Reducing Variance and the Connection to Value-Based Methods

Martha White
Associate Professor
University of Alberta

Recap about variance

- Variance is particularly problematic in PG
- Even if the true objective and gradient is well-behaved (good curvature), the returns can be **high variance**
  - e.g., many samples of the returns can be near zero

Outline

- Explain the sources of variance due to sampling
- Two key strategies to mitigate variance in the gradient estimator
  - control variates using a baseline
  - estimates of the expected return (using value-based methods)
- Data re-use to improve accuracy of value estimates (off-policy learning)
- Data re-use for mini-batch policy updating (and more off-policy issues)

Recall the Policy Objective and Gradient

Policy Objective: $\sum_{s} \mu(s) V^{\pi}(s)$

Policy Gradient $\sum_{s} d_{\pi}(s) \sum_{a} Q^{\pi}(s, a) \nabla \pi(a \mid s)$

To allow for on-policy sampling, we typically write:

$$\sum_{a} Q^{\pi}(s, a) \nabla \pi(a \mid s) = \sum_{a} \pi(a \mid s) Q^{\pi}(s, a) \frac{1}{\pi(a \mid s)} \nabla \pi(a \mid s)$$

$$= \sum_{a} \pi(a \mid s) Q^{\pi}(s, a) \nabla \ln \pi(a \mid s)$$

Sampling the Policy Gradient

- Sample $s \sim d_\pi$
  - Sample start state $s_0 \sim \mu$
  - Run the policy π and accept state s with probability $1 - \gamma$
- Sample $a \sim \pi(a \mid s)$
- Sample return $R(\tau)$ by sampling a trajectory τ from (s, a), following π
- Stochastic gradient: $g(\theta, S, A, \tau = R, S', \ldots) \doteq R(\tau) \nabla \ln \pi(A \mid S)$

Three sources of variance

- Variance from state sampling
- Variance from action sampling
- Variance from sampling returns from a state

Reducing variance due to state sampling

$$g(\theta, s) \doteq \mathbb{E}[g(\theta, S, A, \tau) \mid S = s] = \sum_a Q^\pi(s, a) \nabla \pi(a \mid s)$$

$$g(\theta) \doteq \nabla J(\theta) = \mathbb{E}[g(\theta, S)] = \sum_s d_\pi(s) \sum_a Q^\pi(s, a) \nabla \pi(a \mid s)$$

Simple idea: use a mini-batch, where average $g(\theta, s_i)$ for multiple $s_i \sim d_\pi$

In practice: we use mini-batch samples from the replay buffer

- Sampling states from the buffer likely does not give $s_i \sim d_\pi$

Reducing variance due to action sampling: The All Actions Gradient

- Assume we sample $A \sim \pi(\cdot \,|\, S)$

$$g(\theta, s, a) \doteq \mathbb{E}[g(\theta, S, A, \tau) \,|\, S = s, A = a] = Q^{\pi}(s, a) \nabla \ln \pi(a \,|\, s)$$

- Given $Q^{\pi}(s, a)$, the **simplest way** to reduce variance is to completely remove this stochasticity by summing over all actions

$$\sum_a \pi(a \,|\, s) Q^{\pi}(s, a) \nabla \ln \pi(a \,|\, s) = \sum_a Q^{\pi}(s, a) \nabla \pi(a \,|\, s)$$

The All Actions Gradient may not be feasible in practice

- We **may not have** the function $Q^\pi(s, a)$
    - e.g., might use a sample of the return as an unbiased estimate of $Q^\pi(s, a)$
- If so, we cannot consider all possible actions we could have taken
    - e.g., the sample return from s is for one specific a that was taken

Reducing variance due to action sampling: Using a baseline

$$g(\theta, s, a) \doteq \mathbb{E}[g(\theta, S, A, \tau) \,|\, S = s, A = a] = Q^\pi(s, a)\,\nabla \ln \pi(a \,|\, s)$$

$$g(\theta, s) \doteq \mathbb{E}[g(\theta, S, A) \,|\, S = s] = \sum_a Q^\pi(s, a)\,\nabla \pi(a \,|\, s)$$

Given any state-dependent baseline $b(s)$, define control variate

$$z(s, a) \doteq b(s)\,\nabla \ln \pi(a \,|\, s)$$

To get gradient estimator

$$g(\theta, S, A) - z(S, A) = \left(Q^\pi(s, a) - b(s)\right)\nabla \ln \pi(a \,|\, s)$$

Reducing variance due to action sampling: Using a baseline

$$g(\theta, s, a) \doteq \mathbb{E}[g(\theta, S, A, \tau) \,|\, S = s, A = a] = Q^{\pi}(s, a)\, \nabla \ln \pi(a \,|\, s)$$

$$g(\theta, s) \doteq \mathbb{E}[g(\theta, S, A) \,|\, S = s] = \textstyle\sum_a Q^{\pi}(s, a)\, \nabla \pi(a \,|\, s)$$

$$z(s, a) \doteq b(s)\, \nabla \ln \pi(a \,|\, s)$$
$$g(\theta, S, A) - z(S, A) = \left(Q^{\pi}(s, a) - b(s)\right) \nabla \ln \pi(a \,|\, s)$$

We can show that $\mathbb{E}[z(s, A)] = \displaystyle\sum_a \pi(a \,|\, s) z(s, a) = 0$

Therefore the gradient estimator $g(\theta, S, A) - z(S, A)$ is unbiased
$\mathbb{E}[g(\theta, s, A) - z(s, A)] = g(\theta, s)$

What baseline gives the minimum variance?

We can solve for $b(s)$ such that $\sum_j \text{Var}[g_j(\theta, s, A) - z_j(s, A)]$ is **minimal**

$$b^*(s) = \frac{\sum_j \sum_a Q^\pi(s,a) p_{ja}^2}{\sum_j \sum_a p_{ja}^2} \quad \text{where } p_{ja} = \frac{\partial}{\partial \theta_j} \pi(a \mid s)$$

What baseline gives the minimum variance?

We can solve for $b(s)$ such that $\sum_j \text{Var}[g_j(\theta, s, A) - z_j(s, A)]$ is **minimal**

$$b^*(s) = \frac{\sum_j \sum_a Q^\pi(s, a) p_{ja}^2}{\sum_j \sum_a p_{ja}^2} \quad \text{where } p_{ja} = \frac{\partial}{\partial \theta_j} \pi(a \mid s)$$

Typical choice **in practice** is $b(s) \approx V^\pi(s)$

- Corresponds to using the advantage function $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$

Reducing variance due to return sampling

- Assuming $S \sim d_\pi, A \sim \pi(\cdot \mid S)$ and future actions taken according to π

$$g(\theta, S, A, \tau = R, S', \ldots) \doteq R(\tau) \nabla \ln \pi(A \mid S)$$

- Returns $R(\tau)$ can be very high variance
- **Simple idea**: estimate $Q^\pi(s, a) = \mathbb{E}[R(\tau) \mid S = s, A = a]$
- Define estimator $\hat{Q}(s_t, a_t)$, which could be stochastic
  - e.g., $\hat{Q}(s_t, a_t) = R(\tau)$
  - what else can we choose?

Option 1: Directly approximate $Q^{\pi}(s, a)$

- Approximate action-values using any policy evaluation method
  - e.g., Expected Sarsa for prediction
- Using a sampled transition (S, A, R, S'), update action-values q_w using
  - Sarsa: Sample $a' \sim \pi(\cdot \mid S')$
    $w \leftarrow w + \alpha(R + \gamma q_w(S', a') - q_w(S, A)) \nabla q_w(S, A)$
  - Expected Sarsa:
    $w \leftarrow w + \alpha\left(R + \gamma \sum_{a'} \pi(a' \mid S') q_w(S', a') - q_w(S, A)\right) \nabla q_w(S, A)$

Resulting update using q_w

- Assume $S \sim d_\pi, A \sim \pi(\cdot \mid S)$ to get transition (S, A, R, S')
- The All Actions update is

$$\theta \leftarrow \theta + \alpha \sum_a q_w(s, a) \nabla \pi(a \mid s)$$

Option 2: One-step Returns

- $\hat{Q}(s_t, a_t) = R_{t+1} + \gamma v_w(S_{t+1})$ is a stochastic one-step bootstrapped return
    - Estimator $\hat{Q}(s_t, a_t)$ stochastic due to stochasticity in reward, next state
- We approximate the value function $v_w(s) \approx V^\pi(s)$
- These values can be learned using temporal difference learning

$$w \leftarrow w + \alpha \delta_t \nabla v_w(S_t) \qquad \text{for } \delta_t = R_{t+1} + \gamma v_w(S_{t+1}) - v_w(S_t)$$

The Original Actor-Critic uses Option 2

- $\hat{Q}(s_t, a_t) = R_{t+1} + \gamma v_w(S_{t+1})$
- Assume we use baseline $b(s) = v_w(s)$
- This elegantly provides a gradient estimator that uses the same TD error as the critic, because $\delta_t = \hat{Q}(s_t, a_t) - v_w(s_t)$
- The gradient estimator for the policy is $\delta_t \nabla \ln \pi(a_t \mid s_t)$

$$\theta \leftarrow \theta + \alpha \delta_t \nabla \ln \pi(a_t \mid s_t)$$

Option 3: n-step Returns

$$\hat{Q}(s_t, a_t) = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{n-1} R_{t+n+1} + \gamma^n v_w(S_{t+n+1})$$

- The original Actor-Critic uses n = 1
- Assume $s_t \sim d_\pi$, $a_t \sim \pi(\cdot \mid s_t)$ and the following policy π for n steps, storing rewards R_{t+1+i} along the way
- Define n-step TD error: $\delta_t^{(n)} \doteq \hat{Q}(s_t, a_t) - v_w(s_t)$

$$\theta \leftarrow \theta + \alpha \delta_t^{(n)} \nabla \ln \pi(a_t \mid s_t)$$

$$w \leftarrow w + \alpha \delta_t^{(n)} \nabla v_w(S_t)$$

Option 4: Averaging n-step returns

- If we weight all n-step returns, proportionally to λ^n, we get λ-returns
- Let $R^{(n)}(\tau) \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{n-1} R_{t+n+1} + \gamma^n v_w(S_{t+n+1})$

$$\lambda\text{-return} \doteq (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} R^{(n)}(\tau)$$

How do we pick between these options?

- And why are there so many? Can't we just use the direct approximation q_w?
- Our goal was to **reduce variance** of the gradient estimator $(R(\tau) - b(s))\nabla \ln \pi(A \mid S)$
- But using gradient estimator $(\hat{Q}(S, A) - b(s))\nabla \ln \pi(A \mid S)$ **introduces bias**
- The different options result in different bias and variance properties
- Bias = $\mathbb{E}[\hat{Q}(s, a)\nabla \ln \pi(a \mid s)] - Q^\pi(s, a)\nabla \ln \pi(a \mid s)$

$$= \left(\mathbb{E}[\hat{Q}(s, a)] - Q^\pi(s, a)\right)\nabla \ln \pi(a \mid s)$$

Bias-variance with n-step returns

- **Direct** approximation $\hat{Q}(s_t, a_t) = q_w(s_t, a_t)$ is effectively a 0-step return
  - **Zero Variance**, but potentially **High Bias** if $q_w(s_t, a_t)$ is inaccurate
  - If $q_w(s_t, a_t)$ is accurate, this estimator has Zero Variance and Zero Bias
- **1-step** return $\hat{Q}(s_t, a_t) = R_{t+1} + \gamma v_w(S_{t+1})$
  - $\mathbb{E}[\hat{Q}(s, a)] = \mathbb{E}[R_{t+1} \,|\, S_t = s, A_t = a] \;+\; \gamma\, \mathbb{E}[v_w(S_{t+1}) \,|\, S_t = s, A_t = a]$
  - **Low Variance**, but still potentially **High Bias** due to inaccuracy in $v_w(S_{t+1})$

Bias-variance with n-step returns

- **Direct** approximation $\hat{Q}(s_t, a_t) = q_w(s_t, a_t)$ is effectively a 0-step return
  - **Zero Variance**, but potentially **High Bias** if $q_w(s_t, a_t)$ is inaccurate
- **1-step** return $\hat{Q}(s_t, a_t) = R_{t+1} + \gamma v_w(S_{t+1})$
  - **Low Variance**, but still potentially **High Bias** due to inaccuracy in $v_w(S_{t+1})$
- **n-step** return has **More Variance**, but **Less Bias**, depending on n
- If n is longer than the episode length, then $\hat{Q}(s_t, a_t)$ is a sample of the return
  - Zero Bias, but likely High Variance

A Preliminary Table on the Bias-Variance Properties of the Gradient Estimator

	Low Variance	High Variance
Low Bias	**n-step** estimator, for an **interim n** (?) **n-step** estimator with **accurate value** estimates, even for small n Sampled returns, if variance of returns is low	**Sampled returns**, which are typically high variance if policy and/or environment are stochastic **n-step** estimator, with **large n**
High Bias	**n-step** estimator, for **small n**, with **inaccurate value** estimates	:(

Small nuance about updating v_w

- Estimator $\hat{Q}(s_t, a_t) = R_{t+1} + \gamma v_w(S_{t+1})$ uses a given value estimate v_w
- The bias-variance discussion assumes v_w does not update with the data R_{t+1}, S_{t+1} until after updating the policy
- Example bad outcome:
  - Imagine we perfectly fit v_w using a trajectory sampled under π
  - $v_w(s_t) = R(\tau)$ the return from state s_t in the trajectory, with unique states
  - Our policy updates for this trajectory are zero!

$$(R(\tau) - v_w(s_t)) \nabla \ln \pi(a \mid s) = 0$$

Outline

- Explain the sources of variance due to sampling
- Two key strategies to mitigate variance in the gradient estimator
  - control variates using a baseline
  - estimates of the expected return (using value-based methods)
- **Data re-use** to improve accuracy of value estimates (off-policy learning)
- Data re-use for mini-batch policy updating (and more off-policy issues)

Why are value estimates inaccurate?

- The bias in the gradient estimator comes from inaccurate value estimates
- **Inaccuracy** arises from
  - function approximation
  - insufficient samples (even if value estimator itself is an unbiased estimator)
- Strategies to **reduce bias**
  - improve function approximator
  - re-use old data

Reducing bias with more powerful function approximation

- Bias can be reduced by using **more powerful FA** for q_w or v_w, so that the true expected returns can be represented
    - e.g., use a larger neural network
- But with more powerful FA comes the **need for more data**
    - we need to be as sample efficient as possible with the data that we do see

Sample Efficiency and Data Re-use

- Store all observed data, and extract as much as possible from it
- Can **re-use data** from past experience, to better estimate q_w or v_w
- But! The policy is changing, so updates are **off-policy**
  - Is this a problem?

Updating q_w or v_w off-policy is straightforward

- Assume you have a buffer of past interaction (replay buffer)

- Update q_w with mini-batch sample, using Expected Sarsa for transition (s, a, r, s')

$$\left(r + \sum_{a'} \pi(a' \mid s') q_w(s', a') - q_w(s, a)\right) \nabla q_w(s, a)$$

- Update v_w with mini-batch sample, using update for transition $(s, a, r, s', b(a \mid s))$

$$\rho\left(r + v_w(s') - v_w(s)\right) \nabla v_w(s) \qquad \rho \doteq \frac{\pi(a \mid s)}{b(a \mid s)}$$

where $b(a \mid s)$ was the probability of taking action a in state s

Nuanced issues in using replay for values

- TD methods are known to have **divergence** issues
- Reason: divergence can occur if we do not correct the state distribution
  - implicit weighting is $d(s)$ given by state distribution in replay buffer
  - d is not equal to the on-policy state visitation distribution
- One fix: use **prior corrections** that reweight the update

Prior corrections

- Imagine have s_t from a sampled trajectory under behavior b
- We want to reweight the update so it is as if we had run π
- Update $w \leftarrow w + \alpha \rho_t \delta_t \nabla v_w(S_t)$ only corrects distribution over A_t
- Instead we need to adjust all action probabilities back to s_0

$$w \leftarrow w + \alpha \rho_0 \rho_1 \ldots \rho_t \delta_t \nabla v_w(S_t)$$

More practical alternatives

- There is an growing body of work on sound methods for policy evaluation
- Gradient TD methods provide convergence, without needing prior corrections
- Emphatic TD methods provide soundness with lower variance prior corrections
- With powerful function approximators (really good features), some of these issues also disappear

Overall Conclusion about Data Re-use

- Off-policy methods allow data re-use for more accurate value estimates
- More accurate value estimates can reduce bias in the gradient estimator
- On-policy gradient estimates, that use on-policy returns, cannot exploit this data
    - we get better gradient estimates by using the structure of our problem (the fact that we are in an MDP and have a Bellman equation for values)
    - a smart way to estimate the gradient

Summary Table for Gradient Estimator

	Low Variance	High Variance
Low Bias	**n-step** estimator with **strong value FA** and **many samples**, even for small n Sampled returns, if variance of returns is low	**Sampled returns**, which are typically high variance if policy and/or environment are stochastic **n-step** estimator, with **large n**
High Bias	**n-step** estimator with **weak value FA** and relatively **small n** **n-step** estimator with **strong value FA** but **few samples**	Hopefully not your algorithm (e.g., Random numbers sampled from a high variance distribution)

Data re-use opens up other variance reduction strategies for the gradient estimator

- Can **re-use old data** to sample the policy update
- Two benefits:
    - reduce variance in update using mini-batches
    - increase number of updates (replay)
- But! The policy is changing, so updates are **off-policy**
    - Is this a problem?

A Common Approach to Use Replay for Policy Updates: Ignore State Weighting

- Sample state s from the buffer and sample $a \sim \pi(\cdot \mid s)$
- Update using $(\hat{Q}(s, a) - b(s)) \nabla \ln \pi(a \mid s)$
- Implicit state weighting d for frequency of s in the buffer
  - $s \sim d$ and $d \neq d_\pi$
- Even if we get $g(\theta, s) \doteq \mathbb{E}[g(\theta, S, A, \tau) \mid S = s] = \sum_a Q^\pi(s, a) \nabla \pi(a \mid s)$

$$\sum_s d(s) g(\theta, s) \neq \sum_s d_\pi(s) g(\theta, s) = \nabla J(\theta)$$

Summary about bias in gradient estimator

- Most Actor-Critic algorithms have bias in the update due to
  - State sampling bias
  - Biased estimates of expected returns
- But how much does it matter?

Summary about bias in gradient estimator

- Most Actor-Critic algorithms have bias in the update due to
  - State sampling bias
  - Biased estimates of expected returns
- Bias in the gradient estimator is transient
  - A biased update might even get us to a solution faster
  - e.g., can minimize a lower bound that uses gradient samples according to the behavior (see “An operator view of policy gradient methods”)
- But, bias in the gradient has also been shown to produce poor solutions
  - see “An Off-policy Policy Gradient Theorem Using Emphatic Weightings”

Concluding remarks about bias and variance

- The impact of this bias on our solutions remains poorly understood
  - optimization literature has some theory about biased gradients, but assumes that bias decays with time
- Variance in our gradient estimates can even be good!
  - e.g., potentially avoid getting stuck in flat regions or suboptimal solutions
- A first step is to at least understand the bias-variance properties of our gradient estimators
  - and then more to understand how this impacts policy optimization