POLITECNICO
MILANO 1863

Beyond Maximum Likelihood Model Estimation in Model-based Policy Search

Pierluca D'Oro

Supervisor: Prof. Marcello Restelli
Assistant Supervisors: Dott. Metelli, Dott. Tirinzoni, Dott. Papini

Table of Contents

Reinforcement Learning

Reinforcement Learning

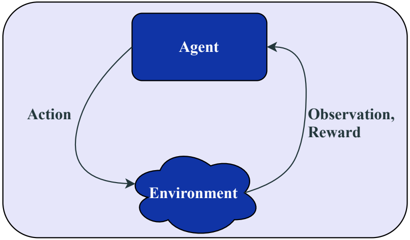
Overview

Solving Sequential Decision Making Problems

Sequential decision making is a core capability of intelligent agents.

Reinforcement Learning (RL) studies how an agent can learn to interact with an environment, guided by a reinforcement signal he wants to maximize.

No knowledge of the environment dynamics is assumed.



Markov Decision Processes

A Markov Decision Process (MDP) [Puterman, 2014] is described by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, p, \mu, \gamma)$, where:

- $\mathcal{S}$ is the space of possible states
- $\mathcal{A}$ is the space of possible actions
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function
- $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the transition model
- $\mu : \mathcal{S} \to \mathbb{R}$ is the distribution of the initial state
- $\gamma \in [0, 1)$ is a discount factor

We assume r is known and uniformly bounded by $|r(s, a)| \leq R_{\max} < +\infty$. The behavior of an agent is described by a policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

Value Functions and the RL objective

Given a state-action pair (s, a) we define the action-value function [Sutton and Barto, 2018], or Q-function, by using the dynamic-programming-based Bellman equation:

$$Q^{\pi,p}(s,a) = r(s,a) + \gamma \int_{\mathcal{S}} p(s'|s,a) \int_{\mathcal{A}} \pi(a'|s')\, Q^{\pi,p}(s',a') \mathrm{d}s' \mathrm{d}a'$$

Value Functions and the RL objective

Given a state-action pair (s, a) we define the action-value function [Sutton and Barto, 2018], or Q-function, by using the dynamic-programming-based Bellman equation:

$$Q^{\pi,p}(s, a) = r(s, a) + \gamma \int_{\mathcal{S}} p(s'|s, a) \int_{\mathcal{A}} \pi(a'|s') Q^{\pi,p}(s', a') \mathrm{d}s' \mathrm{d}a'$$

and the state-value function, or V-function, as:

$$V^{\pi,p}(s) = \mathop{\mathbb{E}}_{a \sim \pi(\cdot|s)} [Q^{\pi,p}(s, a)].$$

Value Functions and the RL objective

Given a state-action pair (s, a) we define the action-value function [Sutton and Barto, 2018], or Q-function, by using the dynamic-programming-based Bellman equation:

$$Q^{\pi,p}(s,a) = r(s,a) + \gamma \int_{\mathcal{S}} p(s'|s,a) \int_{\mathcal{A}} \pi(a'|s') Q^{\pi,p}(s',a') \mathrm{d}s' \mathrm{d}a'$$

and the state-value function, or V-function, as:

$$V^{\pi,p}(s) = \mathop{\mathbb{E}}_{a \sim \pi(\cdot|s)} [Q^{\pi,p}(s,a)].$$

The goal of the agent is to find an optimal policy π^*, maximizing the expected return:

$$J^{\pi,p} = \mathbb{E}_{s_0 \sim \mu} \left[V^{\pi,p}(s_0) \right], \quad \pi^* = \arg\max_{\pi} J^{\pi,p}$$

Successes in Model-free RL

Most RL approaches do not explicitly learn about the transition model p.

Biggest successes in model-free RL in the last years were in games. For instance, super-human performance was reached in:

- ATARI games [Mnih et al., 2015]
- Go [Silver et al., 2016, Silver et al., 2017]
- Starcraft [Vinyals et al., 2019]

Shared traits: very efficient simulator available, no safety, transfer or data efficiency concerns.

In others words, not easy to adapt these methods to the real world.

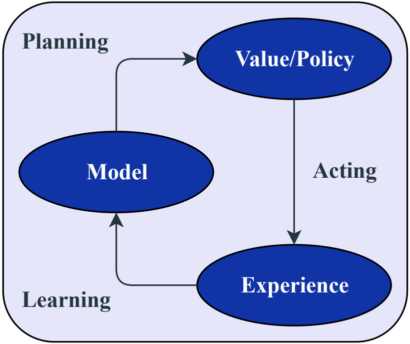
Reinforcement Learning

Model-based Reinforcement Learning

The model-based loop

Model-based Reinforcement Learning (MBRL) uses estimated models of the dynamics of the environment to learn a policy. Synonyms: world model, forward model, (just) model.

The policy is obtained by planning with the learned model.



Motivation

The main advantages of MBRL can be summarized as:

Motivation

The main advantages of MBRL can be summarized as:

- Sample-efficiency

Motivation

The main advantages of MBRL can be summarized as:

- Sample-efficiency
- Easier transfer

Motivation

The main advantages of MBRL can be summarized as:

- Sample-efficiency
- Easier transfer
- More effective exploration

Motivation

The main advantages of MBRL can be summarized as:

- Sample-efficiency
- Easier transfer
- More effective exploration
- Safety

The three key questions in MBRL

The three key questions in MBRL

- Which model class to use?

The three key questions in MBRL

- Which model class to use?
- How to learn the model?

The three key questions in MBRL

- Which model class to use?
- How to learn the model?
- How to use the learned model?

The three key questions in MBRL

- Which model class to use?
- How to learn the model?
- How to use the learned model?

How to learn the model?

Different choices can be made on the loss function (e.g., depending on the type of generative model).

On a higher level, two approaches can be taken for model learning:

How to learn the model?

Different choices can be made on the loss function (e.g., depending on the type of generative model).

On a higher level, two approaches can be taken for model learning:

- Task-agnostic - Maximum Likelihood. Assume no prior knowledge. Commonly used (e.g., by MSE minimization on observed transitions).

How to learn the model?

Different choices can be made on the loss function (e.g., depending on the type of generative model).

On a higher level, two approaches can be taken for model learning:

- Task-agnostic - Maximum Likelihood. Assume no prior knowledge. Commonly used (e.g., by MSE minimization on observed transitions).
- Decision-aware. Leverage knowledge about task, policy or learning algorithm to decide which one of the observed transitions are more important. E.g. Minimize error on Bellman operator in value-based methods [Farahmand et al., 2017]:

$$c\left(\hat{p}, p; Q\right)(s, a) = \left| \int \left[p\left(s'|s, a\right) - \hat{p}\left(s'|s, a\right)\right] \max_{a'} Q(s', a') \mathrm{d}s' \right|$$

How to learn the model?

Different choices can be made on the loss function (e.g., depending on the type of generative model).

On a higher level, two approaches can be taken for model learning:

- Task-agnostic - Maximum Likelihood. Assume no prior knowledge. Commonly used (e.g., by MSE minimization on observed transitions).
- Decision-aware. Leverage knowledge about task, policy or learning algorithm to decide which one of the observed transitions are more important. E.g. Minimize error on Bellman operator in value-based methods [Farahmand et al., 2017]:

$$c\left(\hat{p}, p; Q\right)(s, a) = \left| \int \left[p\left(s' | s, a\right) - \hat{p}\left(s' | s, a\right) \right] \max_{a'} Q(s', a') \mathrm{d}s' \right|$$

RESEARCH GOAL: learn the model of the dynamics that is optimal for improving a policy by using its gradient.

Policy Gradients

Policy Gradients

Overview

A gradient ascent approach to policy search

Policy Gradient methods are among the most popular policy-based RL methods:

A gradient ascent approach to policy search

Policy Gradient methods are among the most popular policy-based RL methods:

- We consider $\pi_{\boldsymbol{\theta}} \in \Pi_\Theta$, with Π_Θ a parametric space of stochastic and differentiable policies

A gradient ascent approach to policy search

Policy Gradient methods are among the most popular policy-based RL methods:

- We consider $\pi_{\boldsymbol{\theta}} \in \Pi_\Theta$, with Π_Θ a parametric space of stochastic and differentiable policies
- The performance on the task J is therefore a function of $\boldsymbol{\theta}$

A gradient ascent approach to policy search

Policy Gradient methods are among the most popular policy-based RL methods:

- We consider $\pi_{\boldsymbol{\theta}} \in \Pi_\Theta$, with Π_Θ a parametric space of stochastic and differentiable policies
- The performance on the task J is therefore a function of $\boldsymbol{\theta}$
- We can differentiate it w.r.t. the parameters of $\pi_{\boldsymbol{\theta}}$

A gradient ascent approach to policy search

Policy Gradient methods are among the most popular policy-based RL methods:

- We consider $\pi_{\boldsymbol{\theta}} \in \Pi_\Theta$, with Π_Θ a parametric space of stochastic and differentiable policies
- The performance on the task J is therefore a function of $\boldsymbol{\theta}$
- We can differentiate it w.r.t. the parameters of $\pi_{\boldsymbol{\theta}}$
- Then, policy can be improved with an update of the kind:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where α is a small step-size (a.k.a. learning rate)

Policy Gradients

A taxonomy of Policy Gradients

Model-Free Gradient

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic and differentiable policies, $\pi \in \Pi_\Theta$.

Definition (Model-Free Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

This is just a renaming of the formulation provided by the Policy Gradient Theorem [Sutton et al., 2000]. Three elements into play:

Model-Free Gradient

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic and differentiable policies, $\pi \in \Pi_\Theta$.

Definition (Model-Free Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

This is just a renaming of the formulation provided by the Policy Gradient Theorem [Sutton et al., 2000]. Three elements into play:

- $\delta_{\mu}^{\pi,p}(s,a)$, the density function of the state-action distribution

Model-Free Gradient

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic and differentiable policies, $\pi \in \Pi_\Theta$.

Definition (Model-Free Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

This is just a renaming of the formulation provided by the Policy Gradient Theorem [Sutton et al., 2000]. Three elements into play:

- $\delta_{\mu}^{\pi,p}(s,a)$, the density function of the state-action distribution
- $\nabla_{\boldsymbol{\theta}} \log \pi(a|s)$, the score, linked to the chance of improvement

Model-Free Gradient

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic and differentiable policies, $\pi \in \Pi_\Theta$.

Definition (Model-Free Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_\mu^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

This is just a renaming of the formulation provided by the Policy Gradient Theorem [Sutton et al., 2000]. Three elements into play:

- $\delta_\mu^{\pi,p}(s,a)$, the density function of the state-action distribution
- $\nabla_{\boldsymbol{\theta}} \log \pi(a|s)$, the score, linked to the chance of improvement
- $Q^{\pi,p}(s,a)$, the value of the state-action couple

Model-Free Gradient

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic and differentiable policies, $\pi \in \Pi_\Theta$.

Definition (Model-Free Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

This is just a renaming of the formulation provided by the Policy Gradient Theorem [Sutton et al., 2000]. Three elements into play:

- $\delta_{\mu}^{\pi,p}(s,a)$, the density function of the state-action distribution
- $\nabla_{\boldsymbol{\theta}} \log \pi(a|s)$, the score, linked to the chance of improvement
- $Q^{\pi,p}(s,a)$, the value of the state-action couple

Unbiased estimators are available.

Fully Model-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Fully Model-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\text{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

As the MFG, but in an imagination MDP in which the true model p is substituted with $\hat{p}$. Advantages:

Fully Model-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Fully Model-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

As the MFG, but in an imagination MDP in which the true model p is substituted with $\hat{p}$. Advantages:

- Cheap to estimate: no environment interaction required

Fully Model-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Fully Model-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

As the MFG, but in an imagination MDP in which the true model p is substituted with $\hat{p}$. Advantages:

- Cheap to estimate: no environment interaction required
- Flexible: any model free algorithm can be adapted, plus others

Fully Model-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Fully Model-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

As the MFG, but in an imagination MDP in which the true model p is substituted with $\hat{p}$. Advantages:

- Cheap to estimate: no environment interaction required
- Flexible: any model free algorithm can be adapted, plus others
- It can reduce variance at the cost of the bias introduced by an estimated model

Model-Value-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Model-Value-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

It uses trajectories from the real environment model p, but solves credit assignment by using the estimated model $\hat{p}$. Examples: SVG(∞) [Heess et al., 2015], MVE [Feinberg et al., 2018].
Advantages:

Model-Value-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Model-Value-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

It uses trajectories from the real environment model p, but solves credit assignment by using the estimated model $\hat{p}$. Examples: SVG(∞) [Heess et al., 2015], MVE [Feinberg et al., 2018].
Advantages:

- Grounded Gradient: real trajectories avoid catastrophic errors

Model-Value-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Model-Value-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\text{MVG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

It uses trajectories from the real environment model p, but solves credit assignment by using the estimated model $\hat{p}$. Examples: SVG(∞) [Heess et al., 2015], MVE [Feinberg et al., 2018].

Advantages:

- Grounded Gradient: real trajectories avoid catastrophic errors
- Still model-based: many model-based advantages are retained

Model-Value-based Gradient

Let Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Definition (Model-Value-based Gradient)

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

It uses trajectories from the real environment model p, but solves credit assignment by using the estimated model $\hat{p}$. Examples: SVG(∞) [Heess et al., 2015], MVE [Feinberg et al., 2018].
Advantages:

- Grounded Gradient: real trajectories avoid catastrophic errors
- Still model-based: many model-based advantages are retained
- A compromise in bias and variance w.r.t. MFG and FMG

Gradients Recap

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Gradients Recap

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Model-Free Gradient $\Rightarrow$ High Variance

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_\mu^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

Gradients Recap

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Model-Free Gradient $\Rightarrow$ High Variance

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s \mathrm{d}a.$$

Fully Model-based Gradient $\Rightarrow$ High Bias

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s \mathrm{d}a.$$

Gradients Recap

Let p be the transition model of an MDP, Π_Θ a parametric space of stochastic policies, $\mathcal{P}$ a class of transition models, $\pi \in \Pi_\Theta$ and $\widehat{p} \in \mathcal{P}$.

Model-Free Gradient $\Rightarrow$ High Variance

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MFG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,p}(s,a) \mathrm{d}s\mathrm{d}a.$$

Fully Model-based Gradient $\Rightarrow$ High Bias

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{FMG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,\widehat{p}}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s\mathrm{d}a.$$

Model-Value-based Gradient $\Rightarrow$ Compromise on bias/variance

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta}) = \frac{1}{1-\gamma} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_{\mu}^{\pi,p}(s,a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s) Q^{\pi,\widehat{p}}(s,a) \mathrm{d}s\mathrm{d}a.$$

Gradient-Aware Model-based Policy Search

Gradient-Aware Model-based Policy Search

Analysis of the Model-Value-based Gradient

An upper bound on the MVG bias

Proposition

Let $q \in [1, +\infty]$ and $\widehat{p} \in \mathcal{P}$. If $\|\nabla_{\boldsymbol{\theta}} \log \pi(a|s)\|_q \leq K$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, then, the L^q-norm of the difference between the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ and the corresponding MVG $\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})$ can be upper bounded as:

$$\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})\|_q \leq c_1 \sqrt{\mathbb{E}_{s,a \sim \delta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \widehat{p}(\cdot|s,a)) \right]}.$$

This proposition justifies maximum likelihood model estimation for MVG-based approaches.

The importance of the error on state-action couple (s, a) only depends upon its visitation $\delta_\mu^{\pi,p}(s, a)$.

An upper bound on the MVG bias

Proposition

Let $q \in [1, +\infty]$ and $\widehat{p} \in \mathcal{P}$. If $\|\nabla_{\boldsymbol{\theta}} \log \pi(a|s)\|_q \leq K$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, then, the L^q-norm of the difference between the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ and the corresponding MVG $\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})$ can be upper bounded as:

$$\begin{aligned}\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})\|_q &\leq c_2 \sqrt{\mathbb{E}_{s,a \sim \eta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \widehat{p}(\cdot|s,a)) \right]} \\ &\leq c_1 \sqrt{\mathbb{E}_{s,a \sim \delta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \widehat{p}(\cdot|s,a)) \right]}.\end{aligned}$$

An upper bound on the MVG bias

Proposition

Let $q \in [1, +\infty]$ and $\widehat{p} \in \mathcal{P}$. If $\|\nabla_{\boldsymbol{\theta}} \log \pi(a|s)\|_q \leq K$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, then, the L^q-norm of the difference between the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ and the corresponding MVG $\nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})$ can be upper bounded as:

$$\begin{aligned} \|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}^{\mathrm{MVG}} J(\boldsymbol{\theta})\|_q &\leq c_2 \sqrt{\mathbb{E}_{s,a \sim \eta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \widehat{p}(\cdot|s,a)) \right]} \\ &\leq c_1 \sqrt{\mathbb{E}_{s,a \sim \delta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \widehat{p}(\cdot|s,a)) \right]}. \end{aligned}$$

The distribution under which model mismatch is measured becomes:

$$\eta_\mu^{\pi,p}(s,a) = \frac{1}{Z} \int_{\mathcal{S}} \int_{\mathcal{A}} \delta_\mu^{\pi,p}(s',a') \left\| \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a'|s') \right\|_q \delta_{s',a'}^{\pi,p}(s,a) \mathrm{d}s' \mathrm{d}a'$$

A decision-aware weighting factor leads to a better bound.

Gradient-Aware Model-based Policy Search

The Algorithm

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based
- Gradient-aware

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based
- Gradient-aware

Gradient-Aware Model-based Policy Search (GAMPS) has 3 steps:

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based
- Gradient-aware

Gradient-Aware Model-based Policy Search (GAMPS) has 3 steps:

1. Learning the model

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based
- Gradient-aware

Gradient-Aware Model-based Policy Search (GAMPS) has 3 steps:

1. Learning the model
2. Computing the value function

Overview

We distill the intuition behind the previous Proposition into a policy search algorithm with the following features:

- Batch (no further environment interaction)
- MVG-based
- Gradient-aware

Gradient-Aware Model-based Policy Search (GAMPS) has 3 steps:

1. Learning the model
2. Computing the value function
3. Estimating the policy gradient

Learning the model

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p}\left(s_{t+1}^i | s_t^i, a_t^i\right)$$

Learning the model

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p} \left(s_{t+1}^i | s_t^i, a_t^i \right)$$

Gradient-Aware Weights

$$\omega_t^i = \gamma^t \rho_{\pi/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \left\| \nabla_{\boldsymbol{\theta}} \log \pi(a_l^i | s_l^i) \right\|_q$$

Learning the model

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p} \left(s_{t+1}^i | s_t^i, a_t^i \right)$$

Important transitions:

Gradient-Aware Weights

$$\omega_t^i = \gamma^t \rho_{\pi/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \left\| \nabla_{\boldsymbol{\theta}} \log \pi(a_l^i | s_l^i) \right\|_q$$

Learning the model

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p}\left(s_{t+1}^i | s_t^i, a_t^i\right)$$

Important transitions:

- Early transitions

Gradient-Aware Weights

$$\omega_t^i = \gamma^t \rho_{\pi/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \left\| \nabla_{\boldsymbol{\theta}} \log \pi(a_l^i | s_l^i) \right\|_q$$

Learning the model

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p} \left(s_{t+1}^i | s_t^i, a_t^i \right)$$

Gradient-Aware Weights

$$\omega_t^i = \gamma^t \rho_{\pi/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \left\| \nabla_{\boldsymbol{\theta}} \log \pi(a_l^i | s_l^i) \right\|_q$$

Important transitions:

- Early transitions
- Likely transitions under π

Gradient-Aware loss function

$$\widehat{p} = \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega_t^i \log \overline{p}\left(s_{t+1}^i | s_t^i, a_t^i\right)$$

Gradient-Aware Weights

$$\omega_t^i = \gamma^t \rho_{\pi/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \left\| \nabla_{\boldsymbol{\theta}} \log \pi(a_l^i | s_l^i) \right\|_q$$

Important transitions:

- Early transitions
- Likely transitions under π
- Transitions at the end of trajectories with large cumulative score magnitude

Computing the value function

To compute $Q^{\pi,\hat{p}}$, we should carry out policy evaluation.

We propose the following Monte-Carlo approach for continuous environments.

Evaluation via Monte-Carlo Imagination Rollouts

$$\widehat{Q}(s,a) = \frac{1}{M}\sum_{j=1}^{M}\sum_{t=0}^{T_j-1}\gamma^t r(s_t^j, a_t^j), \quad \tau^j \sim \zeta_{s,a}^{\pi,\widehat{p}}.$$

Computing the value function

To compute $Q^{\pi,\hat{p}}$, we should carry out policy evaluation.

We propose the following Monte-Carlo approach for continuous environments.

Evaluation via Monte-Carlo Imagination Rollouts

$$\widehat{Q}(s,a) = \frac{1}{M}\sum_{j=1}^{M}\sum_{t=0}^{T_j-1}\gamma^t r(s_t^j, a_t^j), \quad \tau^j \sim \zeta_{s,a}^{\pi,\widehat{p}}.$$

Advantages in avoiding explicit Q-function approximation:

Computing the value function

To compute $Q^{\pi,\hat{p}}$, we should carry out policy evaluation.

We propose the following Monte-Carlo approach for continuous environments.

Evaluation via Monte-Carlo Imagination Rollouts

$$\widehat{Q}(s,a) = \frac{1}{M}\sum_{j=1}^{M}\sum_{t=0}^{T_j-1}\gamma^t r(s_t^j, a_t^j), \quad \tau^j \sim \zeta_{s,a}^{\pi,\widehat{p}}.$$

Advantages in avoiding explicit Q-function approximation:

- No regression targets

Computing the value function

To compute $Q^{\pi,\hat{p}}$, we should carry out policy evaluation.

We propose the following Monte-Carlo approach for continuous environments.

Evaluation via Monte-Carlo Imagination Rollouts

$$\widehat{Q}(s,a) = \frac{1}{M}\sum_{j=1}^{M}\sum_{t=0}^{T_j-1}\gamma^t r(s_t^j, a_t^j), \quad \tau^j \sim \zeta_{s,a}^{\pi,\widehat{p}}.$$

Advantages in avoiding explicit Q-function approximation:

- No regression targets
- No choice for model class

Estimating the policy gradient

Policy Gradient estimator

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \gamma^t \rho_{\pi/\pi_b}(\tau^i_{0:t}) \nabla_{\boldsymbol{\theta}} \log \pi(a^i_t | s^i_t) \widehat{Q}(s^i_t, a^i_t).$$

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega_{t,k}^i \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^t \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_l^i|s_l^i)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p}\in\mathcal{P}} \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T_i-1} \omega_{t,k}^i \log \overline{p}(s_{t+1}^i|s_t^i, a_t^i)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^M \sum_{t=0}^{T_j-1} \gamma^t r(s_t^j, a_t^j)$

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \times \\ \times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_t^i|s_t^i) \widehat{Q}_k(s_t^i, a_t^i)$$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega^i_{t,k} \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \sum_{l=0}^{t} \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_l | s^i_l)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \omega^i_{t,k} \log \overline{p}(s^i_{t+1} | s^i_t, a^i_t)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{T_j - 1} \gamma^t r(s^j_t, a^j_t)$

$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \times$

$\times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_t | s^i_t) \widehat{Q}_k(s^i_t, a^i_t)$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega^i_{t,k} \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \sum_{l=0}^{t} \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_l|s^i_l)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p}\in\mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \omega^i_{t,k} \log \overline{p}(s^i_{t+1}|s^i_t, a^i_t)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{T_j-1} \gamma^t r(s^j_t, a^j_t)$

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \times \times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_t|s^i_t) \widehat{Q}_k(s^i_t, a^i_t)$$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega_{t,k}^i \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^t \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_l^i|s_l^i)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T_i-1} \omega_{t,k}^i \log \overline{p}(s_{t+1}^i|s_t^i, a_t^i)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^M \sum_{t=0}^{T_j-1} \gamma^t r(s_t^j, a_t^j)$

$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \times$
$\times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_t^i|s_t^i) \widehat{Q}_k(s_t^i, a_t^i)$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega^i_{t,k} \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \sum_{l=0}^{t} \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_l|s^i_l)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \omega^i_{t,k} \log \overline{p}(s^i_{t+1}|s^i_t, a^i_t)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{T_j-1} \gamma^t r(s^j_t, a^j_t)$

$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \times$
$\times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_t|s^i_t) \widehat{Q}_k(s^i_t, a^i_t)$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Putting all together

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega^i_{t,k} \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \sum_{l=0}^{t} \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_l|s^i_l)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p}\in\mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \omega^i_{t,k} \log \overline{p}(s^i_{t+1}|s^i_t, a^i_t)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{T_j-1} \gamma^t r(s^j_t, a^j_t)$

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau^i_{0:t}) \times \times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a^i_t|s^i_t) \widehat{Q}_k(s^i_t, a^i_t)$$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Gradient-Aware Model-based Policy Search

Input: Trajectory dataset $\mathcal{D}$, behavior policy π_b, initial parameters $\boldsymbol{\theta}_0$

for $k = 0, 1, ..., K-1$ do

$\omega_{t,k}^i \leftarrow \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \sum_{l=0}^{t} \|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_l^i | s_l^i)\|_q$

$\widehat{p}_k \leftarrow \arg\max_{\overline{p} \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \omega_{t,k}^i \log \overline{p}(s_{t+1}^i | s_t^i, a_t^i)$

Generate a dataset of M trajectories for each (s, a) simulating $\widehat{p}_k$

$\widehat{Q}_k(s, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{T_j-1} \gamma^t r(s_t^j, a_t^j)$

$$\widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i-1} \gamma^t \rho_{\pi_{\boldsymbol{\theta}_k}/\pi_b}(\tau_{0:t}^i) \times \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_t^i | s_t^i) \widehat{Q}_k(s_t^i, a_t^i)$$

$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_k \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$

end for

Gradient-Aware Model-based Policy Search

Theoretical Analysis

Goal of the theoretical analysis

Learning theory analysis using tools from [Cortes et al., 2013].

Objectives:

Goal of the theoretical analysis

Learning theory analysis using tools from [Cortes et al., 2013].

Objectives:

- Highlight the important elements in approximation/estimation errors

Goal of the theoretical analysis

Learning theory analysis using tools from [Cortes et al., 2013].

Objectives:

- Highlight the important elements in approximation/estimation errors
- Show that choosing a simple model class can be wise

Goal of the theoretical analysis

Learning theory analysis using tools from [Cortes et al., 2013].

Objectives:

- Highlight the important elements in approximation/estimation errors
- Show that choosing a simple model class can be wise
- Justify the intuition behind gradient-aware MVG estimation

Finite-sample Bound

Theorem

For any $\delta \in (0,1)$, with probability at least $1-4\delta$ it holds that[a]:

$$\left\| \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \right\|_q \leq \underbrace{c_2 \inf_{\overline{p} \in \mathcal{P}} \sqrt{\mathop{\mathbb{E}}_{s,a \sim \eta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \overline{p}(\cdot|s,a)) \right]}}_{\text{approximation error}} + \underbrace{\mathcal{O}\left(\sqrt{\frac{v}{N}}\right)}_{\text{estimation error}}$$

[a] N: num. of trajectories, v: $\mathcal{P}$ pseudo-dimension, d: gradient dimensionality

Finite-sample Bound

Theorem

For any $\delta \in (0,1)$, with probability at least $1-4\delta$ it holds that[a]:

$$\left\| \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \right\|_q \le \underbrace{c_2 \inf_{\overline{p} \in \mathcal{P}} \sqrt{\mathop{\mathbb{E}}_{s,a \sim \eta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \overline{p}(\cdot|s,a)) \right]}}_{\text{approximation error}} + \underbrace{\mathcal{O}\left(\sqrt{\frac{v}{N}} \right)}_{\text{estimation error}}$$

[a] N: num. of trajectories, v: $\mathcal{P}$ pseudo-dimension, d: gradient dimensionality

Finite-sample Bound

Theorem

For any $\delta \in (0,1)$, with probability at least $1-4\delta$ it holds that[a]:

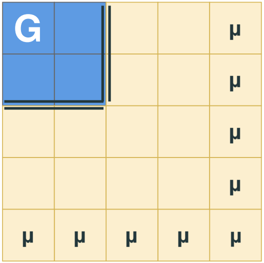
$$\left\| \widehat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \right\|_q \leq \underbrace{c_2 \inf_{\overline{p} \in \mathcal{P}} \sqrt{\mathop{\mathbb{E}}_{s,a \sim \eta_\mu^{\pi,p}} \left[D_{KL}(p(\cdot|s,a) \| \overline{p}(\cdot|s,a)) \right]}}_{\text{approximation error}} + \underbrace{\mathcal{O}\left(\sqrt{\frac{v}{N}} \right)}_{\text{estimation error}}$$

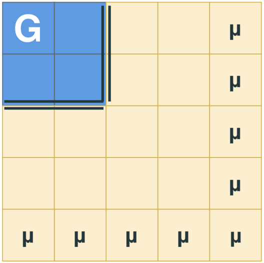
[a] N: num. of trajectories, v: $\mathcal{P}$ pseudo-dimension, d: gradient dimensionality

Gradient-Aware Model-based Policy Search

Experimental Analysis

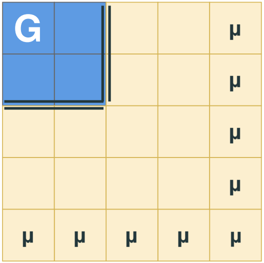
Two-areas Gridworld - Description





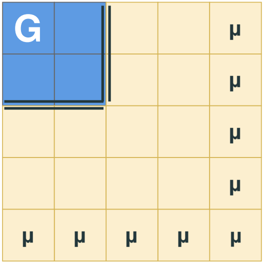
- Swapped action effect in two areas

Two-areas Gridworld - Description



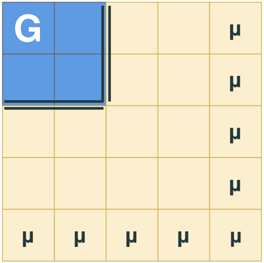
- Swapped action effect in two areas
- Initial states μ

Two-areas Gridworld - Description



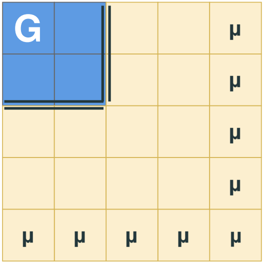
- Swapped action effect in two areas
- Initial states μ
- Reward of -1 everywhere but in G

Two-areas Gridworld - Description



- Swapped action effect in two areas
- Initial states μ
- Reward of -1 everywhere but in G
- One-way wall

Two-areas Gridworld - Description



- Swapped action effect in two areas
- Initial states μ
- Reward of -1 everywhere but in G
- One-way wall
- Batch setting (one time data collection)

Two-areas Gridworld - Policy and Model class

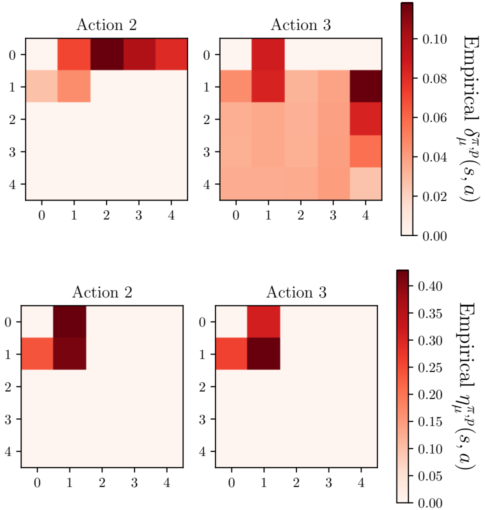
Policy

- Boltzmann (categorical) distribution over actions
- Deterministic in lower part, randomly initialized in upper

Model

- Linear in the sole action (equivalent to a lookup table)
- Able to represent only one of the two environment parts

Two-areas Gridworld - Difference in ML and GAMPS weights



Two-areas Gridworld - Gradient Approximation Experiment

Table 1: Estimation performance on the gridworld environment comparing Maximum Likelihood estimation (ML) and our approach (GAMPS). 1000 training and 1000 validation trajectories per run. Average results on 10 runs with a 95% confidence interval.

Approach	$\widehat{p}$ accuracy	$\widehat{Q}$ MSE	$\widehat{\nabla}_{\boldsymbol{\theta}} J$ cosine similarity
ML			
GAMPS			

Two-areas Gridworld - Gradient Approximation Experiment

Table 1: Estimation performance on the gridworld environment comparing Maximum Likelihood estimation (ML) and our approach (GAMPS). 1000 training and 1000 validation trajectories per run. Average results on 10 runs with a 95% confidence interval.

Approach	$\widehat{p}$ accuracy	$\widehat{Q}$ MSE	$\widehat{\nabla}_{\boldsymbol{\theta}} J$ cosine similarity
ML	0.765 ± 0.001		
GAMPS	0.357 ± 0.004		

Two-areas Gridworld - Gradient Approximation Experiment

Table 1: Estimation performance on the gridworld environment comparing Maximum Likelihood estimation (ML) and our approach (GAMPS). 1000 training and 1000 validation trajectories per run. Average results on 10 runs with a 95% confidence interval.

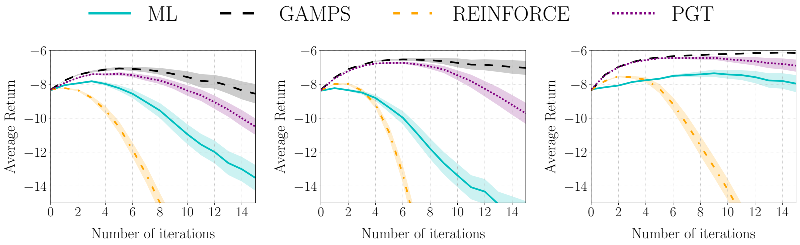
Approach	$\widehat{p}$ accuracy	$\widehat{Q}$ MSE	$\widehat{\nabla}_{\boldsymbol{\theta}} J$ cosine similarity
ML	0.765 ± 0.001	11.803 ± 0.158	
GAMPS	0.357 ± 0.004	633.835 ± 12.697	

Two-areas Gridworld - Gradient Approximation Experiment

Table 1: Estimation performance on the gridworld environment comparing Maximum Likelihood estimation (ML) and our approach (GAMPS). 1000 training and 1000 validation trajectories per run. Average results on 10 runs with a 95% confidence interval.

Approach	$\widehat{p}$ accuracy	$\widehat{Q}$ MSE	$\widehat{\nabla}_{\boldsymbol{\theta}} J$ cosine similarity
ML	0.765 ± 0.001	11.803 ± 0.158	0.449 ± 0.041
GAMPS	0.357 ± 0.004	633.835 ± 12.697	1.000 ± 0.000

Two-areas Gridworld - Policy Improvement Results



(a) 10 trajectories (b) 50 trajectories (c) 100 trajectories

Figure 1: Average return on the Two-areas gridworld with different dataset size. ML is the same as GAMPS but using maximum likelihood model estimation (20 runs, mean $\pm$ std).

Minigolf (Continuous states/actions) - Policy Improvement Results

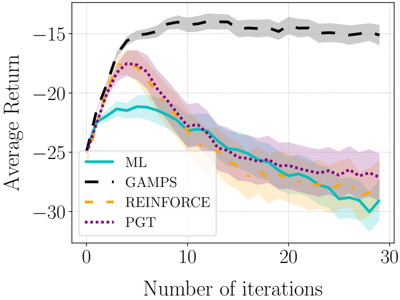


Figure 2: Average return using a 50 trajectories dataset on the minigolf environment (10 runs, mean ± std).

Conclusions and Future Work

Recap

- We analyzed the Model-Value-based Gradient

Recap

- We analyzed the Model-Value-based Gradient
- We showed that ML is suboptimal for model learning when computing the MVG

- We analyzed the Model-Value-based Gradient
- We showed that ML is suboptimal for model learning when computing the MVG
- We built the GAMPS algorithm based on this intuition

Recap

- We analyzed the Model-Value-based Gradient
- We showed that ML is suboptimal for model learning when computing the MVG
- We built the GAMPS algorithm based on this intuition
- We validated the theoretical analysis with experimental evidence

Future Work

Possible extensions and related research directions:

Future Work

Possible extensions and related research directions:

- Online extension

Future Work

Possible extensions and related research directions:

- Online extension
- Different methods for computing $Q^{\pi,\hat{p}}$

Future Work

Possible extensions and related research directions:

- Online extension
- Different methods for computing $Q^{\pi,\hat{p}}$
- Estimation of the reward function

Future Work

Possible extensions and related research directions:

- Online extension
- Different methods for computing $Q^{\pi,\hat{p}}$
- Estimation of the reward function
- Deeper theoretical analysis

Future Work

Possible extensions and related research directions:

- Online extension
- Different methods for computing $Q^{\pi,\hat{p}}$
- Estimation of the reward function
- Deeper theoretical analysis
- Other gradient-aware MVGs (e.g., inspired by SVG)

In Model-based RL

Maximum likelihood is an agnostic way to learn a model, but better loss functions exist when more information is available.

More generally - The meta-learning perspective

If a system uses different internal modules, the learning algorithm of a module can benefit from the knowledge about the learning algorithm of another.

The End[1]

Thank you for the attention!

[1] Paper submitted to AAAI 2020.

Cortes, C., Greenberg, S., and Mohri, M. (2013).
Relative deviation learning bounds and generalization with unbounded loss functions.
arXiv preprint arXiv:1310.5796.

Farahmand, A.-m., Barreto, A., and Nikovski, D. (2017).
Value-aware loss function for model-based reinforcement learning.
In Artificial Intelligence and Statistics, pages 1486–1494.

Feinberg, V., Wan, A., Stoica, I., Jordan, M. I., Gonzalez, J. E., and Levine, S. (2018).
Model-based value estimation for efficient model-free reinforcement learning.
arXiv preprint arXiv:1803.00101.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. (2015).
Learning continuous control policies by stochastic value gradients.

In Advances in Neural Information Processing Systems, pages 2944–2952.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015).
Human-level control through deep reinforcement learning.
Nature, 518(7540):529.

Puterman, M. L. (2014).
Markov decision processes: discrete stochastic dynamic programming.
John Wiley & Sons.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016).
Mastering the game of go with deep neural networks and tree search.
nature, 529(7587):484.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017).

Mastering the game of go without human knowledge.
Nature, 550(7676):354.

Sutton, R. S. and Barto, A. G. (2018).
Reinforcement learning: An introduction.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000).
Policy gradient methods for reinforcement learning with function approximation.
In Advances in Neural Information Processing Systems, pages 1057–1063.

Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W., Dudzik, A., Huang, A., Georgiev, P., Powell, R., Ewalds, T., Horgan, D., Kroiss, M., Danihelka, I., Agapiou, J., Oh, J., Dalibard, V., Choi, D., Sifre, L., Sulsky, Y., Vezhnevets, S., Molloy, J., Cai, T., Budden, D., Paine, T., Gulcehre, C., Wang, Z., Pfaff, T., Pohlen, T., Yogatama, D., Cohen, J., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Apps, C., Kavukcuoglu, K., Hassabis, D., and Silver, D. (2019).

AlphaStar: Mastering the Real-Time Strategy Game StarCraft II.

https://deepmind.com/blog/
alphastar-mastering-real-time-strategy-game-starcraf