An Introduction to Policy Gradient Methods

Matias Bayas-Erazo

University of Zürich

Roadmap

1. **Setup** – Markov Decision Processes and the RL objective
2. **Policy gradient methods** – REINFORCE, reward-to-go, baselines
3. **Modern trust-region methods** – GAE, TRPO, PPO
4. **Application** – solving a consumption-savings problem with PPO

Policy gradient methods

Setup

- Consider a Markov Decision Process (MDP) defined by $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ where:
  - $\mathcal{S}$ is the set of states and $\mathcal{A}$ is the set of actions
  - $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition kernel
  - $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function
  - $\gamma \in [0, 1)$ is the discount factor

- Let $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A})$ be a *stochastic* policy, parameterized by θ.

- Let $\boldsymbol{\tau} = (s_0, a_0, s_1, a_1, \ldots)$ be trajectory of states and actions
  - generated by sampling actions from π_θ and transitioning according to P.

- **Goal:** maximize the expected return of policy π_θ:

$$J(\pi_\theta) = \mathbb{E}_{p(s_0)}\left[V^\pi(s_0)\right], \quad \text{where} \quad V^\pi(s) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s\right]$$

Basic idea behind policy gradient methods

1. Write value of policy π_θ as:

$$J(\theta) = J(\pi_\theta) = \mathbb{E}_{p_\theta(\boldsymbol{\tau})}\left[R(\boldsymbol{\tau})\right],$$

   where:

   - $R(\boldsymbol{\tau}) \equiv \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)$ is the return of trajectory $\boldsymbol{\tau}$
   - $p_\theta(\boldsymbol{\tau})$ is the distribution over trajectories induced by policy π_θ *and the world model*:

$$p_\theta(\boldsymbol{\tau}) = p(s_0) \prod_{t=0}^{T} \hat{P}(s_{t+1}|s_t, a_t)\, \pi_\theta(a_t|s_t)$$

2. Optimizing with respect to θ gives a simple FOC:

$$\nabla_\theta J(\theta) = 0$$

3. Suggests a simple algorithm to find θ^*: $\quad \theta^{(k+1)} \leftarrow \theta^{(k)} + \alpha\, \nabla_\theta J(\theta^{(k)})$

Computing the policy gradient $\nabla_\theta J(\theta)$

- Straightforward differentiation gives:

$$\begin{aligned}\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{p_\theta(\boldsymbol{\tau})}\left[R(\boldsymbol{\tau})\right] &= \int_{\boldsymbol{\tau}} R(\boldsymbol{\tau})\, \nabla_\theta p_\theta(\boldsymbol{\tau})\, d\boldsymbol{\tau} \\ &= \mathbb{E}_{p_\theta(\boldsymbol{\tau})}\left[R(\boldsymbol{\tau})\, \nabla_\theta \log p_\theta(\boldsymbol{\tau})\right]\end{aligned}$$

- **Key step**: the world model $\hat{P}(s_{t+1}|s_t, a_t)$ does not depend on θ, so:

$$\nabla_\theta \log p_\theta(\boldsymbol{\tau}) = \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- So we obtain a simple expression for the policy gradient: [Williams, 1992]

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

Estimating the policy gradient $\nabla_\theta J(\theta)$

- The expectation can be estimated with Monte Carlo sampling.
- Let $\mathcal{D} = \{\boldsymbol{\tau}_i\}_{\boldsymbol{\tau}_i \in \mathcal{D}}$ denote a set of N trajectories generated using policy π_θ.

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} R(\boldsymbol{\tau}_i) \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t^{(i)} | s_t^{(i)})$$

- So now our learning rule becomes:

$$\theta^{(k+1)} \leftarrow \theta^{(k)} + \alpha \left[\frac{1}{N} \sum_{i=1}^{N} R(\boldsymbol{\tau}_i) \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t^{(i)} | s_t^{(i)}) \right]$$

- Often referred to as **Vanilla Policy Gradient** or **REINFORCE**

Improving estimate of the policy gradient $\nabla_\theta J(\theta)$

- The basic REINFORCE estimator has high variance, which slows down learning
  - The "weight" on $\nabla_\theta \log \pi_\theta(a_t|s_t)$ is total return $R(\boldsymbol{\tau})$, which can be very noisy
- We can do better by using **reward-to-go**, $G_t = \sum_{j=t}^{\infty} \gamma^{j-t} R(s_j, a_j)$
- Can show that the policy gradient can be written as

Proof

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

or

$$\nabla_\theta J(\theta) = \mathbb{E}_{p_\theta(\boldsymbol{\tau})} \left[\sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t)\, Q^\pi(s_t, a_t) \right]$$

- $Q^\pi(s_t, a_t)$ is not known and must be estimated, this leads to **actor-critic** methods

Improving estimate of the policy gradient $\nabla_\theta J(\theta)$ (cont.)

- Subtracting a **baseline** $b(s_t)$ leaves the gradient unbiased:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t)\left(Q^\pi(s_t, a_t) - b(s_t)\right)\right]$$

  because conditional on s_t we have $b(s_t)$ fixed and

$$\mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)}[\nabla_\theta \log \pi_\theta(a_t|s_t) b(s_t)] = b(s_t)\nabla_\theta \sum_a \pi_\theta(a|s_t) = 0,$$

- Natural choice: fit $b(s_t) \approx V^\pi(s_t)$ with a critic; this helps to further reduce variance

- The term $A^\pi(s_t, a_t) \equiv Q^\pi(s_t, a_t) - V^\pi(s_t)$ is the **advantage**.
  - Only update likelihood of policies that lead to better-than-average outcomes.
  - In practice, there is a variance-bias trade off
  - see Sutton et al. (1999) for a convergence result.

Generalized advantage estimation

- Introduce n-step advantage estimates:

$$\hat{A}_t^{(n)} = \Big(\sum_{j=0}^{n-1} \gamma^j r_{t+j} \Big) + \gamma^n \hat{V}_\phi(s_{t+n}) - \hat{V}_\phi(s_t),$$

  interpolates b/w high-variance Monte Carlo ($n \to \infty$) and biased one-step TD ($n = 1$).

- **GAE** averages these targets with geometrically decaying weights $w_n \propto \lambda^{n-1}$:

$$\hat{A}_t^{\mathrm{GAE}} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \hat{A}_t^{(n)}.$$

- $\lambda \in [0, 1]$ tunes the bias–variance trade-off.

Modern trust-region policy gradients

- **TRPO** maximizes the surrogate

$$L^{\text{TRPO}}(\theta) = \hat{\mathbb{E}}\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right]$$

  subject to a trust-region constraint $\hat{\mathbb{E}}[D_{\text{KL}}(\pi_{\theta_{\text{old}}} \,\|\, \pi_\theta)] \leq \epsilon$;

- Solving the resulting constrained problem yields the gradient step

- **PPO** keeps trust-region flavor but bypasses constrained with clipped objective:

$$L^{\text{PPO}}(\theta) = \hat{\mathbb{E}}\Big[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\Big]$$

  with $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$.

- Both methods pair these with GAE-style advantages and value-function baselines

Application: solving a consumption-savings problem with PPO

Using PPO to solve an income fluctuation problem

- Households observe (a_t, y_t, r_t), choose $a_{t+1} \in [0, \bar{a}]$, and consume $c_t = (1 + r_t)a_t + y_t - a_{t+1}$ with utility $u(c_t) = \log(c_t + \varepsilon)$.

- **Twist**: agents do not know the law of motion of r_t. The true process is a clipped AR(1):

$$r_{t+1} = \mathrm{clip}\big(\mu_r + \rho_r(r_t - \mu_r) + \sigma_r \varepsilon_{t+1},\ [\underline{r}, \bar{r}]\big), \qquad \varepsilon_{t+1} \sim \mathcal{N}(0, 1),$$

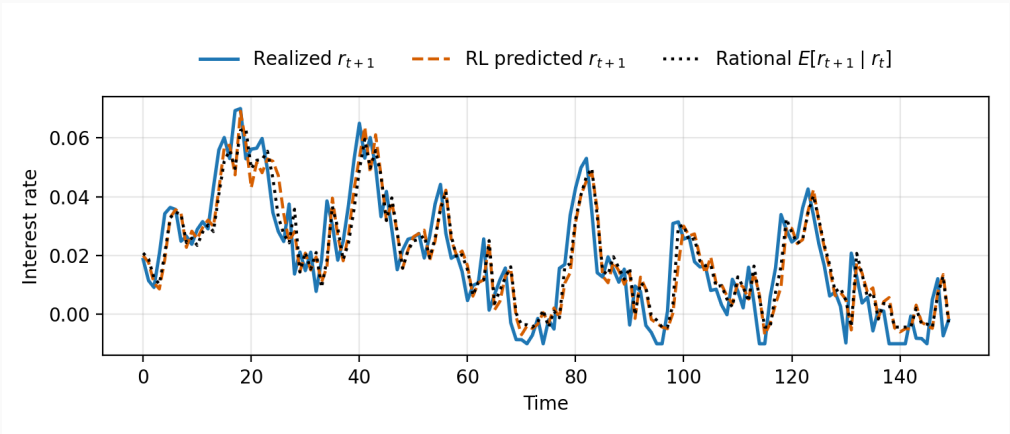
  but only (r_t) is observed, so agents learn a forecasting rule for r_{t+1}

- Policy network outputs both a_{t+1} and $\hat{r}_\theta(s_t) \approx \mathbb{E}[r_{t+1} \mid s_t]$, leading to the actor loss

$$\mathcal{L}_{\text{actor}} = \mathcal{L}_{\text{PPO}} - \alpha_{\text{ent}}\mathcal{L}_{\text{ent}} + \alpha_{\text{rate}}\,\mathbb{E}\big[(\hat{r}_\theta(s_t) - r_{t+1})^2\big],$$

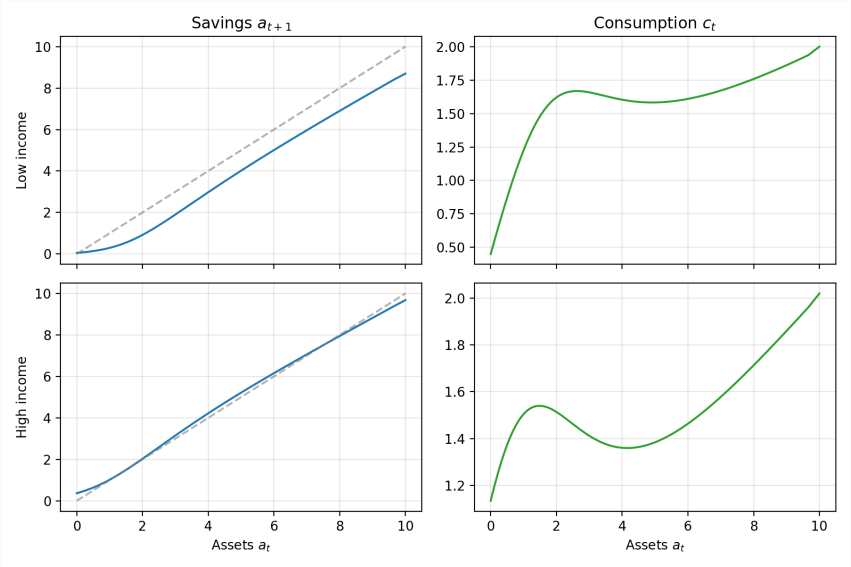
  while the critic minimizes $\mathcal{L}_{\text{critic}} = \mathbb{E}\Big[(V_\phi(s_t) - \hat{V}_t^{\text{GAE}})^2\Big]$.

Learning the interest-rate process

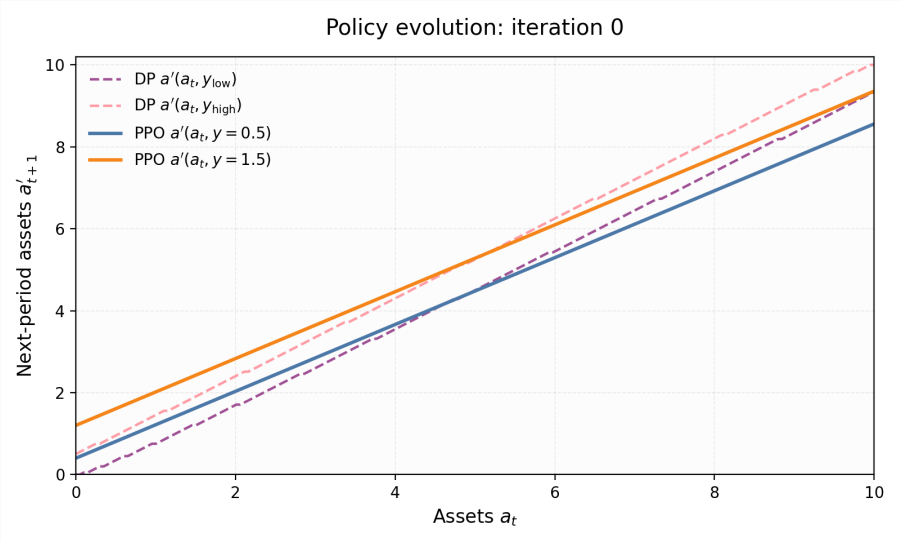


- Agent learns to forecast pretty well, pretty close to RE benchmark

Not so fast...


Savings a_{t+1}
Consumption c_t
Low income
High income
Assets a_t
Assets a_t

Policy evolution under PPO



References

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Proving that $\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\, G_t\right]$

1. Decompose total return as $R(\tau) = \sum_{j=0}^{t-1} \gamma^j R(s_j, a_j) + \gamma^t G_t$

2. Substitute into the policy gradient expression to get:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{j=0}^{t-1} \gamma^j R(s_j, a_j)\right] + \mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \gamma^t G_t\right].$$

3. Now apply the law of iterated expectations to the first term:

$$\mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{j=0}^{t-1} \gamma^j R(s_j, a_j)\right] = \sum_{t=0}^{T} \mathbb{E}\left[\sum_{j=0}^{t-1} \gamma^j R(s_j, a_j)\, \underbrace{\mathbb{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \mid \tau_{1:t}\right]}_{=0}\right]$$

4. The first term is zero, so we get the desired result.

Back to reward-to-go slide

Policy gradient theorem

- The policy gradient can also be expressed as an expectation over states rather than trajectories.

- Let $d^{\pi}(s)$ denote the discounted state visitation distribution under policy π:

$$d^{\pi}(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^{t}P(s_t = s|\pi)$$

- Then the **Policy Gradient Theorem** states that:

$$\nabla_{\theta} J(\theta) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi}, a\sim\pi_{\theta}}\left[\nabla_{\theta}\log\pi_{\theta}(a|s)\, Q^{\pi}(s,a)\right]$$