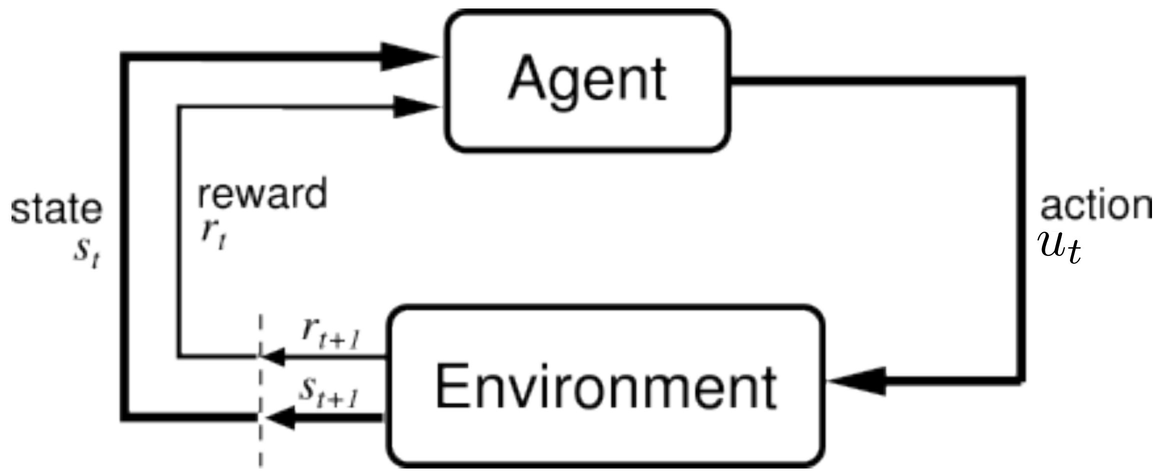
Reinforcement Learning – Policy Optimization

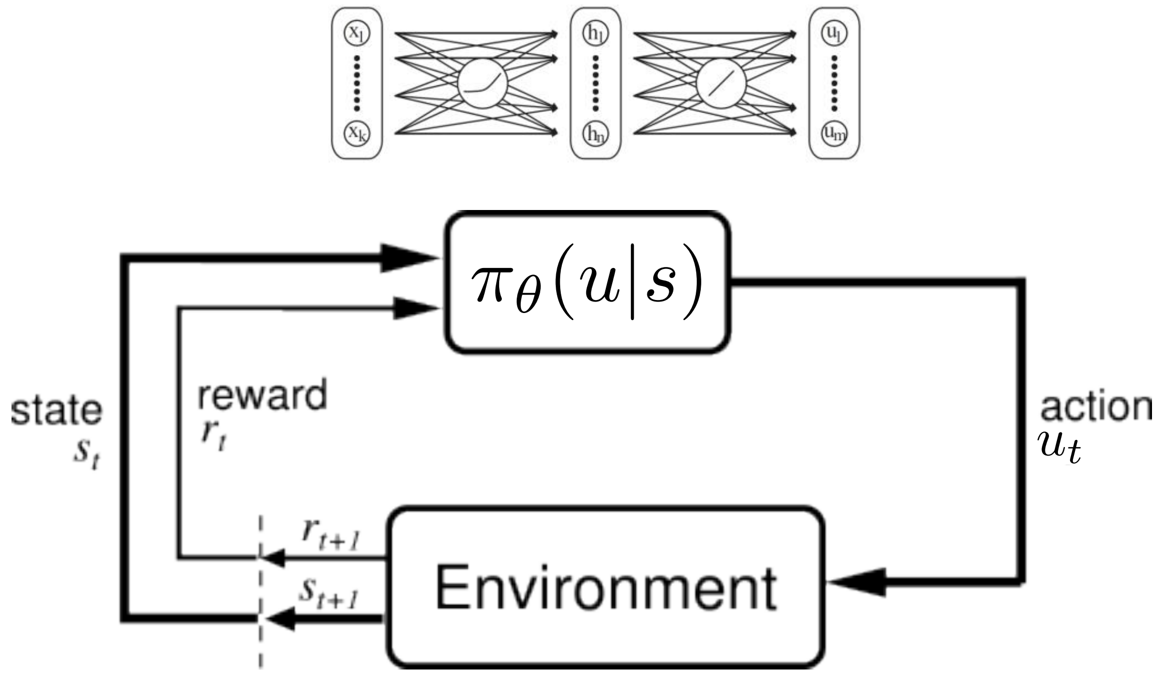
Pieter Abbeel
UC Berkeley / OpenAI / Gradescope

Reinforcement Learning



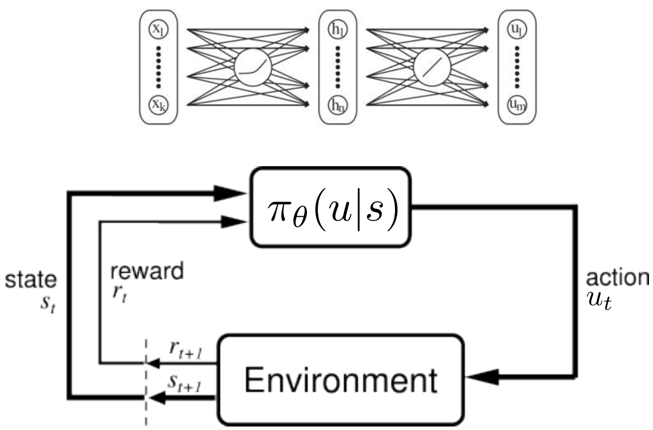
[Figure source: Sutton & Barto, 1998]

Policy Optimization



[Figure source: Sutton & Barto, 1998]

Policy Optimization



- Consider control policy parameterized by parameter vector θ

$$\max_\theta \quad \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

- Often stochastic policy class (smooths out the problem):

  $\pi_\theta(u|s)$: probability of action u in state s

[Figure source: Sutton & Barto, 1998]

Why Policy Optimization

- Often π can be simpler than Q or V
  - E.g., robotic grasp
- V: doesn't prescribe actions
  - Would need dynamics model (+ compute 1 Bellman back-up)
- Q: need to be able to efficiently solve $\arg\max_u Q_\theta(s, u)$
  - Challenge for continuous / high-dimensional action spaces*

*some recent work (partially) addressing this:
NAF: Gu, Lillicrap, Sutskever, Levine ICML 2016
Input Convex NNs: Amos, Xu, Kolter arXiv 2016

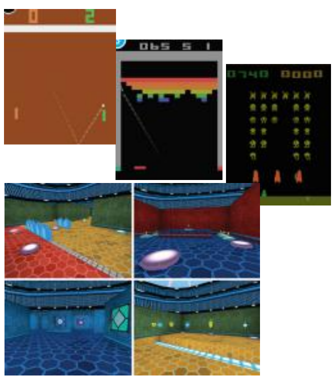
Example Policy Optimization Success Stories

Kohl and Stone, 2004

Ng et al, 2004

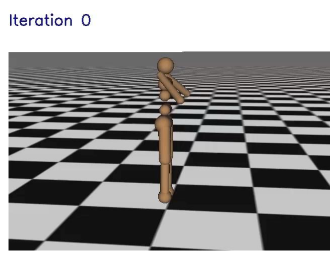
Tedrake et al, 2005

Kober and Peters, 2009

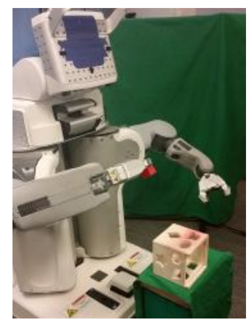


Mnih et al, 2015
(A3C)

Silver et al, 2014
(DPG)
Lillicrap et al, 2015
(DDPG)



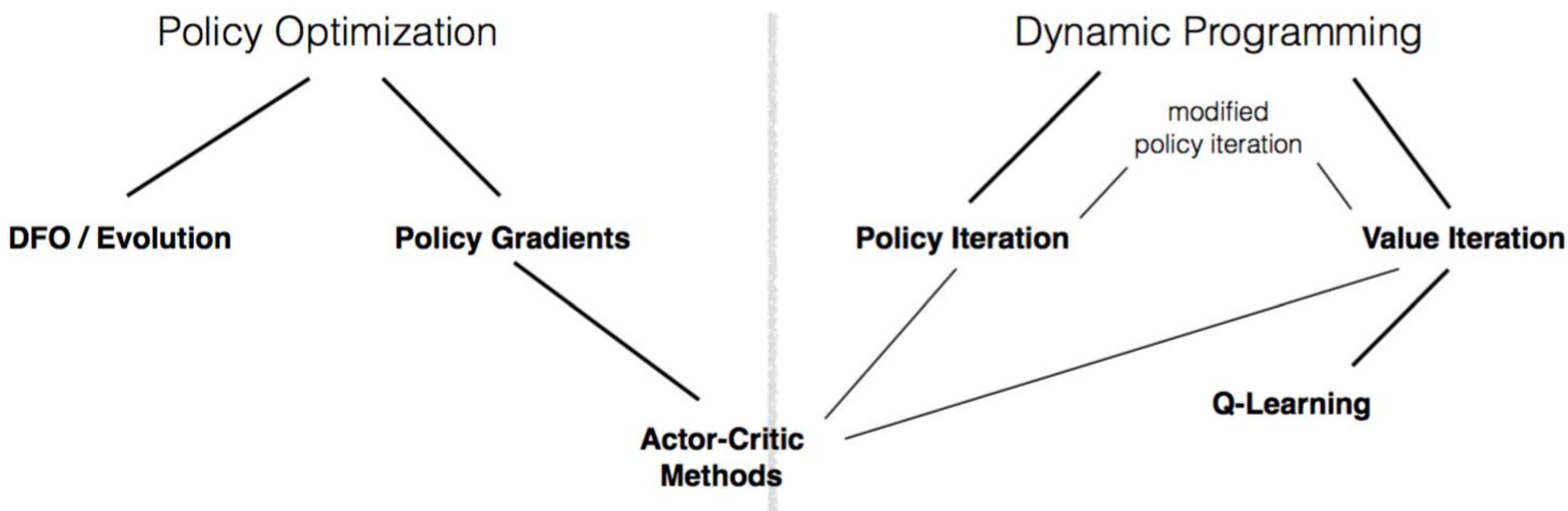
Schulman et al,
2016 (TRPO + GAE)



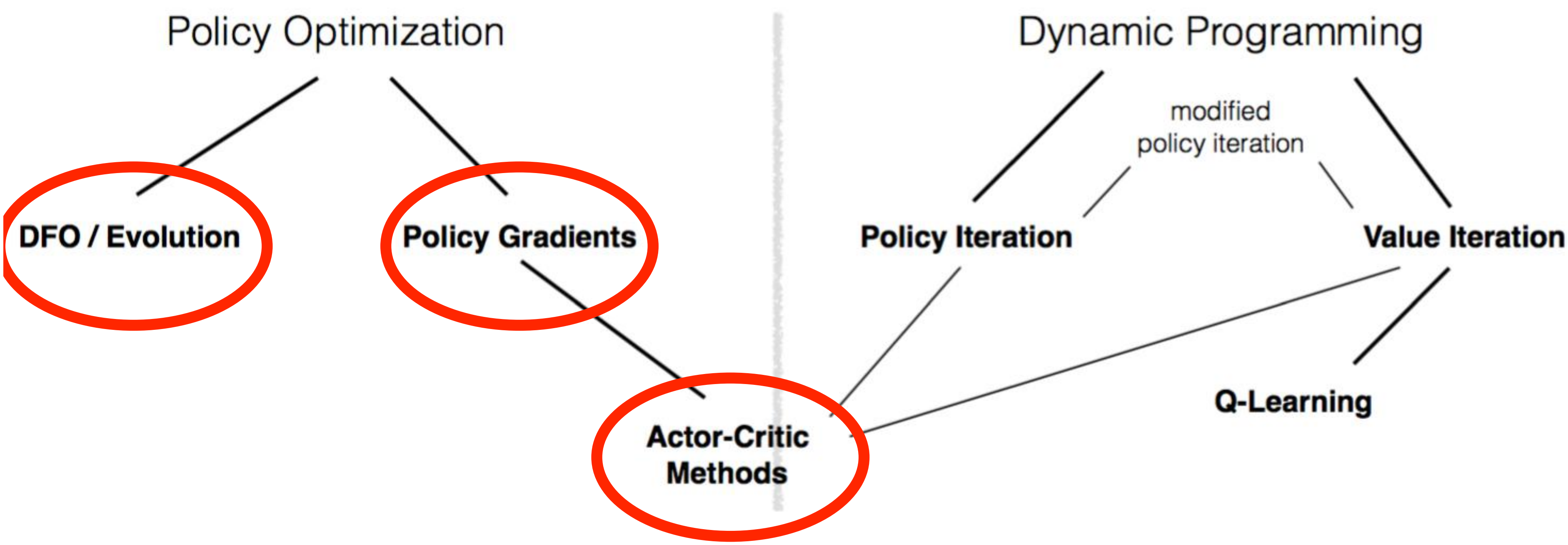
Levine*, Finn*, et
al, 2016
(GPS)

Silver*, Huang*, et
al, 2016
(AlphaGo**)

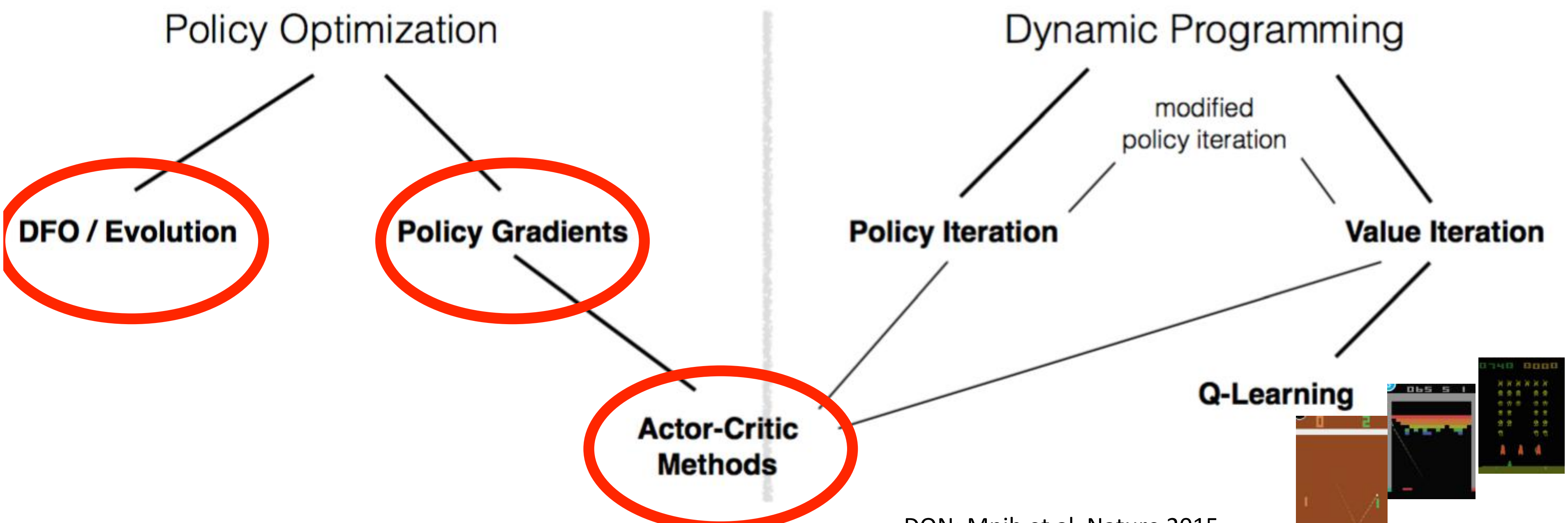
Policy Optimization in the RL Landscape



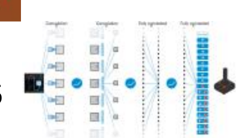
Policy Optimization in the RL Landscape



Policy Optimization in the RL Landscape



DQN: Mnih et al, Nature 2015
Double DQN: Van Hasselt et al, AAAI 2015
Dueling Architecture: Wang et al, ICML 2016
Prioritized Replay: Schaul et al, ICLR 2016
David Silver ICML 2016 tutorial



Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

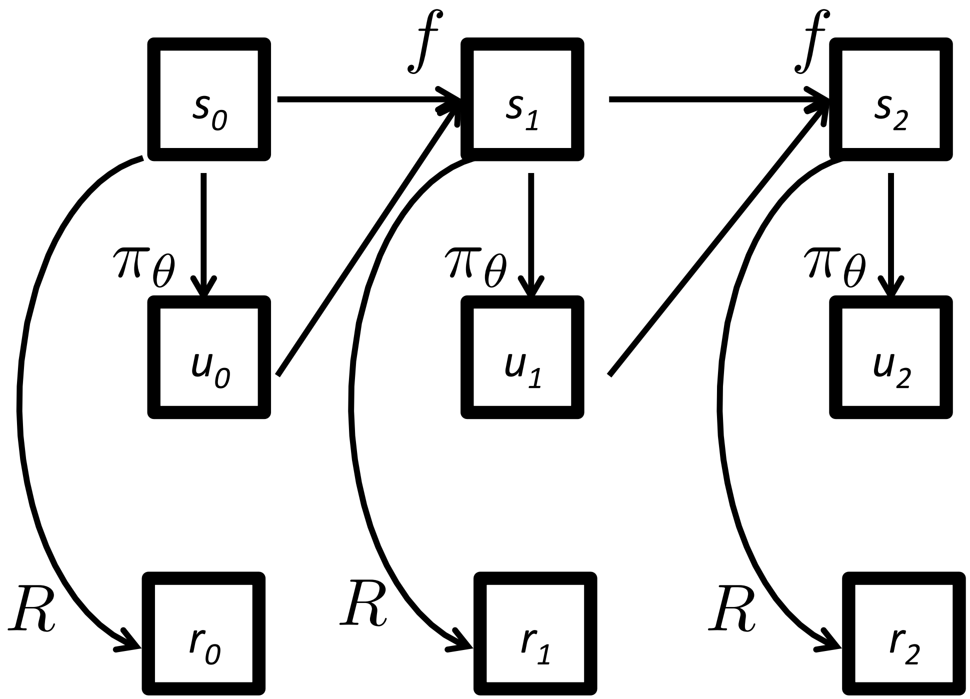
Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

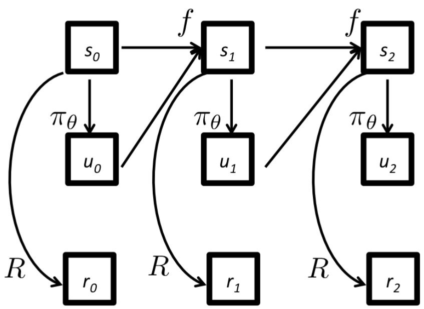
- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$r_t = R(s_t)$$
$$u_t = \pi_\theta(s_t)$$
$$s_{t+1} = f(s_t, u_t)$$

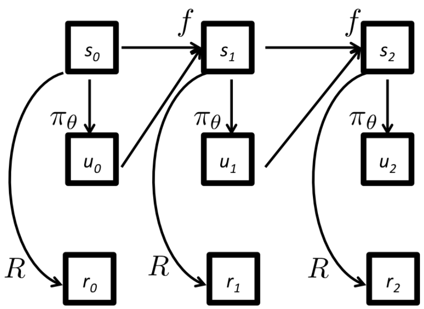
Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \ \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$

- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



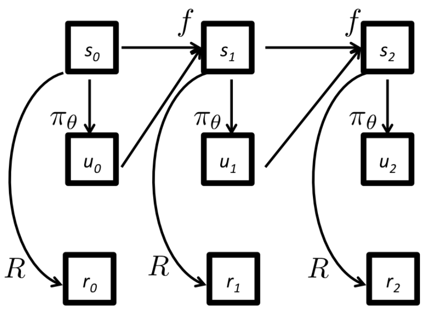
$$\max_\theta U(\theta) = \max_\theta \ \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$

$$= \max_\theta \ r_0 + r_1 + r_2$$

- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

- fixed s0

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \ \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$
$$= \max_\theta \ r_0 + r_1 + r_2$$

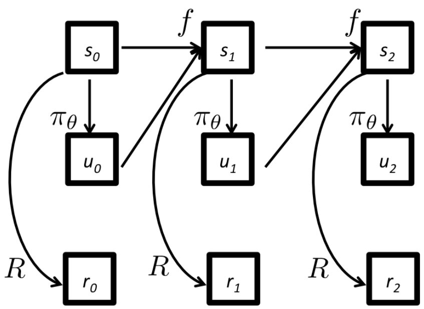
- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

- fixed s0

- Can compute gradient estimate along roll-out from s0:

$$\frac{\partial U}{\partial \theta_i} = \sum_{t=0}^{H} \frac{\partial R}{\partial s}(s_t)\frac{\partial s_t}{\partial \theta_i}$$
$$\frac{\partial s_t}{\partial \theta_i} = \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial s_{t-1}}{\partial \theta_i} + \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial u_{t-1}}{\partial \theta_i}$$
$$\frac{\partial u_t}{\partial \theta_i} = \frac{\partial \pi_\theta}{\partial \theta_i}(s_t, \theta) + \frac{\partial \pi_\theta}{\partial s}(s_t, \theta)\frac{\partial s_t}{\partial \theta_i}$$

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \; \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$
$$= \max_\theta \; r_0 + r_1 + r_2$$

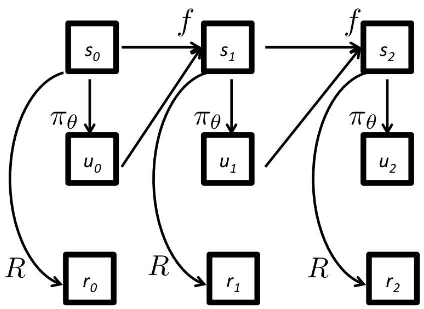
- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

- fixed s0

- Can compute gradient estimate along roll-out from s0:

$$\frac{\partial U}{\partial \theta_i} = \sum_{t=0}^{H} \frac{\partial R}{\partial s}(s_t)\frac{\partial s_t}{\partial \theta_i}$$

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \; \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$
$$= \max_\theta \; r_0 + r_1 + r_2$$

- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

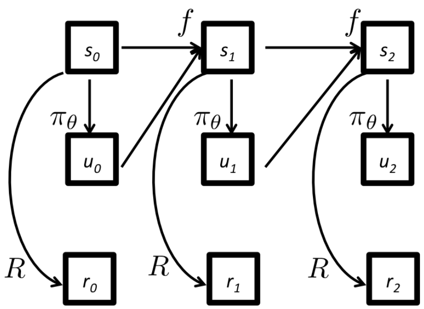
- fixed s0

- Can compute gradient estimate along roll-out from s0:

$$\frac{\partial U}{\partial \theta_i} = \sum_{t=0}^{H} \frac{\partial R}{\partial s}(s_t) \frac{\partial s_t}{\partial \theta_i}$$

$$\frac{\partial s_t}{\partial \theta_i} = \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1}) \frac{\partial s_{t-1}}{\partial \theta_i} + \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1}) \frac{\partial u_{t-1}}{\partial \theta_i}$$

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \ \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$

$$= \max_\theta \ r_0 + r_1 + r_2$$

- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

- fixed s0

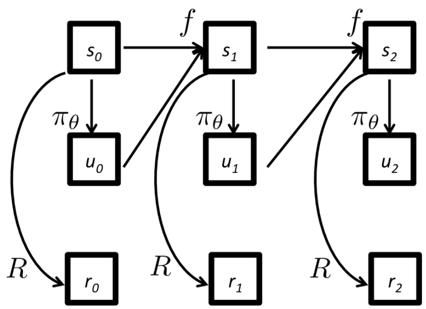
- Can compute gradient estimate along roll-out from s0:

$$\frac{\partial U}{\partial \theta_i} = \sum_{t=0}^{H} \frac{\partial R}{\partial s}(s_t)\frac{\partial s_t}{\partial \theta_i}$$

$$\frac{\partial s_t}{\partial \theta_i} = \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial s_{t-1}}{\partial \theta_i} + \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial u_{t-1}}{\partial \theta_i}$$

$$\frac{\partial u_t}{\partial \theta_i} = \frac{\partial \pi_\theta}{\partial \theta_i}(s_t, \theta) + \frac{\partial \pi_\theta}{\partial s}(s_t, \theta)\frac{\partial s_t}{\partial \theta_i}$$

Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)



$$\max_\theta U(\theta) = \max_\theta \; \mathbb{E}\left[\sum_{t=0}^{H} r_t | \pi_\theta\right]$$
$$= \max_\theta \; r_0 + r_1 + r_2$$

- f known, det., diff.
- R known, det., diff.
- π_θ (known), det., diff.

- fixed s0

- Can compute gradient estimate along roll-out from s0:

$$\frac{\partial U}{\partial \theta_i} = \sum_{t=0}^{H} \frac{\partial R}{\partial s}(s_t)\frac{\partial s_t}{\partial \theta_i}$$
$$\frac{\partial s_t}{\partial \theta_i} = \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial s_{t-1}}{\partial \theta_i} + \frac{\partial f}{\partial s}(s_{t-1}, u_{t-1})\frac{\partial u_{t-1}}{\partial \theta_i}$$
$$\frac{\partial u_t}{\partial \theta_i} = \frac{\partial \pi_\theta}{\partial \theta_i}(s_t, \theta) + \frac{\partial \pi_\theta}{\partial s}(s_t, \theta)\frac{\partial s_t}{\partial \theta_i}$$

- Roll-out = forward prop
- Gradient = back-prop through time
- Multiple s0 → multiple roll-outs / bptt

Path Derivative for Stochastic f – Additive Noise

$$s_{t+1} = f(s_t, u_t) + w_t$$

for any given roll-out, simply consider w_0, w_1,…, w_H fixed (just like we considered s_0 fixed)

→ run backpropagation through time just like for deterministic f

Path Derivative for Stochastic f – Reparameterization Trick

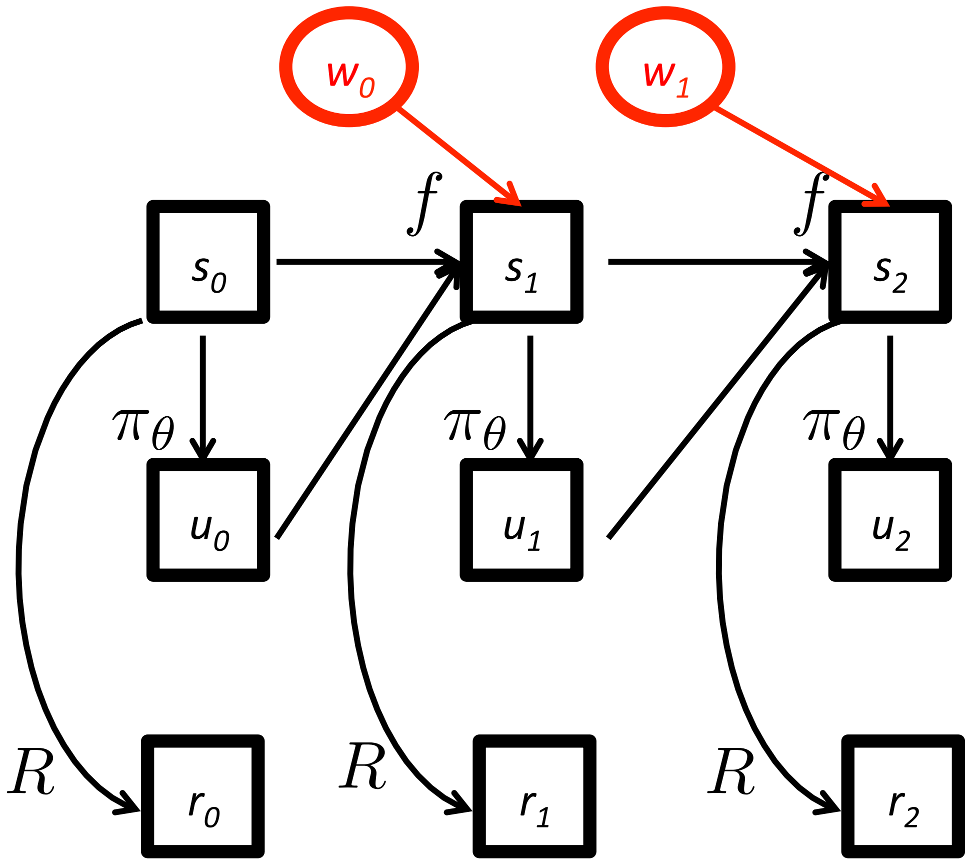
Path Derivative for Stochastic f – Reparameterization Trick

- Original: $$s_{t+1} = f_{\text{STOCH}}(s_t, u_t)$$
- Reparameterized: $$s_{t+1} = f_{\text{DET}}(s_t, u_t, w_t)$$
- E.g. $$s_{t+1} \sim \mathcal{N}(g(s_t, u_t), \sigma^2)$$

→ $$s_{t+1} = g(s_t, u_t) + \sigma w_t \qquad w_t \sim \mathcal{N}(0, I)$$

Stochastic Dynamics f

- f known, ~~det.~~, diff.
- R known, det., diff.
- π_θ (known), det., diff.



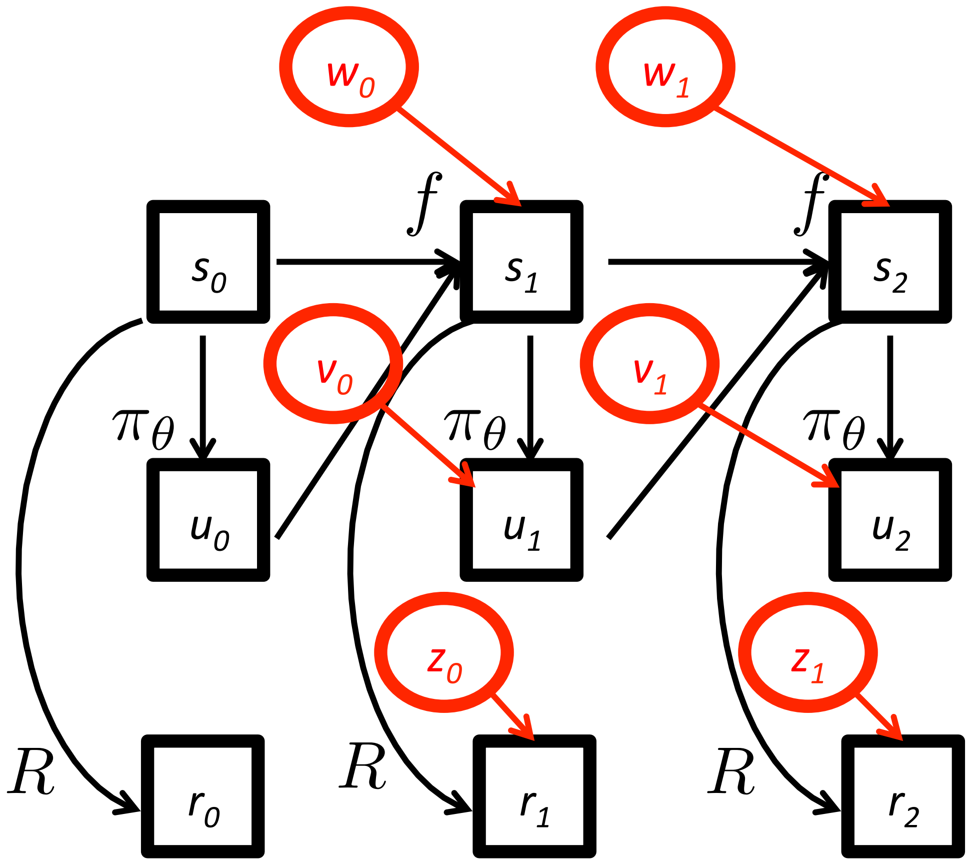
$$r_t = R(s_t)$$

$$u_t = \pi_\theta(s_t)$$

$$s_{t+1} = f(s_t, u_t, w_t)$$

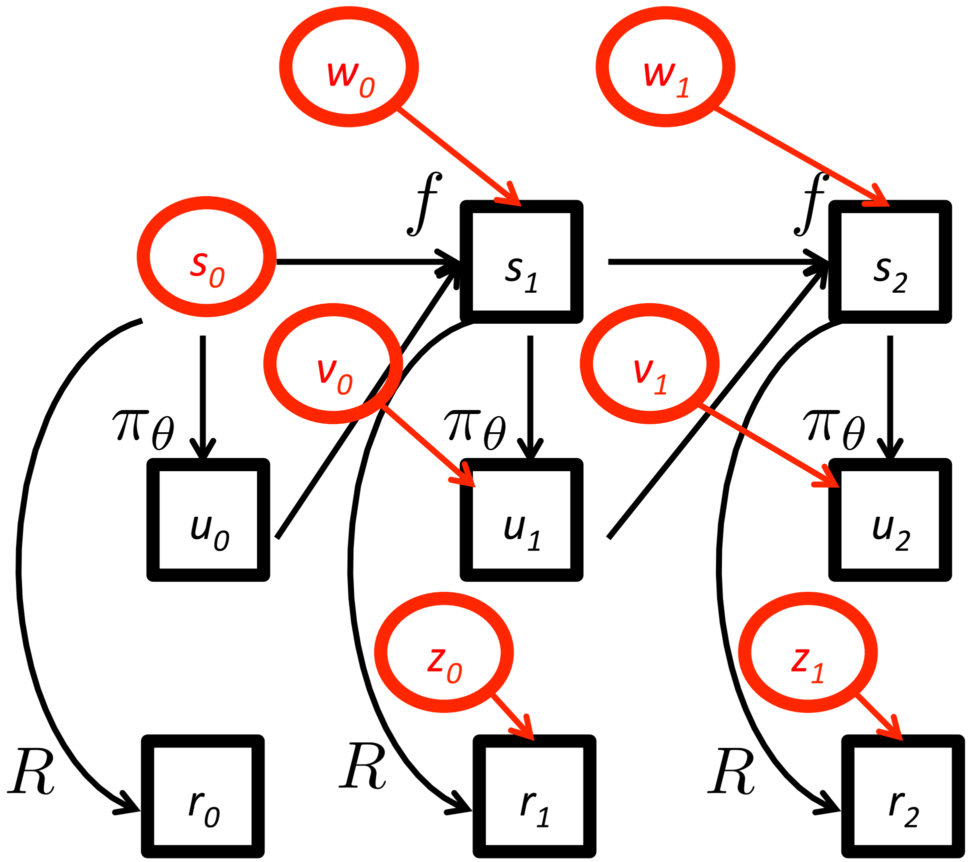
Stochastic f, R and π_θ

- f known, ~~det.~~, diff.
- R known, ~~det.~~, diff.
- π_θ (known), ~~det.~~, diff.



$$r_t = R(s_t, z_t)$$
$$u_t = \pi_\theta(s_t, v_t)$$
$$s_{t+1} = f(s_t, u_t, w_t)$$

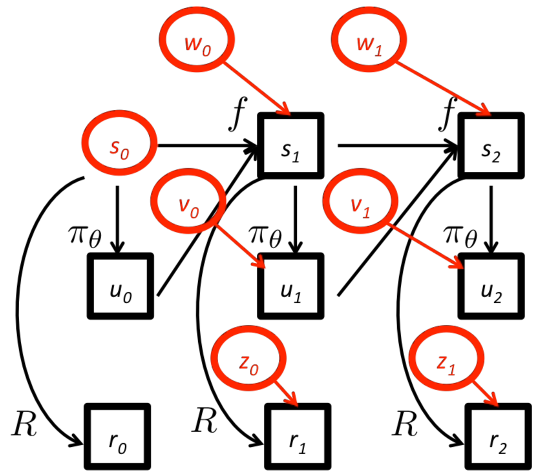
Stochastic f, R and π_θ and s0



- f known, ~~det.~~, diff.
- R known, ~~det.~~, diff.
- π_θ (known), ~~det.~~, diff.

$$r_t = R(s_t, z_t)$$
$$u_t = \pi_\theta(s_t, v_t)$$
$$s_{t+1} = f(s_t, u_t, w_t)$$

PD/BPTT Policy Gradients: Complete Algorithm



$$r_t = R(s_t, z_t)$$

$$u_t = \pi_\theta(s_t, v_t)$$

$$s_{t+1} = f(s_t, u_t, w_t)$$

- f known, ~~det.~~, diff.
- R known, ~~det.~~, diff.
- π_θ (known), ~~det.~~, diff.

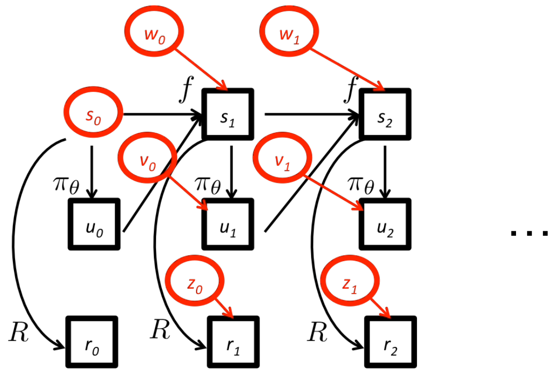
Algorithm:

- for iter = 1, 2, …
  - for roll-out r = 1, 2, …
    - sample s0, w0, w1,…, v0, v1,…, z0, z1, …
    - Forward-pass (=execute roll-out)
    - Backprop to compute gradient estimate
  - average all gradient estimates
  - take step in gradient direction

f, *R* not known

→ could learn from roll-outs (= model-based RL)

PD Policy Gradients -- Variance



- For long horizon H, variance of gradient estimate can be impractically high
- Solutions: instead of $r_t + r_{t+1} + \ldots + r_H$
  - Discounting: $r_t + \gamma r_{t+1} + \ldots + \gamma^{H-t} r_H$
  - Learn value function (SVG): $r_t + \gamma r_{t+1} + \ldots + \gamma^k r_{t+k} + \gamma^{k+1} V^{\pi_\theta}(s_{t+k+1})$
  - Learn Q function (DDPG): $Q^{\pi_\theta}(s_t, u_t)$

[SVG: Heess et al, 2015; DPG: Silver, 2014, DDPG Lillicrap et al, 2015]

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - ***Parameter Perturbation / Evolutionary Strategies***
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Black Box Gradient Computation

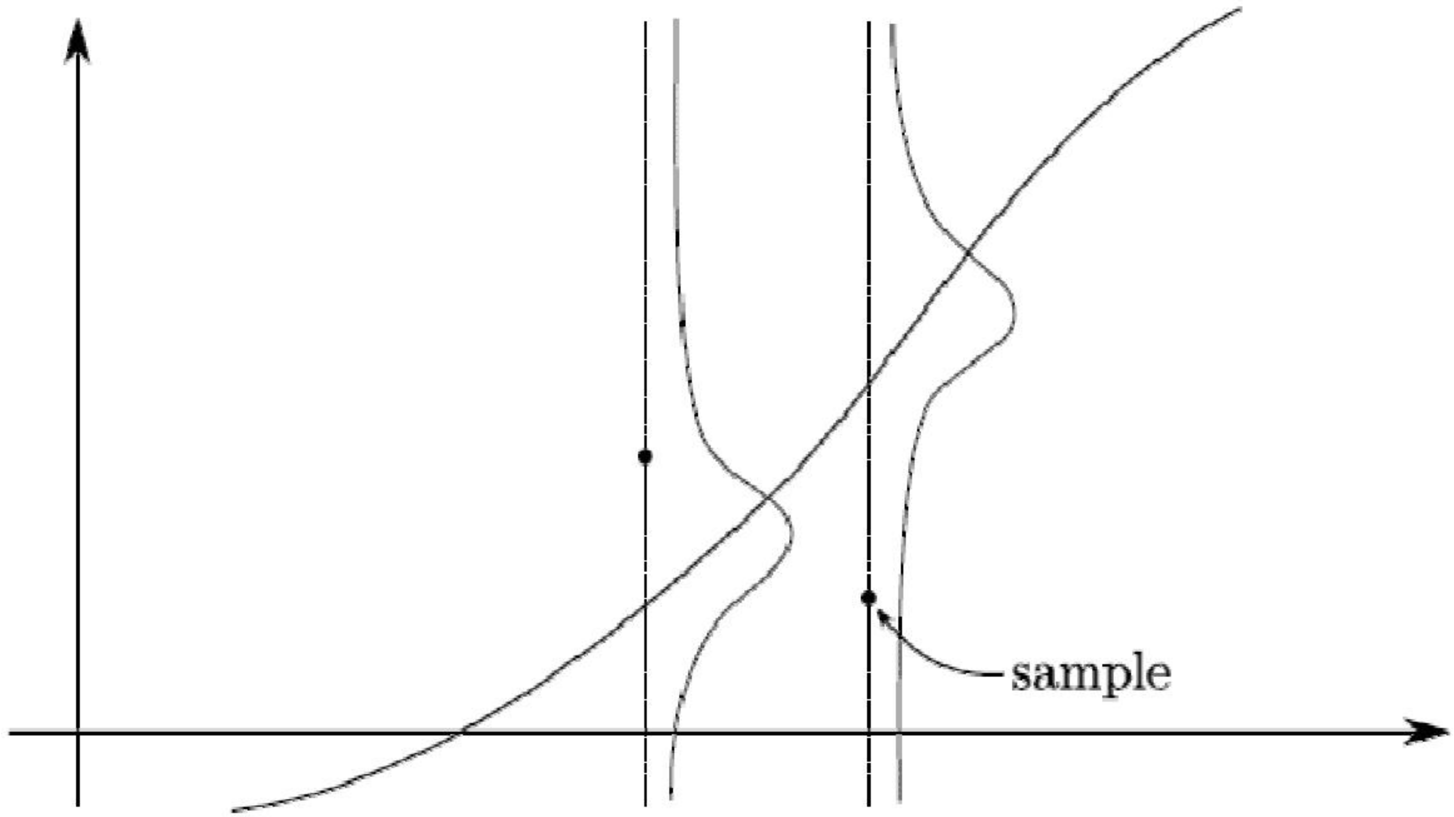
We can compute the gradient g using standard finite difference methods, as follows:

$$\frac{\partial U}{\partial \theta_j}(\theta) = \frac{U(\theta + \epsilon e_j) - U(\theta - \epsilon e_j)}{2\epsilon}$$

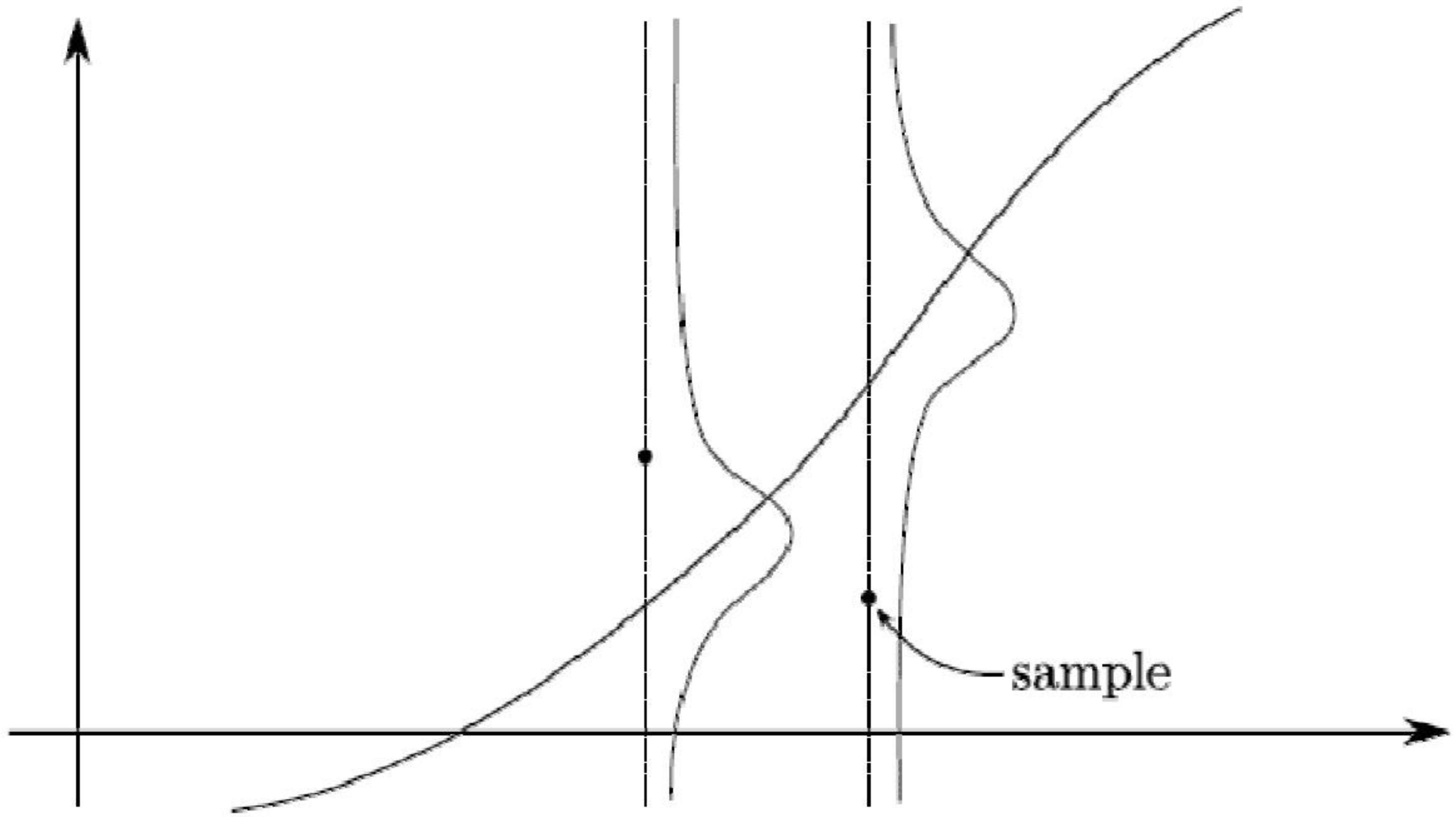
Where:

$$e_j = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{matrix} \\ \\ \\ \\ \leftarrow \text{ } j\text{'th entry} \\ \\ \\ \\ \end{matrix}$$

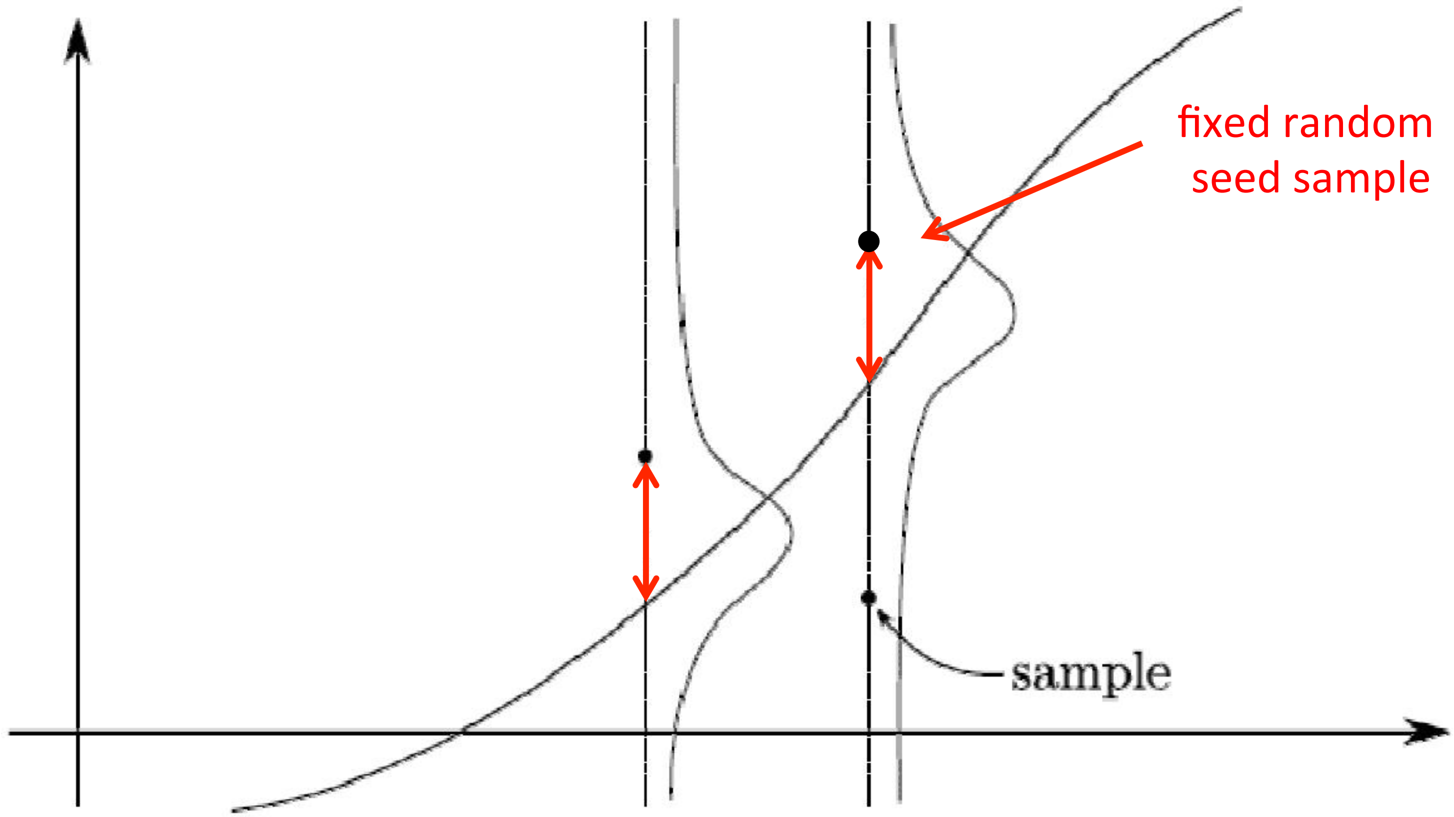
Challenge: Noise Can Dominate



Solution 1: Average over many samples



Solution 2: Fix random seed



Solution 2: Fix random seed

- Randomness in policy and dynamics
  - But can often only control randomness in policy..
- Example: wind influence on a helicopter is stochastic, but if we assume the same wind pattern across trials, this will make the different choices of θ more readily comparable
- *Note: equally applicable to evolutionary methods*

[Ng & Jordan, 2000] provide theoretical analysis of gains from fixing randomness ("pegasus")

[Policy search was done in simulation] [Ng + al, ISER 2004]

Learning to Hover

x, y, z: x points forward along the helicopter, y sideways to the right, z downward.

n_x, n_y, n_z: rotation vector that brings helicopter back to "level" position (expressed in the helicopter frame).

$$u_{collective} = \theta_1 \cdot f_1(z^* - z) + \theta_2 \cdot \dot{z}$$

$$u_{elevator} = \theta_3 \cdot f_2(x^* - x) + \theta_4 f_4(\dot{x}) + \theta_5 \cdot q + \theta_6 \cdot n_y$$

$$u_{aileron} = \theta_7 \cdot f_3(y^* - y) + \theta_8 f_5(\dot{y}) + \theta_9 \cdot p + \theta_{10} \cdot n_x$$

$$u_{rudder} = \theta_{11} \cdot r + \theta_{12} \cdot n_z$$

Gradient-Free Methods

$$\max_{\theta} \; U(\theta) = \max_{\theta} \; \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

- Cross-Entropy Method (CEM)
- Covariance Matrix Adaptation (CMA)

Cross-Entropy Method

$$\max_\theta U(\theta) = \max_\theta \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

- Views U as a black box
- Ignores all other information other than U collected during episode

= evolutionary algorithm

population: $P_{\mu^{(i)}}(\theta)$

<u>CEM:</u>

for iter i = 1, 2, …
  for population member e = 1, 2, …
    sample $\theta^{(e)} \sim P_{\mu^{(i)}}(\theta)$
    execute roll-outs under $\pi_{\theta^{(e)}}$
    store $(\theta^{(e)}, U(e))$
  endfor
  $\mu^{(i+1)} = \arg\max_\mu \sum_{\bar{e}} \log P_\mu(\theta^{(\bar{e})})$
    where $\bar{e}$ indexes over top p %
endfor

Cross-Entropy Method

- Can work embarrassingly well

Method	Mean Score	Reference
Nonreinforcement learning		
Hand-coded	631,167	Dellacherie (Fahey, 2003)
Genetic algorithm	586,103	(Böhm et al., 2004)
Reinforcement learning		
Relational reinforcement learning+kernel-based regression	≈50	Ramon and Driessens (2004)
Policy iteration	3183	Bertsekas and Tsitsiklis (1996)
Least squares policy iteration	<3000	Lagoudakis, Parr, and Littman (2002)
Linear programming + Bootstrap	4274	Farias and van Roy (2006)
Natural policy gradient	≈6800	Kakade (2001)
CE+RL	21,252	
CE+RL, constant noise	72,705	
CE+RL, decreasing noise	348,895	

István Szita and András Lörincz. "Learning Tetris using the noisy cross-entropy method". In: *Neural computation* 18.12 (2006), pp. 2936–2941

Approximate Dynamic Programming Finally Performs Well in the Game of Tetris

[NIPS 2013]

Victor Gabillon
INRIA Lille - Nord Europe,
Team SequeL, FRANCE
victor.gabillon@inria.fr

Mohammad Ghavamzadeh*
INRIA Lille - Team SequeL
& Adobe Research
mohammad.ghavamzadeh@inria.fr

Bruno Scherrer
INRIA Nancy - Grand Est,
Team Maia, FRANCE
bruno.scherrer@inria.fr

Closely Related Approaches

<u>CEM:</u>

for iter i = 1, 2, …
  for population member e = 1, 2, …
    sample $\theta^{(e)} \sim P_{\mu^{(i)}}(\theta)$
    execute roll-outs under $\pi_{\theta^{(e)}}$
    store $(\theta^{(e)}, U(e))$
  endfor
  $$\boxed{\mu^{(i+1)} = \arg\max_{\mu} \sum_{\bar{e}} \log P_{\mu}(\theta^{(\bar{e})}) \quad \text{where } \bar{e} \text{ indexes over top p \%}}$$
endfor

- Reward Weighted Regression (RWR)
  - Dayan & Hinton, NC 1997; Peters & Schaal, ICML 2007

$$\boxed{\mu^{(i+1)} = \arg\max_{\mu} \sum_{e} q(U(e), P_{\mu}(\theta^{(e)})) \log P_{\mu}(\theta^{(e)})}$$

- Policy Improvement with Path Integrals (PI^2)
  - PI2: Theodorou, Buchli, Schaal JMLR2010; Kappen, 2007; (PI2-CMA: Stulp & Sigaud ICML2012)

$$\boxed{\mu^{(i+1)} = \arg\max_{\mu} \sum_{e} \exp(\lambda U(e)) \log P_{\mu}(\theta^{(e)})}$$

- Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES)
  - CMA: Hansen & Ostermeier 1996; (CMA-ES: Hansen, Muller, Koumoutsakos 2003)

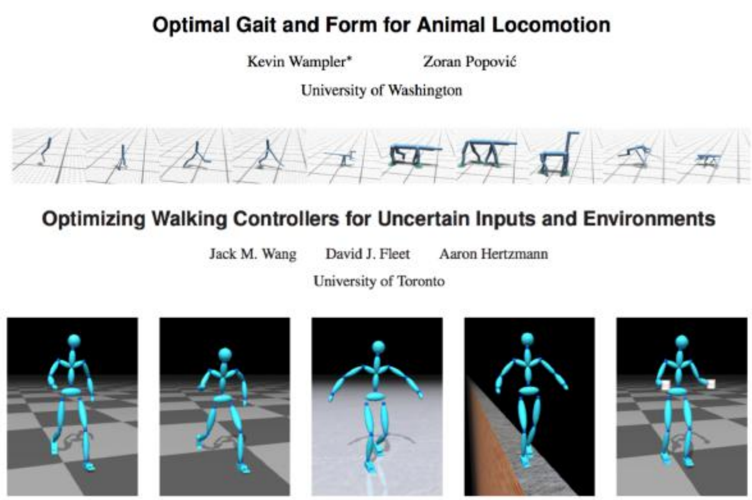
$$\boxed{(\mu^{(i+1)}, \Sigma^{(i+1)}) = \arg\max_{\mu,\Sigma} \sum_{\bar{e}} w(U(\bar{e})) \log \mathcal{N}(\theta^{(\bar{e})}; \mu, \Sigma)}$$

- PoWER
  - Kober & Peters, NIPS 2007 (also applies importance sampling for sample re-use)

$$\boxed{\mu^{(i+1)} = \mu^{(i)} + \left(\sum_{e} (\theta^{(e)} - \mu^{(i)}) U(e)\right) \Big/ \left(\sum_{e} U(e)\right)}$$

Applications

Covariance Matrix Adaptation (CMA) has become standard in graphics [Hansen, Ostermeier, 1996]



PoWER [Kober&Peters, MLJ 2011]

Cross-Entropy / Evolutionary Methods

- Full episode evaluation, parameter perturbation
- Simple
- Main caveat: best when intrinsic dimensionality not too high
  - i.e., number of population members comparable to or larger than number of (effective) parameters

  → in practice OK if low-dimensional θ and willing to do do many runs

  → Easy-to-implement baseline, great for comparisons!

Considerations

- Pros:
  - Work with arbitrary parametrization, even non-differentiable
  - Embarrassingly easy to parallelize
- Cons:
  - Not very sample efficient since ignores temporal structure

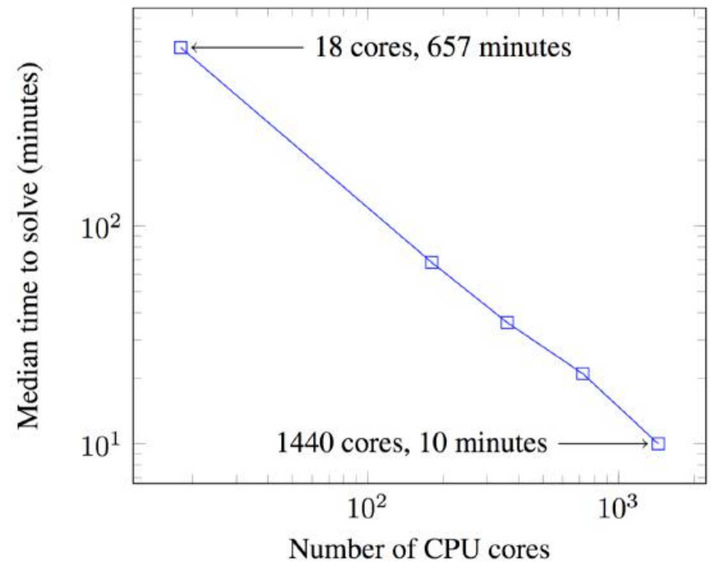


Figure 1. Time to reach a score of 6000 on 3D Humanoid with different number of CPU cores. Experiments are repeated 7 times and median time is reported.

[Salimans, Ho, Chen, Sutskever, 2017]

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - ***Likelihood Ratio (LR) Policy Gradient***
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Likelihood Ratio Policy Gradient

We let τ denote a state-action sequence $s_0, u_0, \ldots, s_H, u_H$. We overload notation: $R(\tau) = \sum_{t=0}^{H} R(s_t, u_t)$.

$$U(\theta) = \mathrm{E}[\sum_{t=0}^{H} R(s_t, u_t); \pi_\theta] = \sum_{\tau} P(\tau; \theta) R(\tau)$$

In our new notation, our goal is to find θ:

$$\max_{\theta} U(\theta) = \max_{\theta} \sum_{\tau} P(\tau; \theta) R(\tau)$$

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\nabla_\theta U(\theta) = \nabla_\theta \sum_{\tau} P(\tau; \theta) R(\tau)$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\begin{aligned}\nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau; \theta) R(\tau) \\ &= \sum_{\tau} \nabla_\theta P(\tau; \theta) R(\tau)\end{aligned}$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau;\theta)R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau;\theta)R(\tau) \\
&= \sum_{\tau} \nabla_\theta P(\tau;\theta)R(\tau) \\
&= \sum_{\tau} \frac{P(\tau;\theta)}{P(\tau;\theta)} \nabla_\theta P(\tau;\theta)R(\tau)
\end{aligned}$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau;\theta)R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau;\theta)R(\tau) \\
&= \sum_{\tau} \nabla_\theta P(\tau;\theta)R(\tau) \\
&= \sum_{\tau} \frac{P(\tau;\theta)}{P(\tau;\theta)} \nabla_\theta P(\tau;\theta)R(\tau) \\
&= \sum_{\tau} P(\tau;\theta) \frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} R(\tau)
\end{aligned}$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau;\theta) R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} \frac{P(\tau;\theta)}{P(\tau;\theta)} \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} P(\tau;\theta) \frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} R(\tau) \\
&= \sum_{\tau} P(\tau;\theta) \nabla_\theta \log P(\tau;\theta) R(\tau)
\end{aligned}$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Likelihood Ratio Policy Gradient

$$U(\theta) = \sum_{\tau} P(\tau;\theta) R(\tau)$$

Taking the gradient w.r.t. θ gives

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta \sum_{\tau} P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} \frac{P(\tau;\theta)}{P(\tau;\theta)} \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_{\tau} P(\tau;\theta) \frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} R(\tau) \\
&= \sum_{\tau} P(\tau;\theta) \nabla_\theta \log P(\tau;\theta) R(\tau)
\end{aligned}$$

Approximate with the empirical estimate for m sample paths under policy π_θ:

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)};\theta) R(\tau^{(i)})$$

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]

Derivation from Importance Sampling

$$U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

[Tang&Abbeel, NIPS 2011]

Derivation from Importance Sampling

$$U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$\nabla_\theta U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

[Tang&Abbeel, NIPS 2011]

Derivation from Importance Sampling

$$U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$\nabla_\theta U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$\nabla_\theta \left. U(\theta) \right|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{\nabla_\theta \left. P(\tau|\theta) \right|_{\theta_{\text{old}}}}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

[Tang&Abbeel, NIPS 2011]

Derivation from Importance Sampling

$$U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$\nabla_\theta U(\theta) = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$\nabla_\theta \; U(\theta)|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\frac{\nabla_\theta \; P(\tau|\theta)|_{\theta_{\text{old}}}}{P(\tau|\theta_{\text{old}})} R(\tau) \right]$$

$$= \mathbb{E}_{\tau \sim \theta_{\text{old}}} \left[\nabla_\theta \log P(\tau|\theta)|_{\theta_{\text{old}}} \; R(\tau) \right]$$

[Tang&Abbeel, NIPS 2011]

Derivation from Importance Sampling

$$U(\theta) = \mathbb{E}_{\tau\sim\theta_{\text{old}}}\left[\frac{P(\tau|\theta)}{P(\tau|\theta_{\text{old}})}R(\tau)\right]$$

$$\nabla_\theta U(\theta) = \mathbb{E}_{\tau\sim\theta_{\text{old}}}\left[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{\text{old}})}R(\tau)\right]$$

$$\nabla_\theta\; U(\theta)|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{\tau\sim\theta_{\text{old}}}\left[\frac{\nabla_\theta\; P(\tau|\theta)|_{\theta_{\text{old}}}}{P(\tau|\theta_{\text{old}})}R(\tau)\right]$$

$$= \mathbb{E}_{\tau\sim\theta_{\text{old}}}\left[\nabla_\theta \log P(\tau|\theta)|_{\theta_{\text{old}}}\; R(\tau)\right]$$

Suggests we can also look at more than just gradient!
E.g., can use importance sampled objective as "surrogate loss" (locally)

[Tang&Abbeel, NIPS 2011]

Likelihood Ratio Gradient: Validity

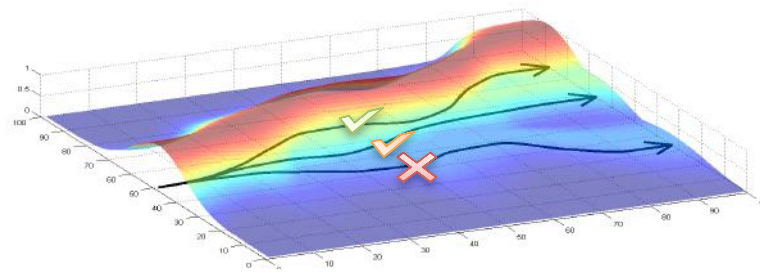
$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

- Valid even if R is discontinuous, and unknown, or sample space (of paths) is a discrete set

Likelihood Ratio Gradient: Intuition

$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

- Gradient tries to:
  - Increase probability of paths with positive R
  - Decrease probability of paths with negative R



! Likelihood ratio changes probabilities of experienced paths, does not try to change the paths (<-> Path Derivative)

Let's Decompose Path into States and Actions

$$\nabla_\theta \log P(\tau^{(i)};\theta) = \nabla_\theta \log \left[\prod_{t=0}^{H} \underbrace{P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)})}_{\text{dynamics model}} \cdot \underbrace{\pi_\theta(u_t^{(i)}|s_t^{(i)})}_{\text{policy}} \right]$$

Let's Decompose Path into States and Actions

$$\nabla_\theta \log P(\tau^{(i)};\theta) = \nabla_\theta \log \left[\prod_{t=0}^{H} \underbrace{P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)})}_{\text{dynamics model}} \cdot \underbrace{\pi_\theta(u_t^{(i)}|s_t^{(i)})}_{\text{policy}} \right]$$

$$= \nabla_\theta \left[\sum_{t=0}^{H} \log P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)}) + \sum_{t=0}^{H} \log \pi_\theta(u_t^{(i)}|s_t^{(i)}) \right]$$

Let's Decompose Path into States and Actions

$$
\begin{aligned}
\nabla_\theta \log P(\tau^{(i)};\theta) &= \nabla_\theta \log \left[\prod_{t=0}^{H} \underbrace{P(s_{t+1}^{(i)} | s_t^{(i)}, u_t^{(i)})}_{\text{dynamics model}} \cdot \underbrace{\pi_\theta(u_t^{(i)} | s_t^{(i)})}_{\text{policy}} \right] \\
&= \nabla_\theta \left[\sum_{t=0}^{H} \log P(s_{t+1}^{(i)} | s_t^{(i)}, u_t^{(i)}) + \sum_{t=0}^{H} \log \pi_\theta(u_t^{(i)} | s_t^{(i)}) \right] \\
&= \nabla_\theta \sum_{t=0}^{H} \log \pi_\theta(u_t^{(i)} | s_t^{(i)})
\end{aligned}
$$

Let's Decompose Path into States and Actions

$$\nabla_\theta \log P(\tau^{(i)};\theta) = \nabla_\theta \log \left[\prod_{t=0}^{H} \underbrace{P(s_{t+1}^{(i)}|s_t^{(i)},u_t^{(i)})}_{\text{dynamics model}} \cdot \underbrace{\pi_\theta(u_t^{(i)}|s_t^{(i)})}_{\text{policy}} \right]$$

$$= \nabla_\theta \left[\sum_{t=0}^{H} \log P(s_{t+1}^{(i)}|s_t^{(i)},u_t^{(i)}) + \sum_{t=0}^{H} \log \pi_\theta(u_t^{(i)}|s_t^{(i)}) \right]$$

$$= \nabla_\theta \sum_{t=0}^{H} \log \pi_\theta(u_t^{(i)}|s_t^{(i)})$$

$$= \sum_{t=0}^{H} \underbrace{\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})}_{\text{no dynamics model required!!}}$$

Likelihood Ratio Gradient Estimate

The following expression provides us with an unbiased estimate of the gradient, and we can compute it without access to a dynamics model:

$$\hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

Here:

$$\nabla_\theta \log P(\tau^{(i)}; \theta) = \sum_{t=0}^{H} \underbrace{\nabla_\theta \log \pi_\theta(u_t^{(i)} | s_t^{(i)})}_{\text{no dynamics model required!!}}$$

Unbiased means:

$$\mathrm{E}[\hat{g}] = \nabla_\theta U(\theta)$$

Likelihood Ratio Gradient Estimate

- As formulated thus far: unbiased but very noisy
- Fixes that lead to real-world practicality
  - Baseline
  - Temporal structure
  - Also: KL-divergence trust region / natural gradient (= general trick, equally applicable to perturbation analysis and finite differences)

Likelihood Ratio Gradient: Baseline

$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

- To build intuition, let's assume R > 0
  - Then tries to increase probabilities of all paths

→ Consider baseline b:

$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta)(R(\tau^{(i)}) - b)$$

still unbiased

[Williams 1992]

$$\begin{aligned}
&\mathbb{E}\left[\nabla_\theta \log P(\tau;\theta) b\right] \\
=& \sum_\tau P(\tau;\theta) \nabla_\theta \log P(\tau;\theta) b \\
=& \sum_\tau P(\tau;\theta) \frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} b \\
=& \sum_\tau \nabla_\theta P(\tau;\theta) b \\
=& \nabla_\theta \left(\sum_\tau P(\tau) b \right) \\
=& \nabla_\theta \left(b \right) \\
=& 0
\end{aligned}$$

Good choices for b?

$$b = \mathbb{E}\left[R(\tau)\right] \approx \frac{1}{m} \sum_{i=1}^{m} R(\tau^{(i)})$$

$$b = \frac{\sum_i \left(\nabla_\theta \log P(\tau^{(i)};\theta)\right)^2 R(\tau^{(i)})}{\sum_i \left(\nabla_\theta \log P(\tau^{(i)};\theta)\right)^2}$$

[See: Greensmith, Bartlett, Baxter, JMLR 2004 for variance reduction techniques.]

Likelihood Ratio and Temporal Structure

- Current estimate:

$$\hat{g} = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)};\theta)(R(\tau^{(i)}) - b)$$

$$= \frac{1}{m}\sum_{i=1}^{m} \left(\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\right)\left(\sum_{t=0}^{H-1} R(s_t^{(i)}, u_t^{(i)}) - b\right)$$

- Future actions do not depend on past rewards, hence can lower variance by instead using:

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)}) \left(\sum_{k=t}^{H-1} R(s_k^{(i)}, u_k^{(i)}) - b(s_k^{(i)})\right)$$

- Good choice for b?

Expected return: $b(s_t) = \mathbb{E}\left[r_t + r_{t+1} + r_{t+2} + \ldots + r_{H-1}\right]$

→ Increase logprob of action proportionally to how much its returns are better than the expected return under the current policy

[Policy Gradient Theorem: Sutton et al, NIPS 1999; GPOMDP: Bartlett & Baxter, JAIR 2001; Survey: Peters & Schaal, IROS 2006]

Pseudo-code Reinforce aka Vanilla Policy Gradient

Algorithm 1 "Vanilla" policy gradient algorithm

Initialize policy parameter θ, 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 $\nabla_\theta \log \pi(a_t \mid s_t, \theta)\hat{A}_t$
end for

~ [Williams, 1992]

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - ***Step-sizing / Natural Gradient / Trust Regions (TRPO)***
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Step-sizing and Trust Regions

- Step-sizing necessary as gradient is only first-order approximation

What's in a step-size?

- Terrible step sizes, always an issue, but how about just not so great ones?

- Supervised learning
  - Step too far → next update will correct for it

- Reinforcement learning
  - Step too far → terrible policy
  - Next mini-batch: collected under this terrible policy!
  - Not clear how to recover short of going back and shrinking the step size

Step-sizing and Trust Regions

- Simple step-sizing: Line search in direction of gradient
  - Simple, but expensive (evaluations along the line)
  - Naïve: ignores where the first-order approximation is good/poor

Step-sizing and Trust Regions

- Advanced step-sizing: Trust regions
- First-order approximation from gradient is a good approximation within "trust region"

→ Solve for best point within trust region:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

Evaluating the KL

- Our problem:
$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:
$$P(\tau;\theta) = P(s_0)\prod_{t=0}^{H-1}\pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:

$$P(\tau;\theta) = P(s_0)\prod_{t=0}^{H-1} \pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)$$

- Hence:

$$KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) = \sum_\tau P(\tau;\theta)\log\frac{P(\tau;\theta)}{P(\tau;\theta+\delta\theta)}$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:

$$P(\tau;\theta) = P(s_0) \prod_{t=0}^{H-1} \pi_\theta(u_t|s_t) P(s_{t+1}|s_t,u_t)$$

- Hence:

$$KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) = \sum_\tau P(\tau;\theta) \log \frac{P(\tau;\theta)}{P(\tau;\theta+\delta\theta)}$$
$$= \sum_\tau P(\tau;\theta) \log \frac{P(s_0) \prod_{t=0}^{H-1} \pi_\theta(u_t|s_t) P(s_{t+1}|s_t,u_t)}{P(s_0) \prod_{t=0}^{H-1} \pi_{\theta+\delta\theta}(u_t|s_t) P(s_{t+1}|s_t,u_t)}$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:

$$P(\tau;\theta) = P(s_0)\prod_{t=0}^{H-1} \pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)$$

- Hence:

$$KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) = \sum_\tau P(\tau;\theta)\log\frac{P(\tau;\theta)}{P(\tau;\theta+\delta\theta)}$$
$$= \sum_\tau P(\tau;\theta)\log\frac{P(s_0)\prod_{t=0}^{H-1}\pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)}{P(s_0)\prod_{t=0}^{H-1}\pi_{\theta+\delta\theta}(u_t|s_t)P(s_{t+1}|s_t,u_t)}$$

dynamics cancels out! ☺

$$= \sum_\tau P(\tau;\theta)\log\frac{\prod_{t=0}^{H-1}\pi_\theta(u_t|s_t)}{\prod_{t=0}^{H-1}\pi_{\theta+\delta\theta}(u_t|s_t)}$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$

$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:

$$P(\tau;\theta) = P(s_0) \prod_{t=0}^{H-1} \pi_\theta(u_t|s_t) P(s_{t+1}|s_t, u_t)$$

- Hence:

$$KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) = \sum_\tau P(\tau;\theta) \log \frac{P(\tau;\theta)}{P(\tau;\theta+\delta\theta)}$$

$$= \sum_\tau P(\tau;\theta) \log \frac{P(s_0)\prod_{t=0}^{H-1} \pi_\theta(u_t|s_t) P(s_{t+1}|s_t,u_t)}{P(s_0)\prod_{t=0}^{H-1} \pi_{\theta+\delta\theta}(u_t|s_t) P(s_{t+1}|s_t,u_t)}$$

dynamics cancels out! ☺

$$= \sum_\tau P(\tau;\theta) \log \frac{\prod_{t=0}^{H-1} \pi_\theta(u_t|s_t)}{\prod_{t=0}^{H-1} \pi_{\theta+\delta\theta}(u_t|s_t)}$$

$$\approx \frac{1}{M} \sum_{s \text{ in roll-outs under } \theta} \sum_u \pi_\theta(u|s) \log \frac{\pi_\theta(u|s)}{\pi_{\theta+\delta\theta}(u|s)}$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Recall:

$$P(\tau;\theta) = P(s_0)\prod_{t=0}^{H-1} \pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)$$

- Hence:

$$KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) = \sum_\tau P(\tau;\theta)\log\frac{P(\tau;\theta)}{P(\tau;\theta+\delta\theta)}$$
$$= \sum_\tau P(\tau;\theta)\log\frac{P(s_0)\prod_{t=0}^{H-1}\pi_\theta(u_t|s_t)P(s_{t+1}|s_t,u_t)}{P(s_0)\prod_{t=0}^{H-1}\pi_{\theta+\delta\theta}(u_t|s_t)P(s_{t+1}|s_t,u_t)}$$

dynamics cancels out! ☺

$$= \sum_\tau P(\tau;\theta)\log\frac{\prod_{t=0}^{H-1}\pi_\theta(u_t|s_t)}{\prod_{t=0}^{H-1}\pi_{\theta+\delta\theta}(u_t|s_t)}$$
$$\approx \frac{1}{M}\sum_{s \text{ in roll-outs under } \theta}\sum_u \pi_\theta(u|s)\log\frac{\pi_\theta(u|s)}{\pi_{\theta+\delta\theta}(u|s)}$$
$$\approx \frac{1}{M}\sum_{s \text{ in roll-outs under } \theta} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s))$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Has become:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \frac{1}{M}\sum_{s\sim\pi_\theta} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s)) \leq \varepsilon$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Has become:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \frac{1}{M}\sum_{s\sim\pi_\theta} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s)) \leq \varepsilon$$

- 2nd order approximation to KL:

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Has become:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \frac{1}{M}\sum_{s\sim\pi_\theta} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s)) \leq \varepsilon$$

- 2nd order approximation to KL:

$$KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s) \approx \delta\theta^\top \left(\sum_{(s,u)\sim\theta} \nabla_\theta \log \pi_\theta(u|s) \nabla_\theta \log \pi_\theta(u|s)^\top \right) \delta\theta$$

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Has become:

$$\max_{\delta\theta} \ \hat{g}^\top \delta\theta$$
$$\text{s.t.} \ \ \frac{1}{M}\sum_{(s,u)\sim\theta} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s) \leq \varepsilon$$

- 2nd order approximation to KL:

$$\begin{aligned} KL(\pi_\theta(u|s)||\pi_{\theta+\delta\theta}(u|s) &\approx \delta\theta^\top \left(\sum_{(s,u)\sim\theta} \nabla_\theta \log \pi_\theta(u|s) \nabla_\theta \log \pi_\theta(u|s)^\top \right) \delta\theta \\ &= \delta\theta^\top F_\theta \delta\theta \end{aligned}$$

→ Fisher matrix F_θ easily computed from gradient calculations

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- If constraint moved to objective → natural policy gradient
  - [Kakade 2002, Bagnell & Schneider 2003, Peters & Schaal 2003]

- But keeping as constraint tends to be beneficial [Schulman et al 2015]
  - Can be done through dual gradient descent on Lagrangian

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$

$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

Evaluating the KL

- Our problem:
$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- Done?

Evaluating the KL

- Our problem:
$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- Done?
  - Deep RL → θ high-dimensional, and building / inverting F_θ impractical

Evaluating the KL

- Our problem: $$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$ $$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- Done?
  - Deep RL → θ high-dimensional, and building / inverting F_θ impractical
    - Efficient scheme through conjugate gradient [Schulman et al, 2015, TRPO]

Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- Done?
  - Deep RL → θ high-dimensional, and building / inverting F_θ impractical
    - Efficient scheme through conjugate gradient [Schulman et al, 2015, TRPO]
  - Can we do even better?

Evaluating the KL

- Our problem: $$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$ $$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

- Done?
  - Deep RL → θ high-dimensional, and building / inverting F_θ impractical
    - Efficient scheme through conjugate gradient [Schulman et al, 2015, TRPO]
  - Can we do even better?
    - Replace objective by surrogate loss that's higher order approximation yet equally efficient to evaluate [Schulman et al, 2015, TRPO]

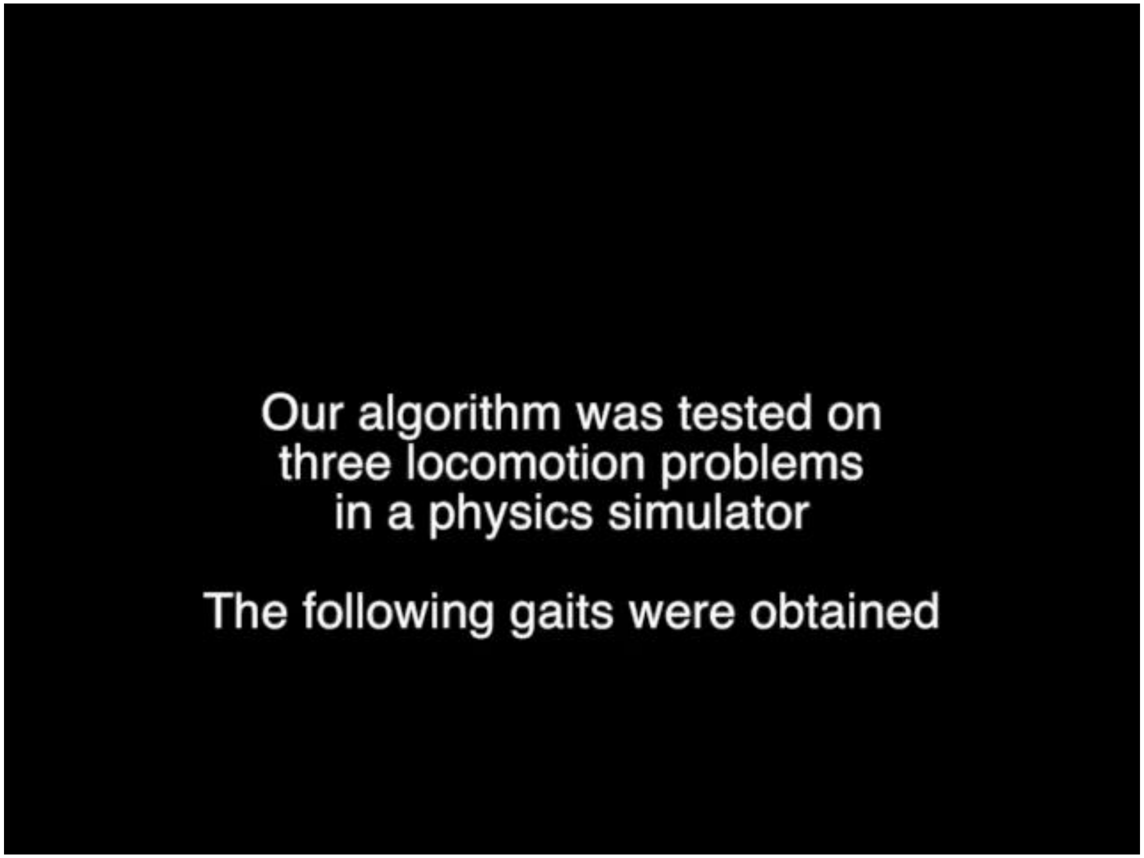
Evaluating the KL

- Our problem:

$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$
$$\text{s.t.} \quad \delta\theta^\top F_\theta \delta\theta \leq \varepsilon$$

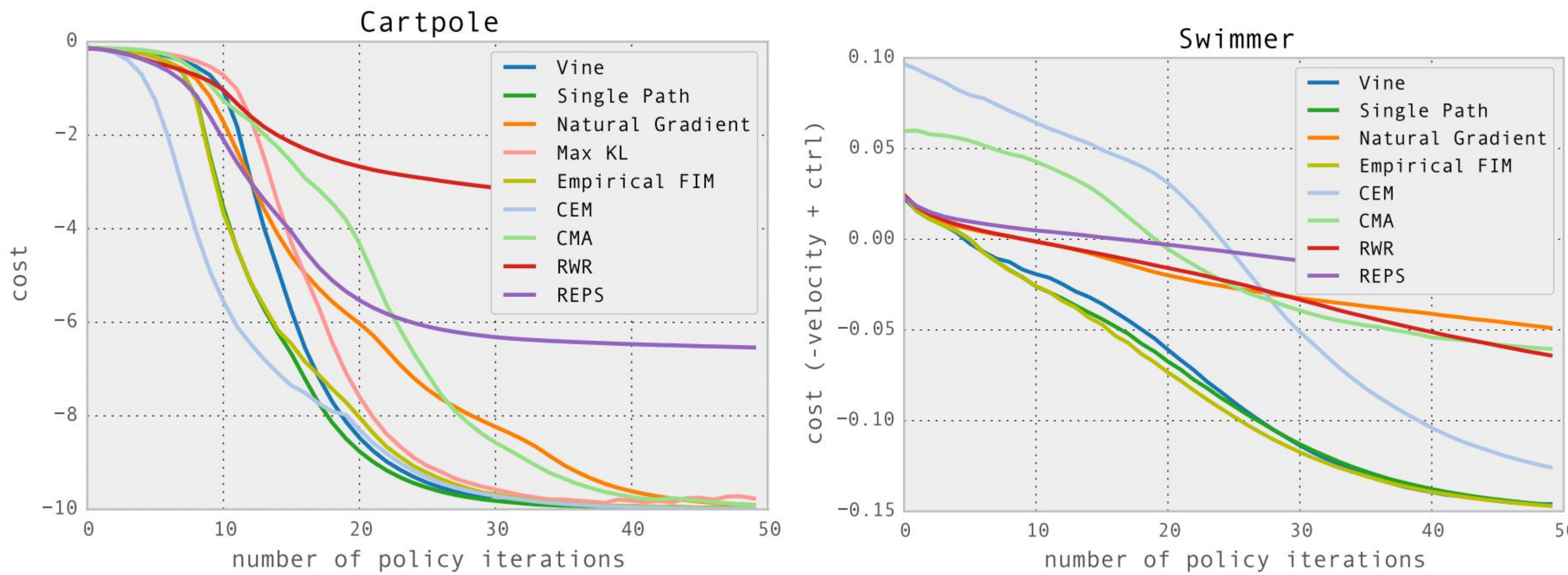
- Done?
  - Deep RL → θ high-dimensional, and building / inverting F_θ impractical
    - Efficient scheme through conjugate gradient [Schulman et al, 2015, TRPO]
  - Can we do even better?
    - Replace objective by surrogate loss that's higher order approximation yet equally efficient to evaluate [Schulman et al, 2015, TRPO]
    - Note: the surrogate loss idea is generally applicable when likelihood ratio gradients are used

Experiments in Locomotion

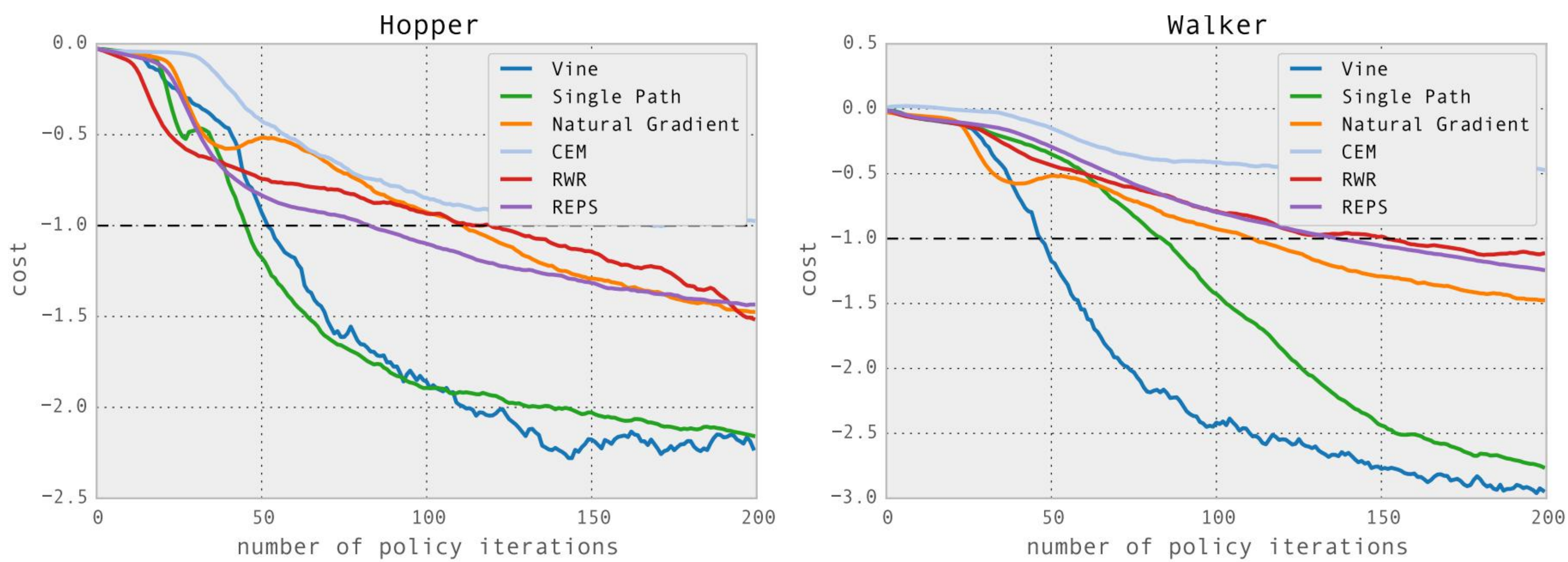


[Schulman, Levine, Moritz, Jordan, Abbeel, 2014]

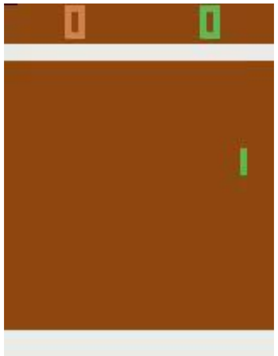
Learning Curves -- Comparison



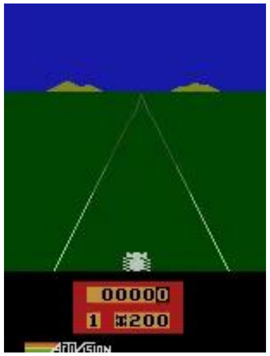
Learning Curves -- Comparison



Atari Games

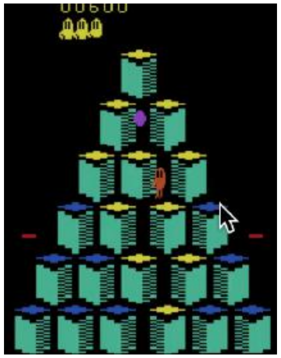


Pong



Enduro

Beamrider



Q*bert

- Deep Q-Network (DQN) [Mnih et al, 2013/2015]
- Dagger with Monte Carlo Tree Search [Xiao-Xiao et al, 2014]
- Trust Region Policy Optimization [Schulman, Levine, Moritz, Jordan, Abbeel, 2015]
- …

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - ***Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)***

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

<u>Assumes:</u>
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

<u>Assumes:</u>
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\sum_{k=t}^{H-1} R(s_k^{(i)}, u_k^{(i)}) - \underbrace{V^\pi(s_k^{(i)})}_{\text{How to estimate?}}\right)$$

Estimation of V^π

- Bellman Equation for V^π

$$V^\pi(s) = \sum_u \pi(u|s) \sum_{s'} P(s'|s,u)[R(s,u,s') + \gamma V^\pi(s')]$$

- Fitted V iteration:
  - Init $V^\pi_{\phi_0}$
  - Collect data {s, u, s', r}
  - $\phi_{i+1} \leftarrow \min_\phi \sum_{(s,u,s',r)} \|r + V^\pi_{\phi_i}(s') - V_\phi(s)\|_2^2 + \lambda \|\phi - \phi_i\|_2^2$

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\sum_{k=t}^{H-1}R(s_k^{(i)},u_k^{(i)})-V^\pi(s_k^{(i)})\right)$$

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\underbrace{\sum_{k=t}^{H-1} R(s_k^{(i)},u_k^{(i)})} - V^\pi(s_k^{(i)})\right)$$

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\underbrace{\sum_{k=t}^{H-1} R(s_k^{(i)},u_k^{(i)})} - V^\pi(s_k^{(i)})\right)$$

- Estimation of Q from *single* roll-out

$$Q^\pi(s,u) = \mathbb{E}[r_0 + r_1 + r_2 + \cdots | s_0 = s, a_0 = a]$$

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\underbrace{\sum_{k=t}^{H-1} R(s_k^{(i)}, u_k^{(i)})} - V^\pi(s_k^{(i)})\right)$$

- Estimation of Q from *single* roll-out

$$Q^\pi(s,u) = \mathbb{E}[r_0 + r_1 + r_2 + \cdots | s_0 = s, a_0 = a]$$

- = high variance per sample based / no generalization

Further Refinements

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\underbrace{\sum_{k=t}^{H-1}R(s_k^{(i)},u_k^{(i)})} - V^\pi(s_k^{(i)})\right)$$

- Estimation of Q from *single* roll-out

$$Q^\pi(s,u) = \mathbb{E}[r_0 + r_1 + r_2 + \cdots | s_0 = s, a_0 = a]$$

- = high variance per sample based / no generalization
  - Reduce variance by discounting

Recall Our Likelihood Ratio PG Estimator

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(u_t^{(i)}|s_t^{(i)})\left(\underbrace{\sum_{k=t}^{H-1} R(s_k^{(i)}, u_k^{(i)})} - V^\pi(s_k^{(i)})\right)$$

- Estimation of Q from *single* roll-out

$$Q^\pi(s,u) = \mathbb{E}[r_0 + r_1 + r_2 + \cdots | s_0 = s, a_0 = a]$$

- = high variance per sample based / no generalization
  - Reduce variance by discounting
  - Reduce variance by function approximation (=critic)

Variance Reduction by Discounting

$$Q^\pi(s,u) = \mathbb{E}[r_0 + r_1 + r_2 + \cdots | s_0 = s, a_0 = a]$$

→ introduce discount factor as a hyperparameter to improve estimate of Q:

$$Q^{\pi,\gamma}(s,u) = \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots | s_0 = s, a_0 = a]$$

Reducing Variance by Function Approximation

$$Q^{\pi,\gamma}(s,u) = \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u]$$

Reducing Variance by Function Approximation

$$
\begin{aligned}
Q^{\pi,\gamma}(s,u) &= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma V^{\pi}(s_1) \mid s_0 = s, u_0 = u]
\end{aligned}
$$

Reducing Variance by Function Approximation

$$
\begin{aligned}
Q^{\pi,\gamma}(s,u) &= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma V^{\pi}(s_1) \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 V^{\pi}(s_2) \mid s_0 = s, u_0 = u]
\end{aligned}
$$

Reducing Variance by Function Approximation

$$
\begin{aligned}
Q^{\pi,\gamma}(s,u) &= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma V^{\pi}(s_1) \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 V^{\pi}(s_2) \mid s_0 = s, u_0 = u] \\
&= \mathbb{E}[r_0 + \gamma r_1 + +\gamma^2 r_2 + \gamma^3 V^{\pi}(s_3) \mid s_0 = s, u_0 = u] \\
&= \cdots
\end{aligned}
$$

- ***Async Advantage Actor Critic (A3C)*** uses one choice of k > 1

Reducing Variance by Function Approximation

$$
\begin{aligned}
Q^{\pi,\gamma}(s,u) &= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)} \\
&= \mathbb{E}[r_0 + \gamma V^\pi(s_1) \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)\lambda} \\
&= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 V^\pi(s_2) \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)\lambda^2} \\
&= \mathbb{E}[r_0 + \gamma r_1 + +\gamma^2 r_2 + \gamma^3 V^\pi(s_3) \mid s_0 = s, u_0 = u] \\
&= \cdots && {\color{red}(1-\lambda)\lambda^3}
\end{aligned}
$$

- ***Generalized Advantage Estimation (GAE)*** uses $\hat{Q}^{GAE}(s,u)$
  - = lambda exponentially weighted average of all the above [Schulman et al, ICLR 2016]

Reducing Variance by Function Approximation

$$
\begin{aligned}
Q^{\pi,\gamma}(s,u) &= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)} \\
&= \mathbb{E}[r_0 + \gamma V^\pi(s_1) \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)\lambda} \\
&= \mathbb{E}[r_0 + \gamma r_1 + \gamma^2 V^\pi(s_2) \mid s_0 = s, u_0 = u] && {\color{red}(1-\lambda)\lambda^2} \\
&= \mathbb{E}[r_0 + \gamma r_1 + +\gamma^2 r_2 + \gamma^3 V^\pi(s_3) \mid s_0 = s, u_0 = u] \\
&= \cdots && {\color{red}(1-\lambda)\lambda^3}
\end{aligned}
$$

- ***Generalized Advantage Estimation (GAE)*** uses $\hat{Q}^{GAE}(s,u)$
  - = lambda exponentially weighted average of all the above [Schulman et al, ICLR 2016]
- ~ TD(lambda) / eligibility traces [Sutton and Barto, 1990]

Actor-Critic with GAE

- Policy Gradient + Generalized Advantage Estimation:
  - Init π_{θ_0} $V^{\pi}_{\phi_0}$
  - Collect roll-outs {s, u, s', r} and $\hat{Q}_i^{GAE}(s,u)$
  - Update: $\phi_{i+1} \leftarrow \min_{\phi} \sum_{(s,u,s',r)} \|\hat{Q}_i^{\text{GAE}}(s,u) - V_{\phi}(s)\|_2^2 + \kappa\|\phi - \phi_i\|_2^2$

$$\theta_{i+1} \leftarrow \theta_i + \alpha \; \frac{1}{m}\sum_{k=1}^{m}\sum_{t=0}^{H-1} \nabla_\theta \log \pi_{\theta_i}(u_t^{(k)}|s_t^{(k)}) \left(\hat{Q}_i^{\text{GAE}}(s_t^{(k)}, u_t^{(k)}) - V^{\pi}_{\phi_i}(s_t^{(k)})\right)$$

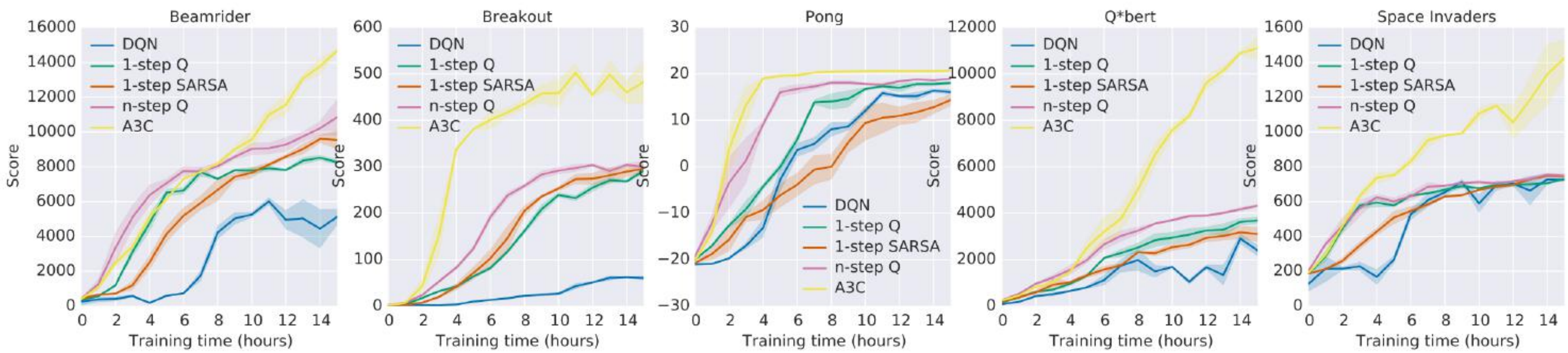
Note: many variations, e.g. could instead use 1-step for V, full roll-out for pi:

$$\phi_{i+1} \leftarrow \min_{\phi} \sum_{(s,u,s',r)} \|r + V^{\pi}_{\phi_i}(s') - V_{\phi}(s)\|_2^2 + \lambda\|\phi - \phi_i\|_2^2$$

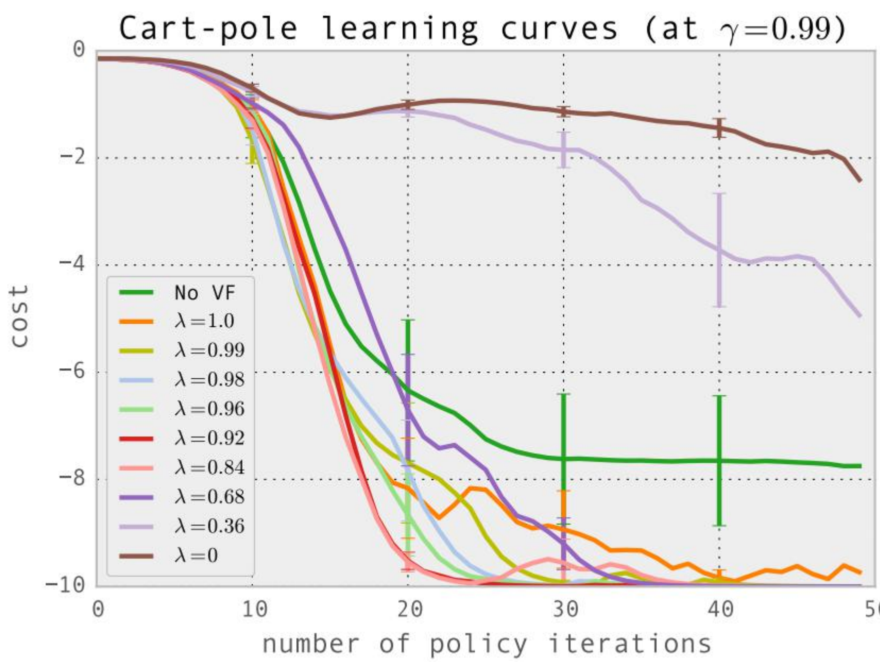
$$\theta_{i+1} \leftarrow \theta_i + \alpha \; \frac{1}{m}\sum_{k=1}^{m}\sum_{t=0}^{H-1} \nabla_\theta \log \pi_{\theta_i}(u_t^{(k)}|s_t^{(k)}) \left(\sum_{t'=t}^{H-1} r_{t'}^{(k)} - V^{\pi}_{\phi_i}(s_{t'}^{(k)})\right)$$

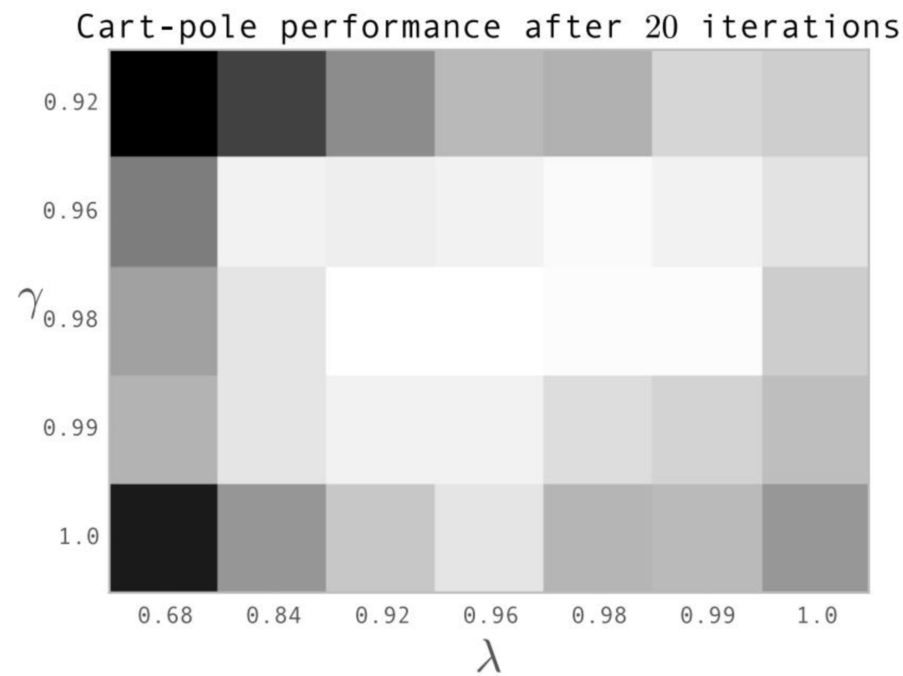
Async Advantage Actor Critic (A3C)

- [Mnih et al, ICML 2016]
  - Likelihood Ratio Policy Gradient
  - n-step Advantage Estimation



Effect of gamma and lambda

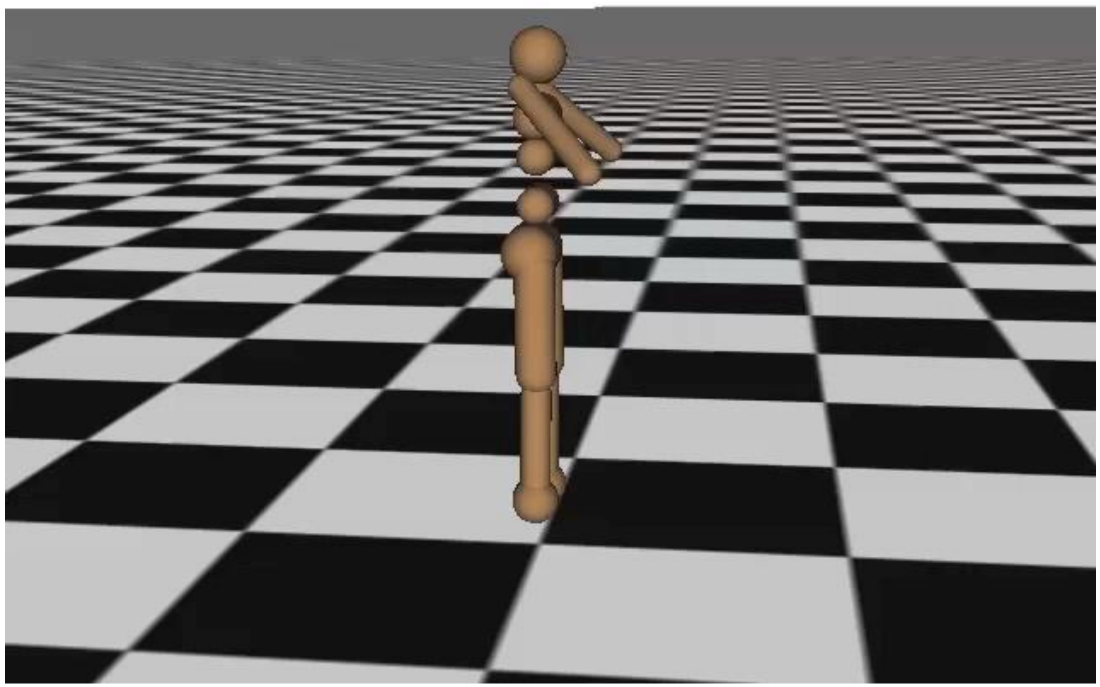




[Schulman et al, 2016 -- GAE]

Learning Locomotion (TRPO + GAE)

Iteration 0



[Schulman, Moritz, Levine, Jordan, Abbeel, 2016]

Outline

- **Model-based**
  - *Pathwise Derivatives (PD) / BackPropagation Through Time (BPTT)*
    - Deterministic dynamics
    - Stochastic dynamics / Reparameterization trick
    - Variance reduction (-> SVG, DDPG)

Assumes:
- f known, differentiable
- R known, differentiable
- π_θ (known), differentiable

- **Model-free**
  - *Parameter Perturbation / Evolutionary Strategies*
  - *Likelihood Ratio (LR) Policy Gradient*
    - Derivation
    - Connection w/Importance Sampling
    - Variance reduction
    - Step-sizing / Natural Gradient / Trust Regions (TRPO)
    - Generalized Advantage Estimation (GAE) / Asynchronous Actor Critic (A3C)

Assumes:
- f -- no assumptions
- R -- no assumptions
- π_θ -- (known), stochastic

- **Stochastic Computation Graphs:** general framework for PD / LR gradients

Stochastic Computation Graphs

	Stochastic Computation Graph	Objective	Gradient Estimator	Surrogate Loss Computation Graph
(1)	$\theta \longrightarrow \boxed{x} \longrightarrow y \longrightarrow \boxed{f}$	$\mathbb{E}_y\left[f(y)\right]$		

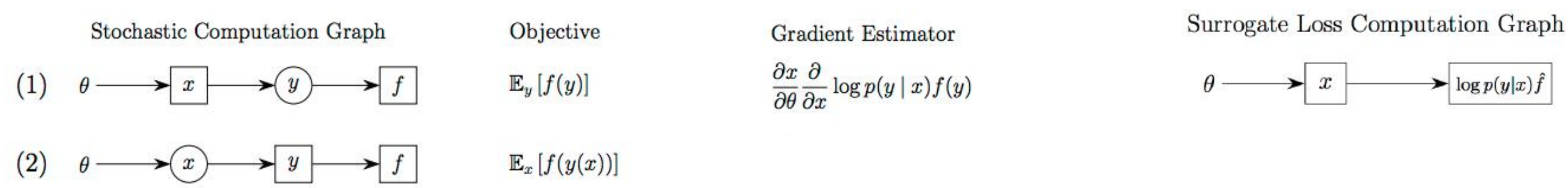
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs

	Stochastic Computation Graph	Objective	Gradient Estimator	Surrogate Loss Computation Graph
(1)	$\theta \longrightarrow \boxed{x} \longrightarrow y \longrightarrow \boxed{f}$	$\mathbb{E}_y [f(y)]$	$\frac{\partial x}{\partial \theta} \frac{\partial}{\partial x} \log p(y \mid x) f(y)$	$\theta \longrightarrow \boxed{x} \longrightarrow \boxed{\log p(y \mid x) \hat{f}}$

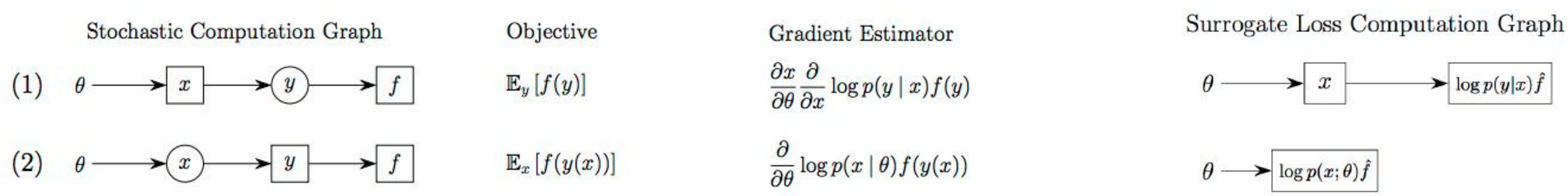
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs



[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs



[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs

	Stochastic Computation Graph	Objective	Gradient Estimator	Surrogate Loss Computation Graph
(1)	$\theta \to \boxed{x} \to y \to \boxed{f}$	$\mathbb{E}_y\left[f(y)\right]$	$\frac{\partial x}{\partial \theta}\frac{\partial}{\partial x}\log p(y \mid x) f(y)$	$\theta \to \boxed{x} \to \boxed{\log p(y \mid x)\hat{f}}$
(2)	$\theta \to x \to \boxed{y} \to \boxed{f}$	$\mathbb{E}_x\left[f(y(x))\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(y(x))$	$\theta \to \boxed{\log p(x;\theta)\hat{f}}$
(3)	$\theta \to x \to y \to \boxed{f}$	$\mathbb{E}_{x,y}\left[f(y)\right]$		

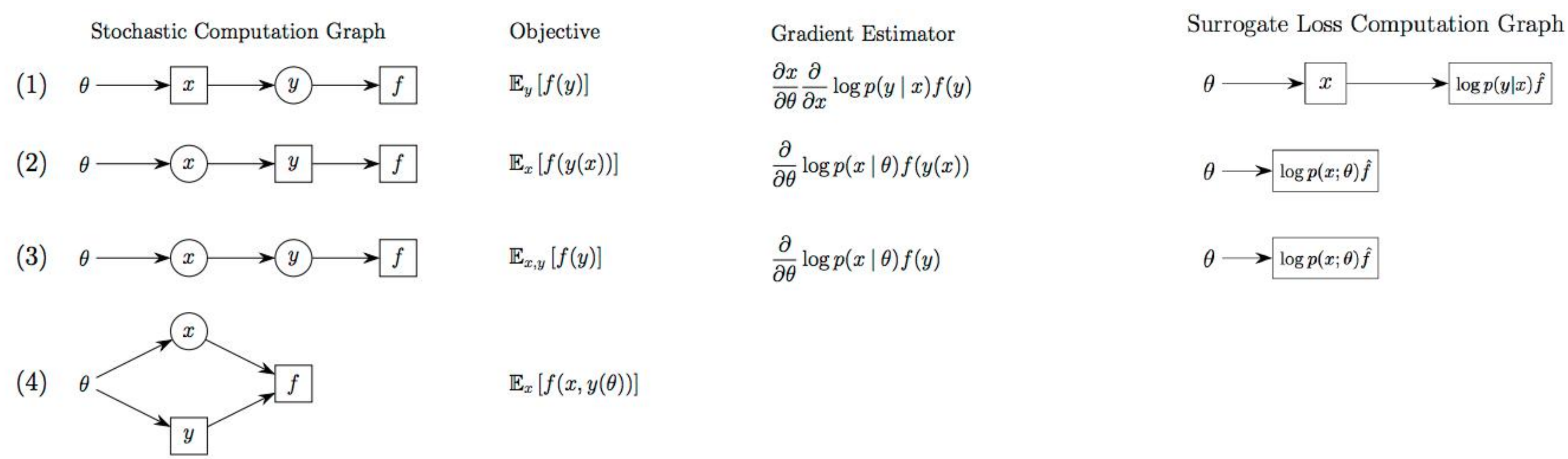
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs

	Stochastic Computation Graph	Objective	Gradient Estimator	Surrogate Loss Computation Graph
(1)	$\theta \longrightarrow \boxed{x} \longrightarrow y \longrightarrow \boxed{f}$	$\mathbb{E}_y\left[f(y)\right]$	$\frac{\partial x}{\partial \theta}\frac{\partial}{\partial x}\log p(y \mid x) f(y)$	$\theta \longrightarrow \boxed{x} \longrightarrow \boxed{\log p(y\|x)\hat{f}}$
(2)	$\theta \longrightarrow x \longrightarrow \boxed{y} \longrightarrow \boxed{f}$	$\mathbb{E}_x\left[f(y(x))\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(y(x))$	$\theta \longrightarrow \boxed{\log p(x;\theta)\hat{f}}$
(3)	$\theta \longrightarrow x \longrightarrow y \longrightarrow \boxed{f}$	$\mathbb{E}_{x,y}\left[f(y)\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(y)$	$\theta \longrightarrow \boxed{\log p(x;\theta)\hat{f}}$

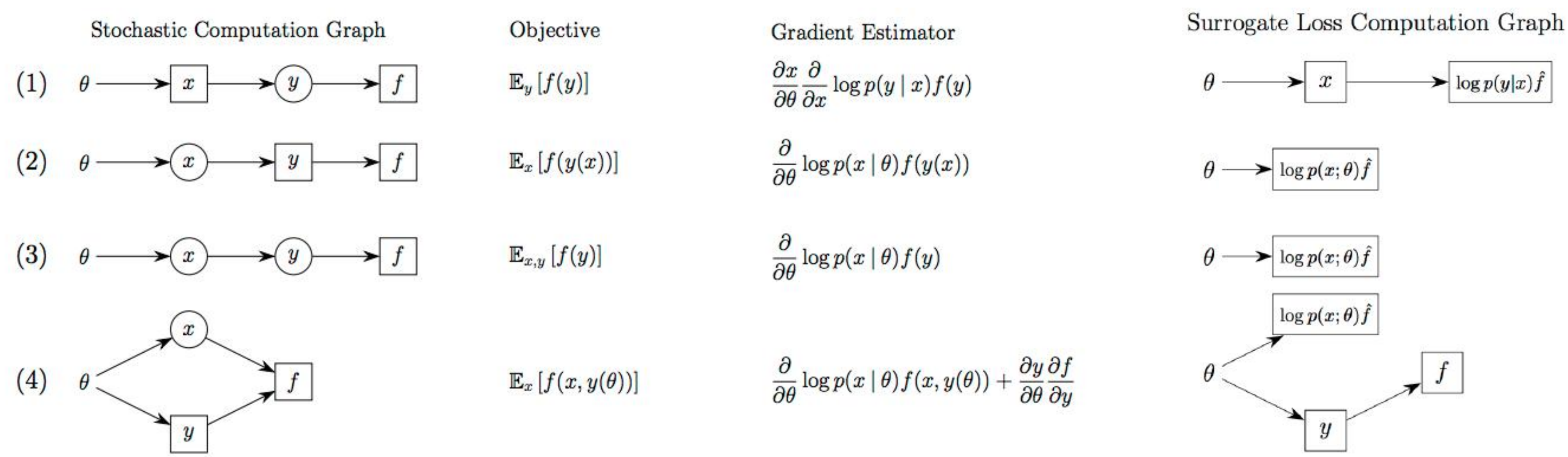
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs



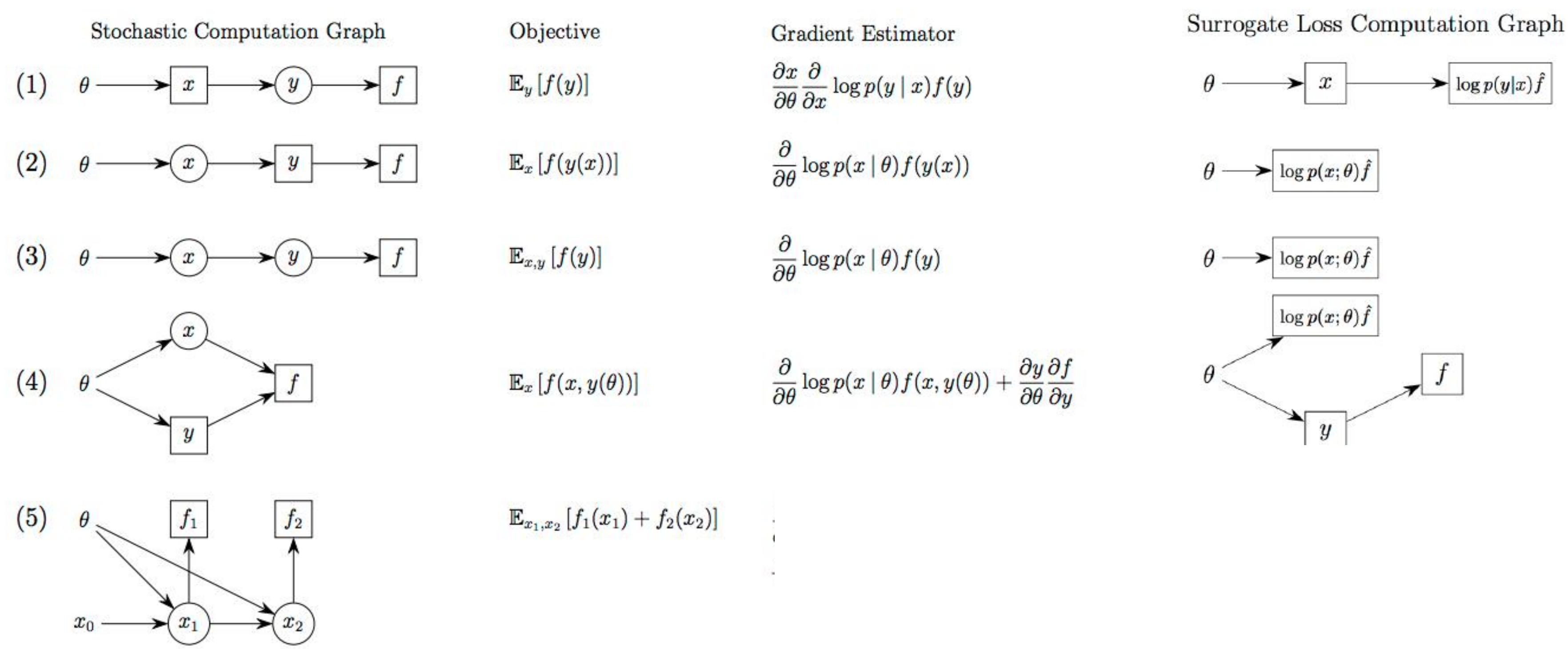
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs



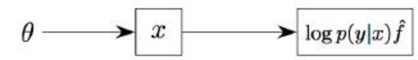
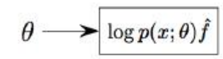
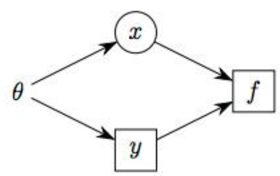
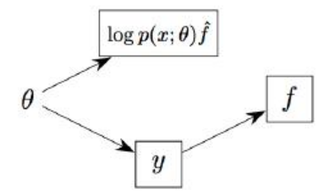
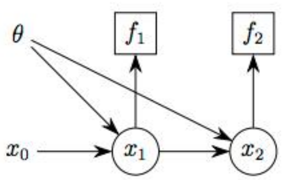
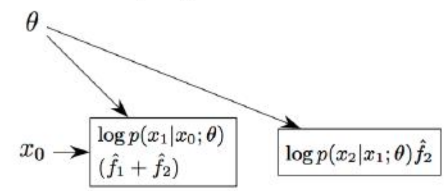
[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs



[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Stochastic Computation Graphs

	Stochastic Computation Graph	Objective	Gradient Estimator	Surrogate Loss Computation Graph
(1)		$\mathbb{E}_y\left[f(y)\right]$	$\frac{\partial x}{\partial \theta}\frac{\partial}{\partial x}\log p(y \mid x) f(y)$	
(2)		$\mathbb{E}_x\left[f(y(x))\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(y(x))$	
(3)		$\mathbb{E}_{x,y}\left[f(y)\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(y)$	
(4)		$\mathbb{E}_x\left[f(x, y(\theta))\right]$	$\frac{\partial}{\partial \theta}\log p(x \mid \theta) f(x, y(\theta)) + \frac{\partial y}{\partial \theta}\frac{\partial f}{\partial y}$	
(5)		$\mathbb{E}_{x_1,x_2}\left[f_1(x_1) + f_2(x_2)\right]$	$\frac{\partial}{\partial \theta}\log p(x_1 \mid \theta, x_0)(f_1(x_1) + f_2(x_2))$ $+\frac{\partial}{\partial \theta}\log p(x_2 \mid \theta, x_1) f_2(x_2)$	

[Schulman, Heess, Weber, Abbeel, NIPS 2015]

Food for Thought

- When more than gradient computation is applicable, which one is best?
- When dynamics is only available as black-box, but derivatives aren't available – finite difference based derivatives on the dynamics black box?
  - OR: directly finite differences / gradient-free on the policy
  - Finite difference tricky (impractical?) when can't control random seed…
- What if model is unknown, but estimate available?