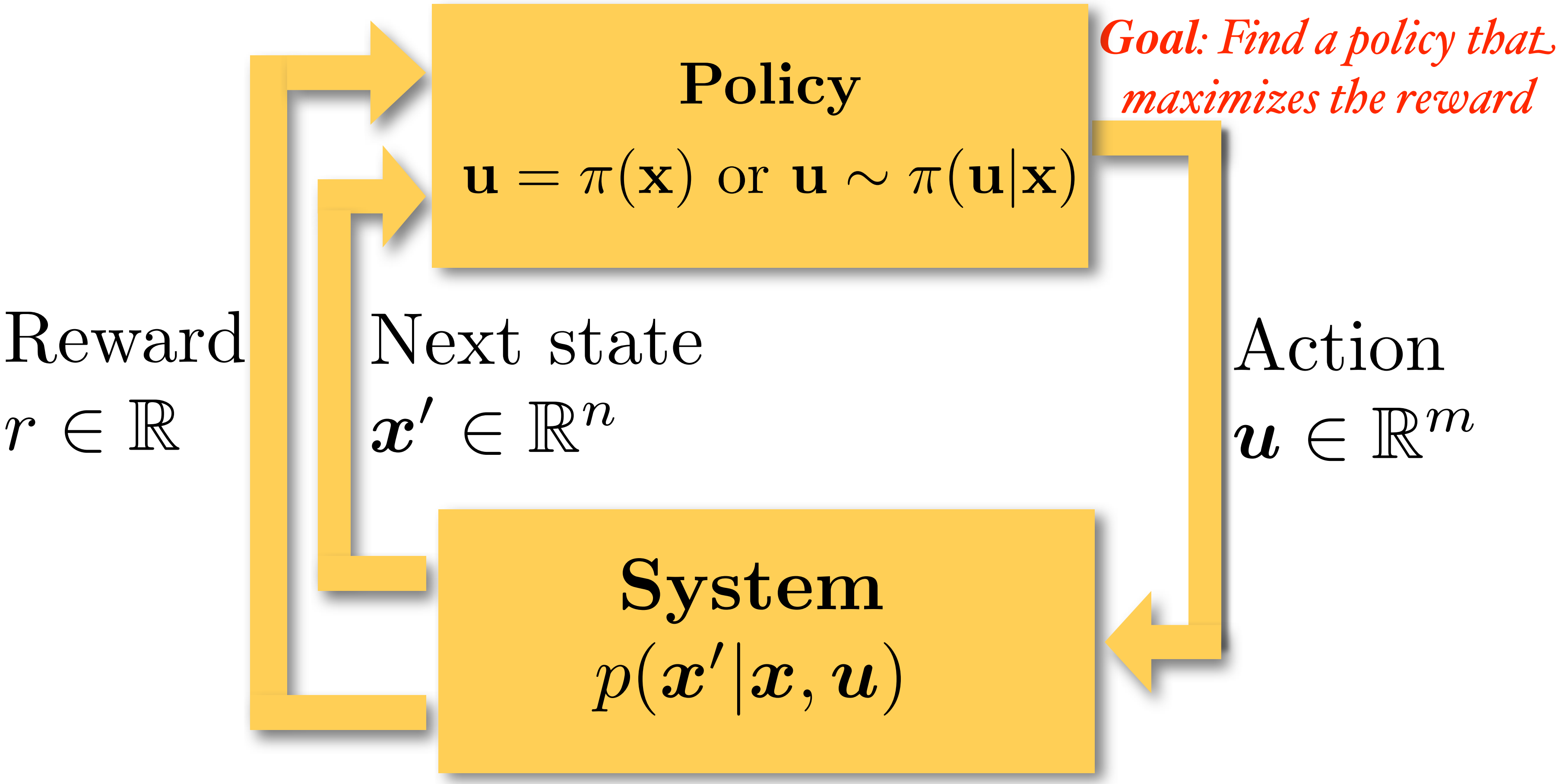
Tutorial on Policy Gradient Methods

Jan Peters

Outline

1. **Reinforcement Learning**
2. Finite Difference vs Likelihood-Ratio Policy Gradients
3. Likelihood-Ratio Policy Gradients
4. Conclusion

General Setup



Goal of RL

What does maximizing your rewards mean?

$$J(\pi) = \frac{1}{T}\sum_{t=1}^{T} r(\mathbf{x}_t, \mathbf{u}_t, t) \rightarrow E\{r(\mathbf{x}, \mathbf{u}, t)\}$$

Find a policy that maximizes the rewards!

A policy tells you which actions to use for each state!

Policy Search vs Value Function Methods

Value Function View!

Critic: Policy Evaluation

$$Q(\mathbf{x}_t, \mathbf{u}_t, t) = E\left\{ \sum_{\tau=t}^{T} r(\mathbf{x}_\tau, \mathbf{u}_\tau, \tau) \middle| \mathbf{x}_t, \mathbf{u}_t \right\}$$

Actor: Compute Optimal Policy

$$\mathbf{u}_t = \pi(\mathbf{x}_t, t) = \operatorname{argmax} Q(\mathbf{x}_t, \mathbf{u}, t)$$

Policy Search View!

Critic: Policy Sensitivity

$$J(\pi) = E\left\{ \sum_{t=0}^{T} r(\mathbf{x}_t, \mathbf{u}_t, t) \right\}$$

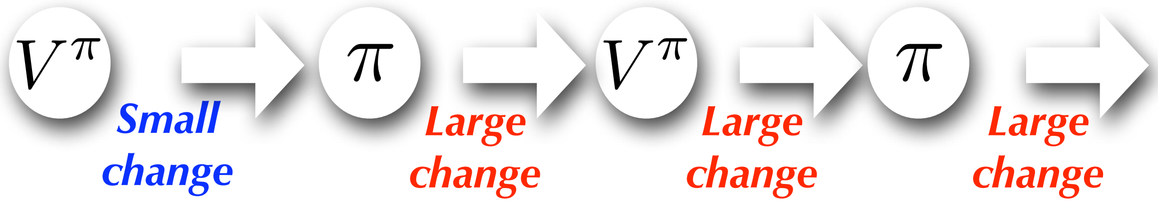
Actor: Policy Improvement

$$\pi' = \operatorname{argmax}_{\pi'} \{ J(\pi') - J(\pi) \}$$

Greedy vs Gradients

Greedy Updates:

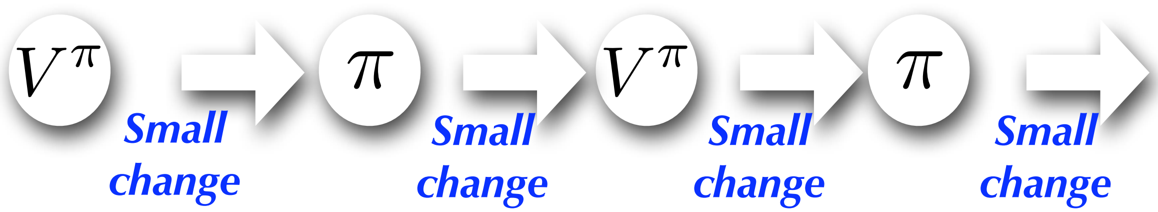
$$\boldsymbol{\theta}_{\pi'} = \text{argmax}_{\tilde{\boldsymbol{\theta}}} E_{\pi_{\tilde{\theta}}} \left\{ Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) \right\}$$



potentially unstable learning process with large policy jumps

Policy Gradient Updates:

$$\boldsymbol{\theta}_{\pi'} = \boldsymbol{\theta}_{\pi} + \alpha \left. \frac{dJ(\boldsymbol{\theta})}{d\boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_\pi}$$



stable learning process with smooth policy improvement

Outline

1. Reinforcement Learning for Motor Control?
2. **Finite Difference vs Likelihood-Ratio Policy Gradients**
3. Likelihood-Ratio Policy Gradients
4. Conclusion

Why Policy Gradient Methods?

Why Policy Gradients?

- Smooth changes in the parameters result into stability.
- Prior information can be incorporated with ease.
- Works with incomplete information.
- Exploration-Exploitation Dilemma implicitly treated.
- Is unbiased!
- Only requires much fewer samples.

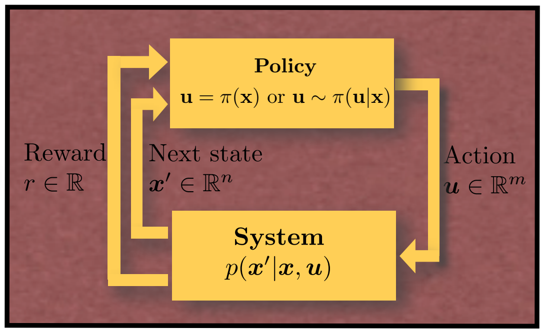
3. FD vs LR gradients

Finite Difference Gradients

Blackbox-Approach

Perturb the Parameters of your Policy

$\theta + \delta\theta$ →



→ $J(\theta + \delta\theta) - J(\theta)$

$$\frac{dJ}{d\theta} \approx \frac{J(\theta + \delta\theta) - J(\theta)}{\delta\theta}$$

Finite Difference Gradients

Why use Finite Difference Gradients?

- Only needs a black box!
- Works on any parameterization and deterministic policy.
- Fast to estimate for deterministic systems.
- State of the art in the simulation community

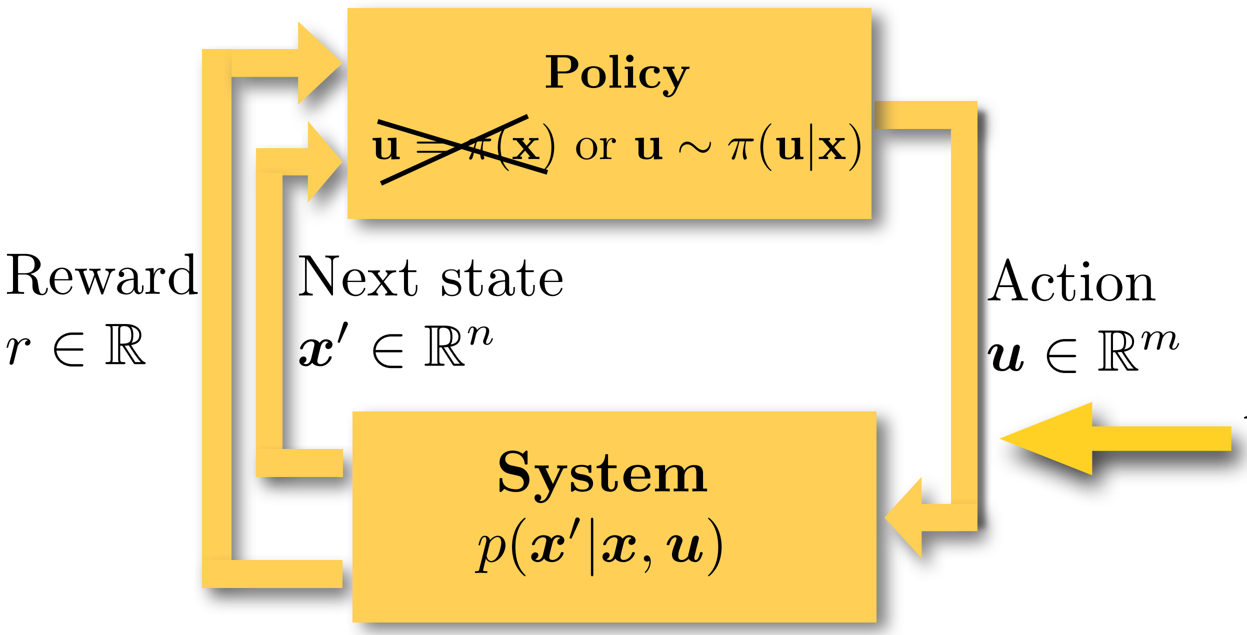
Why not?

- Parameter perturbation can destroy your robot.
- Exploration is hard to include.
- For stochastic systems it is very **slow**.

Likelihood Ratio Gradients

Whitebox-Approach

Perturb the actions using a stochastic policy



Actions with Noise

Likelihood Ratio Gradients

Whitebox-Approach: Likelihood Ratio Trick

$$
\begin{aligned}
\frac{d}{d\theta}J(\theta) &= \frac{d}{d\theta}\int_{\mathbb{U}} \pi\left(u\right) r\left(u\right) du, && (1)\\
&= \int_{\mathbb{U}} \frac{d\pi\left(u\right)}{d\theta} r\left(u\right) du, && (2)\\
&= \int_{\mathbb{U}} \pi\left(u\right) \frac{1}{\pi\left(u\right)} \frac{d\pi\left(u\right)}{d\theta} r\left(u\right) du, && (3)\\
&= \int_{\mathbb{U}} \pi\left(u\right) \frac{d\log\pi\left(u\right)}{d\theta} r\left(u\right) du, && (4)\\
&= E\left\{\frac{d\log\pi\left(u\right)}{d\theta} r\left(u\right)\right\} \approx \sum_{i=1}^{N} \frac{d\log\pi\left(u_i\right)}{d\theta} r\left(u_i\right) && (5)
\end{aligned}
$$

Likelihood Ratio Gradients

Why use Likelihood Ratio Gradients?

- Fastest Gradient Method!
- We know the policy derivative - thus they are more efficient than for the simulation community.
- Only perturb the motor command -> policies will remain stable!

Why not?

- Stochastic policy.
- Injection of noise into the system.
- Theory much more complex!

Outline

1. Reinforcement Learning for Motor Control?
2. Finite Difference vs Likelihood-Ratio Policy Gradients
3. **Likelihood-Ratio Policy Gradients**
4. Conclusion

Goal of RL Revisited

Goal: Optimize the expected return

$$J(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \pi(\boldsymbol{u}|\boldsymbol{x}) r(\boldsymbol{x}, \boldsymbol{u}) d\boldsymbol{u} d\boldsymbol{x},$$

State distribution

Policy
(we can choose it)

Reward

$$= (1-\gamma) E\left\{ \sum_{t=0}^{\infty} \gamma^t r_t \right\}$$

Policy Gradient Methods

According to the Policy Gradient Theorem, the gradient can be computed as

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \nabla_{\boldsymbol{\theta}} \pi(\boldsymbol{u}|\boldsymbol{x}) (Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) - b^{\pi}(\boldsymbol{x})) d\boldsymbol{u} d\boldsymbol{x}.$$

Gradient of the expected return

Derivative only of the policy

State-action value function

Arbitrary baseline function

Problems: High variance, very slow convergence, dependence on baseline!

Originally discovered: Aleksandrov, 1968; Glynn, 1986.
Examples: episodic REINFORCE, SRV, GPOMDP, ...

Policy Gradient Methods

According to the Policy Gradient Theorem, the gradient can be computed as

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \nabla_{\boldsymbol{\theta}} \pi(\boldsymbol{u}|\boldsymbol{x})(Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) - b^{\pi}(\boldsymbol{x})) d\boldsymbol{u} d\boldsymbol{x}.$$

Gradient of the expected return

Derivative only of the policy

State-action value function

Arbitrary baseline function

Problems: High variance, very slow convergence, dependence on baseline!

Originally discovered: Aleksandrov, 1968; Glynn,1986.
Examples: episodic REINFORCE, SRV, GPOMDP, ...

4. Likelihood Ratio Gradients

Compatible Function Approximation

The state-action value function can be replaced by

$$Q^\pi(\boldsymbol{x}, \boldsymbol{u}) \equiv f^\pi_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{u}) = \frac{d\log\pi(\boldsymbol{u}|\boldsymbol{x})}{d\boldsymbol{\theta}}^T \boldsymbol{w}$$

State-action value function

Compatible function approximation

Log-policy derivative

Parameters of the function approximator

without biasing the gradient.

Thus, the policy gradient becomes

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^\pi(\boldsymbol{x}) \int_{\mathbb{U}} \nabla_{\boldsymbol{\theta}} \pi(\boldsymbol{u}|\boldsymbol{x}) (f^\pi_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{u}) - b^\pi(\boldsymbol{x})) d\boldsymbol{u} d\boldsymbol{x}.$$

(Sutton et al., 2000; Konda& Tsitsiklis, 2000)

All-Action Gradient

By integrating over all possible actions in a state, the baseline can be integrated out, and the gradient becomes:

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \nabla_{\boldsymbol{\theta}} \pi(\boldsymbol{u}|\boldsymbol{x})(f_w^{\pi}(\boldsymbol{x},\boldsymbol{u}) - b(\boldsymbol{x})) d\boldsymbol{u} d\boldsymbol{x}, \\
&= \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \pi(\boldsymbol{u}|\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x})^T \boldsymbol{w} d\boldsymbol{u} d\boldsymbol{x}, \\
&= \boldsymbol{F}(\boldsymbol{\theta}) \boldsymbol{w}.
\end{aligned}
$$

All Action Matrix

Parameters

(Peters et al., 2003)

Natural Gradients

Natural Gradients:

A more efficient gradient in learning problems is the natural gradient (Amari, 1998):

Natural gradient

$$\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = G^{-1}(\boldsymbol{\theta}) \boldsymbol{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Inverse of the Fisher Information Matrix

'Vanilla` gradient

where

$$\boldsymbol{G}(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \pi(\boldsymbol{u}|\boldsymbol{x}) \boldsymbol{\nabla}_{\boldsymbol{\theta}} \log\left(d^{\pi}(\boldsymbol{x})\pi(\boldsymbol{u}|\boldsymbol{x})\right) \boldsymbol{\nabla}_{\boldsymbol{\theta}} \log\left(d^{\pi}(\boldsymbol{x})\pi(\boldsymbol{u}|\boldsymbol{x})\right) d\boldsymbol{u} d\boldsymbol{x}.$$

$$\boldsymbol{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \boldsymbol{\nabla}_{\boldsymbol{\theta}} \pi(\boldsymbol{u}|\boldsymbol{x})(Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) - b^{\pi}(\boldsymbol{x})) d\boldsymbol{u} d\boldsymbol{x}.$$

All Action = Fisher Information!

So how does the All-Action Matrix

$$\boldsymbol{F}(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \pi(\boldsymbol{u}|\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x}) d\boldsymbol{u} d\boldsymbol{x}.$$

relate to the Fisher Information Matrix

$$\boldsymbol{G}(\boldsymbol{\theta}) = \int_{\mathbb{X}} d^{\pi}(\boldsymbol{x}) \int_{\mathbb{U}} \pi(\boldsymbol{u}|\boldsymbol{x}) \nabla_{\boldsymbol{\theta}} \log \left(d^{\pi}(\boldsymbol{x}) \pi(\boldsymbol{u}|\boldsymbol{x})\right) \nabla_{\boldsymbol{\theta}} \log \left(d^{\pi}(\boldsymbol{x}) \pi(\boldsymbol{u}|\boldsymbol{x})\right) d\boldsymbol{u} d\boldsymbol{x}.$$

While Kakade (2002) suggested that ***F*** is an ‘average of point Fisher information matrices’, we could prove that

***F* = *G*.**

(Peters et al., 2003; 2005; Bagnel et al., 2003)

4. Likelihood Ratio Gradients

Natural Gradient Updates

As ***G*** = ***F***, the gradient simplifies to

$$\tilde{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{G}^{-1}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{G}^{-1}(\boldsymbol{\theta}) \boldsymbol{F}(\boldsymbol{\theta}) \boldsymbol{w} = \boldsymbol{w},$$

and the policy parameter update becomes

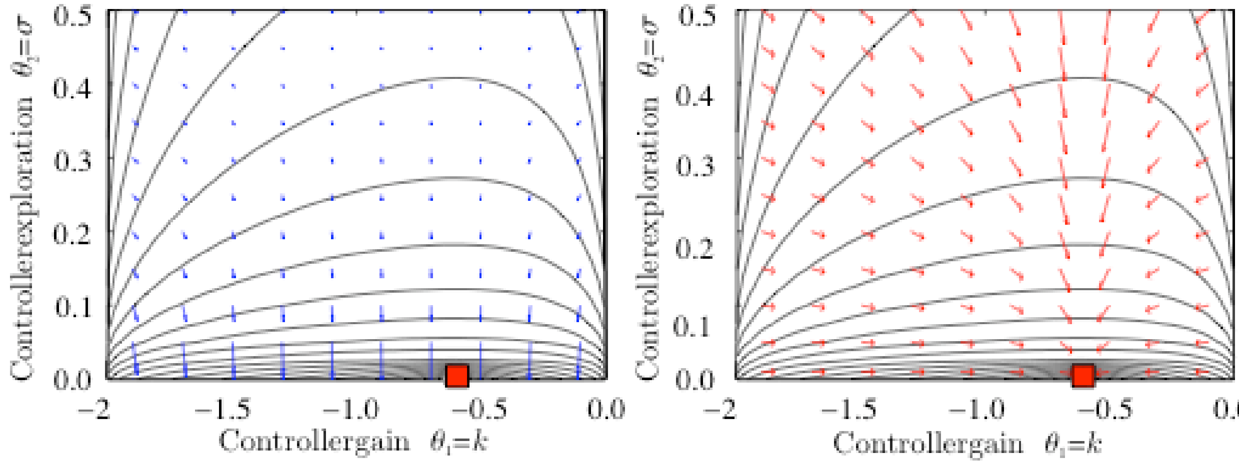
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha_t \boldsymbol{w}_t.$$

Important: The estimation of the gradient has simplified upon estimating the compatible function approximation / critic!!!

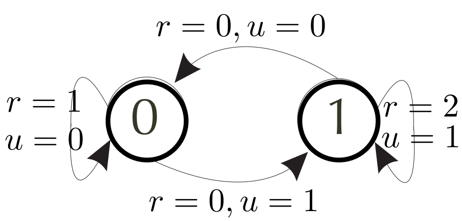
(Kakade, 2002; Peters et al., 2003, 2005; Bagnell&Schneider, 2003)

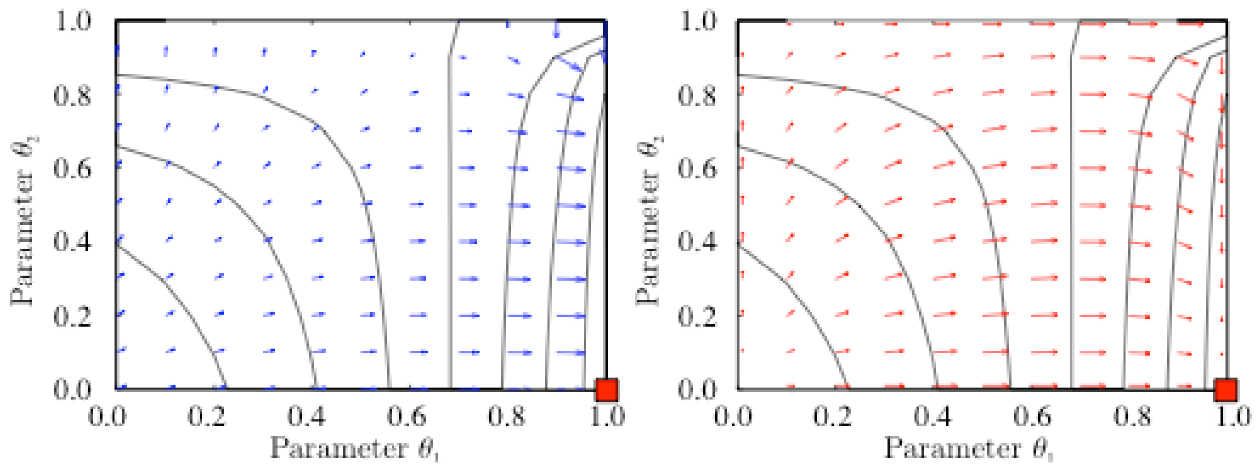
Natural Policy Gradients

Linear Quadratic Regulation



Two-State Problem





4. Likelihood Ratio Gradients

Compatible Function Approximation

To obtain the natural gradient

$$\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{w}$$

we need to estimate the compatible function approximation

$$f_{\boldsymbol{w}}^{\pi}(\boldsymbol{x}, \boldsymbol{u}) = \frac{d\log\pi(\boldsymbol{u}|\boldsymbol{x})}{d\boldsymbol{\theta}}^{T} \boldsymbol{w}$$

This function approximation is mean zero! Therefore it can ONLY represent the Advantage Function

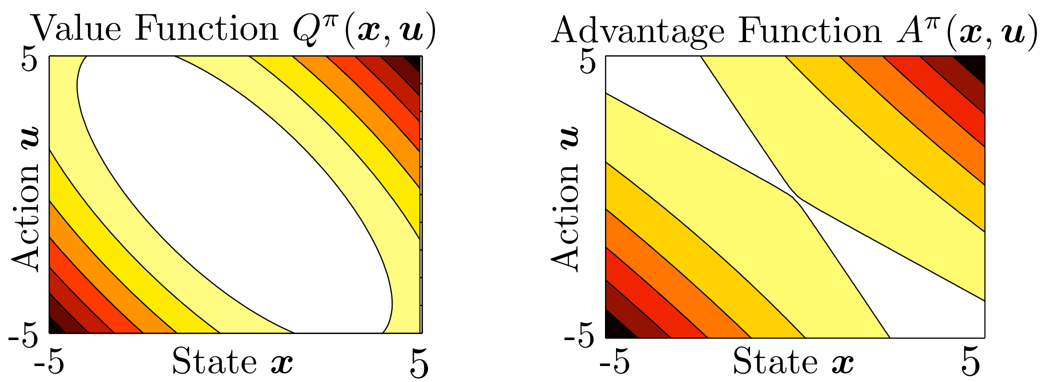
$$f_{\boldsymbol{w}}^{\pi}(\boldsymbol{x}, \boldsymbol{u}) = Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) - V^{\pi}(\boldsymbol{x}) = A^{\pi}(\boldsymbol{x}, \boldsymbol{u}).$$

Compatible Function Approximation

The advantage function

$$f_{\boldsymbol{w}}^{\pi}(\boldsymbol{x}, \boldsymbol{u}) = Q^{\pi}(\boldsymbol{x}, \boldsymbol{u}) - V^{\pi}(\boldsymbol{x}) = A^{\pi}(\boldsymbol{x}, \boldsymbol{u}).$$

is very different from the value functions!



...and we cannot directly do Temporal Difference Learning on this representation!

4. Likelihood Ratio Gradients

Natural Actor-Critic

We cannot do TD learning with

$$f_{\boldsymbol{w}}^{\pi}(\boldsymbol{x},\boldsymbol{u}) = Q^{\pi}(\boldsymbol{x},\boldsymbol{u}) - V^{\pi}(\boldsymbol{x}) = A^{\pi}(\boldsymbol{x},\boldsymbol{u}).$$

But when we add further basis function approximators

$$V^{\pi}(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^{T}\boldsymbol{v}$$

into the Bellman equation

$$V^{\pi}(\boldsymbol{x}_t) + \boldsymbol{\nabla}_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}_t|\boldsymbol{x}_t)^{T}\boldsymbol{w} = r(\boldsymbol{x}_t,\boldsymbol{u}_t) + \gamma V^{\pi}(\boldsymbol{x}_{t+1}) + \epsilon_t$$

we get **a linear regression problem which can be solved with the LSTD(λ) algorithm (Boyan, 1996) in one step!**

Natural Actor-Critic

New basis functions

Boyan's LSTD(λ)

Critic: LSTD-Q(λ) Evaluation

$$\boldsymbol{\Gamma}_t = [\boldsymbol{\phi}(\boldsymbol{x}_{t+1})^T, \mathbf{0}^T]^T$$

$$\boldsymbol{\Phi}_t = [\boldsymbol{\phi}(\boldsymbol{x}_t)^T, \boldsymbol{\nabla}_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}_t|\boldsymbol{x}_t)^T]^T$$

$$\boldsymbol{z}_{t+1} = \lambda \boldsymbol{z}_t + \boldsymbol{\Phi}_t$$

$$\boldsymbol{A}_{t+1} = \boldsymbol{A}_t + \boldsymbol{z}_{t+1}(\boldsymbol{\Phi}_t - \gamma \boldsymbol{\Gamma}_t)$$

$$\boldsymbol{b}_{t+1} = \boldsymbol{b}_t + \boldsymbol{z}_{t+1} r_{t+1}$$

$$[\boldsymbol{w}_{t+1}^T, \boldsymbol{v}_{t+1}^T]^T = \boldsymbol{A}_{t+1}^{-1} \boldsymbol{b}_{t+1}$$

Actor: Natural Policy Gradient Improvement

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha_t \boldsymbol{w}_t.$$

4. Likelihood Ratio Gradients

Algorithms Derivable from this Framework

Gibbs Policy

$$\pi(u_t|x_t) = e^{\theta_{xu}} / \sum_b e^{\theta_{xb}}$$

Additional Basis Functions

$$\phi(x) = [0, ..., 0, 1, 0, ..., 0]^T$$

NAC Framework → **Sutton et al.'s (1983) Actor Critic**

Linear Gauss-Policy

$$\pi(\boldsymbol{u}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{u} - \boldsymbol{\theta}_{\text{gain}}{}^T \boldsymbol{x}, \boldsymbol{\theta}_{\text{explore}})$$

Additional Basis Functions

$$\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P} \boldsymbol{x} + p$$

NAC Framework with Learning Rate

$$\alpha_i = \frac{1}{\|J(\theta_i)\|}$$

→ **Bradtke&Bartos (1993) Q-Learning for LQR**

Episodic Natural Actor Critic

...but in many cases we don't have any good additional basis functions!

$$V^{\pi}(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^{T}\boldsymbol{v}$$

In this case, we can sum up the advantages along a trajectory and obtain one data point for a linear regression problem

$$\underbrace{V^{\pi}(\mathbf{x}_0)}_{J} + \underbrace{\left(\sum_{t=0}^{T} \nabla_{\theta} \log \pi(\mathbf{u}_t|\mathbf{x}_t)\right)^{T}}_{\varphi_i} \mathbf{w} = \underbrace{\sum_{t=0}^{T} \gamma^{t} r(\mathbf{x}_t, \mathbf{u}_t)}_{R_i} + \gamma^{T+1} \underbrace{V^{\pi}(\mathbf{x}_{T+1})}_{0}$$

...and an additional basis function of 1 suffices!

Episodic Natural Actor-Critic

Critic: Episodic Evaluation

Sufficient Statistics

$$\mathbf{\Phi} = \begin{bmatrix} \varphi_1, & \varphi_2, & \ldots, & \varphi_N \\ 1, & 1, & \ldots, & 1 \end{bmatrix}^T$$

$$\mathbf{R} = \left[R_1, R_2^T, \ldots, R_N^T\right]^T$$

Linear Regression

$$\begin{bmatrix} \boldsymbol{w} \\ J \end{bmatrix} = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{R}$$

Actor: Natural Policy Gradient Improvement

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha_t \boldsymbol{w}_t.$$

Improving Motor Primitives

Ijspeert et al. (2002) suggested a nonlinear dynamics approach for motor primitives in imitation learning:

Trajectory Plan Dynamics

$$\begin{cases} \dot{z} = \alpha_z\left(\beta_z\left(g-y\right)-z\right) \\ \dot{y} = \alpha_y\left(f\left(x,v\right)+z\right) \end{cases}$$

where

Canonical Dynamics

$$\begin{cases} \dot{v} = \alpha_v\left(\beta_v\left(g-x\right)-v\right) \\ \dot{x} = \alpha_x v \end{cases}$$

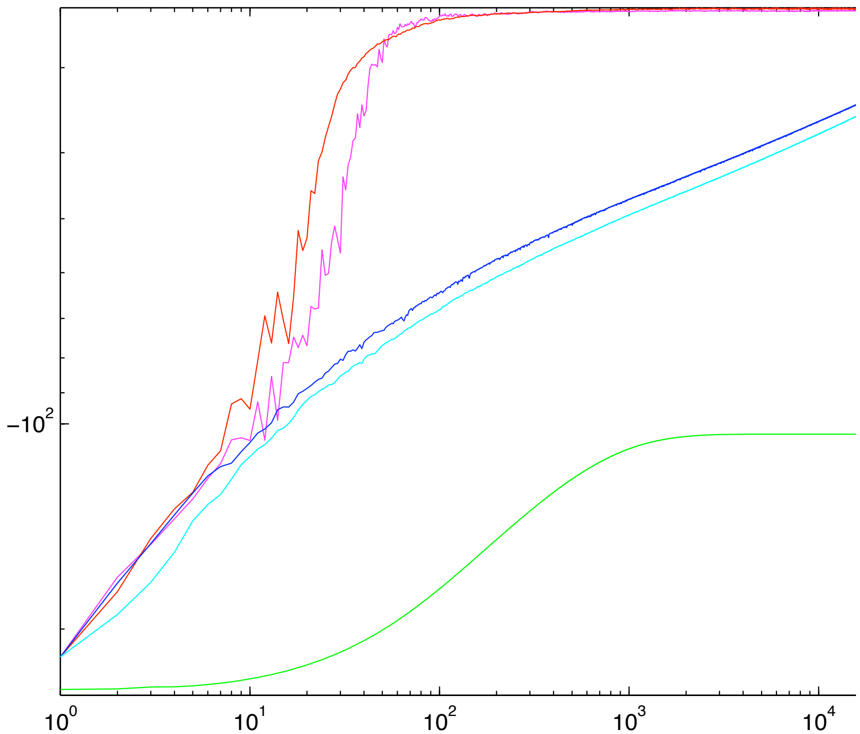
Local Linear Model Approx.

$$\begin{cases} f\left(x,v\right) = \dfrac{\sum_{i=1}^{k} w_i b_i v}{\sum_{i=1}^{k} w_i} \\ w_i = \exp\left(-\dfrac{1}{2} d_i\left(\bar{x}-c_i\right)^2\right) \text{ and } \bar{x} = \dfrac{x-x_0}{g-x_0} \end{cases}$$

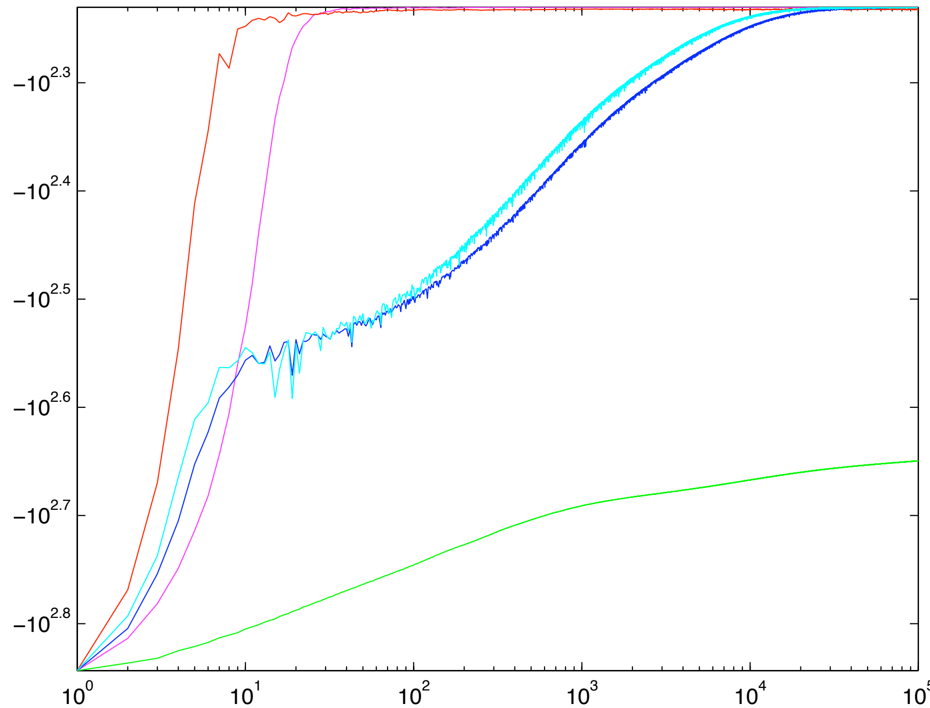
The parameters b can also be improved by Reinforcement Learning

Improving Motor Primitives

Minimum Motor Command



Two Goals Policies



4. Evaluations

Outline

1. Reinforcement Learning for Motor Control?
2. Finite Difference vs Likelihood-Ratio Policy Gradients
3. Likelihood-Ratio Policy Gradients
4. **Conclusion**

Conclusions

- *If you can explore your complete state-action space sufficiently … use value function methods.*
- *If you have access to policy and its derivatives… use likelihood ratio policy gradient methods.*
  - *If you have good additional basis function… use Natural Actor-Critic.*
  - *If not … use Episodic Natural Actor-Critic.*
  - *If you want to explore a problem fastly … use 'vanilla' likelihood ratio policy gradient methods.*
- *If you can only access your parameters… use finite difference policy gradient methods.*
- *If you can only access your parameters… use finite difference policy gradient methods.*

Projects...

- **Learning Motor Primitives with Reward-Weighted Regression**
  - *This will be a nearly new method... very enthusiastic people needed!*
- ***Applying Policy Gradients (methods of your choice!) to Oscillator Optimization!***
  - *Here we can create several projects if you like :)*