Model-based Safe Deep Reinforcement Learning via a Constrained Proximal Policy Optimization Algorithm

Ashish Kumar Jayant, Prof. Shalabh Bhatnagar

Dept of CSA, IISc Bangalore

October 13, 2022

Table of Contents

Markov Decision Processes (MDP) Framework

- (S, A, R, μ, P) tuple where -
  - S denotes State Space
  - A denotes Action Space
  - $\mu : S \to [0, 1]$ denotes initial state distribution
  - $P : S \times A \times S \to [0, 1]$ denotes transition probability function
  - $R : S \times A \times S \to \mathbb{R}$ denotes single-stage reward function
- $\pi : S \to P(A)$ denotes stationary policy, which maps states to probability distribution over actions
- $\pi(a|s)$ denotes probability of selecting action a in state s
- a_t : Action taken by agent at timestep t
- $s_{t+1} \sim P(.|(s_t, a_t))$, r_t denotes reward received at timestep t

Reinforcement Learning Problem Formulation

- Policy optimization approach
- Policy is parameterized by θ denoted by π_θ
- Search for optimal policy within the set $\prod_\theta \subseteq \prod$ where $\prod$ denotes set of all stationary policies
- Let objective function be -

$$J^R(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu, a_t \sim \pi_\theta, \forall t\right]. \quad (1)$$

- Optimization problem is -

$$\max_{\pi_\theta \in \prod_\theta} J^R(\pi_\theta) \quad (2)$$

- Desirable property during optimization - $\mathbb{E}_t[KL(\pi_{\theta_{old}}(.|s_t), \pi_{\theta_{new}}(.|s_t))] \leq \epsilon$[1]

[1] Sham M. Kakade. "A Natural Policy Gradient". In: *NIPS*. 2001; John Schulman et al. "Trust Region Policy Optimization". In: *ICML* (2015); John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: 1707.06347 [cs.LG].

Unsafe Exploration

- Significant number of random exploratory steps.
- This can lead to potentially dangerous behaviour.
- This limits it application in real-world for e.g Autonomous Driving, Robotics in healthcare, Financial Sequential Decision modelling etc.
- Simulation to real world transfer suffers from out of distribution data.
- Can we limit this unsafe exploration upto some extent?

Constrained Markov Decision Process

- (S, A, R, C_i, μ, P) tuple where -
  - S denotes State Space
  - A denotes Action Space
  - $\mu : S \rightarrow [0, 1]$ denotes initial state distribution
  - $P : S \times A \times S \rightarrow [0, 1]$ denotes transition probability function
  - $R : S \times A \times S \rightarrow \mathbb{R}$ denotes single-stage reward function
  - Cost function is also specified to give an idea about hazardous action.
  - $C_i : S \times A \times S \rightarrow \mathbb{R}^+$ denotes single-stage ith non-negative cost function

Constrained Reinforcement Learning Problem

- Reward objective function is -

$$J^R(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu, a_t \sim \pi_\theta, \forall t\right]. \quad (3)$$

- ith constraint objective function is -

$$J^{C_i}(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu, a_t \sim \pi_\theta, \forall t\right]. \quad (4)$$

- Constrained Optimization Problem -

$$\max_{\pi_\theta \in \prod_\theta} J^R(\pi_\theta) \; s.t \; J^{C_i}(\pi_\theta) \le d_i \; \forall i = 1 \; to \; n,$$

s.t $E_t[KL(\pi_{\theta_{old}}(.|s_t), \pi_{\theta_{new}}(.|s_t))] \le \epsilon$ ..(5)

Real-world Examples

- CMDP framework is more applicable to settings where violations over period of timesteps needs to be constrained.
- e.g Financial settings[2], Wireless Communication[3], Routing[4].

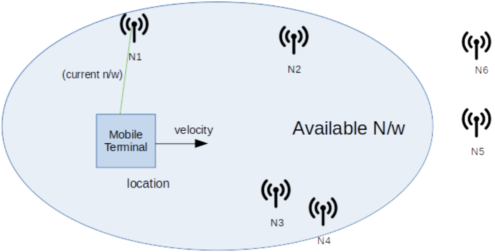
---

[2] Naoki Abe et al. "Optimizing Debt Collections Using Constrained Reinforcement Learning". In: *16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2010.

[3] Alexander Zadorojniy. "Constrained Markov Decision Processes with Application to Wireless Communications". Technion – Israel Institute of Technology, 2004; C. Sun, E. Stevens-Navarro, and V. W. S. Wong. "A Constrained MDP-Based Vertical Handoff Decision Algorithm for 4G Wireless Networks". In: *2008 IEEE International Conference on Communications*. 2008.

[4] Shalabh Bhatnagar and K. Lakshmanan. "An Online Actor-Critic Algorithm with Function Approximation for Constrained Markov Decision Processes". In: *Journal of Optimization Theory and Applications* 153 (June 2012).

Real-world Example : 4G Vertical Handoff Decision Algorithm[5]



State = [location,velocity, battery_status, current_network, available_networks, bandwidths, delays]

Action = Available network Ids to choose from

Reward r(s_t,a_t) = f(gain in bandwidth, benefit in delay, switching cost)

Cost c(s_t,a_t) = access_cost of network i (c_i)

Goal : maximize expected discounted sum of rewards
s.t keeping expected discounted sum of access cost below cost budget

Figure: 4G Vertical Handoff Decision Problem

[5] Sun, Stevens-Navarro, and Wong, "A Constrained MDP-Based Vertical Handoff Decision Algorithm for 4G Wireless Networks".

Lagrangian-based formulation

- Lagrangian of problem 7 is given as (Assuming 1 constraint function for simplicity) -

$$L(\theta, \lambda) = J^R(\pi_\theta) - \lambda(J^C(\pi_\theta) - d) \tag{6}$$

- $\lambda \in \mathbb{R}^+$ is the Lagrange parameter.
- Goal : find a tuple (θ^*, λ^*) of the policy and Lagrange parameters such that[6] -

$$L(\theta^*, \lambda^*) = \max_\theta \min_\lambda L(\theta, \lambda) \tag{7}$$

- Solving the above max-min problem is equivalent to finding a saddle point (θ^*, λ^*) such that $\forall(\theta, \lambda)$ we have -

$$L(\theta^*, \lambda) \geq L(\theta^*, \lambda^*) \geq L(\theta, \lambda^*) \tag{8}$$

[6] Eitan Altman. *Constrained Markov Decision Processes*. Vol. 1. Taylor & Francis, 1998.

Lagrangian-based formulation : Contd.

- Finding a globally optimal saddle point is computationally hard.
- Look for locally optimal saddle point.
- Find $L(\theta^*, \lambda^*)$ such that condition in (8) is satisfied in its local neighbourhood H defined as -

$$H_{\epsilon_1,\epsilon_2} \triangleq \{(\theta, \lambda) |\ \|\theta - \theta^*\| \leq \epsilon_1, \|\lambda - \lambda^*\| \leq \epsilon_2\} \tag{9}$$

  for some $\epsilon_1, \epsilon_2 > 0$
- Gradient Descent Ascent procedure is used to solve for locally optimal saddle point of max-min problem in (7) -

$$\theta_{n+1} = \theta_n + \eta_1(n)\nabla_{\theta_n}(L(\theta_n, \lambda_n)), \tag{10}$$

$$\lambda_{n+1} = [\lambda_n - \eta_2(n)\nabla_{\lambda_n}(L(\theta_n, \lambda_n))]_+. \tag{11}$$

Lagrangian-based Policy Optimization I

- Convergence properties have been studied in tabular CMDP setting[7], Linear Function Approximation setting[8] using two-timescale stochastic approximation.
- Combining updates (10), (11) with PPO[9] gives us PPO-Lagrangian.[10]
- Simple to implement in practice.
- Requires first order optimization which is ideal for training Deep Neural Network based polices.

[7] Vivek S Borkar. "An actor-critic algorithm for constrained Markov decision processes". In: *Systems & control letters* 54.3 (2005), pp. 207–213.

[8] Shalabh Bhatnagar. "An actor-critic algorithm with function approximation for discounted cost constrained Markov decision processes". In: *Systems & Control Letters* 59 (Dec. 2010), pp. 760–766.

[9] Schulman et al., *Proximal Policy Optimization Algorithms*.

[10] Alex Ray, Joshua Achiam, and Dario Amodei. "Benchmarking Safe Exploration in Deep Reinforcement Learning". In: arxiv, 2019.

PPO-Lagrangian[12] Preliminaries I

- Preliminaries
  - Compute reward-to-go, cost-to-go estimates -

$$\hat{R}_t = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{T-t-1} r_T, \tag{12}$$
$$\hat{C}_t = c_{t+1} + \gamma c_{t+2} + \cdots + \gamma^{T-t-1} c_T. \tag{13}$$

  - Let Value function and Cost-Value functions be (V_{ψ_r}, V_{ψ_c})
  - Critic optimization (V_{ψ_r}, V_{ψ_c}) -

$$Loss(\psi_r) = \sum_{t=0}^{T} (V^R_{\psi_r}(s_t) - \hat{R}_t)^2, \tag{14}$$
$$Loss(\psi_c) = \sum_{t=0}^{T} (V^C_{\psi_c}(s_t) - \hat{C}_t)^2. \tag{15}$$

- Further, A^R_t and A^C_t are the estimated advantages based on the reward and cost returns using GAE[11], respectively, by time t

PPO-Lagrangian[12] Preliminaries II

- Estimation of J^R_θ and J^C_θ -

$$J^R_\theta = \mathbb{E}_t[\min(r_t(\theta)A^R_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A^R_t)], \quad (16)$$

$$J^C_\theta = \mathbb{E}_t[\min(r_t(\theta)A^C_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A^C_t)], \quad (17)$$

  where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the ratio of probability of selecting action a_t in state s_t under parameter θ as opposed to θ_{old}.
- ϵ is the clip-ratio which clips $r_t(\theta)$ to $(1-\epsilon)$ if it is less than $(1-\epsilon)$ and clips to $(1+\epsilon)$ if it is greater than $(1+\epsilon)$.

[11] John Schulman et al. "High-Dimensional Continuous Control Using Generalized Advantage Estimation". In: *ICLR* (2016).

PPO-Lagrangian Algorithm[12]

Algorithm PPO Lagrangian

Input: Initialize actor neural net parameter θ_0, critic parameters ψ_{r0}, ψ_{c0}, Lagrange parameter $\lambda_0 \geq 0$, cost threshold = d

for $t = 0, 1, \ldots, T$ **do**

  for *worker* $= 1, \ldots, N$ **do**

    Run policy $\pi_{\theta_{old}}$ in environment for T time steps.

    Compute reward return estimate $\hat{R}_t$, cost return $\hat{C}_t$, advantage estimates A_t^R, A_t^C using (12)-(15)

  end for

  for $k = 1, \ldots K$ **do**

    Compute $J_{\theta_k}^R, J_{\theta_k}^C$ as in (16)-(17)

    Compute $L = \frac{1}{1+\lambda}(J_{\theta_k}^R - \lambda(J_{\theta_k}^C - d))$

    Update parameters θ, λ as in (10)-(11)

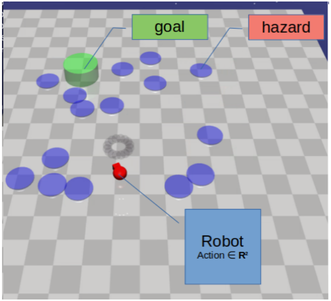
    Update critic parameters ψ_r, ψ_c using gradient of (14)-(15) wrt ψ_r, ψ_c

  end for

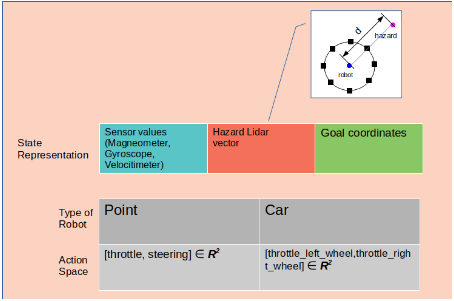
end for

Safety Gym[12]

- We test our and baseline algorithms on Open AI Safety Gym Benchmark. Finite horizon ($T = 1000$) setting.
- Reward Function (R)=($prev\_dist\_goal - curr\_dist\_goal$)
- Cost Function (C)=$\mathbb{1}(hazard\_distance < hazard\_size)$



(a) Safety Gym



(b) Details

[12] Ray, Achiam, and Amodei, "Benchmarking Safe Exploration in Deep Reinforcement Learning".

Model-based RL

- In model based RL we learn the model of environment as well.

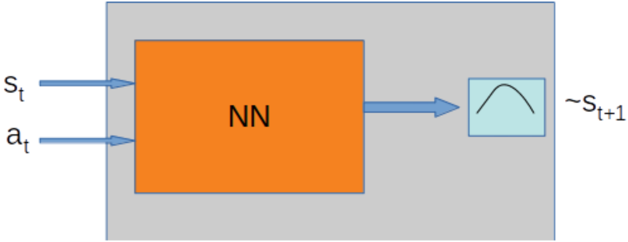


Figure: Environment Model

- Learn an approximate model of true transition (density) $P(s_{t+1}|s_t, a_t)$ say, $P_\alpha(s_{t+1}|s_t, a_t)$ which is parametrized by α.
- Then use it to create "imaginary" rollouts i.e, $s_0 \sim \mu$, $\hat{s_{t+1}} \sim P_\alpha(s_{t+1}|s_t, a_t)$ *for* $t > 0$ while training.

Model-based RL : Contd.

- Problem in (7) becomes :

$$\max_{\pi_\theta \in \Pi_\theta} J_m^R(\pi_\theta) \ s.t \ J_m^{C_i}(\pi_\theta) \le d_i, \ \forall i = 1 \ to \ n, \tag{18}$$

- where,

$$\begin{aligned} J_m^R(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu, \\ s_{t+1} \sim P_\alpha(.|s_t, a_t), \ a_t \sim \pi_\theta, \forall t], \end{aligned} \tag{19}$$

$$\begin{aligned} J_m^{C_i}(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu, \\ s_{t+1} \sim P_\alpha(.|s_t, a_t), a_t \sim \pi_\theta, \forall t], \end{aligned} \tag{20}$$

Model-Based RL Flow : Learning in dreams!k

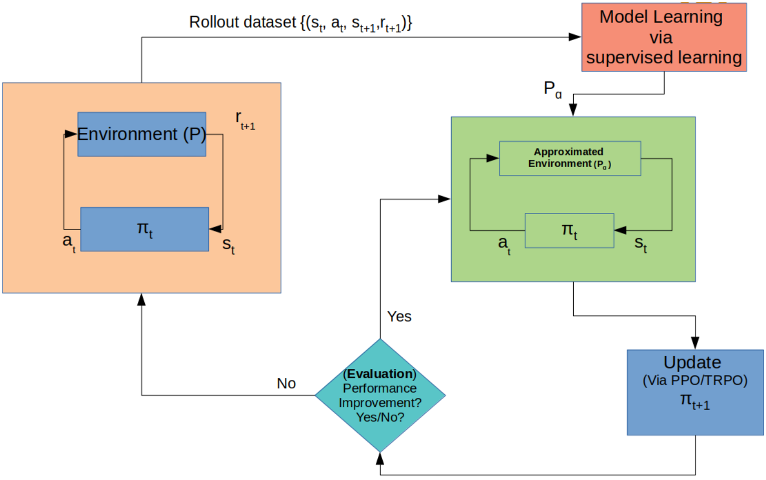


Figure: Model-Based RL

Advantages

- This approach is useful in improving sample efficiency in terms of real environment interactions.[13][14]
- In other words, we can achieve similar level of performance as model-free approach with considerably less real-environment interactions.
- This is a valuable feature to have in Safe RL settings as well.

[13] Michael Janner et al. "When to Trust Your Model: Model-Based Policy Optimization". In: *NIPS* (2019).

[14] Thanard Kurutach et al. "Model-Ensemble Trust-Region Policy Optimization". In: *ICLR* (2018).

Learning environment dynamics: Challenges

- **Aleatoric Uncertainty** - Inherent stochasticity of system.
- After taking action a_t at s_t agent goes to state s_{t+1} with some probability. e.g observation noise
- **Epistemic Uncertainty** - Lack of sufficient knowledge/data. Dataset size $\rightarrow \infty$, epistemic uncertainty $\rightarrow 0$.
- **Model Bias** - Aggregation of error over horizon.

Learning Environment Dynamics: Solutions in Literature

- Train an ensemble of N_m neural networks with different initializations.
- Where each neural network's outputs parametrize a Multivariate Gaussian distribution with diagonal covariance matrix.[15][16]

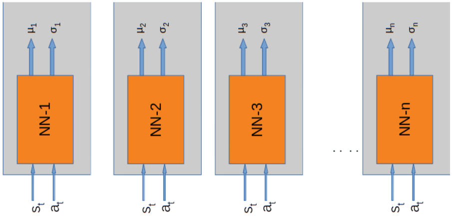


Figure: Ensemble of Uncertainty-aware NNs

[15] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles". In: NIPS. 2017.

[16] Kurtland Chua et al. "Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models". In: *NIPS* (2018).

Learning Environment Dynamics : Contd

- Every ith neural network parametrized by α_i, we learn parametrized Multivariate normal distribution with diagonal covariance matrix Σ whose diagonal entries are $\sigma_i, \ \forall i = 1 \ to \ N_m$:
$P_{\alpha_i}(s_{t+1}|(s_t, a_t)) = \mathcal{N}(\mu_{\alpha_i}(s_t, a_t), \Sigma_{\alpha_i}(s_t, a_t))$.
- Using negative log-likelihood minimization, our loss function for ith neural net becomes -

$$L_{\alpha_i} = \sum_{i=1}^{N} [\mu_{\alpha_i}(s_t, a_t) - s_{t+1})]^T \Sigma_{\alpha_i}^{-1}(s_t, a_t)[\mu_{\alpha_i}(s_t, a_t) - s_{t+1}] + \log |\Sigma_{\alpha_i}(s_t, a_t)| \tag{21}$$

- Collect data tuples $(s_t, a_t, s_{t+1})_{i=1}^{n}$ using current policy.
- Update α_i by minimizing (21) using SGD/ADAM.

Model-based PPO-Lagrangian

- Idea : Combine the benefits of model-based RL and Lagrangian-relaxation based methods.
- Small detail : Evaluating performance of policy without interacting with real environment. We define Performance Ratio (PR) -

$$PR = \frac{1}{N_m} \sum_{i=1}^{N_m} \mathbb{1}(\zeta^R(\alpha_i, \theta_t) > \zeta^R(\alpha_i, \theta_{t-1})) \tag{22}$$

- where $\zeta^R(\alpha_i, \theta_t) = \sum_{t=0}^{T} \gamma^t R(s_t, a_t, s_{t+1}),\ s_0 \sim \mu, \forall t > 0\ :$ $s_{t+1} \sim P_{\alpha_i}(.|s_t, a_t), a_t \sim \pi_{\theta_t}(.|s_t)$
- Reward and cost function are available in closed form. i.e, $r_t = f_1(s_t, a_t, s_{t+1}), c_t = f_2(s_t, a_t, s_{t+1})$ and f_1 and f_2 are available with us.

Solutions to aggregation of error and its issues

- We have a finite horizon setting, horizon T.
- **Approach 1** : Using a truncated horizon ($H < T$).[17]

$$\max_{\pi_\theta \in \Pi_\theta} J^R_{m,H}(\pi_\theta) \ s.t \ J^C_{m,H}(\pi_\theta) \leq d^*, \tag{23}$$

- where,

$$J^R_{m,H}(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{H-1} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu,$$
$$s_{t+1} \sim P_\alpha(.|s_t, a_t), \ a_t \sim \pi_\theta, \forall t], \tag{24}$$

$$J^C_{m,H}(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{H-1} \gamma^t C(s_t, a_t, s_{t+1}) \mid s_0 \sim \mu,$$
$$s_{t+1} \sim P_\alpha(.|s_t, a_t), a_t \sim \pi_\theta, \forall t], \tag{25}$$

[17] Chua et al., "Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models"; Kurutach et al., "Model-Ensemble Trust-Region Policy Optimization"; Janner et al., "When to Trust Your Model:

Contribution : Dealing with underestimation of cost-return

- We have knowledge of initially prescribed threshold d for a finite horizon T.
- Now we have truncated horizon $H < T$.
- Need new and stricter threshold $d^* =?$
- Naive estimation, $d^* = d * \frac{H}{T}$: Leads to cost-limit violations for original horizon T!
- Use a hyperparamter β :

$$\lambda_n = [\lambda_n - \eta_2(n)(J^C(\pi_\theta) - d * \beta)]_+ \tag{26}$$

and we tune β empirically.

Effect of β

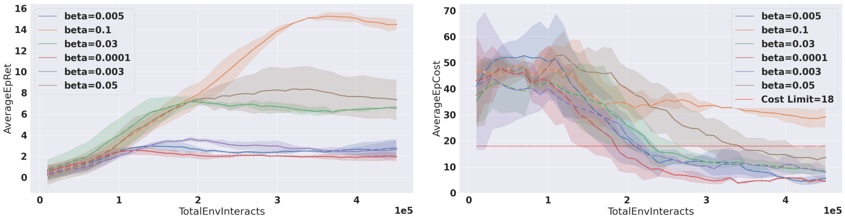


Figure: effect of β in PointGoal

Another approach in literature to deal with aggregation of error : **safeLOOP**

- **Approach 2** : Using approximation of value (or costvalue) function[18]

$$\max_{\pi_\theta \in \Pi_\theta} J^R_{m,L}(\pi_\theta) \ s.t \ J^C_{m,L}(\pi_\theta) \leq d^*, \tag{27}$$

- where,

$$J^R_{m,L}(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{H-1} \gamma^t R(s_t, a_t, s_{t+1}) + \gamma^H V^R(s_H) \mid s_0 \sim \mu, \\ s_{t+1} \sim P_\alpha(.|s_t, a_t), \ a_t \sim \pi_\theta, \forall t], \tag{28}$$

$$J^C_{m,L}(\pi_\theta) = \mathbb{E}[\sum_{t=0}^{H-1} \gamma^t C(s_t, a_t, s_{t+1}) + \gamma^H V^C(s_H) \mid s_0 \sim \mu, \\ s_{t+1} \sim P_\alpha(.|s_t, a_t), a_t \sim \pi_\theta, \forall t], \tag{29}$$

[18] Harshit Sikchi, Wenxuan Zhou, and David Held. "Learning Off-Policy with

Algorithm : Model based PPO-Lagrangian I

Input: Initialize actor neural net parameter θ_0, critic parameters ψ_{r0}, ψ_{c0}, Ensemble models $[P_{\alpha_i}]_{i=1}^{n}$, Lagrange parameter $\lambda_0 \geq 0$, cost threshold = d, Environment Horizon = T, Model Horizon = H

for i =1 . . . N training epochs **do**

  Collect data tuples $(s_t, a_t, s_{t+1})_{i=1}^{n}$ using policy π_{θ_i} in environment for T time steps over multiple episodes $|E|$

  Train $[P_{\alpha_i}]_{i=1}^{n}$ by minimizing (21)

  while Performance ratio $>$ 70% **do**

    $s_0 \sim \mu$

    Collect data rollouts as $a_t \sim \pi_{\theta_i}(.|s_t),\ s_t \sim P_{\alpha_q}(.|s_t, a_t)$ (At each timestep 'q' is randomly selected from 1,2,5..n) for H timesteps $(H < T)$

    Compute $J^{C}_{sample}(\pi_{\theta_t}) = \frac{1}{|E|} \sum_{p=1}^{H} \gamma^{p} C(s_t, a_t)$ where $|E|$ is no. of episodes

Algorithm : Model based PPO-Lagrangian II

```
        Update $\lambda$ by substituting $J^C(\pi_\theta)$ by $J^C_{sample}(\pi_{\theta_i})$ in (26).
        {Multiple gradient updates for actor and critic}
        for k=1...K do
            Compute $J^R(\pi_{\theta_k}), J^C(\pi_{\theta_k})$ as in (28) and (29) respectively.
            Update parameters $\theta_k$ using (10)
            Update critic parameters $\psi_r, \psi_c$ by minimizing (14) and (15) respectively.
        end for
        Compute Performance Ratio (PR) using (22)
    end while
end for
```

Baselines

We compare our approach with the following baselines :

- Unconstrained model-free algorithm - PPO[19]
- PPO-Lagrangian (Model-Free)[20]
- Constrained Policy Optimization (Model-Free)[21]
- Learning Off-Policy with Online Planning (Model-Based)[22]

---

[19] Schulman et al., *Proximal Policy Optimization Algorithms*.

[20] Ray, Achiam, and Amodei, "Benchmarking Safe Exploration in Deep Reinforcement Learning".

[21] Joshua Achiam et al. "Constrained Policy Optimization". In: *ICML* (2017).

[22] Sikchi, Zhou, and Held, "Learning Off-Policy with Online Planning".

Results

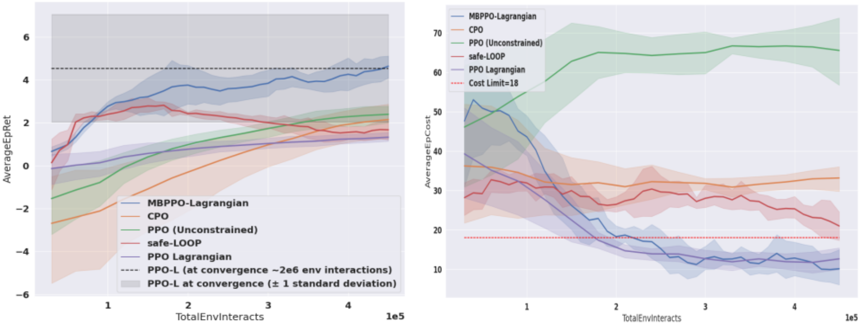


Figure: Performance in PointGoal

Results

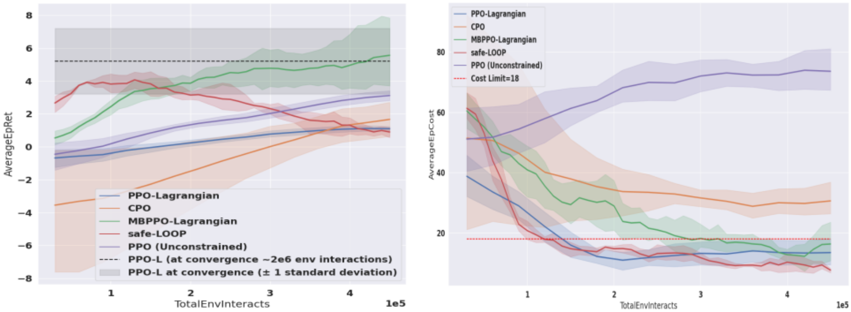


Figure: Performance in CarGoal

Results

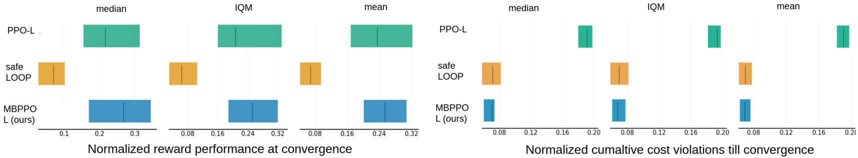


Figure: Normalized Reward Returns at Convergence (left) with median, inter-quartile mean (IQM), mean estimates and Normalized Cumulative Violations (right) with median, inter-quartile mean (IQM), mean estimates. Top rows (in green) represent PPO-Lagrangian, middle rows (in orange) represent safe-LOOP and bottom rows (in blue) represent our approach using rliable[23]

[23] Rishabh Agarwal et al. "Deep reinforcement learning at the edge of the statistical precipice". In: *Advances in Neural Information Processing Systems* 34 (2021).

Results

- Cumulative violations :

$$Cumulative \;\; Violations = \sum_{Till \;\; convergence} \mathbb{1}(C(s_t, a_t) == 1) \tag{30}$$

Approach	Cumulative Violations till convergence
PPO-L	38424 ± 670
MBPPO-L (ours)	12209 ± 798

Table: Cumalative Violations till Convergence on modified PointGoal1

Approach	Cumulative Violations till convergence
PPO-L	38356 ± 2217
MBPPO-L (ours)	15732 ± 1345

Table: Cumulative Violations till Convergence on modified CarGoal1

Limitations

- High running time

Table: Running time for 450k steps (in seconds)

Algorithm	Running time (in s)
PPO-Lagrangian	187.95 ± 7.56
CPO	266.25 ± 6.46
MBPPO-Lagrangian	21420.91 ± 554.449
safe-LOOP	183156.33 ± 19083.43

- High-dimensional spaces envs (e.g in quadrupled robots etc)

Table: Reward Performance at convergence in DoggoGoal

Algorithm	Episodic reward performance of final policy
PPO	21.3 ± 1.23
PPO-Lagrangian	1.6275 ± 0.46
safe-LOOP	-0.14 ± 0.05
MBPPO-Lagrangian	-0.69 ± 0.06

Small insight : Model learning seems to be more challenging in constrained setting

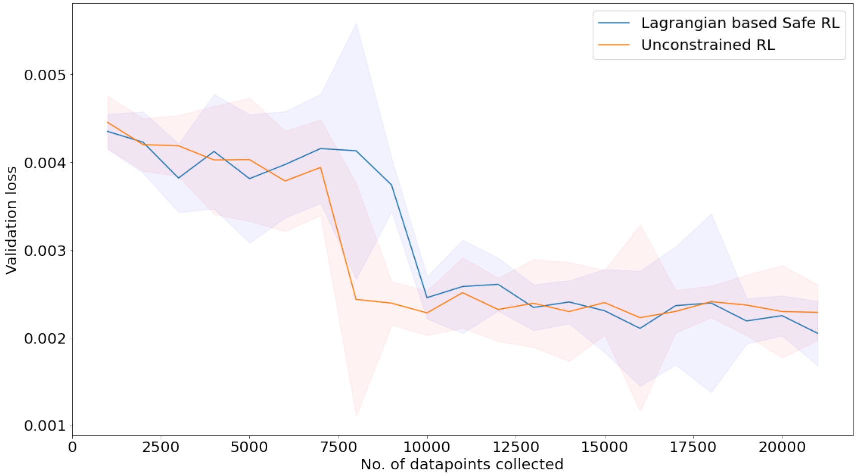


Figure: Quicker convergence of Validation loss in unconstrained setting as compared to constrained setting

Future Work

- Improving reward returns: off-policy[24] + cmdp + model-based?
- Any innovation/novelty in saddle point optimization research will apply for Constrained RL settings too!
- Due to popularity of GANs, this area (saddle point optimization) is being studied in comprehensive way in theoretical settings too. E.g Local convergence of GANs using two-timescale approximation[25]
- Model learning in high-dimensional settings still remains a challenge and even more in constrained exploration!

[24] Raghuram Bharadwaj Diddigi et al. *Neural Network Compatible Off-Policy Natural Actor-Critic Algorithm*. 2022. DOI: 10.48550/ARXIV.2110.10017. URL: https://arxiv.org/abs/2110.10017.

[25] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.