Advanced Policy Gradients - II

Easwar Subramanian

TCS Innovation Labs, Hyderabad

Email : easwar.subramanian@tcs.com / cs5500.2020@iith.ac.in

Novemer 20, 2021

Overview of this Lecture

1. Review
2. Approximations to Trust Region Formulation
3. Natural Policy Gradient
4. Fisher Information Matrix and KL Divergence
5. Relationship of Natural Gradient to Policy Gradient
6. Other Algorithms
7. Proximal Policy Optimization

Review

Policy Optimization Problem

The performance of a policy π_θ is given by

$$J(\theta) = V^{\pi_\theta}(s_0) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s\right]$$

where $\gamma < 1$ is the discount factor of the MDP

General form for gradient of the performance measure is given by

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)\Psi_t\right]$$

Disadvantages are :

- Sample Inefficiency : **on-policy expectation**
- Distance in parameter space $\neq$ policy space

Surrogate Loss Function

We recast the optimization problem using a **surrogate loss function**

$$\arg\max_{\pi'} J(\pi') = \arg\max_{\pi'} \left[J(\pi') - J(\pi_0) \right] \approx \mathcal{L}_{\pi_0}(\pi')$$

where

$$\mathcal{L}_{\pi_0}(\pi') = \mathop{\mathbb{E}}_{\tau \sim \pi_0} \left[\sum_{t=0}^{\infty} \gamma^t \frac{\pi'(a_t|s_t)}{\pi_0(a_t|s_t)} A^{\pi_0}(s_t, a_t) \right]$$

The approximation is valid if policies π' and π_0 are **'close'** in terms of their KL divergence

Relative Policy Performance Bound

We can have a **relative policy performance bound** using KL divergence to measure the goodness of the approximation obtained

$$\left[J(\pi') - \left(J(\pi_0) + \mathcal{L}_{\pi_0}(\pi')\right)\right] \leq C\sqrt{\mathop{\mathbb{E}}_{s\sim d^{\pi_0}}\left[D_{KL}(\pi'||\pi_0)[s]\right]}$$

This gives rise to an optimization routine with the following iterative procedure with π_{k+1} and π_k are related by

$$\pi_{k+1} = \arg\max_{\pi'}\left[\mathcal{L}_{\pi_k}(\pi') - C\sqrt{\mathop{\mathbb{E}}_{s\sim d^{\pi_k}}\left[D_{KL}(\pi'||\pi_k)[s]\right]}\right]$$

Performance guarantee

$$[J(\pi_{k+1}) - J(\pi_k)] \geq 0$$

- C is quite high when γ is close to 1 and hence choosing step size becomes an issue

A First-Cut Algorithm

1: Initialize π_0
2: **for** $k = 0, 1, 2, \cdots$ until convergence **do**
3: Sample a trajectory τ from policy π_k
4: Compute advantage function $A^{\pi_{\theta_k}}(a_t, s_t)$ for all (s_t, a_t) pairs in the trajectory τ
5: Solve the optimization problem

$$\pi_{k+1} = \arg\max_{\pi'} L_{\pi_k}(\pi') - C\sqrt{\mathbb{E}_{s\sim d^{\pi_k}}\left[D_{KL}(\pi'||\pi_k)[s]\right]}$$

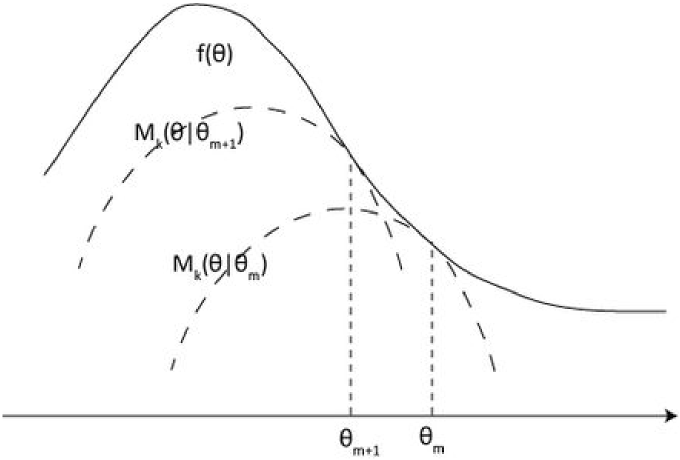
6: **end for**

Issues are :

- C is quite high when γ is close to 1 $\left(C = \frac{4\varepsilon\gamma}{1-\gamma^2}\alpha^2\right)$
- Consequently, step size becomes too small

Majorize Maximize Framework

Majorize-Maximize framework is used to solve the optimization step



Approximate Monotone Improvement

- Instead of KL penalty, use KL constraint
- Can control worst case error through constraint upper limit

$$\pi_{k+1} = \arg\max_{\pi'} \left[L_{\pi_k}(\pi') \right]$$
$$\text{such that} \quad \mathop{\mathbb{E}}_{s \sim d^{\pi_k}} D_{KL}(\pi' || \pi_k)[s] \leq \delta$$

- From the constraint, **steps respect a notion of distance in policy space**
- Above constrained optimization is basis of many algorithms, Natural Policy Gradient (NPG), truncated NPG, TRPO and PPO
- The objective and the constraint can be estimated from the roll-out of old policies – **sample efficient**
- Update is **invariant** to parametrization

Approximations to Trust Region Formulation

Trust Region Formulation

We have the following optimization problem

$$\pi_{k+1} = \arg\max_{\pi'} \left[\mathcal{L}_{\pi_k}(\pi')\right]$$
$$\text{such that } \bar{D}_{KL}(\pi'||\pi_k) \leq \delta$$

The constraint on the optimization problem is the trust region with size δ and some guarantees on performance improvement are there within the trust region

For parametrized policies the optimization can be written as

$$\pi_{\theta_{k+1}} = \arg\max_{\pi_\theta} \left[\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta)\right]$$
$$\text{such that } \bar{D}_{KL}(\pi_\theta||\pi_{\theta_k}) \leq \delta$$

How do we solve it ?

- Linear approximation for the objective
- Quadratic approximation for the constraint

Approximation of Objective Function

Taylor series expansion for function $f(x)$ around point a is given by

$$f(x) = f(a) + f'(a)(x-a) + \frac{1}{2} f''(a)(x-a)^2 + \cdots$$

- Using Taylor series expansion on objective function $\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta)$ around θ_k (upto first order term) gives us

$$\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta) \approx \cancelto{0}{\mathcal{L}_{\pi_{\theta_k}}(\pi_{\theta_k})} + g^T(\theta - \theta_k) \qquad \text{where } g \doteq \nabla_\theta \mathcal{L}_{\pi_{\theta_k}}(\pi_\theta)\ |_{\theta=\theta_k}$$

- Recall that g is exactly the policy gradient (from previous lecture !)

$$\nabla_\theta \mathcal{L}_{\pi_{\theta_k}}(\pi_\theta)|_{\theta=\theta_k} = \mathop{\mathbb{E}}_{\tau \sim \pi_{\theta_k}} \left[\sum_{t=0}^{\infty} \nabla_\theta \log(\pi_{\theta_k}(a_t|s_t)|_{\theta=\theta_k}\ \gamma^t A^{\pi_{\theta_k}}(s_t, a_t) \right]$$

- Objective function is simplified to

$$\theta_{k+1} = \arg\max_\theta g^T(\theta - \theta_k)$$

Approximation of Trust Region Constraint

Using Taylor series expansion on the constraint (around θ_k; upto second order) gives us

$$\bar{D}_{KL}(\pi_\theta||\pi_{\theta_k}) \approx \cancelto{0}{\bar{D}_{KL}(\pi_{\theta_k}||\pi_{\theta_k})} + \cancelto{0}{\nabla_\theta \bar{D}_{KL}(\pi_\theta||\pi_{\theta_k})|_{\theta=\theta_k}} + \nabla^2_\theta \bar{D}_{KL}(\pi_\theta||\pi_{\theta_k})\ |_{\theta=\theta_k}$$

The first order term $\nabla_\theta \bar{D}_{KL}(\pi_\theta||\pi_{\theta_k})$ evaluates to zero since the expectation of the score function is zero

$$\nabla_\theta \bar{D}_{KL}(\pi_\theta \,\|\, \pi_{\theta_k}) = \nabla_\theta \mathop{\mathbb{E}}_{\pi_\theta}[\log \pi_\theta] - \nabla_\theta \mathop{\mathbb{E}}_{\pi_\theta}[\log \pi_{\theta_k}] = \mathop{\mathbb{E}}_{\pi_\theta}[\nabla_\theta \log \pi_\theta] = 0$$

Therefore, we are left only with the second order term

$$\bar{D}_{KL}(\pi_\theta||\pi_{\theta_k}) \approx \frac{1}{2}(\theta - \theta_k)^T H\ (\theta - \theta_k) \quad \text{where } H \doteq \nabla^2_\theta \bar{D}_{KL}(\pi_\theta||\pi_{\theta_k})\ |_{\theta=\theta_k}$$

Natural Policy Gradient

Natural Policy Gradient

The optimization problem is now simplified as

$$\theta_{k+1} = \arg\max_{\theta} g^T(\theta - \theta_k)$$
$$\text{such that } \frac{1}{2}(\theta - \theta_k)^T H \, (\theta - \theta_k) \leq \delta$$

Linear objective with quadratic constraint

Solution to the approximate problem obtained using Lagrange multiplier method

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g$$

The term $H^{-1}g$ is called the Natural gradient

Algorithm : Natural Policy Gradient

Algorithm Natural Policy Gradient

1: Initialize π_0
2: **for** $k = 0, 1, 2, \cdots$ **do**
3: Collect trajectories D_k on policy $\pi_k = \pi_{\theta_k}$
4: Estimate all advantages $A^{\pi_{\theta_k}}(s_t, a_t)$
5: Form sample estimates for policy gradients $\hat{g}_k$ (using advantage estimates)
6: Form sample estimates for the Hessian of KL divergence
7: Compute the Natural Policy Gradient update

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g_k^T H_k^{-1} g_k}} H_k^{-1} g_k$$

8: **end for**

Fisher Information Matrix and KL Divergence

Fisher Information Matrix and KL Divergence

- Let $p(x|\theta)$ be a probability distribution parameterized by θ.
- Score function of a parameterized probability distribution is given by

$$s(\theta) = \nabla_\theta \log p(x|\theta) \,,$$

- For a parameter vector θ, Fisher Information Matrix is given by,

$$\mathrm{F} = \mathop{\mathbb{E}}_{p(x|\theta)} \left[\nabla_\theta \log p(x|\theta) \, \nabla_\theta \log p(x|\theta)^{\mathrm{T}} \right] .$$

- The sample estimate of the above expectation is given by,

$$\mathrm{F} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log p(x_i|\theta) \, \nabla_\theta \log p(x_i|\theta)^{\mathrm{T}} . \tag{1}$$

- **Claim** : Fisher Information Matrix F is the Hessian of KL-divergence between two probability distributions $p(x|\theta')$ and $p(x|\theta)$ evaluated at $\theta' = \theta$

$$\mathrm{KL}[p(x|\theta') \,\|\, p(x|\theta)] = \mathop{\mathbb{E}}_{p(x|\theta)} [\mathrm{H}_{\log p(x|\theta)}] = \mathrm{F}$$

Properties of Natural Policy Gradient

- Natural policy gradient algorithm gives an update-rule in which updates are pre-multiplied by H^{-1}
- The Hessian of the KL-divergence is the Fischer Information Matrix given by

$$\mathrm{F} = \mathop{\mathbb{E}}_{\pi_\theta}\left[\nabla \log \pi_\theta(\cdot|s)\, \nabla \log \pi_\theta(\cdot|s)^{\mathrm{T}}\right]$$

- The NPG direction $H^{-1}g$ is **co-variant**; i.e. it points in same direction irrespective of the parametrization that is used to compute it

Relationship of Natural Gradient to Policy Gradient

Policy Gradient Formulation

Consider the following optimization problem

$$\pi_{\theta_{k+1}} = \arg\max_{\pi_\theta} \left[\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta)\right]$$
$$\text{such that } \left\|\theta - \theta_k)\right\|^2 \leq \delta$$

After **linearising** the objective, the optimization problem is now,

$$\theta_{k+1} = \arg\max_{\theta} g^T(\theta - \theta_k) \text{ such that } (\theta - \theta_k)^2 \leq \delta$$

This is the original policy gradient problem !!

We move a small distance in parameter space in the direction of the gradient

Natural Gradient Formulation

Natural policy gradient problem is given by,

$$\pi_{\theta_{k+1}} = \arg\max_{\pi_\theta} \left[\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta) \right]$$
$$\text{such that } \bar{D}_{KL}(\pi_\theta || \pi_{\theta_k}) \leq \delta$$

After **linearising** the objective and **quadratifying** the constraint, fhe optimization problem is then given by,

$$\theta_{k+1} = \arg\max_{\theta} g^T(\theta - \theta_k) \text{ such that } \frac{1}{2}(\theta - \theta_k)^T F \, (\theta - \theta_k) \leq \delta$$

Relationship between Formulations

- Vanilla policy gradient has the right objective but *"incorrect"* constraint (Euclidean penalty instead of KL penalty)
- Recall that, policy iteration (from MDP lectures) obtain policy improvement with no constraint

Other Algorithms

Truncated Natural Policy Gradient

- **Problem** : For neural networks, the dimensionality of parameter θ are high. High computational cost in inverting the matrix H
- **Solution** : Use the **conjugate gradient algorithm** to compute $H^{-1}g$ without inverting H
- Resultant algorithm : Truncated Natural Policy Gradient
- ACTKR algorithm uses KFAC technique to solve the inverse Hessian computation problem

Problems with Natural Policy Gradient Update

- Another problem with NPG update is that - might not be robust to trust region size δ
  - δ may be too large in some iterations and can degrade the performance
- Because of quadratic approximation, the KL-divergence constraint may be violated
- Monotonic improvement may not occur in all iterations

TRPO : Line Search Algorithm

- Enforce improvement in surrogate (i.e. $\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta) \geq 0$)
- Enforce KL constraint
- How ? Backtracking line search with exponential decay

Algorithm Line Search for TRPO

1: Compute the proposed policy step $\Delta_k = \sqrt{\frac{2\delta}{g_k^T H_k^{-1} g_k}} H_k^{-1} g_k$
2: **for** $j = 0, 1, 2, \cdots N$ **do**
3: Compute proposed update $\theta = \theta_k + \alpha_j \Delta_k$
4: **If** $\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta) \geq 0$ and $\bar{D}_{KL}(\theta || \theta_k) \leq \delta$
5: Accept the update $\theta = \theta_k + \alpha_j \Delta_k$
6: **Else**
7: Find another α_j (Reduce α_j)
8: **end for**

TCS Research & Innovation

Algorithm Trust Region Policy Optimization

1: Initialize π_0
2: **for** $k = 0, 1, 2, \cdots$ **do**
3: Collect trajectories D_k on policy $\pi_k = \pi_{\theta_k}$
4: Estimate all advantages $A^{\pi_{\theta_k}}(s_t, a_t)$
5: Form sample estimates for policy gradients $\hat{g}_k$ (using advantage estimates)
6: Form sample estimates for the Hessian of KL divergence / FIM
7: Use conjugate gradient to obtain FIM estimate H^{-1}
8: Estimate step size α using backtracking line search to enforce KL constraint and monotonic improvement
9: Compute the Natural Policy Gradient update

$$\theta_{k+1} = \theta_k + \alpha \, \Delta_k$$

10: **end for**

TATA CONSULTANCY SERVICES

Proximal Policy Optimization

Proximal Policy Optimization

Proximal Policy Optimization is a family of methods that approximately enforce without actually computing the natural gradient

- **Adaptive KL Penalty**

$$\pi_{\theta_{k+1}} = \arg\max_{\pi_\theta} \left[\mathcal{L}_{\pi_{\theta_k}}(\pi_\theta) - \beta \bar{D}_{KL}(\pi_\theta || \pi_{\theta_k}) \right]$$

  Penalty co-efficient β is changed between iterations to approximately enforce KL constraint

- **Clipped Objective** (Simpler to implement, no need to check KL constraint; works well)

$$\mathcal{L}^{CLIP}_{\pi_{\theta_k}}(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_{\theta_k}} [\sum_{t=0}^{T} \min(r_t(\theta) A_t^{\pi_{\theta_k}}, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon) A_t^{\pi_{\theta_k}}]$$

  where $r_t(\theta)$ is the importance sampling ratio between target policy π_θ and behaviour policy π_{θ_k} and policy update takes place as

$$\pi_{\theta_{k+1}} = \arg\max_{\pi_\theta} \mathcal{L}^{CLIP}_{\pi_{\theta_k}}(\pi_\theta)$$