Policy-based methods/Policy gradient methods

Value-based methods

- assign a value to a state (value function) OR action given a state (action value function)
- policy is *derived* from these values
  - chose the action leading to the greatest return
  - \argmax\act to implement the policy

Policy-based methods

- by contrast, construct the policy π directly
- as a parameterized (by parameters Θ) function

π_θ(\actseq|\stateseq) = \prc\actseq_\act\stateseq_\state

i.e., the policy is a probability distribution of actions \act , conditional on the state \state)

Policy-based methods are *necessary* it those cases in which Value-based methods are not possible:

- Continuous (versus discrete) action
  - the $\backslash\mathrm{argmax}\backslash\mathrm{act}$ that implements policy in Value based methods is not possible
- Stochastic policy necessary
  - games against an adversary: when an adversary can take advantage of Agent predictability

Policy-based methods are *desirable/preferable* when Value-based methods are impractical

- Large number of possible actions
- High dimensional state spaces
  - state is characterized by a (long) vector of characteristics

In both these cases:

- tables are impractical representations of Value function or Action value function

Scenario	Policy-Based Required	Policy-Based Desirable
Continuous action spaces	Yes	Yes
Stochastic strategies needed	Yes	Yes
Aliased or partially observable states	Yes	Yes
High-dimensional spaces	Sometimes	Yes
Discrete/simple environments	No	Sometimes

Here is a brief comparison of Value-based and Policy-based methods.

Aspect	**Value-Based Methods**	**Policy-Based Methods**
Output	State/action value functions	Directly parameterized policy
Policy Representation	Implicit (via greedy/exploratory actions)	Explicit (probability/distribution mapping)
Learning Objective	Value prediction loss minimization	Expected return maximization (gradient ascent)
Typical Example Algorithms	DQN, Q-learning, SARSA	REINFORCE, PPO, vanilla policy gradient
Action Space	Discrete (practical)	Handles continuous and discrete
Stochastic Policies	Limited	Natural/efficient
Exploration Strategies	Decoupled from policy (e.g. epsilon-greedy)	Inherent (stochastic policy outputs)

Policy Gradient methods

The predominant class of Policy based methods are those based on the *Policy Gradient* method.

Policy Gradient methods create a sequence of improving policies
$$\pi_0, \ldots, \pi_p, \ldots$$
by creating a sequence of improved parameter estimates
$$\theta_0, \ldots, \theta_p, \ldots$$
using Gradient Ascent on some objective function $J(\theta)$ to improve θ_p
$$\theta_{p+1} = \theta_p + \alpha * \nabla_\theta J(\theta_p)$$

- gradient of an Objective Function $J(\theta)$
- with respect to parameters Θ

Since we are trying to maximize objective function $J(\theta)$ rather than minimize a loss objective

- we use Gradient Ascent rather than Gradient Descent
- hence we add the gradient rather than subtract it, in the update

[RL Book Chapt 12 (http://incompleteideas.net/book/RLbook2020.pdf#page=343)](http://incompleteideas.net/book/RLbook2020.pdf#page=343)

Aside

There a a few Policy base methods that *don't* use Policy Gradient

- in the module on Value based methods, we learned about Policy Iteration
- Policy iteration alternates
    - Policy Evaluation: improving the estimate of a Value function
    - Policy Improvement: improving the policy
        - use $\backslash\text{argmax}\backslash\text{act}$ to implement the current policy

Thus, Policy Iteration is both Value based and Policy based

- but does not evolve policy via gradients

Stochastic policy and environment

With Value based methods

- the Environment can be stochastic
- but the Policy is usually deterministic
  - $\backslash\mathrm{argmax}\backslash\mathrm{act}$ to implement the policy

With Policy Gradient methods

- the policy can be stochastic (action is a probability distribution)
  $\pi(\backslash\mathrm{act}|\backslash\mathrm{state};\theta) = \backslash\mathrm{pr}\backslash\mathrm{actseq}_{=}\backslash\mathrm{act}\,|\,\backslash\mathrm{stateseq}_{=}\backslash\mathrm{state}\,,\ \theta_{=}\theta$
- for example: Non-greedy policy that trades off Exploitation vs Exploration

The environment can *also* be stochastic $$

\transp({ \state', \rew | \state, \act })

\transp({ \stateseq*{\tt+1}*, *rewseq*{\tt+1} | \state = \stateseq*tt*, *act* = *actseq*\tt }) $$

- the response $(\state', \rew)$ by the environment is not deterministic

This poses a challenge to Value-based methods

- a single observation of $(\state', \rew)$ is a *high variance* estimate of $\transp(\state', \rew | \state, \act)$

Objective function

The performance measure $J(\theta)$ that we seek to maximize is the

- *expected value* (across each possible episode τ) of
- the return $G_{0,\tau}$ from initial state $\stateseq_0$ of the episode

$$J(\theta) = \Exp\tau \sim \pr\theta(G_{0,\tau})$$

- where $\pr\theta$ is the probability distribution of episodes
  - is a function of the policy parameters θ

Recall: the return from state $\boxed{\text{\stateseq\_\tt}}$ of the episode is

$$\begin{aligned} G_{,\tau} &= \sum_{k=0}^{\gamma^{k} * \rewseq_{+k+1}} \\ &= \rewseq_{+1} + \gamma * G_{+1,\tau} \end{aligned}$$

Taking the gradient of the Objective

For Gradient Ascent/Descent

- We need to be able to compute
$$\nabla_\theta J(\theta_p)$$

the gradient of the Objective w.r.t the parameters

However: there is an issue in computing $J(\theta)$.

- Letting $\backslash\mathrm{pr}\tau; \theta$ denote the probability of episode τ occurring
- the expectation can be re-written as a probability-weighted sum
$$\backslash\mathrm{Exp}\tau \sim \backslash\mathrm{pr}\theta(G_{0,\tau}) = \sum_\tau \backslash\mathrm{pr}\tau; \theta * G_{0,\tau}$$

The issue is that $\backslash\mathrm{pr}\tau; \theta$ depends on

- the Environment's response at each step of τ
  - to the agent choosing actions \actseq_\tt in state \stateseq_\tt at step
- and the response is governed by probability \transp({ \stateseq_{\tt+1}, \rewseq_{\tt+1} | \state = \stateseq_\tt, \act = \actseq_\tt })

BUT under the assumption of *Model-free* methods

- the Environment's Transition Probability is unknown

So,

- how can we compute the gradient of an expectation
- when don't know the probability distribution

Policy Gradient Theorem

The *Policy Gradient Theorem* tells us how to compute the Gradient of the Expectation.

Most importantly

- the Environment's Transition probability *does not* appear
- so this results in an operational way to compute the Gradient needed for maximization of $J(\theta)$

Notes

- We simplify the presentation by assuming discount factor $\gamma = 1$

We can write the Policy Gradient Theorem in two mathematically equivalent forms

Episode Reward form

$$\nabla_\theta J(\theta) = \backslash\mathrm{Exp}\tau \sim \pi_\theta \sum_{=0}^{|\tau|} \nabla_\theta \log \pi(\backslash\mathrm{actseq}_{\tau,} | \backslash\mathrm{stateseq}_{\tau,})) \ \backslash\mathrm{rewseq}(\tau)$$

where

$$\backslash\mathrm{rewseq}(\tau) = G_{0,\tau}$$

denotes the *episode reward*

This form is particularly useful

- when there is a *single reward* received at the end of the trajectory

Notes

- To clarify that states, actions, rewards, returns, etc. depend on the specific trajectory τ
  - we add an extra subscript when necessary for clarification

$$\backslash\mathrm{rewseq}_{\tau,}$$

Periodic Reward form

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{=0}^{T-1} \nabla_\theta \log \pi_\theta(\backslash\mathrm{actseq}_{\tau,} \,|\, \backslash\mathrm{stateseq}_{\tau,t}) \; G_{\tau,}\right]$$

where

$$G_{\tau,}$$

is the return-to-go of the trajectory τ from step (state $\backslash\mathrm{stateseq}_{\tau,}$).

This form is particularly useful

- when there are rewards at intermediate steps of the episode

Whichever way we write it

- the Policy Gradient Theorem is the foundation for all policy-based methods
- it tells us how to change the parameters θ of the Policy NN in a direction leading to optimality

We will study a few of these methods in a later section.

Computing $\nabla_\theta J(\theta)$

Our first step will be to turn the expectation

$$\backslash\text{Exp}\tau \sim \backslash\text{prc}\tau\theta \ \backslash\text{rewseq}(\tau)$$

into a sum

$$\sum_{\tau \sim \backslash\text{prc}\tau\theta} \backslash\text{prc}\tau\theta * \backslash\text{rewseq}(\tau)$$

where

$$\backslash\text{prc}\tau\theta$$

is the probability of trajectory τ

With stochastic policy and Environment

- for trajectory τ

$$\tau = \backslash\mathrm{stateseq}_{\tau,0}, \backslash\mathrm{actseq}_{\tau,0}, \backslash\mathrm{rewseq}_{\tau,1}, \backslash\mathrm{stateseq}_{\tau,1} \ldots \backslash\mathrm{stateseq}_{\tau,}, \backslash\mathrm{actseq}_{\tau,}, \backslash\mathrm{rewseq}_{\tau,+1}, \backslash\mathrm{stateseq}_{\tau,+1}, \ldots$$

we can compute $\backslash\mathrm{prc}\tau\theta$

The presence of Environment Transition probability

$$\backslash\mathrm{transp}(\backslash\mathrm{stateseq}_{\tau,+1}, \backslash\mathrm{rewseq}_{\tau,+1} | \backslash\mathrm{stateseq}_{\tau,}, \backslash\mathrm{actseq}_{\tau,})$$

- is problematic
- as it is generally unknown
    - we would like the Model-free assumption to hold

Turning the expectation into a sum and taking the gradient of the expected Episode Reward

$$\begin{aligned}
\nabla_\theta \sum_{\tau \sim \backslash\mathrm{prc}\tau\theta} \backslash\mathrm{prc}\tau\theta * \backslash\mathrm{rewseq}(\tau) &= \sum_{\tau \sim \backslash\mathrm{prc}\tau\theta} \nabla_\theta \ \backslash\mathrm{prc}\tau\theta * \backslash\mathrm{rewseq}(\tau) \\
&= \sum_{\tau \sim \backslash\mathrm{prc}\tau\theta} (\backslash\mathrm{prc}\tau\theta \nabla_\theta \log \backslash\mathrm{prc}\tau\theta) \ \backslash\mathrm{rews}
\end{aligned}$$

We now substitute the chained probability previously derived for

$$\backslash \mathrm{prc}\tau\theta$$

into the

$$\log \backslash \mathrm{prc}\tau\theta$$

term above.

To summarize the proof:

$$\begin{aligned}
\nabla_\theta \backslash\mathrm{Exp}\tau \sim \backslash\mathrm{pr}\theta\ \backslash\mathrm{rewseq}(\tau) &= \nabla_\theta \sum_{\tau\sim\backslash\mathrm{pr}\theta} \backslash\mathrm{prc}\tau\theta * \backslash\mathrm{rewseq}(\tau)\\
&= \sum_{\tau\sim\backslash\mathrm{pr}\theta} (\backslash\mathrm{prc}\tau\theta\nabla_\theta \log \backslash\mathrm{prc}\tau\theta) * \backslash\mathrm{rewseq}(\\
&= \sum_{\tau\sim\backslash\mathrm{pr}\theta} \Big(\backslash\mathrm{prc}\tau\theta * \sum_{=0}^{|\tau|} \nabla_\theta \log \pi(\backslash\mathrm{actseq}_{\tau,} | \backslash\\
&= \backslash\mathrm{Exp}\tau \sim \backslash\mathrm{prc}\tau\theta \sum_{=0}^{|\tau|} \nabla_\theta \log \pi(\backslash\mathrm{actseq}_{\tau,} | \backslash\mathrm{st}
\end{aligned}$$

Key result

- We can evaluate the Gradient *without knowing* the model
    - i.e., Environment Transition Probability terms $\transp(\ldots)$
- The expectation can be approximated by *sampling* trajectories
    - observe the *effect* of the Environment's distribution
    - without *knowing* it

The convention is to write the expectation

$\Exp{\tau \sim \pi_\theta}$ rather than $\Exp{\tau \sim \prc{\tau}{\theta}}$

This is merely convention: they refer to the same distribution of episodes.

Notes on Proof of Policy Gradient theorem

Likelihood ratio trick

The likelihood ratio trick states that

- for a parameterized probability distribution $p_\theta(x)$
- and a function $f(x)$:

the gradient of an expectation can be converted into an expectation over the gradient.

It is a simple consequence of the Derivative of a Log rule of calculus.

$$
\begin{aligned}
\nabla_\theta \mathbb{E}_{x\sim p_\theta}[f(x)] &= \nabla_\theta \int p_\theta(x) f(x) dx && \text{convert expectation to int} \\
&= \int \nabla_\theta p_\theta(x) f(x) dx && \text{move grad inside the integ} \\
&= \int f(x) \nabla_\theta p_\theta(x) dx && \text{rearrange term} \\
&= \int f(x) \left(p_\theta(x) \nabla_\theta \log p_\theta(x) \right) dx && \text{log trick:} \\
& && \nabla_\theta p_\theta(x) = p_\theta(x) \nabla_\theta \log p_\theta(\\
&= \mathbb{E}_{x\sim p_\theta} \left[f(x) \nabla_\theta \log p_\theta(x) \right] && \text{convert integral back to ex}
\end{aligned}
$$

The "log trick" follows from the rules of calculus

$$
\begin{aligned}
\nabla_\Theta \log p_\theta(x) &= \frac{1}{p_\theta(x)} * \nabla_\Theta p_\theta(x) && \text{Calculus: grad of log, chain rule} \\
\nabla_\Theta p_\theta(x) &= p_\theta(x) * \nabla_\Theta \log p_\theta(x) && \text{re-arranging terms}
\end{aligned}
$$

Why substituting $\textcolor{red}{\backslash\mathbf{rewseq}}_{\tau,}$ for $G_{\tau,0}$ is algebraically the same

Consider the two equivalent forms for expressing the Policy Gradient Theorem

$$\sum_{=0}^{T-1} \nabla_\theta \log \pi_\theta(a_| \mathtt{s}_) \cdot \mathtt{R}(\tau)$$

and

\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | s_t) \cdot G_\tt

where:

$$G_= \sum_{\mathtt{k}=}^{\mathtt{T}-1} \gamma^{\mathtt{k}-} \mathbf{r}_\mathtt{k}$$

and

$$R(\tau) = G_0$$

How can these two forms be mathematically equivalent ?

- since the first form involves G_0
    - the rewards *over all steps* of the episode
- and the second form involves $\boxed{\text{G\_\textbackslash tt}}$
    - the *future* rewards from step onward
    - and $\boxed{\text{G\_\textbackslash tt}}$ and $G_{'}$ for $'$ > include the same rewards

Algebraically they appear different.

The answer is that

- the two forms appear *within an expectation*
- which is evaluated over *future* time steps
- so the part of $\boxed{\text{G\_\textbackslash tt}}$ that reference *past rewards* is equal to 0 under the expectation

Here are the details:

- Step 1: Start with the total return $R(\tau) = G_0$

Define L to be the expression for the first form of the Theorem:

$$L = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot G_0 = G_0 \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- Step 2: Expand G_0 as the sum over rewards

$$G_0 = \sum_{k=0}^{T-1} \gamma^k r_k$$

Substitute into L:

$$L = \left(\sum_{k=0}^{T-1} \gamma^k r_k\right) \left(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\right)$$

- Step 3: Express as a double sum

$$L = \sum_{t=0}^{T-1} \sum_{k=0}^{T-1} \gamma^k r_k \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- Step 4: Separate sums over past and future rewards relative to t

$$L = \sum_{t=0}^{T-1} \left(\sum_{k=0}^{t-1} \gamma^k r_k + \sum_{k=t}^{T-1} \gamma^k r_k \right) \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- Step 5: Rewrite future rewards shifted by t

Define $j = k - t$:

$$\sum_{k=t}^{T-1} \gamma^k r_k = \gamma^t \sum_{j=0}^{T-1-t} \gamma^j r_{t+j} = \gamma^t G_t$$

- Step 6: Substitute back into L

$$L = \sum_{t=0}^{T-1} \left(\sum_{k=0}^{t-1} \gamma^k r_k + \gamma^t G_t \right) \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- Step 7: Expectation zeroes out past rewards term

Because rewards before time t do not depend on action a_t, their expected contribution is zero:

$$\mathbb{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \sum_{k=0}^{t-1} \gamma^k r_k\right] = 0$$

- Step 8: Final form of the policy gradient

Thus,

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \gamma^t G_t\right]$$

which is the second form of expressing the Policy Gradient Theorem.

Alternate Proof of the Policy Gradient Theorem (Per-Step Reward Perspective)

Let $J(\theta)$ be the expected discounted sum of rewards under policy π_θ:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right]$$

- Step 1: Expand the Expectation

Rewrite the expectation explicitly:

$$J(\theta) = \sum_\tau P_\theta(\tau) \left(\sum_{t=0}^{T-1} \gamma^t r_t \right)$$

- Step 2: Differentiation w.r.t. θ

$$\nabla_\theta J(\theta) = \sum_\tau \nabla_\theta P_\theta(\tau) \left(\sum_{t=0}^{T-1} \gamma^t r_t \right)$$

Apply the **likelihood ratio trick**: $\nabla_\theta P_\theta(\tau) = P_\theta(\tau) \nabla_\theta \log P_\theta(\tau)$ So, $\nabla_\theta J(\theta) = \sum_\tau P_\theta(\tau) \nabla_\theta \log P_\theta(\tau) \left(\sum_{t=0}^{T-1} \gamma^t r_t \right)$ Or equivalently, $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\nabla_\theta \log P_\theta(\tau) \left(\sum_{t=0}^{T-1} \gamma^t r_t \right) \right]$

- Step 3: Break Down $\log P_\theta(\tau)$

Recall, $\log P_\theta(\tau) = \sum_{t=0}^{T-1} \log \pi_\theta(a_t|s_t) + \textbf{terms independent of } \theta$ So, $\nabla_\theta \log P_\theta(\tau) = \sum_{t'=0}^{T-1} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'})$ Substitute:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t'=0}^{T-1} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'}) \cdot \left(\sum_{t=0}^{T-1} \gamma^t r_t\right)\right]$$

- Step 4: Swap Order of Summation

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t'=0}^{T-1} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'})\gamma^t r_t\right]$$

Switch the order:

$$= \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \sum_{t'=0}^{T-1} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'})\gamma^t r_t\right]$$

- Step 5: Analyze the causal relationship

The gradient w.r.t. $a_{t'}$ can only affect rewards *from* t' onward (not earlier rewards due to the Markov property), so for $t < t'$ the expectation is zero.

Thus, the only contributing terms are where $t' \leq t$:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t'=0}^{T-1} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'}) \left(\sum_{t=t'}^{T-1} \gamma^t r_t\right)\right]$$

- Step 6: Recognize the return-to-go term

$$\sum_{t=t'}^{T-1} \gamma^t r_t = \gamma^{t'} \sum_{j=0}^{T-1-t'} \gamma^j r_{t'+j} = \gamma^{t'} G_{t'}$$

So, we may write:

$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \gamma^t G_t\right]$ Often, γ^t is absorbed into the definition of G_t.

- Step 7: Final form (policy gradient theorem)

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) G_t\right]$$

Conclusion:
By starting with the expectation of per-step rewards and applying the likelihood ratio trick, we arrive at the same policy gradient theorem:
each action's gradient is weighted by the return-to-go from that time (not just the immediate reward).

Preview of Policy-Based Reinforcement Learning Methods

We will subsequently present a number of Policy based methods.

Actor-Critic

Value-based methods learn a function approximation of the *value* of a state or a state/action pair.

- policy is chosen based on the value of successor states

Simple Policy-based methods learn a parameterized policy function.

- using a NN to learn the policy
- using an objective function $J(\theta)$ that depends on an approximation of either
  - the value $\statevalfun(\state)$ or $\boxed{\text{G\_\textbackslash tt}}$
  - or action/value function $\actvalfun(\state, \act)$

Actor-Critic-Policy-based methods used Neural Networks to learn

- *both* the value function and policy function approximations
- the agent is called the *Actor*
- the NN providing estimates of G_t or $\actvalfun(\state, \act)$ is called the *Critic*

- RL tips and tricks (https://stable-baselines3.readthedocs.io/en/master/guide/rl_tips.html)
- RL book contents (http://incompleteideas.net/book/RLbook2020.pdf#page=7)
- RL book notation (http://incompleteideas.net/book/RLbook2020.pdf#page=20)

In [2]:

```
print("Done")
```

```
Done
```