Maximize to Explore (MEX): One Objective Function Fusing Estimation, Planning, and Exploration

Zhihan Liu[1], Miao Lu[2], Wei Xiong[3],
Han Zhong[4], Hao Hu[5], Shenao Zhang[1],
Sirui Zheng[1], Zhuoran Yang[6], and Zhaoran Wang[1]

[1]Northwestern University [2]Stanford University [3]University of Illinois Urbana-Champaign [4]Peking University [5]Tsinghua University [6]Yale University

November 15, 2023

1 Background and Our Contributions

2 Algorithm design: Maximize to Explore (MEX)

3 Deep RL implementations

Challenge of Online Reinforcement Learning

How to maintain a balance between exploration and exploitation?

Typically, a sample-efficient algorithm undertakes three tasks:

1. Estimation: from data to encapsulated knowledge of env.
2. Planning: exploiting the current knowledge
3. Exploration: further exploring the unknown env.

To handle large state space: function approximation. But needs

- solve constrained optimization in data-dependent level-sets
- or sample from complicated posterior over hypotheses

to achieve provable sample-efficiency with general FA.

Incompatible with modern deep RL methods :(

Question

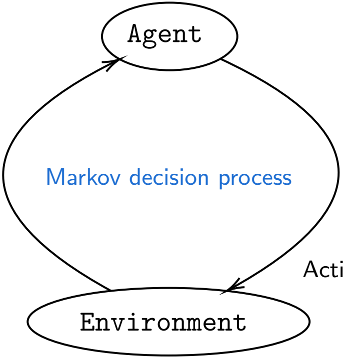
Under general function approximation, can we design a provably sample-efficient and easy-to-implement RL framework to trade off between exploration and exploitation?

Reinforcement Learning 101

One episode: step $h = 1, \cdots, H$

Reward $r_h = r(s_h, a_h)$

Next state $s_{h+1} \sim \mathbb{P}_h(\cdot|s_h, a_h)$



Action $a_h = \pi_h(\cdot|s_h)$

- Goal: learn policy $\pi^\star$ to maximize the expected total reward: $\pi^\star = \arg\max_\pi \{V_1^\pi(s) = \mathbb{E}_{\mathbb{P},\pi}[\sum_{h\in[H]} r_h(s_h, a_h) \mid s_1 = s]\}$.
- Online RL: learn by online interaction $\pi^1, \cdots, \pi^K$.
- Sample efficiency? Regret$(K) = \sum_{k\in[K]} V_1^{\pi^\star}(s_1) - V_1^{\pi^k}(s_1)$.

Contributions

1. Easy-to-implement framework Maximize to Explore (`MEX`):
   - unconstrainedly maximizes a single objective to fuse estimation and planning while automatically trade off between exploration and exploitation.
   - under mild assumptions, `MEX` achieves an $\widetilde{\mathcal{O}}(\sqrt{K})$-regret.
2. Cover various known model-free/model-based tracktable MDP instances. Extension to two-player zero-sum Markov game.
3. Deep RL implementations (both model-free/model-based styles). Experiments on sparse reward MuJoCo environments demonstrate the effectiveness of `MEX`.

1 Background and Our Contributions

2 Algorithm design: Maximize to Explore (MEX)

3 Deep RL implementations

Maximize to Explore (MEX)

At each episode $k \in [K]$, solve $f^k \in \mathcal{H}$ via

$$f^k = \operatorname*{argsup}_{f \in \mathcal{H}} \left\{ V_{1,f}(s_1) - \eta \cdot \sum_{h=1}^{H} L_h^{k-1}(f) \right\}. \tag{1}$$

Then set $\pi^k = \pi_{f^k}$ (optimal policy w.r.t. f^k) to collect data.

- $V_{1,f}(s_1)$: exploration for a higher return
- $-\sum_{h=1}^{H} L_h^{k-1}(f)$: exploitation of agent's current knowledge
- balanced through a fixed coefficient $\eta > 0$.
- Unconstrained optimization problem!
- Theoretically, `MEX` achieves regret of

$$\widetilde{\mathcal{O}}\Big(\texttt{Poly}(H) \cdot d_{\mathsf{GEC}}(1/\sqrt{HK})^{\frac{1}{2}} \cdot K^{\frac{1}{2}}\Big), \tag{2}$$

d_{GEC} is generalized eluder coefficient [1] (Zhong et al, 2022).

1 Background and Our Contributions

2 Algorithm design: Maximize to Explore (MEX)

3 Deep RL implementations

Deep RL Implementations (Model-Based MEX)

- Adapted from MBPO ([2]), Model-Based MEX solves:
$$\max_{\phi}\max_{\pi} \underbrace{\mathbb{E}_{(x,a,r,x')\sim\mathcal{D}}\left[\log \mathbb{P}_\phi(x', r \mid x, a)\right]}_{\text{MBPO Objective}} + \eta' \cdot \underbrace{\mathbb{E}_{x\sim\sigma}\left[V^{\pi}_{\mathbb{P}_\phi}(x)\right]}_{\text{Model Value}},$$
where we denote by $\sigma(\cdot)$ the initial state distribution, and $\mathcal{D}$ the replay buffer.
- We calculate the model gradient $\nabla_\phi \mathbb{E}_{x\sim\sigma}\left[V^{\pi}_{\mathbb{P}_\phi}(x)\right]$ as
$$\mathbb{E}_{\tau^{\pi}_{\phi}}\left[\left(r + \gamma V^{\pi}_{\mathbb{P}_\phi}(x') - Q^{\pi}_{\mathbb{P}_\phi}(x, a)\right) \cdot \nabla_\phi \log \mathbb{P}_\phi(x', r \mid x, a)\right],$$
where τ^{π}_{ϕ} is the trajectory under policy π and transition $\mathbb{P}_\phi$, starting from σ.
- Update the policy π and the model parameter ϕ, iteratively.

Deep RL Implementations (Model-Free MEX)

- Adapted from TD-3 ([3]), Model-Free MEX solves:

$$\max_{\theta} \max_{\pi} \underbrace{-\mathbb{E}_{\beta}\left[\left(r + \gamma Q_{\theta}(x', a') - Q_{\theta}(x, a)\right)^{2}\right]}_{\text{negative TD Loss}}$$
$$+ \eta' \cdot \mathbb{E}_{\beta}\Bigg[\underbrace{\mathbb{E}_{a\sim\pi} Q_{\theta}(x, a)}_{\text{Q-Function}} \underbrace{- \log \sum_{a\in\mathcal{A}} \exp\left(Q_{\theta}(x, a)\right)}_{\text{Stabilizes Training}}\Bigg].$$

- Here, β is the distribution for the off-policy replay buffer.
- Similar to CQL ([4]), term $\log \sum_{a\in\mathcal{A}} \exp\left(Q_{\theta}(x, a)\right)$ is used to stabilize the training.
- Update the policy π and the Q-Function parameter θ, iteratively.

Experiment Setup

- We evaluate the effectiveness of `MEX` by assessing its performance in both standard gym locomotion tasks and sparse reward locomotion and navigation tasks within the MuJoCo ([5]) environment.
- For sparse reward tasks, we select `cheetah-vel`, `walker-vel`, `hopper-vel`, `ant-vel`, and `ant-goal`, where the agent receives a reward *only* when it successfully attains the desired velocity or goal.

Empirical Performance

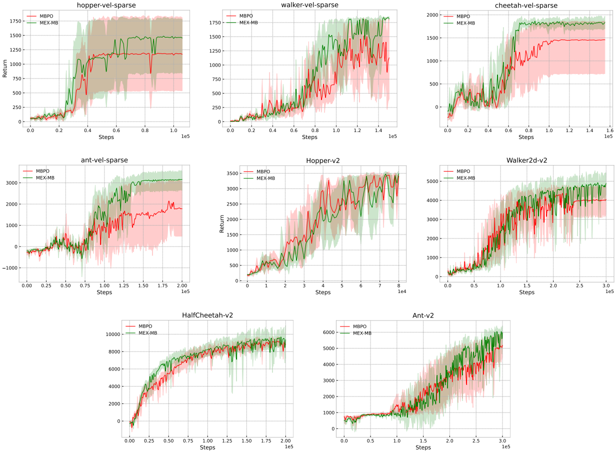


Figure: Model-based MEX-MB in sparse and standard MuJoCo locomotion tasks. (Green line depicts the performance of MEX-MB.)

Empirical Performance

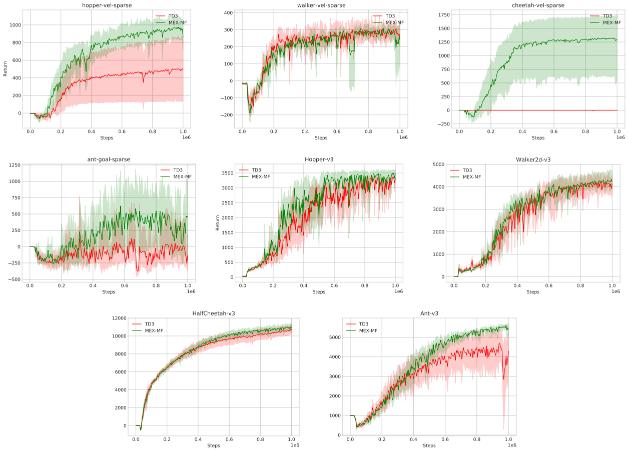


Figure: Model-free `MEX-MF` in sparse and standard MuJoCo locomotion tasks. (Green line depicts the performance of `MEX-MF`.)

Thank You!

Reference

[1] Zhong, Han, et al. "Gec: A unified framework for interactive decision making in mdp, pomdp, and beyond." arXiv preprint arXiv:2211.01962 (2022).

[2] Janner, Michael, et al. "When to trust your model: Model-based policy optimization." Advances in neural information processing systems 32 (2019).

[3] Wu, Jiaolv, et al. "A-TD3: An Adaptive Asynchronous Twin Delayed Deep Deterministic for Continuous Action Spaces." IEEE Access 10 (2022): 128077-128089.

[4] Kumar, Aviral, et al. "Conservative q-learning for offline reinforcement learning." Advances in Neural Information Processing Systems 33 (2020): 1179-1191.

[5] Todorov, Emanuel, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control." 2012 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2012.