Policy Gradient Methods: Pathwise Derivative Methods and Wrap-up

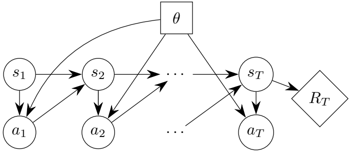
March 15, 2017

Pathwise Derivative Policy Gradient Methods

Policy Gradient Estimators: Review

Deriving the Policy Gradient, Reparameterized

- Episodic MDP:

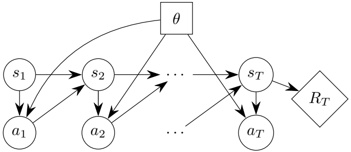


  Want to compute $\nabla_\theta \mathbb{E}[R_T]$. We'll use $\nabla_\theta \log \pi(a_t \mid s_t; \theta)$
- Reparameterize: $a_t = \pi(s_t, z_t; \theta)$. z_t is noise from fixed distribution.

- Only works if $P(s_2 \mid s_1, a_1)$ is known $\ddot{\frown}$

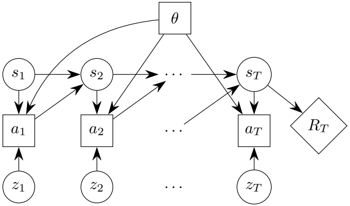
Deriving the Policy Gradient, Reparameterized

- Episodic MDP:



  Want to compute $\nabla_\theta \mathbb{E}[R_T]$. We'll use $\nabla_\theta \log \pi(a_t \mid s_t; \theta)$

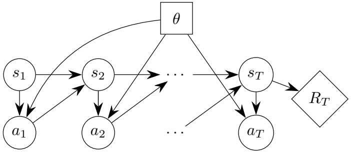
- Reparameterize: $a_t = \pi(s_t, z_t; \theta)$. z_t is noise from fixed distribution.



- Only works if $P(s_2 \mid s_1, a_1)$ is known $\ddot\frown$

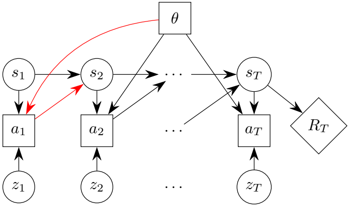
Deriving the Policy Gradient, Reparameterized

- Episodic MDP:



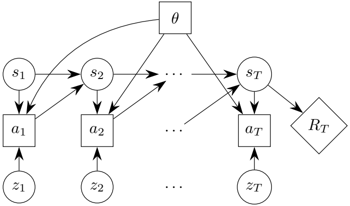
Want to compute $\nabla_\theta \mathbb{E}[R_T]$. We'll use $\nabla_\theta \log \pi(a_t \mid s_t; \theta)$

- Reparameterize: $a_t = \pi(s_t, z_t; \theta)$. z_t is noise from fixed distribution.



- Only works if $P(s_2 \mid s_1, a_1)$ is known $\ddot{\frown}$

Using a Q-function



$$\frac{\mathrm{d}}{\mathrm{d}\theta}\mathbb{E}\left[R_T\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}R_T}{\mathrm{d}a_t}\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}}{\mathrm{d}a_t}\mathbb{E}\left[R_T \mid a_t\right]\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right]$$

$$= \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}Q(s_t, a_t)}{\mathrm{d}a_t}\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}}{\mathrm{d}\theta}Q(s_t, \pi(s_t, z_t; \theta))\right]$$

SVG(0) Algorithm

- Learn Q_ϕ to approximate $Q^{\pi,\gamma}$, and use it to compute gradient estimates.

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)

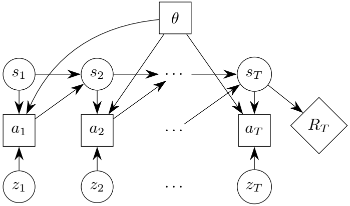
SVG(0) Algorithm

- Learn Q_ϕ to approximate $Q^{\pi,\gamma}$, and use it to compute gradient estimates.
- Pseudocode:

for iteration=$1, 2, \ldots$ **do**
  Execute policy π_θ to collect T timesteps of data
  Update π_θ using $g \propto \nabla_\theta \sum_{t=1}^{T} Q(s_t, \pi(s_t, z_t; \theta))$
  Update Q_ϕ using $g \propto \nabla_\phi \sum_{t=1}^{T} (Q_\phi(s_t, a_t) - \hat{Q}_t)^2$, e.g. with TD($\lambda$)
end for

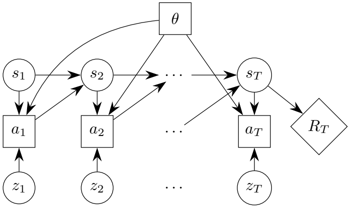
N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, et al. "Learning Continuous Control Policies by Stochastic Value Gradients". *arXiv preprint arXiv:1510.09142* (2015)

SVG(1) Algorithm



- Instead of learning Q, we learn
  - State-value function $V \approx V^{\pi,\gamma}$
  - Dynamics model f, approximating $s_{t+1} = f(s_t, a_t) + \zeta_t$
- Given transition (s_t, a_t, s_{t+1}), infer $\zeta_t = s_{t+1} - f(s_t, a_t)$
- $Q(s_t, a_t) = \mathbb{E}[r_t + \gamma V(s_{t+1})] = \mathbb{E}[r_t + \gamma V(f(s_t, a_t) + \zeta_t)]$, and $a_t = \pi(s_t, \theta, \zeta_t)$

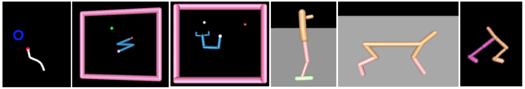
SVG(∞) Algorithm



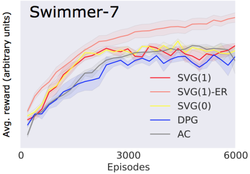
- Just learn dynamics model f
- Given whole trajectory, infer all noise variables
- Freeze all policy and dynamics noise, differentiate through entire deterministic computation graph

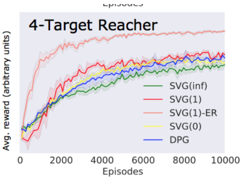
SVG Results

- Applied to 2D robotics tasks



- Overall: different gradient estimators behave similarly





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)

Deterministic Policy Gradient

- For Gaussian actions, variance of score function policy gradient estimator goes to infinity as variance goes to zero
  - Intuition: finite difference gradient estimators
- But SVG(0) gradient is fine when $\sigma \to 0$

$$\nabla_\theta \sum_t Q(s_t, \pi(s_t, \theta, \zeta_t))$$

- Problem: there's no exploration.
- Solution: add noise to the policy, but estimate Q with TD(0), so it's valid off-policy
- Policy gradient is a little biased (even with $Q = Q^\pi$), but only because state distribution is off—it gets the right gradient at every state

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, et al. "Deterministic Policy Gradient Algorithms". *ICML*. 2014.

Deep Deterministic Policy Gradient

- Incorporate replay buffer and target network ideas from DQN for increased stability
- Use lagged (Polyak-averaging) version of Q_ϕ and π_θ for fitting Q_ϕ (towards $Q^{\pi,\gamma}$) with TD(0)

$$\hat{Q}_t = r_t + \gamma Q_{\phi'}(s_{t+1}, \pi(s_{t+1}; \theta'))$$

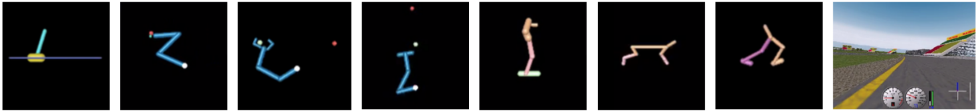
- Pseudocode:

for iteration$=1, 2, \ldots$ **do**
  Act for several timesteps, add data to replay buffer
  Sample minibatch
  Update π_θ using $g \propto \nabla_\theta \sum_{t=1}^{T} Q(s_t, \pi(s_t, z_t; \theta))$
  Update Q_ϕ using $g \propto \nabla_\phi \sum_{t=1}^{T} (Q_\phi(s_t, a_t) - \hat{Q}_t)^2$,
end for

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, et al. "Continuous control with deep reinforcement learning". *arXiv preprint arXiv:1509.02971* (2015)

DDPG Results

Applied to 2D and 3D robotics tasks and driving with pixel input



T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, et al. "Continuous control with deep reinforcement learning". *arXiv preprint arXiv:1509.02971* (2015)

Policy Gradient Methods: Comparison

- Two kinds of policy gradient estimator
  - REINFORCE / score function estimator: $\nabla \log \pi(a \mid s)\hat{A}$.
    - Learn Q or V for variance reduction, to estimate $\hat{A}$
  - Pathwise derivative estimators (differentiate wrt action)
    - SVG(0) / DPG: $\frac{\mathrm{d}}{\mathrm{d}a} Q(s, a)$ (learn Q)
    - SVG(1): $\frac{\mathrm{d}}{\mathrm{d}a}(r + \gamma V(s'))$ (learn f, V)
    - SVG(∞): $\frac{\mathrm{d}}{\mathrm{d}a_t}(r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots)$ (learn f)
- Pathwise derivative methods more sample-efficient when they work (maybe), but work less generally due to high bias

Policy Gradient Methods vs Q-Function Regression Methods

- Q-function regression methods are more sample-efficient when they work, but don't work as generally
- Policy gradients are easier to debug and understand
  - Don't have to deal with "burn-in" period
  - When it's working, performance should be monotonically increasing
  - Diagnostics like KL, entropy, baseline's explained variance
- Q-function regression methods are more compatible with exploration and off-policy learning
- Policy-gradient methods are more compatible with recurrent policies
- Q-function regression methods CAN be used with continuous action spaces (e.g., S. Gu, T. Lillicrap, I. Sutskever, and S. Levine. "Continuous deep Q-learning with model-based acceleration". (2016)) but final performance is worse (so far)

Recent Papers on Connecting Policy Gradients and Q-function Regression

- B. O'Donoghue, R. Munos, K. Kavukcuoglu, and V. Mnih. "PGQ: Combining policy gradient and Q-learning". (2016)
- Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, et al. "Sample Efficient Actor-Critic with Experience Replay". (2016)
  - Uses adjusted returns: A. Harutyunyan, M. G. Bellemare, T. Stepleton, and R. Munos. "Q(λ) with Off-Policy Corrections". 2016, N. Jiang and L. Li. "Doubly robust off-policy value evaluation for reinforcement learning". 2016
- O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans. "Bridging the Gap Between Value and Policy Based Reinforcement Learning". (2017)
- T. Haarnoja, H. Tang, P. Abbeel, and S. Levine. "Reinforcement Learning with Deep Energy-Based Policies". (2017)
- O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans. "Bridging the Gap Between Value and Policy Based Reinforcement Learning". (2017)