Aalto University
School of Electrical Engineering

ELEC-E8125 Reinforcement Learning
Policy gradient

Ville Kyrki

22.10.2019

Today

- Direct policy learning via policy gradient.

Learning goals

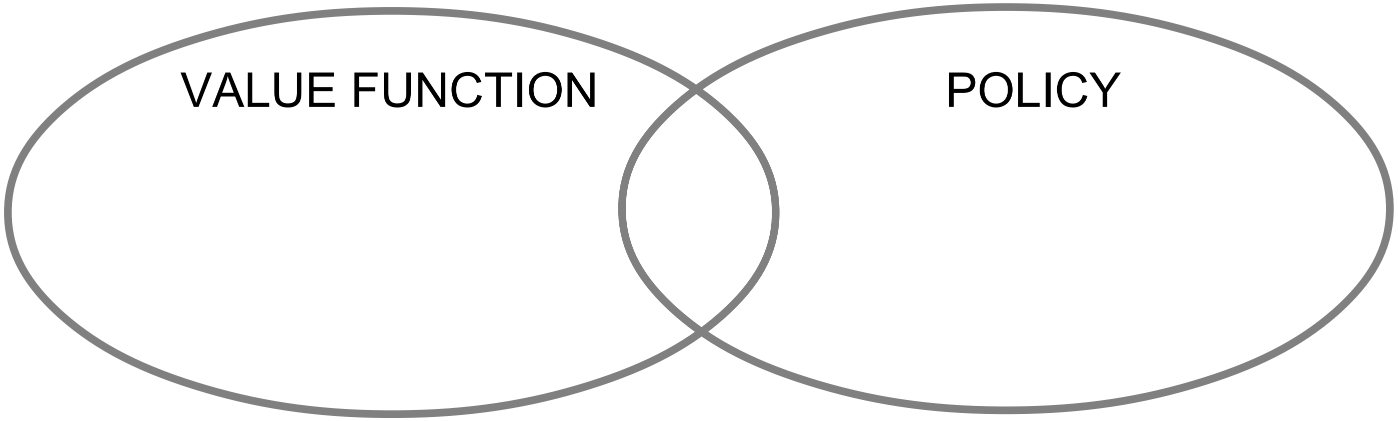
- Understand basis and limitations of policy gradient approaches.

Motivation

https://www.youtube.com/watch?v=xyJAvghtqIM

- Even with value function approximation, large state spaces can be problematic.
- Learning parametric policies $\pi(u|x,\theta)$ directly without learning value functions sometimes easier.
- Non-Markov (partially observable) or adversarial situations might benefit from stochastic policies.

Value-based vs policy-based RL



Value-based
- Learned value function.
- Implicit policy.

Actor-critic
- Learned value function.
- Learned policy.

Policy-based
- No value function.
- Learned policy.

- Can learn stochastic policies.
- Usually locally optimal.

Stochastic policies

- Discrete actions: Soft-max policy

$$\pi_{\theta}(\boldsymbol{u}_t|\boldsymbol{x}_t) = 1/Z\, e^{\theta^T \varphi(\boldsymbol{x}_t, \boldsymbol{u}_t)}$$

Probability portional to expontiated linear combination of features.

Normalization constant

$$Z = \sum_u e^{\theta^T \varphi(\boldsymbol{x}_t, \boldsymbol{u}_t)}$$

- Continuous actions: Gaussian policy

$$\pi_{\theta}(u_t|\boldsymbol{x}_t) \sim N(\boldsymbol{\theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_t), \sigma^2)$$

Mean is linear combination of features.

Can also be understood as linear policy plus exploration uncertainty

$$\pi_{\theta}(u_t|\boldsymbol{x}_t) = \boldsymbol{\theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_t) + \epsilon \quad \epsilon \sim N(0, \sigma^2)$$

Note: Policies include exploration!

But how to fit these?

Note: This is not RL!

Supervised policy learning – behavioral cloning

- Assume examples of policy are given in form of (x,u) pairs.
- How to fit a stochastic policy to these?

$$\pi_{\boldsymbol{\theta}}(u_t | \boldsymbol{x}_t) \sim N(\boldsymbol{\theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_t), \sigma^2)$$ ← Example

Note: This is not RL!

Supervised policy learning – behavioral cloning

- Assume examples of policy are given in form of (x,u) pairs. Assume independent examples.
- How to fit a stochastic policy to these?

$$\pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t) \sim N(\boldsymbol{\theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_t), \sigma^2)$$ ← Example

- Maximum likelihood parameter estimation
  - Here: maximize probability of actions given states and parameters.

$$P(U|X;\theta) = \prod_t \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t)$$

How to proceed?

Example: Maximum likelihood estimation

- Maximize log-likelihood

$$P(U|X;\theta) = \prod_t \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t)$$

$$N(\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(u-\mu)^2}{\sqrt{2}\sigma}}$$

Example: Maximum likelihood estimation

- Maximize log-likelihood

$$P(U|X;\theta)=\prod_t \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t)$$

$$N(\mu,\sigma^2)=\frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(u-\mu)^2}{2\sigma}}$$

$$\log P(U|X;\theta)=\sum_t \log \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t)$$
$$\nabla \log P(U|X;\theta)=\sum_t \nabla \log \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t)$$

But we don't have examples!

What is a good policy?

- How to measure policy quality?

$$R(\boldsymbol{\theta}) = E\left[\sum_{t=0}^{T} \gamma^t r_t\right]$$

- More generally,

$$R(\boldsymbol{\theta}) = E\left[\sum_{t=0}^{T} a_t r_t\right]$$

Can also represent average reward per time step.

How to optimize parameters?

Policy gradient

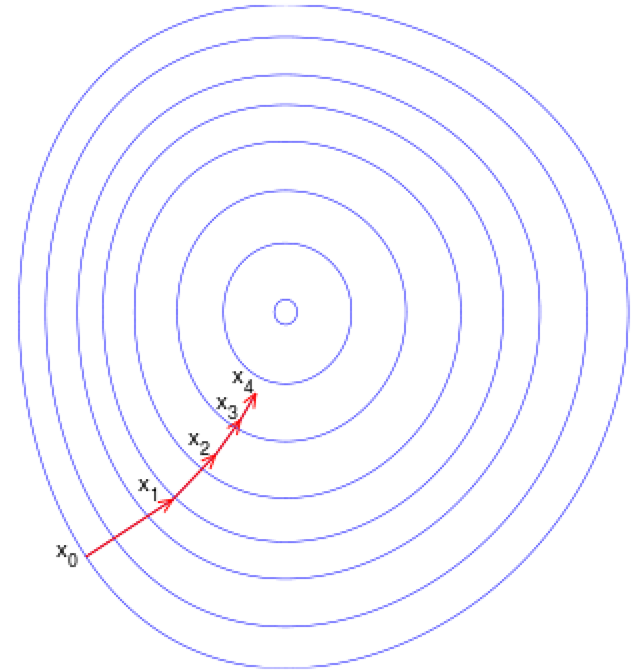
- Use gradient ascent on *R(θ)*.
- Update policy parameters by

$$\theta_{m+1} = \theta_m + \alpha_m \nabla_\theta R|_{\theta=\theta_m}$$

- How to calculate gradient?

$$R(\theta) = E\left[\sum_{t=0}^{T} a_t r_t\right]$$

Depends on θ.



$$\sum_{m=0}^{\infty} \alpha_m > 0 \quad \sum_{m=0}^{\infty} \alpha_m^2 < \infty$$

Guarantees convergence to local minimum.

How to estimate gradient from data (if we have a chance to try different policies)?

Finite difference gradient estimation

- What is gradient?
  - Vector of partial derivatives.
- How to estimate derivative?
  - Finite difference: $f'(x) \approx \frac{f(x+dx)-f(x)}{dx}$
- For policy gradient:
  - Generate variation $\Delta\boldsymbol{\theta}_i$
  - Estimate experimentally $R(\boldsymbol{\theta}+\Delta\boldsymbol{\theta}_i) \approx \hat{R}_i = \sum_{t=0}^{H} a_t r_t$
  - Compute gradient $\left[\boldsymbol{g}_{FD}^T, R_{ref}\right]^T = \left(\Delta\boldsymbol{\Theta}^T \Delta\boldsymbol{\Theta}\right)^{-1} \Delta\boldsymbol{\Theta}^T \hat{\boldsymbol{R}}$
  - Repeat until estimate converged

Not easy to choose.

$$\Delta\boldsymbol{\Theta}^T = \begin{bmatrix} \Delta\boldsymbol{\theta}_1, \ldots, \Delta\boldsymbol{\theta}_I \\ 1, \ldots, 1 \end{bmatrix}$$

$$\hat{\boldsymbol{R}}^T = [\hat{R}_1, \ldots, \hat{R}_I]$$

Where does this come from?

$$\hat{R}_i \approx R_{ref} + \boldsymbol{g}^T \Delta\boldsymbol{\theta}_i$$

Likelihood-ratio approach

- Assume trajectories tau are generated by roll-outs, thus

$$\boldsymbol{\tau} \sim p_{\theta}(\boldsymbol{\tau}) = p(\boldsymbol{\tau}|\boldsymbol{\theta}) \quad R(\boldsymbol{\tau}) = \sum_{t=0}^{H} a_t r_t$$

- Expected return can then be written

$$R(\boldsymbol{\theta}) = E_{\boldsymbol{\tau}}[R(\boldsymbol{\tau})] = \int p_{\theta}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau}$$

- Gradient is thus

$$\nabla_{\theta} R(\boldsymbol{\theta}) = \int \nabla_{\theta} p_{\theta}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau}$$

$$= \int p_{\theta}(\boldsymbol{\tau}) \nabla_{\theta} \log p_{\theta}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau}$$

← Likelihood ratio "trick": Substitute

$$\nabla_{\theta} p_{\theta}(\boldsymbol{\tau}) = p_{\theta}(\boldsymbol{\tau}) \nabla_{\theta} \log p_{\theta}(\tau)$$

- Why do that?

$$= E_{\boldsymbol{\tau}}[\nabla_{\theta} \log p_{\theta}(\boldsymbol{\tau}) R(\boldsymbol{\tau})]$$

$$p_{\theta}(\boldsymbol{\tau}) = p(\boldsymbol{x}_0) \prod_{t=0}^{H} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t) \pi_{\theta}(\boldsymbol{u}_t|\boldsymbol{x}_t)$$

Try substitution for log-gradient! $\nabla_{\theta} \log p_{\theta}(\boldsymbol{\tau}) = \sum_{t=0}^{H} \nabla_{\theta} \log \pi_{\theta}(\boldsymbol{u}_t|\boldsymbol{x}_t)$

↑ We know this!

Example differentiable policies

Normalization constant missing.

Probability proportional to exponentiated linear combination of features.

- Soft-max policy

$$\pi_\theta(\boldsymbol{u}_t|\boldsymbol{x}_t) \propto e^{\theta^T \varphi(\boldsymbol{x}_t, \boldsymbol{u}_t)}$$

  - Log-policy (*score function*)

$$\nabla_\theta \log \pi_\theta(\boldsymbol{u}_t|\boldsymbol{x}_t) = \varphi(\boldsymbol{x}_t, \boldsymbol{u}_t) - E_{\pi_\theta}[\varphi(\boldsymbol{x}_t, \cdot)]$$

- Gaussian policy

$$\pi_\theta(u_t|\boldsymbol{x}_t) \sim N(\theta^T \varphi(\boldsymbol{x}_t), \sigma^2)$$

Mean is linear combination of features.

  - Log-policy

$$\nabla_\theta \log \pi_\theta(u_t|\boldsymbol{x}_t) = \frac{(u_t - \theta^T \varphi(\boldsymbol{x}_t))\varphi(\boldsymbol{x}_t)}{\sigma^2}$$

Can also be understood as linear policy plus exploration uncertainty

$$\pi_\theta(u_t|\boldsymbol{x}_t) = \theta^T \varphi(\boldsymbol{x}_t) + \epsilon \quad \epsilon \sim N(0, \sigma^2)$$

Example differentiable policies

Normalization constant missing.

- Discrete neural net policy

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t|\boldsymbol{x}_t) \propto e^{f_\theta(\boldsymbol{x}_t, \boldsymbol{u}_t)}$$

Probability proportional to exponentiated neural network output.

- Gaussian neural network policy

$$\pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t) \sim N(f_\theta(\boldsymbol{x}_t), \sigma^2)$$

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(u_t|\boldsymbol{x}_t) = \frac{(u_t - f_\theta(\boldsymbol{x}_t)) \nabla_\theta f_\theta(\boldsymbol{x}_t)}{\sigma^2}$$

Aalto University
School of Electrical
Engineering

OK, now to applying the policy gradient:

$$\nabla_{\boldsymbol{\theta}} R(\boldsymbol{\theta}) = E_{\boldsymbol{\tau}}[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau})]$$

MC policy gradient – REINFORCE

- Episodic version shown here.
- Approach:
  - Perform episode *J (=1,2,3,...)*.
  - Estimate gradient $g_{RE} = E_{\tau}\left[\left(\sum_{t=0}^{H} \nabla_{\theta} \log \pi_{\theta}(\boldsymbol{u}_t | \boldsymbol{x}_t)\right) R(i)\right]$

$$\approx \frac{1}{J} \sum_{i=1}^{J} \left[\left(\sum_{t=0}^{H} \nabla_{\theta} \log \pi_{\theta}(\boldsymbol{u}_t^{[i]} | \boldsymbol{x}_t^{[i]})\right)\left(\sum_t r_{t,i}\right)\right]$$

Reward for trial *i*.

Use empirical mean.

  - Repeat with new trial(s) until convergence.
- No need to generate policy variations because of stochastic policy.

Limitations so far

- High variance in gradient estimate because of stochastic policy.
- Slow convergence, hard to choose learning rate.
  - Parametrization dependent gradient estimate.
- On-policy method.

Decreasing variance by adding baseline

- Constant baseline can be added to reduce *variance* of gradient estimate.

$$\nabla_{\boldsymbol{\theta}} R(\boldsymbol{\theta}) = E_{\boldsymbol{\tau}}\left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{\tau})(R(\boldsymbol{\tau}) - b)\right]$$
$$= E_{\boldsymbol{\tau}}\left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau})\right]$$

- Does not cause bias because

$$E_{\boldsymbol{\tau}}\left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) b\right] = \int \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) b \, d\boldsymbol{\tau} = b \nabla_{\boldsymbol{\theta}} \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) d\boldsymbol{\tau} = b \nabla_{\boldsymbol{\theta}} 1 = 0$$

Intuition:
Modifying rewards by a constant
does not change optimal policy.

Episodic REINFORCE with optimal baseline

- Optimal baseline for episodic REINFORCE (minimize variance of estimator):

$$b_h = \frac{E_{\boldsymbol{\tau}}\left[\left(\sum_{t=0}^{H} \nabla_{\theta_h} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t)\right)^2 R_{\boldsymbol{\tau}}\right]}{E_{\boldsymbol{\tau}}\left[\left(\sum_{t=0}^{H} \nabla_{\theta_h} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t)\right)\right]^2}$$

In practice, approximate by empirical mean (average over trials).

- Approach:
  - Perform trial *J (=1,2,3,...).*
  - For each gradient element *h*
    - Estimate optimal baseline b_h
    - Estimate gradient
  - Repeat until convergence.

Componentwise!

$$g_h = \frac{1}{J} \sum_{i=1}^{J} \left[\left(\sum_{t=0}^{H} \nabla_{\theta_h} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t^{[i]} | \boldsymbol{x}_t^{[i]})\right)(R(i) - b_h^{[i]})\right]$$

Even with optimal baseline, variance can be an issue.

Policy gradient theorem

- Observation: Future actions do not depend on past rewards.

$$E\left[\nabla_{\theta}\log\pi_{\theta}(\boldsymbol{u}_t|\boldsymbol{x}_t)r_k\right]=0 \quad \forall\, t>k$$

"don't take into account past rewards when evaluating the effect of an action" (causality, taking an action can only affect future rewards)

- PGT:
  - Reduces variance of estimate → Fewer samples needed on average.

$$\boldsymbol{g}_{PGT}=E_{\boldsymbol{\tau}}\left[\sum_{k=0}^{H}\left(\sum_{t=0}^{k}\nabla_{\theta_h}\log\pi_{\theta}(\boldsymbol{u}_t|\boldsymbol{x}_t)\right)(a_k r_k-b_k^h)\right]$$

Note: If only rewards at final time step, this is equivalent to REINFORCE.

Off-policy policy gradient

- What if we have samples from another policy (e.g. earlier timesteps?

  Optimize $E_{\tau \sim \pi_\theta(\tau)}[R(\tau)]$

  using samples from $\pi'(\tau)$

- Use importance sampling!

$$E_{x \sim p(x)}[f(x)] = \int p(x) f(x) dx$$

$$= E_{x \sim q(x)}\left[\frac{p(x)}{q(x)} f(x)\right]$$

Where does this come from?

Off-policy policy gradient

- What if we have samples from another policy (e.g. earlier timesteps?

  Optimize $E_{\tau \sim \pi_\theta(\tau)}[R(\tau)]$

  using samples from $\pi'(\tau)$

- Use importance sampling!

$$E_{x \sim p(x)}[f(x)] = \int p(x) f(x)\,dx$$

$$= E_{x \sim q(x)}\left[\frac{p(x)}{q(x)} f(x)\right]$$

Where does this come from?

Thus, optimize $E_{\tau \sim \pi'(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi'(\tau)} R(\tau)\right]$

Off-policy policy gradient

$$E_{\tau \sim \pi'(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi'(\tau)} R(\tau)\right]$$

- We had earlier

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = p(\boldsymbol{x}_0) \prod_{t=0}^{H} p(\boldsymbol{x}_{t+1} | \boldsymbol{x}_t, \boldsymbol{u}_t) \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t)$$

- Thus

$$\frac{\pi_\theta(\tau)}{\pi'(\tau)} = \frac{p(\boldsymbol{x}_0) \prod_{t=0}^{H} p(\boldsymbol{x}_{t+1} | \boldsymbol{x}_t, \boldsymbol{u}_t) \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t)}{p(\boldsymbol{x}_0) \prod_{t=0}^{H} p(\boldsymbol{x}_{t+1} | \boldsymbol{x}_t, \boldsymbol{u}_t) \pi'(\boldsymbol{u}_t | \boldsymbol{x}_t)} = \frac{\prod_{t=0}^{H} \pi_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t)}{\prod_{t=0}^{H} \pi'(\boldsymbol{u}_t | \boldsymbol{x}_t)}$$

Off-policy policy gradient

- Now the gradient

$$\nabla_\theta E_{\tau \sim \pi'(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi'(\tau)} R(\tau)\right] = E_{\tau \sim \pi'(\tau)}\left[\frac{\nabla_\theta \pi_\theta(\tau)}{\pi'(\tau)} R(\tau)\right]$$

$$= E_{\tau \sim \pi'(\tau)}\left[\frac{\pi_\theta(\tau)}{\pi'(\tau)} \nabla_\theta \log \pi_\theta(\tau) R(\tau)\right]$$

$$= E_{\tau \sim \pi'(\tau)}\left[\left(\prod_t \frac{\pi_\theta(\tau)}{\pi'(\tau)}\right)\left(\sum_t \nabla_\theta \log \pi_\theta(u_t|x_t)\right)\left(\sum_t r_t\right)\right]$$

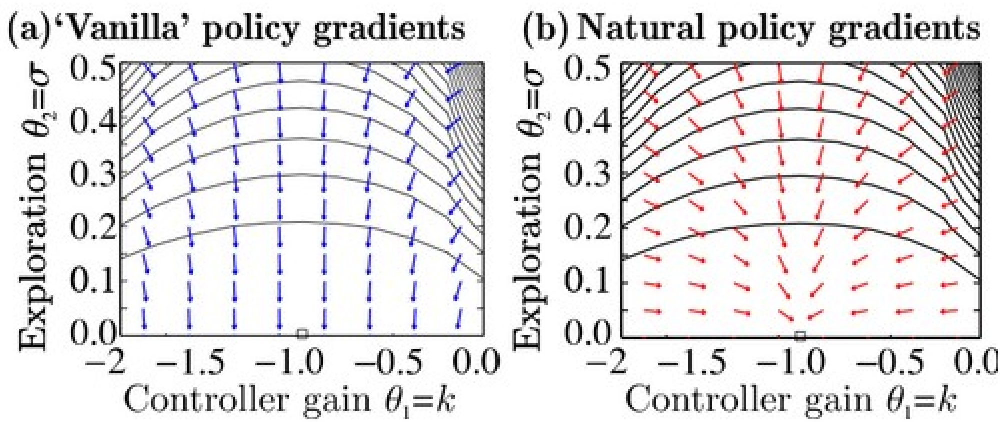
Compare to on-policy (REINFORCE)

$$\nabla_\theta E_{\tau \sim \pi_\theta(\tau)}[R(\tau)] = E_{\tau \sim \pi_\theta(\tau)}\left[\left(\sum_t \nabla_\theta \log \pi_\theta(u_t|x_t)\right)\left(\sum_t r_t\right)\right]$$

Will be used later!

Gradient vs natural gradient

- Gradient depends on parametrization.
- Natural gradient parametrization independent.



$$\nabla_\theta^{NG} \pi_\theta(u|\boldsymbol{x}) = \boldsymbol{F}_\theta^{-1} \nabla_\theta \pi_\theta(u|\boldsymbol{x})$$

Intuition: Divide gradient update by second derivative.

Normalizes parameter influence.

- Fisher information matrix

$$\boldsymbol{F}_\theta = E\left[\nabla_\theta \log \pi_\theta(u|\boldsymbol{x}) \nabla_\theta \log \pi_\theta(u|\boldsymbol{x})^T\right]$$

Potentially improves convergence significantly,
in practice sample-based approximation less useful.

Summary

- Policy gradient methods can be used for stochastic policies and continuous action spaces.
- Finite-difference approaches approximate gradient by policy adjustments.
- Likelihood ratio-approaches calculate gradient through known policy.
- Policy gradient often requires very many updates because of noisy gradient and small update steps.

Next: Actor-critic approaches

- Can we combine policy learning with value-based methods?