10703 Deep Reinforcement Learning

Policy Gradient Methods – Part 3

Tom Mitchell

October 8, 2018

Recommended readings: next slide.
(not covered in Barto & Sutton)

Used Materials

- **Disclaimer**: Much of the material and slides for this lecture were borrowed from Ruslan Salakhutdinov, who in turn borrowed from Rich Sutton's RL class and David Silver's Deep RL tutorial

Recommended Readings on Natural Policy Gradient and Convergence of Actor-Critic Learning

Bhatnagar, S., Sutton, R., Ghavamzadeh, M., Lee, M. (2009). Natural actor–critic algorithms. *Automatica, 45*(11).

Grondman, I., Busoniu, L., Lopes, G. A., Babuska, R. (2012). A survey of actor–critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42*(6):1291–1307.

Kakade, S. M. (2002). A natural policy gradient. In *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, pp. 1531–1538. MIT Press, Cambridge, MA.

Peters, J., Schaal, S. (2008). Natural actor–critic. *Neurocomputing, 71*(7):1180–1190.

Recall the Policy Gradient Theorem:

$$\nabla_\theta J(\pi_\theta) = \sum_s d^{\pi_\theta}(s) \sum_a Q^{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(s,a)$$

where $d^{\pi_\theta}(s)$ is distribution over states generated by following $\pi_\theta(s,a)$

We can rewrite the Policy Gradient Theorem in several forms:

$$\begin{aligned}
\nabla_\theta J(\pi_\theta) &= \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} \\
&= E_{\pi_\theta}\left[\ Q^{\pi_\theta}(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)}\right] \\
&= E_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\ \nabla_\theta \ln \pi_\theta(s,a)\right]
\end{aligned}$$

Actor-Critic

- Monte-Carlo policy gradient still has high variance
- We can use a critic to estimate the action-value function:

$$Q_w(s, a) \approx Q^{\pi_\theta}(s, a)$$

- Actor-critic algorithms maintain two sets of parameters
  - Critic Updates action-value function parameters w
  - Actor Updates policy parameters θ, in direction suggested by critic
- Actor-critic algorithms follow an approximate policy gradient

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s, a)\ Q_w(s, a)\right]$$
$$\Delta\theta = \alpha \nabla_\theta \log \pi_\theta(s, a)\ Q_w(s, a)$$

Reducing Variance Using a Baseline

- We can subtract a baseline function B(s) from the policy gradient
- This can reduce variance, without changing expectation!

$$\mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) B(s)\right] = 0$$

- A good baseline is the state value function $B(s) = V^{\pi_\theta}(s)$
- So we can rewrite the policy gradient using the advantage function:

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

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

- Note that it is the exact same policy gradient:

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

Estimating the Advantage Function

- For the true value function $V^{\pi_\theta}(s)$ the TD error:

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

  is an unbiased estimate of the advantage function:

$$\begin{aligned}
\mathbb{E}_{\pi_\theta}\left[\delta^{\pi_\theta} | s, a\right] &= \mathbb{E}_{\pi_\theta}\left[r + \gamma V^{\pi_\theta}(s') | s, a\right] - V^{\pi_\theta}(s) \\
&= Q^{\pi_\theta}(s, a) - V^{\pi_\theta}(s) \\
&= A^{\pi_\theta}(s, a)
\end{aligned}$$

- So we can use the TD error to compute the policy gradient

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

- Remember the policy gradient

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

Estimating the Advantage Function

- For the true value function $V^{\pi_\theta}(s)$ the TD error:

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

  is an unbiased estimate of the advantage function

- So we can use the TD error to compute the policy gradient

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

- In practice we can use an approximate TD error

$$\delta_v = r + \gamma V_v(s') - V_v(s)$$

- This approach only requires one set of critic parameters v

Dueling Networks

- Split Q-network into two channels
- Action-independent value function V(s,v)
- Action-dependent advantage function A(s, a, w)

$$Q(s,a) = V(s,v) + A(s,a,\mathbf{w})$$

- Advantage function is defined as:

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

Wang et.al., ICML, 2016

Advantage Actor-Critic Algorithm

One-step Actor-Critic (episodic)

Input: a differentiable policy parameterization $\pi(a|s,\boldsymbol{\theta}), \forall a \in \mathcal{A}, s \in \mathcal{S}, \boldsymbol{\theta} \in \mathbb{R}^n$
Input: a differentiable state-value parameterization $\hat{v}(s,\mathbf{w}), \forall s \in \mathcal{S}, \mathbf{w} \in \mathbb{R}^m$
Parameters: step sizes $\alpha > 0$, $\beta > 0$

Initialize policy weights $\boldsymbol{\theta}$ and state-value weights $\mathbf{w}$
Repeat forever:
  Initialize S (first state of episode)
  $I \leftarrow 1$
  While S is not terminal:
    $A \sim \pi(\cdot|S,\boldsymbol{\theta})$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
    $\mathbf{w} \leftarrow \mathbf{w} + \beta\,\delta\,\nabla_{\mathbf{w}}\hat{v}(S,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha I \delta \nabla_{\boldsymbol{\theta}} \log \pi(A|S,\boldsymbol{\theta})$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$

So Far: Summary of PG Algorithms

- The policy gradient has many equivalent forms

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\, G_t\right] && \text{REINFORCE} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; Q^w(s,a)\right] && \text{Q Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; A^w(s,a)\right] && \text{Advantage Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; \delta\right] && \text{TD Actor-Critic}
\end{aligned}$$

- Each leads a stochastic gradient ascent algorithm

- Critic uses policy evaluation (e.g. MC or TD learning) to estimate $Q^\pi(s,a)$, $A^\pi(s,a)$ or $V^\pi(s)$

But will it converge if we use function approximation??

Under what conditions??

Bias in Actor-Critic Algorithms

- Approximating the policy gradient introduces bias

$$Q_w(s,a) \approx Q^{\pi_\theta}(s,a)$$

- A biased policy gradient may not find the right solution
- Luckily, if we choose value function approximation carefully
- Then we can avoid introducing any bias
- i.e. we can still follow the exact policy gradient

Compatible Function Approximation

- If the following two conditions are satisfied:

  1. Value function approximator is compatible with the policy

$$\boxed{\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)}$$

  2 Value function parameters w minimize the mean-squared error

$$\boxed{\varepsilon = \mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_w(s,a)\right)^2\right]}$$

- Then the policy gradient is exact,

$$\boxed{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q_w(s,a)\right]}$$

- Remember:

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

Proof

$$\boxed{\varepsilon = \mathbb{E}_{\pi_\theta}\left[(Q^{\pi_\theta}(s,a) - Q_w(s,a))^2\right]}$$

- If w is chosen to minimize mean-squared error ε, then gradient of ε with respect to w must be zero,

$$\begin{aligned}
\nabla_w \varepsilon &= 0 \\
\mathbb{E}_{\pi_\theta}\left[(Q^\theta(s,a) - Q_w(s,a))\nabla_w Q_w(s,a)\right] &= 0 \\
\mathbb{E}_{\pi_\theta}\left[(Q^\theta(s,a) - Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a)\right] &= 0 \\
\mathbb{E}_{\pi_\theta}\left[Q^\theta(s,a)\nabla_\theta \log \pi_\theta(s,a)\right] &= \mathbb{E}_{\pi_\theta}\left[Q_w(s,a)\nabla_\theta \log \pi_\theta(s,a)\right]
\end{aligned}$$

- So $Q_w(s, a)$ can be substituted directly into the policy gradient,

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

- Remember: $\boxed{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)\right]}$

Proof

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

- If w is chosen to minimize mean-squared error ε, then gradient of ε with respect to w must be zero,

$$\nabla_w \varepsilon = 0$$

$$\mathbb{E}_{\pi_\theta}\left[\left(Q^{\theta}(s,a) - Q_w(s,a)\right)\nabla_w Q_w(s,a)\right] = 0$$

$$\mathbb{E}_{\pi_\theta}\left[\left(Q^{\theta}(s,a) - Q_w(s,a)\right)\nabla_\theta \log \pi_\theta(s,a)\right] = 0$$

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

note error ε need not be zero, just needs to be minimized!

note we only need

$$\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$$

to within a constant!

- So $Q_w(s, a)$ can be substituted directly into the policy gradient,

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

- Remember: $\boxed{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)\right]}$

Compatible Function Approximation

- If the following two conditions are satisfied:

  1. Value function approximator is compatible with the policy

$$\boxed{\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)}$$

How can we achieve this??

  2 Value function parameters w minimize the mean-squared error

$$\boxed{\varepsilon = \mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_w(s,a)\right)^2\right]}$$

- Then the policy gradient is exact,

$$\boxed{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q_w(s,a)\right]}$$

- Remember:

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

Compatible Function Approximation

- If the following two conditions are satisfied:

    1. Value function approximator is compatible with the policy

$$\boxed{\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)}$$

How can we achieve this??

One way: make Q_w and π_θ both be linear functions of same features of s,a

- let $\Phi(s,a)$ be a vector of features describing the pair (s,a)
- let $Q_w(s,a) = \mathbf{w}^T\Phi(s,a)$. let $\log \pi_\theta(s,a) = \boldsymbol{\theta}^T\Phi(s,a)$
- then $\nabla_w Q_w(s,a) = \phi(s,a) = \nabla_\theta \pi_\theta(s,a)$

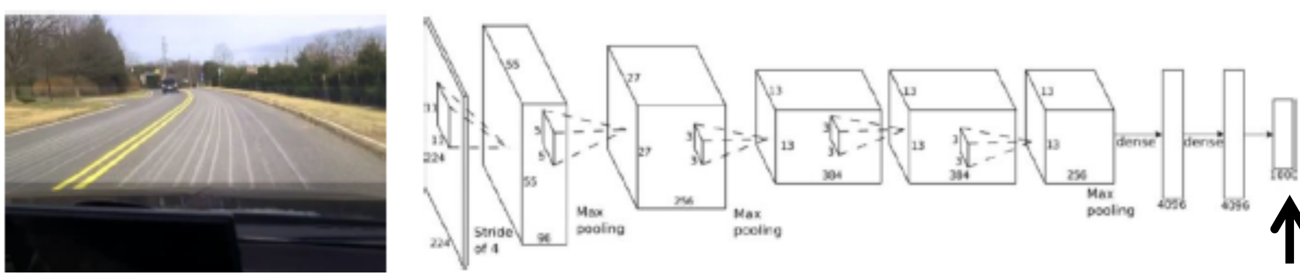
Compatible Function Approximation

$$\boxed{\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)}$$

How can we achieve this??

One way: make Q_w and π_θ both be linear functions of same features of s,a

- let $\Phi(s,a)$ be a vector of features describing the pair (s,a)
- let $Q_w(s,a) = \mathbf{w}^T\Phi(s,a)$. let $\log \pi_\theta(s,a) = \boldsymbol{\theta}^T\Phi(s,a)$
- then $\nabla_w Q_w(s,a) = \phi(s,a) = \nabla_\theta \pi_\theta(s,a)$



$\Phi(s)$

$Q_w(s,a) = \mathbf{w}_a^T\,\Phi(s)$

$\log \pi_\theta(s,a) = \theta_a^T\,\Phi(s)$

Alternative Policy Gradient Directions

- Generalized gradient ascent algorithms can follow any ascent direction
- A good ascent direction can significantly speed convergence
- Also, a policy can often be reparametrized without changing action probabilities
- For example, increasing score of all actions in a softmax policy
- The vanilla gradient is sensitive to these reparametrizations
- but the **natural gradient** *is not!*

Natural Policy Gradient

- The natural policy gradient is parameterization independent (i.e., not influenced by set of parameters you use to define
- it finds ascent direction that is closest to vanilla gradient

$$\boxed{\nabla_\theta^{nat} \pi_\theta(s, a) = G_\theta^{-1} \nabla_\theta \pi_\theta(s, a)}$$

- where G_θ is the Fisher information matrix

$$\boxed{G_\theta = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a)^T\right]}$$

Natural Policy Gradient

- The natural policy gradient is parameterization independent (i.e., not influenced by set of parameters you use to define

$$\boxed{\nabla_\theta^{nat} \pi_\theta(s, a) = G_\theta^{-1} \nabla_\theta \pi_\theta(s, a)}$$

- where G_θ is the Fisher information matrix

$$\boxed{G_\theta = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a)^T\right]}$$

  - what is the <i, j>th element of G_θ ?

  - what is G_θ if we have a parameterization that yields the natural gradient?

Natural Actor-Critic

Under linear model:
$A^{\pi_\theta}(s,a) = \phi(s)^\top \mathrm{w}$

- Using compatible function approximation,

$$\nabla_w A_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$$

- The natural policy gradient simplifies,

$$\begin{aligned}\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) A^{\pi_\theta}(s,a)\right] \\ &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a)^T w\right] \\ &= G_\theta w\end{aligned}$$

$$\nabla_\theta^{nat} \pi_\theta(s,a) = G_\theta^{-1} \nabla_\theta \pi_\theta(s,a)$$

$$\nabla_\theta^{nat} J(\theta) = w$$

- i.e. update actor parameters in direction of critic parameters!

from: Peters and Schaal

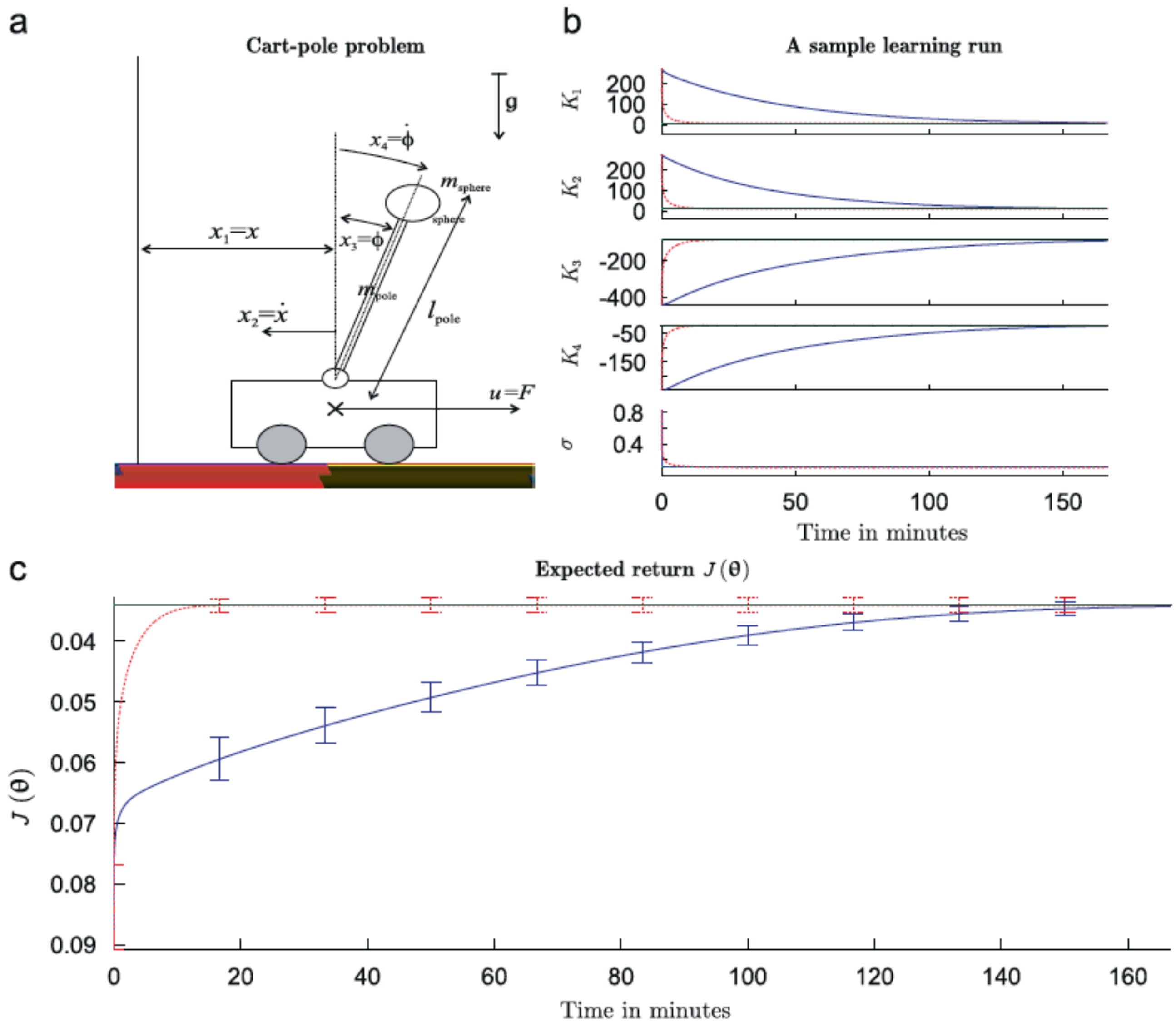


Fig. 3. This figure shows the performance of Natural Actor-Critic in the Cart-Pole Balancing framework. In (a), you can see the general setup of the pole mounted on the cart. In (b), a sample learning run of the both Natural Actor-Critic and the true policy gradient is given. The dashed line denotes the Natural Actor-Critic performance while the solid line shows the policy gradients performance. In (c), the expected return of the policy is shown. This is an average over 100 randomly picked policies as described in Section 4.1.

from: Kakade

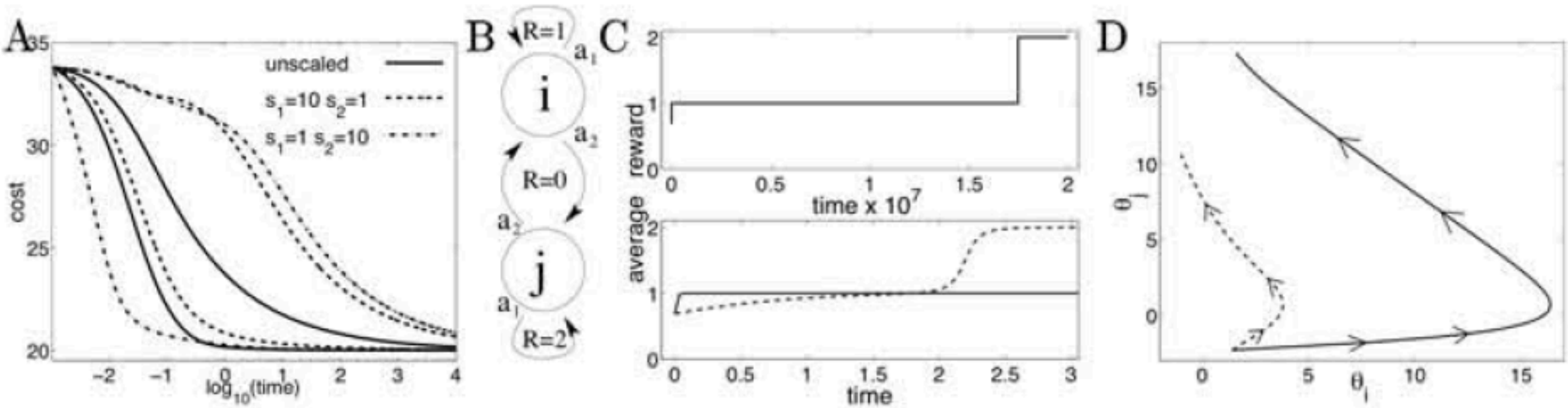


Figure 1: A) The cost vs. $\log_{10}(\text{time})$ for an LQG (with 20 time step trajectories). The policy used was $\pi(u; x, \theta) \propto \exp(\theta_1 s_1 x^2 + \theta_2 s_2 x)$ where the rescaling constants, s_1 and s_2, are shown in the legend. Under equivalent starting distributions ($\theta_1 s_1 = \theta_2 s_2 = -.8$), the right-most three curves are generated using the standard gradient method and the rest use the natural gradient. B) See text. C top) The average reward vs. time (on a 10^7 scale) of a policy under standard gradient descent using the sigmoidal policy parameterization ($\pi(1; s, \theta_i) \propto \exp(\theta_i)/(1 + \exp(\theta_i))$), with the initial conditions $\pi(i, 1) = .8$ and $\pi(j, 1) = .1$. C bottom) The average reward vs. time (unscaled) under standard gradient descent (solid line) and natural gradient descent (dashed line) for an early window of the above plot. D) Phase space plot for the standard gradient case (the solid line) and the natural gradient case (dashed line).

Summary of Policy Gradient Algorithms

- The policy gradient has many equivalent forms

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ v_t\right] && \text{REINFORCE} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q^w(s,a)\right] && \text{Q Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ A^w(s,a)\right] && \text{Advantage Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ \delta\right] && \text{TD Actor-Critic} \\
G_\theta^{-1}\nabla_\theta J(\theta) &= w && \text{Natural Actor-Critic}
\end{aligned}
$$

- Each leads a stochastic gradient ascent algorithm
- Critic uses policy evaluation (e.g. MC or TD learning) to estimate $Q^\pi(s,a)$, $A^\pi(s,a)$ or $V^\pi(s)$