Setting




Objective:

$$\text{maximize}\,\eta(\pi), \ \text{ where}$$
$$\eta(\pi) = E[r_0 + r_1 + \cdots + r_{T-1} \mid \pi]$$

# Parameterized Policies

- A family of policies indexed by parameter vector $\theta \in \mathbb{R}^d$
  - Deterministic: $a = \pi(s, \theta)$
  - Stochastic: $\pi(a \mid s, \theta)$
- Analogous to classification or regression with input $s$, output $a$.
  - Discrete action space: network outputs vector of probabilities
  - Continuous action space: network outputs mean and diagonal covariance of Gaussian

# Policy Gradient Methods: Overview

Problem:

$$\text{maximize } E[R \mid \pi_\theta]$$

Intuitions: collect a bunch of trajectories, and ...

1. Make the good trajectories more probable[1]
2. Make the good actions more probable
3. Push the actions towards good actions (DPG[2], SVG[3])

[1] R. J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. *Machine learning* (1992); R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. “Policy gradient methods for reinforcement learning with function approximation”. *NIPS*. MIT Press, 2000.

[2] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, et al. “Deterministic Policy Gradient Algorithms”. *ICML*. 2014.

[3] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, et al. “Learning Continuous Control Policies by Stochastic Value Gradients”. *arXiv preprint arXiv:1510.09142* (2015).

# Score Function Gradient Estimator

- Consider an expectation $E_{x \sim p(x \mid \theta)}[f(x)]$. Want to compute gradient wrt $\theta$

$$
\begin{aligned}
\nabla_\theta E_x[f(x)] &= \nabla_\theta \int \mathrm{d}x \; p(x \mid \theta) f(x) \\
&= \int \mathrm{d}x \; \nabla_\theta p(x \mid \theta) f(x) \\
&= \int \mathrm{d}x \; p(x \mid \theta) \frac{\nabla_\theta p(x \mid \theta)}{p(x \mid \theta)} f(x) \\
&= \int \mathrm{d}x \; p(x \mid \theta) \nabla_\theta \log p(x \mid \theta) f(x) \\
&= E_x[f(x) \nabla_\theta \log p(x \mid \theta)].
\end{aligned}
$$

- Last expression gives us an unbiased gradient estimator. Just sample $x_i \sim p(x \mid \theta)$, and compute $\hat{g}_i = f(x_i) \nabla_\theta \log p(x_i \mid \theta)$.
- Need to be able to compute and differentiate density $p(x \mid \theta)$ wrt $\theta$

# Derivation via Importance Sampling

## Alternative Derivation Using Importance Sampling[4]

$$\mathbb{E}_{x\sim\theta}\left[f(x)\right] = \mathbb{E}_{x\sim\theta_{\text{old}}}\left[\frac{p(x \mid \theta)}{p(x \mid \theta_{\text{old}})} f(x)\right]$$

$$\nabla_\theta \mathbb{E}_{x\sim\theta}\left[f(x)\right] = \mathbb{E}_{x\sim\theta_{\text{old}}}\left[\frac{\nabla_\theta p(x \mid \theta)}{p(x \mid \theta_{\text{old}})} f(x)\right]$$

$$\nabla_\theta \mathbb{E}_{x\sim\theta}\left[f(x)\right]\Big|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{x\sim\theta_{\text{old}}}\left[\frac{\nabla_\theta p(x \mid \theta)\big|_{\theta=\theta_{\text{old}}}}{p(x \mid \theta_{\text{old}})} f(x)\right]$$

$$= \mathbb{E}_{x\sim\theta_{\text{old}}}\left[\nabla_\theta \log p(x \mid \theta)\big|_{\theta=\theta_{\text{old}}} f(x)\right]$$

[4] T. Jie and P. Abbeel. "On a connection between importance sampling and the likelihood ratio policy gradient". *Advances in Neural Information Processing Systems*. 2010, pp. 1000–1008.

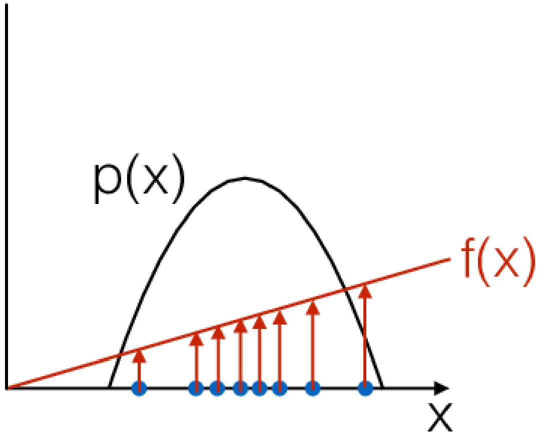
# Score Function Gradient Estimator: Intuition

$$\hat{g}_i = f(x_i) \nabla_\theta \log p(x_i \mid \theta)$$

- Let's say that $f(x)$ measures how good the sample $x$ is.
- Moving in the direction $\hat{g}_i$ pushes up the logprob of the sample, in proportion to how good it is
- *Valid even if $f(x)$ is discontinuous, and unknown, or sample space (containing $x$) is a discrete set*

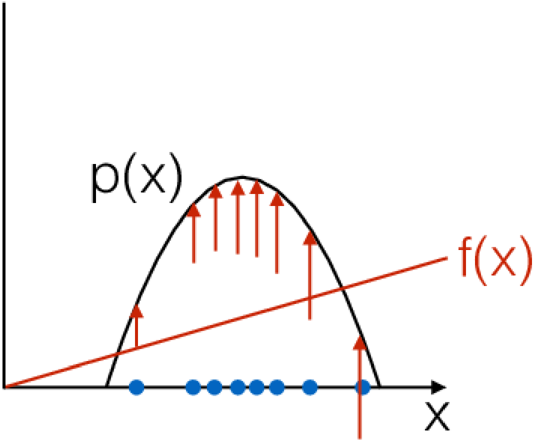
# Score Function Gradient Estimator: Intuition

$$\hat{g}_i = f(x_i)\nabla_\theta \log p(x_i \mid \theta)$$



# Score Function Gradient Estimator: Intuition

$$\hat{g}_i = f(x_i)\nabla_\theta \log p(x_i \mid \theta)$$



# Score Function Gradient Estimator for Policies

- Now random variable $x$ is a whole trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_{T-1}, a_{T-1}, r_{T-1}, s_T)$

$$\nabla_\theta E_\tau[R(\tau)] = E_\tau[\nabla_\theta \log p(\tau \mid \theta) R(\tau)]$$

- Just need to write out $p(\tau \mid \theta)$:

$$p(\tau \mid \theta) = \mu(s_0) \prod_{t=0}^{T-1} [\pi(a_t \mid s_t, \theta) P(s_{t+1}, r_t \mid s_t, a_t)]$$

$$\log p(\tau \mid \theta) = \log \mu(s_0) + \sum_{t=0}^{T-1} [\log \pi(a_t \mid s_t, \theta) + \log P(s_{t+1}, r_t \mid s_t, a_t)]$$

$$\nabla_\theta \log p(\tau \mid \theta) = \nabla_\theta \sum_{t=0}^{T-1} \log \pi(a_t \mid s_t, \theta)$$

$$\nabla_\theta \mathbb{E}_\tau [R] = \mathbb{E}_\tau \left[ R \nabla_\theta \sum_{t=0}^{T-1} \log \pi(a_t \mid s_t, \theta) \right]$$

- Interpretation: using good trajectories (high $R$) as supervised examples in classification / regression

# Policy Gradient: Use Temporal Structure

- Previous slide:

$$\nabla_\theta \mathbb{E}_\tau [R] = \mathbb{E}_\tau \left[ \left( \sum_{t=0}^{T-1} r_t \right) \left( \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \right) \right]$$

- We can repeat the same argument to derive the gradient estimator for a single reward term $r_{t'}$.

$$\nabla_\theta \mathbb{E} [r_{t'}] = \mathbb{E} \left[ r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \right]$$

- Sum this formula over $t$, we obtain

$$\begin{aligned} \nabla_\theta \mathbb{E} [R] &= \mathbb{E} \left[ \sum_{t=0}^{T-1} r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \right] \\ &= \mathbb{E} \left[ \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \sum_{t'=t}^{T-1} r_{t'} \right] \end{aligned}$$

# Policy Gradient: Introduce Baseline

- Further reduce variance by introducing a *baseline* $b(s)$

$$\nabla_\theta \mathbb{E}_\tau [R] = \mathbb{E}_\tau \left[ \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \left( \sum_{t'=t}^{T-1} r_{t'} - b(s_t) \right) \right]$$

- For any choice of $b$, gradient estimator is unbiased.
- Near optimal choice is expected return, $b(s_t) \approx \mathbb{E}[r_t + r_{t+1} + r_{t+2} + \cdots + r_{T-1}]$
- Interpretation: increase logprob of action $a_t$ proportionally to how much returns $\sum_{t'=t}^{T-1} r_{t'}$ are better than expected

# Baseline—Derivation

$$
\begin{aligned}
&\mathbb{E}_{\tau}\left[\nabla_{\theta}\log\pi(a_t \mid s_t,\theta)b(s_t)\right] \\
&= \mathbb{E}_{s_{0:t},a_{0:(t-1)}}\left[\mathbb{E}_{s_{(t+1):T},a_{t:(T-1)}}\left[\nabla_{\theta}\log\pi(a_t \mid s_t,\theta)b(s_t)\right]\right] && \text{(break up expectation)} \\
&= \mathbb{E}_{s_{0:t},a_{0:(t-1)}}\left[b(s_t)\mathbb{E}_{s_{(t+1):T},a_{t:(T-1)}}\left[\nabla_{\theta}\log\pi(a_t \mid s_t,\theta)\right]\right] && \text{(pull baseline term out)} \\
&= \mathbb{E}_{s_{0:t},a_{0:(t-1)}}\left[b(s_t)\mathbb{E}_{a_t}\left[\nabla_{\theta}\log\pi(a_t \mid s_t,\theta)\right]\right] && \text{(remove irrelevant vars.)} \\
&= \mathbb{E}_{s_{0:t},a_{0:(t-1)}}\left[b(s_t)\cdot 0\right]
\end{aligned}
$$

Last equality because $0 = \nabla_{\theta}\mathbb{E}_{a_t\sim\pi(\cdot \mid s_t)}[1] = \mathbb{E}_{a_t\sim\pi(\cdot \mid s_t)}\left[\nabla_{\theta}\log\pi_{\theta}(a_t \mid s_t)\right]$

# Discounts for Variance Reduction

- Introduce discount factor $\gamma$, which ignores delayed effects between actions and rewards

$$\nabla_\theta \mathbb{E}_\tau [R] \approx \mathbb{E}_\tau \left[ \sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \left( \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'} - b(s_t) \right) \right]$$

- Now, we want $b(s_t) \approx \mathbb{E}\left[ r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-1-t} r_{T-1} \right]$

# "Vanilla" Policy Gradient Algorithm

Initialize policy parameter $\theta$, baseline $b$
**for** iteration$=1, 2, \dots$ **do**
  Collect a set of trajectories by executing the current policy
  At each timestep in each trajectory, compute
   the *return* $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}$, and
   the *advantage estimate* $\hat{A}_t = R_t - b(s_t)$.
  Re-fit the baseline, by minimizing $\|b(s_t) - R_t\|^2$,
   summed over all trajectories and timesteps.
  Update the policy, using a policy gradient estimate $\hat{g}$,
   which is a sum of terms $\nabla_\theta \log \pi(a_t \mid s_t, \theta) \hat{A}_t$.
   (Plug $\hat{g}$ into SGD or ADAM)
**end for**

# Practical Implementation with Autodiff

- Usual formula $\sum_t \nabla_\theta \log \pi(a_t \mid s_t; \theta)\hat{A}_t$ is inefficient—want to batch data
- Define "surrogate" function using data from currecnt batch

$$L(\theta) = \sum_t \log \pi(a_t \mid s_t; \theta)\hat{A}_t$$

- Then policy gradient estimator $\hat{g} = \nabla_\theta L(\theta)$
- Can also include value function fit error

$$L(\theta) = \sum_t \left( \log \pi(a_t \mid s_t; \theta)\hat{A}_t - \|V(s_t) - \hat{R}_t\|^2 \right)$$

# Value Functions

$$Q^{\pi,\gamma}(s,a) = \mathbb{E}_{\pi}\left[r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \mid s_0 = s, a_0 = a\right]$$

Called $Q$-function or state-action-value function

$$\begin{aligned} V^{\pi,\gamma}(s) &= \mathbb{E}_{\pi}\left[r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \mid s_0 = s\right] \\ &= \mathbb{E}_{a\sim\pi}\left[Q^{\pi,\gamma}(s,a)\right] \end{aligned}$$

Called state-value function

$$A^{\pi,\gamma}(s,a) = Q^{\pi,\gamma}(s,a) - V^{\pi,\gamma}(s)$$

Called advantage function

# Policy Gradient Formulas with Value Functions

- Recall:

$$\nabla_\theta \mathbb{E}_\tau[R] = \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta)\left(\sum_{t'=t}^{T-1} r_{t'} - b(s_t)\right)\right]$$

$$\approx \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta)\left(\sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'} - b(s_t)\right)\right]$$

- Using value functions

$$\nabla_\theta \mathbb{E}_\tau[R] = \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) Q^\pi(s_t, a_t)\right]$$

$$= \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) A^\pi(s_t, a_t)\right]$$

$$\approx \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) A^{\pi,\gamma}(s_t, a_t)\right]$$

- Can plug in "advantage estimator" $\hat{A}$ for $A^{\pi,\gamma}$
- Advantage estimators have the form $\mathit{Return} - V(s)$

# Value Functions in the Future

- Baseline accounts for and removes the effect of *past* actions
- Can also use the value function to estimate future rewards

$$
\begin{aligned}
\hat{R}_t^{(1)} &= r_t + \gamma V(s_{t+1}) && \text{cut off at one timestep} \\
\hat{R}_t^{(2)} &= r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) && \text{cut off at two timesteps} \\
&\dots \\
\hat{R}_t^{(\infty)} &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots && \infty \text{ timesteps (no } V\text{)}
\end{aligned}
$$

# Value Functions in the Future

- Subtracting out baselines, we get advantage estimators

$$
\begin{aligned}
\hat{A}_t^{(1)} &= r_t + \gamma V(s_{t+1}) - V(s_t) \\
\hat{A}_t^{(2)} &= r_t + r_{t+1} + \gamma^2 V(s_{t+2}) - V(s_t) \\
\cdots \\
\hat{A}_t^{(\infty)} &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots - V(s_t)
\end{aligned}
$$

- $\hat{A}_t^{(1)}$ has low variance but high bias, $\hat{A}_t^{(\infty)}$ has high variance but low bias.
- Using intermediate $k$ (say, 20) gives an intermediate amount of bias and variance

# Discounts: Connection to MPC

- MPC:

$$\underset{a}{\text{maximize}}\, Q^{*,T}(s,a) \approx \underset{a}{\text{maximize}}\, Q^{*,\gamma}(s,a)$$

- Discounted policy gradient

$$\mathbb{E}_{a\sim\pi}\left[Q^{\pi,\gamma}(s,a)\nabla_\theta \log \pi(a \mid s;\theta)\right] = 0 \quad \text{when} \quad a \in \arg\max Q^{\pi,\gamma}(s,a)$$

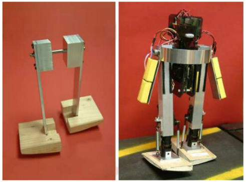
# Application: Robot Locomotion

## Learning to Walk in 20 Minutes

Russ Tedrake
Brain & Cognitive Sciences
Center for Bits and Atoms
Massachusetts Inst. of Technology
Cambridge, MA 02139
russt@csail.mit.edu

Teresa Weirui Zhang
Mechanical Engineering
Department
University of California, Berkeley
Berkeley, CA 94270
resa@berkeley.edu

H. Sebastian Seung
Howard Hughes Medical Institute
Brain & Cognitive Sciences
Massachusetts Inst. of Technology
Cambridge, MA 02139
seung@mit.edu



# Finite-Horizon Methods: Advantage Actor-Critic

- A2C / A3C uses this fixed-horizon advantage estimator. (NOTE: “async” is only for speed, doesn’t improve performance)
- Pseudocode

**for** iteration=$1, 2, \ldots$ **do**

Agent acts for $T$ timesteps (e.g., $T = 20$),

For each timestep $t$, compute

$$\hat{R}_t = r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_t)$$
$$\hat{A}_t = \hat{R}_t - V(s_t)$$

$\hat{R}_t$ is target value function, in regression problem

$\hat{A}_t$ is estimated advantage function

Compute loss gradient $g = \nabla_\theta \sum_{t=1}^{T} \left[ -\log \pi_\theta(a_t \mid s_t) \hat{A}_t + c(V(s) - \hat{R}_t)^2 \right]$

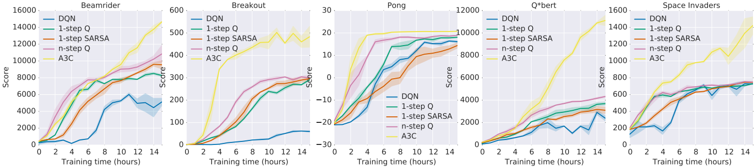
$g$ is plugged into a stochastic gradient descent variant, e.g., Adam.

**end for**

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, et al. “Asynchronous methods for deep reinforcement learning”. (2016)

# A3C Video

# A3C Results



Beamrider
Breakout
Pong
Q*bert
Space Invaders
DQN
1-step Q
1-step SARSA
n-step Q
A3C
Score
Training time (hours)

# Further Reading

- A nice intuitive explanation of policy gradients: http://karpathy.github.io/2016/05/31/rl/
- R. J. Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". *Machine learning* (1992); R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. "Policy gradient methods for reinforcement learning with function approximation". *NIPS.* MIT Press, 2000
- My thesis has a decent self-contained introduction to policy gradient methods: http://joschu.net/docs/thesis.pdf
- A3C paper: V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, et al. "Asynchronous methods for deep reinforcement learning". (2016)