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
1. Reinforcement Learning

**General Setup**

- **Reward**: $r \in \mathbb{R}$
- **Next state**: $x' \in \mathbb{R}^n$
- **Action**: $u \in \mathbb{R}^m$

**System**

$p(x'|x,u)$

**Policy**

$u = \pi(x)$ or $u \sim \pi(u|x)$

**Goal**: Find a policy that maximizes the reward
What does maximizing your rewards mean?

\[ J(\pi) = \frac{1}{T} \sum_{t=1}^{T} r(x_t, u_t, t) \rightarrow E\{r(x, u, t)\} \]

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
\[ Q(x_t, u_t, t) = E \left\{ \sum_{\tau=t}^{T} r(x_{\tau}, u_{\tau}, \tau) \mid x_t, u_t \right\} \]

Actor: Compute Optimal Policy
\[ u_t = \pi(x_t, t) = \arg\max Q(x_t, u, t) \]

Policy Search View!

Critic: Policy Sensitivity
\[ J(\pi) = E \left\{ \sum_{t=0}^{T} r(x_t, u_t, t) \right\} \]

Actor: Policy Improvement
\[ \pi' = \arg\max_{\pi'} \{ J(\pi') - J(\pi) \} \]

1. Reinforcement Learning
Greedy vs Gradients

Greedy Updates:

$$\theta_{\pi'} = \arg\max_{\tilde{\theta}} E_{\pi_{\tilde{\theta}}} \{ Q^\pi(x, u) \}$$

- $V^\pi \rightarrow \pi$ (Small change)
- $\pi \rightarrow V^\pi$ (Large change)
- $V^\pi \rightarrow \pi$ (Large change)
- $\pi \rightarrow \pi$ (Large change)

Policy Gradient Updates:

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

- $V^\pi \rightarrow \pi$ (Small change)
- $\pi \rightarrow V^\pi$ (Small change)
- $V^\pi \rightarrow \pi$ (Small change)
- $\pi \rightarrow \pi$ (Small change)

2. Value Function Methods

- Potentially unstable learning process with large policy jumps
- Stable learning process with smooth policy improvement
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*

\[ \theta + \delta \theta \] \[ \rightarrow \] \[ J(\theta + \delta \theta) - J(\theta) \]

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

3. FD vs LR gradients
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.

3. FD vs LR gradients
Likelihood Ratio Gradients

Whitebox-Approach

Perturb the actions using a stochastic policy

Reward $r \in \mathbb{R}$

Next state $x' \in \mathbb{R}^n$

Action $u \in \mathbb{R}^m$

System $p(x'|x, u)$

Policy $u = \pi(x)$ or $u \sim \pi(u|x)$

Actions with Noise

3. FD vs LR gradients
Whitebox-Approach: Likelihood Ratio Trick

\[
\frac{d}{d\theta} J(\theta) = \frac{d}{d\theta} \int_{U} \pi(u) r(u) du, \quad (1)
\]

\[
= \int_{U} \frac{d\pi(u)}{d\theta} r(u) du, \quad (2)
\]

\[
= \int_{U} \pi(u) \frac{1}{\pi(u)} \frac{d\pi(u)}{d\theta} r(u) du, \quad (3)
\]

\[
= \int_{U} \pi(u) \frac{d \log \pi(u)}{d\theta} r(u) du, \quad (4)
\]

\[
= E \left\{ \frac{d \log \pi(u)}{d\theta} r(u) \right\} \approx \sum_{i=1}^{N} \frac{d \log \pi(u_i)}{d\theta} r(u_i) \quad (5)
\]
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!

3. FD vs LR gradients
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

\[ J(\theta) = \int_X d\pi(x) \int_U \pi(u|x)r(x,u)du\,d\,x, \]

State distribution  
Policy  
(we can choose it)  
Reward

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

4. Likelihood Ratio Gradients
According to the Policy Gradient Theorem, the gradient can be computed as

$$\nabla_\theta J(\theta) = \int_X d\pi(x) \int_U \nabla_\theta \pi(u|x)(Q^\pi(x, u) - b^\pi(x))dudx.$$  

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

Examples: episodic REINFORCE, SRV, GPOMDP, ...

4. Likelihood Ratio Gradients
According to the Policy Gradient Theorem, the gradient can be computed as

\[ \nabla_\theta J(\theta) = \int_X d\pi(x) \int_U \nabla_\theta \pi(u|x)(Q^\pi(x,u) - b^\pi(x))dudx. \]

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

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

### 4. Likelihood Ratio Gradients
The state-action value function can be replaced by

\[
Q^\pi(x, u) \equiv f^\pi_w(x, u) = \frac{d\log \pi(u|x)}{d\theta}^T w
\]

without biasing the gradient.

Thus, the policy gradient becomes

\[
\nabla_\theta J(\theta) = \int_X d\pi(x) \int_U \nabla_\theta \pi(u|x)(f^\pi_w(x, u) - b^\pi(x))du dx.
\]

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

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

\[ \nabla_\theta J(\theta) = \int_X d\pi(x) \int_U \nabla_\theta \pi(u|x)(f_w(x, u) - b(x)) du dx, \]

\[ = \int_X d\pi(x) \int_U \pi(u|x) \nabla_\theta \log \pi(u|x) \nabla_\theta \log \pi(u|x)^T w du dx, \]

\[ = F(\theta)w. \]

4. Likelihood Ratio Gradients

(Peters et al., 2003)
Natural Gradients:

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

\[ \nabla_{\theta} J(\theta) = G^{-1}(\theta) \nabla_{\theta} J(\theta) \]

where

\[ G(\theta) = \int_{X} d\pi(x) \int_{U} \pi(u|x) \nabla_{\theta} \log(d\pi(x)\pi(u|x)) \nabla_{\theta} \log(d\pi(x)\pi(u|x)) du dx. \]

\[ \nabla_{\theta} J(\theta) = \int_{X} d\pi(x) \int_{U} \nabla_{\theta} \pi(u|x)(Q(\pi, x, u) - b(\pi, x)) du dx. \]
So how does the All-Action Matrix

$$F(\theta) = \int_X d\pi(x) \int_U \pi(u|x) \nabla_\theta \log \pi(u|x) \nabla_\theta \log \pi(u|x) dudx.$$  

relate to the Fisher Information Matrix

$$G(\theta) = \int_X d\pi(x) \int_U \pi(u|x) \nabla_\theta \log \pi(u|x) \nabla_\theta \log \pi(u|x) dudx.$$  

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)
Natural Gradient Updates

As $G = F$, the gradient simplifies to

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

and the policy parameter update becomes

$$\theta_{t+1} = \theta_t + \alpha_t w_t.$$

**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)

4. Likelihood Ratio Gradients
Natural Policy Gradients

Linear Quadratic Regulation

Two-State Problem

4. Likelihood Ratio Gradients
Compatible Function Approximation

To obtain the natural gradient

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

we need to estimate the compatible function approximation

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

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

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

4. Likelihood Ratio Gradients
The advantage function

\[ f^\pi_w(x, u) = Q^\pi(x, u) - V^\pi(x) = A^\pi(x, u). \]

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

\[ f_{\pi}^{\pi}(x, u) = Q_{\pi}(x, u) - V_{\pi}(x) = A_{\pi}(x, u). \]

But when we add further basis function approximators

\[ V_{\pi}(x) = \phi(x)^T v \]

into the Bellman equation

\[ V_{\pi}(x_t) + \nabla_\theta \log \pi(u_t|x_t)T w = r(x_t, u_t) + \gamma V_{\pi}(x_{t+1}) + \epsilon_t \]

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

4. Likelihood Ratio Gradients
Natural Actor-Critic

Critic: LSTD-Q(\(\lambda\)) Evaluation

\[
\begin{align*}
\Gamma_t &= [\phi(x_{t+1})^T, 0^T]^T \\
\Phi_t &= [\phi(x_t)^T, \nabla_\theta \log \pi(u_t|x_t)^T]^T \\
\z_{t+1} &= \lambda \z_t + \Phi_t \\
A_{t+1} &= A_t + \z_{t+1}(\Phi_t - \gamma \Gamma_t) \\
b_{t+1} &= b_t + \z_{t+1}r_{t+1} \\
[w_{t+1}^T, v_{t+1}^T]^T &= A_{t+1}^{-1}b_{t+1}
\end{align*}
\]

Actor: Natural Policy Gradient Improvement

\[
\theta_{t+1} = \theta_t + \alpha_t w_t.
\]

New basis functions

Boyan’s LSTD(\(\lambda\))

4. Likelihood Ratio Gradients
Algorithms Derivable from this Framework

**Gibbs Policy**

$$\pi(u_t|x_t) = \frac{e^{x^T u}}{\sum_b e^{x^T b}}$$

**Additional Basis Functions**

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

**Linear Gauss-Policy**

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

**Additional Basis Functions**

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

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

**NAC Framework**

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

**NAC Framework with Learning Rate**

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

4. Likelihood Ratio Gradients
Episodic Natural Actor Critic

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

\[ V^\pi(x) = \phi(x)^T \nu \]

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

\[
\begin{align*}
J & = \sum_{t=0}^{T} \nabla_{\theta} \log \pi(u_t|x_t) \\
\varphi_i & = \sum_{t=0}^{T} \gamma^t r(x_t, u_t) + \gamma^{T+1} V^\pi(x_{T+1}) \\
\end{align*}
\]

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

4. Likelihood Ratio Gradients
Episodic Natural Actor-Critic

### Critic: Episodic Evaluation

$$\Phi = \left[ \varphi_1, \varphi_2, \ldots, \varphi_N \right]^T$$

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

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

### Actor: Natural Policy Gradient Improvement

$$\theta_{t+1} = \theta_t + \alpha_t \mathbf{w}_t.$$
Ijspeert et al. (2002) suggested a nonlinear dynamics approach for motor primitives in imitation learning:

\[ \dot{z} = \alpha_z \beta_z (g - y) - z \]
\[ \dot{y} = \alpha_y (f(x,v) + z) \]

where

\[ \dot{v} = \alpha_v \beta_v (g - x) - v \]
\[ \dot{x} = \alpha_x v \]

\[ f(x,v) = \frac{\sum_{i=1}^{k} w_i b_i v}{\sum_{i=1}^{k} w_i} \]

\[ w_i = \exp \left( -\frac{1}{2} d_i (\bar{x} - c_i)^2 \right) \] and \[ \bar{x} = \frac{x - x_0}{g - x_0} \]

The parameters \( b \) can also be improved by Reinforcement Learning

4. Evaluations
4. Evaluations
Outline

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

6. Conclusion
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 :)