Mean Field Games: Numerical Methods and Applications in Machine Learning

Part 7: Mean Field Reinforcement Learning

Mathieu LAURIÈRE

https://mlauriere.github.io/teaching/MFG-PKU-7.pdf

Peking University
Summer School on Applied Mathematics
July 26 – August 6, 2021

RECAP

Outline

From Optimal Control to MFRL

$$\begin{tikzcd}
\text{Mean-Field Control} \arrow[r, "\textit{unknown model}"] & \text{Mean-Field Reinforcement Learning} \\
\text{Optimal Control} \arrow[u, "\substack{N\to\infty \\ MKV}"'] \arrow[r, "\textit{unknown model}"'] & \text{Reinforcement Learning} \arrow[u, "\substack{N\to\infty \\ MKV}"]
\end{tikzcd}$$

Reinforcement Learning – Setup

- **Markov Decision Process (MDP):** $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where:
  - $\mathcal{S}$: state space, $\mathcal{A}$: action space,
  - $p : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$: transition kernel, $p(\cdot|s, a)$ gives next state's distribution
  - $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: reward function, $\gamma \in (0, 1)$: discount factor
- **Goal:** Find (stationary, mixed) policy $\pi^* : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ maximizing:

$$R(\pi) = \mathbb{E}\left[\sum_{n\geq 0} \gamma^n r(s_n, a_n)\right], \qquad \text{with } a_n \sim \pi(\cdot|s_n), s_{n+1} \sim p(\cdot|s_n, a_n)$$

Reinforcement Learning – Setup

- **Markov Decision Process (MDP):** $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where:
  - $\mathcal{S}$: state space, $\mathcal{A}$: action space,
  - $p : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$: transition kernel, $p(\cdot|s, a)$ gives next state's distribution
  - $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: reward function, $\gamma \in (0, 1)$: discount factor
- **Goal:** Find (stationary, mixed) policy π^* : $\mathcal{S} \to \mathcal{P}(\mathcal{A})$ maximizing:

$$R(\pi) = \mathbb{E}\left[\sum_{n \geq 0} \gamma^n r(s_n, a_n)\right], \qquad \text{with } a_n \sim \pi(\cdot|s_n), s_{n+1} \sim p(\cdot|s_n, a_n)$$

- **Model:** p, r

Reinforcement Learning – Setup

- **Markov Decision Process (MDP):** $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where:
  - $\mathcal{S}$: state space, $\mathcal{A}$: action space,
  - $p : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$: transition kernel, $p(\cdot|s, a)$ gives next state's distribution
  - $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: reward function, $\gamma \in (0, 1)$: discount factor
- **Goal:** Find (stationary, mixed) policy $\pi^* : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ maximizing:

$$R(\pi) = \mathbb{E}\left[\sum_{n \geq 0} \gamma^n r(s_n, a_n)\right], \qquad \text{with } a_n \sim \pi(\cdot|s_n), s_{n+1} \sim p(\cdot|s_n, a_n)$$

- **Model:** p, r
- **Two settings:**

  (1) **Known model** : Optimal control theory & methods

  (2) **Sample transitions & rewards**: Reinforcement Learning (RL) framework

Reinforcement Learning – Paradigm

We want to **learn** the best control by performing **experiments** of the form:

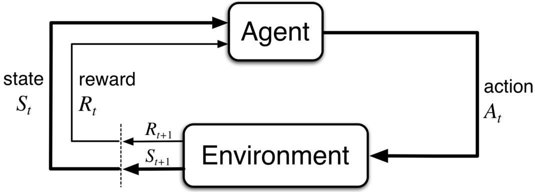
Given the current state S_t,
(1) *Take an action A_t*
(2) *Observe reward R_{t+1} & new state S_{t+1}*

[1] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* MIT press.

Reinforcement Learning – Paradigm

We want to **learn** the best control by performing **experiments** of the form:

Given the current state S_t*,*
(1) *Take an action* A_t
(2) *Observe reward* R_{t+1} *& new state* S_{t+1}



Source: [Sutton, Barto][1]

[1] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* MIT press.

- **Learning the policy:**
  - Policy Gradient

$$\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta^{(\mathtt{k})} \nabla J(\theta^{(\mathtt{k})}), \qquad \pi^{(\mathtt{k})}(a|s) = \pi(s|a, \theta^{(\mathtt{k})})$$

- **Learning the policy:**
  - Policy Gradient

$$\theta^{(\mathtt{k}+1)} = \theta^{(\mathtt{k})} - \eta^{(\mathtt{k})} \nabla J(\theta^{(\mathtt{k})}), \qquad \pi^{(\mathtt{k})}(a|s) = \pi(s|a, \theta^{(\mathtt{k})})$$

  - PPO, TRPO
  - ...

Reinforcement Learning – Methods

- **Learning the policy:**
  - Policy Gradient

$$\theta^{(\mathtt{k}+1)} = \theta^{(\mathtt{k})} - \eta^{(\mathtt{k})} \nabla J(\theta^{(\mathtt{k})}), \qquad \pi^{(\mathtt{k})}(a|s) = \pi(s|a, \theta^{(\mathtt{k})})$$

  - PPO, TRPO
  - ...
- **Learning the value function:**
  - Q-learning

$$Q^*(s, a) = r(s, a) + \gamma \max_{\pi} \mathbb{E}_{a' \sim \pi(\cdot|s), s' \sim p(\cdot|s, a')} \left[Q^*(s', a') \right]$$

Note: $V^*(s) = \max_{a \in \mathcal{A}} Q^*(s, a)$, $v^*(s) = \mathrm{argmax}_{a \in \mathcal{A}} Q^*(s, a)$

Reinforcement Learning – Methods

- **Learning the policy:**
  - Policy Gradient

$$\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta^{(\mathtt{k})} \nabla J(\theta^{(\mathtt{k})}), \qquad \pi^{(\mathtt{k})}(a|s) = \pi(s|a, \theta^{(\mathtt{k})})$$

  - PPO, TRPO
  - ...
- **Learning the value function:**
  - Q-learning

$$Q^*(s, a) = r(s, a) + \gamma \max_{\pi} \mathbb{E}_{a' \sim \pi(\cdot|s), s' \sim p(\cdot|s, a')} \left[Q^*(s', a') \right]$$

    Note: $V^*(s) = \max_{a \in \mathcal{A}} Q^*(s, a)$, $v^*(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s, a)$
  - Deep Q-neural network (DQN)
  - ...

Reinforcement Learning – Methods

- **Learning the policy:**
  - Policy Gradient

$$\theta^{(k+1)} = \theta^{(k)} - \eta^{(k)} \nabla J(\theta^{(k)}), \qquad \pi^{(k)}(a|s) = \pi(s|a, \theta^{(k)})$$

  - PPO, TRPO
  - ...
- **Learning the value function:**
  - Q-learning

$$Q^*(s, a) = r(s, a) + \gamma \max_{\pi} \mathbb{E}_{a' \sim \pi(\cdot|s), s' \sim p(\cdot|s, a')} \Big[Q^*(s', a') \Big]$$

    Note: $V^*(s) = \max_{a \in \mathcal{A}} Q^*(s, a)$, $v^*(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q^*(s, a)$
  - Deep Q-neural network (DQN)
  - ...
- **Hybrid:**
  - Deep Deterministic Policy Gradient (DDPG)
  - Soft Actor Critic (SAC)
  - ...

Outline

Problem Formulation

Generic Mean Field model: for a typical infinitesimal agent

- **Dynamics:** discrete time

$$X_{n+1}^{\alpha,\mu} = F(X_n^{\alpha,\mu}, \alpha_n, \mu_n, \epsilon_{n+1}, \epsilon_{n+1}^0), \quad n \geq 0, \qquad X_0^{\alpha,\mu} \sim \mu_0$$

◇ $X_n^{\alpha,\mu} \in \mathcal{X} \subseteq \mathbb{R}^d$: state, $\alpha_n \in \mathcal{U} \subseteq \mathbb{R}^k$: action

◇ $\epsilon_n \sim \nu$: idiosyncratic noise, $\epsilon_n^0 \sim \nu^0$: common noise (random env.)

◇ $\mu_n \in \mathcal{P}(\mathcal{X} \times \mathcal{A})$: a state-action distribution

◇ π_n: a policy; randomized actions: $\alpha_n \sim \pi_n(\cdot|s_n, \mu_n)$

Problem Formulation

Generic Mean Field model: for a typical infinitesimal agent

- **Dynamics:** discrete time

$$X^{\alpha,\mu}_{n+1} = F(X^{\alpha,\mu}_n, \alpha_n, \mu_n, \epsilon_{n+1}, \epsilon^0_{n+1}), \quad n \geq 0, \qquad X^{\alpha,\mu}_0 \sim \mu_0$$

  ◇ $X^{\alpha,\mu}_n \in \mathcal{X} \subseteq \mathbb{R}^d$: state, $\alpha_n \in \mathcal{U} \subseteq \mathbb{R}^k$: action

  ◇ $\epsilon_n \sim \nu$: idiosyncratic noise, $\epsilon^0_n \sim \nu^0$: common noise (random env.)

  ◇ $\mu_n \in \mathcal{P}(\mathcal{X} \times \mathcal{A})$: a state-action distribution

  ◇ π_n: a policy; randomized actions: $\alpha_n \sim \pi_n(\cdot | s_n, \mu_n)$

- **Cost:** $\mathbb{J}(\pi; \mu) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X^{\alpha,\mu}_n, \alpha_n, \mu_n\right)\right]$

Generic Mean Field model: for a typical infinitesimal agent

- **Dynamics:** discrete time

$$X_{n+1}^{\alpha,\mu} = F(X_n^{\alpha,\mu}, \alpha_n, \mu_n, \epsilon_{n+1}, \epsilon_{n+1}^0), \quad n \geq 0, \qquad X_0^{\alpha,\mu} \sim \mu_0$$

◇ $X_n^{\alpha,\mu} \in \mathcal{X} \subseteq \mathbb{R}^d$: state, $\alpha_n \in \mathcal{U} \subseteq \mathbb{R}^k$: action

◇ $\epsilon_n \sim \nu$: idiosyncratic noise, $\epsilon_n^0 \sim \nu^0$: common noise (random env.)

◇ $\mu_n \in \mathcal{P}(\mathcal{X} \times \mathcal{A})$: a state-action distribution

◇ π_n: a policy; randomized actions: $\alpha_n \sim \pi_n(\cdot|s_n, \mu_n)$

- **Cost:** $\mathbb{J}(\pi; \mu) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^{\alpha,\mu}, \alpha_n, \mu_n\right)\right]$

Two scenarios:

- **Cooperative (MFC):** Find π^* s.t.

$$\pi^* \text{ minimizes } \pi \mapsto J^{MFC}(\pi) = \mathbb{J}(\pi; \mu^\pi) \text{ where } \mu_n^\pi = \mathbb{P}^0_{X_n^{\alpha,\mu^\pi}}$$

Problem Formulation

Generic Mean Field model: for a typical infinitesimal agent

- **Dynamics:** discrete time

$$X_{n+1}^{\alpha,\mu} = F(X_n^{\alpha,\mu}, \alpha_n, \mu_n, \epsilon_{n+1}, \epsilon_{n+1}^0), \quad n \geq 0, \qquad X_0^{\alpha,\mu} \sim \mu_0$$

  - ◇ $X_n^{\alpha,\mu} \in \mathcal{X} \subseteq \mathbb{R}^d$: state, $\alpha_n \in \mathcal{U} \subseteq \mathbb{R}^k$: action
  - ◇ $\epsilon_n \sim \nu$: idiosyncratic noise, $\epsilon_n^0 \sim \nu^0$: common noise (random env.)
  - ◇ $\mu_n \in \mathcal{P}(\mathcal{X} \times \mathcal{A})$: a state-action distribution
  - ◇ π_n: a policy; randomized actions: $\alpha_n \sim \pi_n(\cdot|s_n, \mu_n)$

- **Cost:** $\mathbb{J}(\pi; \mu) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^{\alpha,\mu}, \alpha_n, \mu_n\right)\right]$

Two scenarios:

- **Cooperative (MFC):** Find π^* s.t.

$$\pi^* \text{ minimizes } \pi \mapsto J^{MFC}(\pi) = \mathbb{J}(\pi; \mu^\pi) \text{ where } \mu_n^\pi = \mathbb{P}^0_{X_n^{\alpha,\mu^\pi}}$$

- **Non-Cooperative (MFG):** Find $(\hat{\pi}, \hat{\mu})$ s.t.

$$\begin{cases} \hat{\pi} \text{ minimizes } \pi \mapsto J^{MFG}(\pi; \hat{\mu}) = \mathbb{J}(\pi; \hat{\mu}) \\ \hat{\mu}_n = \mathbb{P}^0_{X_n^{\hat{\alpha},\hat{\mu}}} \end{cases}$$

Mean Field MDP

Key Remark:

$$\alpha^* \in \operatorname*{argmin}_{\alpha} J^{MFC}(\alpha) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^{\alpha}, \alpha_n, \mu_n^{\pi}\right)\right], \qquad \mu_n^{\pi} = \mathbb{P}^0_{X_n^{\alpha}}$$

Mean Field MDP

Key Remark:

$$\alpha^* \in \underset{\alpha}{\operatorname{argmin}}\, J^{MFC}(\alpha) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^{\alpha}, \alpha_n, \mu_n^{\pi}\right)\right], \qquad \mu_n^{\pi} = \mathbb{P}^0_{X_n^{\alpha}}$$

$$= \mathbb{E}_{\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n \underbrace{\int_{\mathcal{X}\times\mathcal{U}} f\left(x, a, \mu_n^{\pi}\right) \nu_n^{\pi}(dx, da)}_{\text{function of } \nu_n^{\pi}}\right]$$

Mean Field MDP

Key Remark:

$$\alpha^* \in \underset{\alpha}{\operatorname{argmin}}\, J^{MFC}(\alpha) = \mathbb{E}_{\epsilon,\epsilon^0}\Big[\sum_{n=0}^{\infty} \gamma^n f\big(X_n^{\alpha}, \alpha_n, \mu_n^{\pi}\big)\Big], \qquad \mu_n^{\pi} = \mathbb{P}^0_{X_n^{\alpha}}$$

$$= \mathbb{E}_{\epsilon^0}\Big[\sum_{n=0}^{\infty} \gamma^n \underbrace{\int_{\mathcal{X}\times\mathcal{U}} f\big(x, a, \mu_n^{\pi}\big)\, \nu_n^{\pi}(dx, da)}_{\text{function of } \nu_n^{\pi}}\Big]$$

- **Lifted problem:** population / social planner's optimization problem:
  - $\rightarrow$ state = population distribution μ_n^{π}
  - $\rightarrow$ value function = function of the distribution μ

Mean Field MDP

Key Remark:

$$\alpha^* \in \underset{\alpha}{\operatorname{argmin}}\, J^{MFC}(\alpha) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^\alpha, \alpha_n, \mu_n^\pi\right)\right], \qquad \mu_n^\pi = \mathbb{P}^0_{X_n^\alpha}$$

$$= \mathbb{E}_{\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n \underbrace{\int_{\mathcal{X}\times\mathcal{U}} f\left(x, a, \mu_n^\pi\right) \nu_n^\pi(dx, da)}_{\text{function of } \nu_n^\pi}\right]$$

- **Lifted problem:** population / social planner's optimization problem:
  - → state = population distribution μ_n^π
  - → value function = function of the distribution μ
- **Mean Field Markov Decision Process (MFMDP):** $(\bar{\mathcal{S}}, \bar{\mathcal{A}}, \bar{p}, \bar{r}, \gamma)$, where:
  - State space: $\bar{\mathcal{S}} = \mathcal{P}(\mathcal{X})$
  - Action space: $\bar{\mathcal{A}} = \mathcal{P}(\mathcal{X} \times \mathcal{U})$ with constraint: $pr_1(\bar{a}) = \mu$
  - Transition function: $\mu' = \bar{F}(\mu, \bar{a}, \epsilon^0) \sim \bar{p}(\mu, \bar{a})$
  - Reward function: $\bar{r}(\mu, \bar{a}) = -\int_{\mathcal{X}\times\mathcal{U}} f(x, a, \mu)\bar{a}(dx, da)$

Mean Field MDP

Key Remark:

$$\alpha^* \in \operatorname*{argmin}_{\alpha} J^{MFC}(\alpha) = \mathbb{E}_{\epsilon,\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n f\left(X_n^\alpha, \alpha_n, \mu_n^\pi\right)\right], \qquad \mu_n^\pi = \mathbb{P}^0_{X_n^\alpha}$$

$$= \mathbb{E}_{\epsilon^0}\left[\sum_{n=0}^{\infty} \gamma^n \underbrace{\int_{\mathcal{X}\times\mathcal{U}} f\left(x, a, \mu_n^\pi\right) \nu_n^\pi(dx, da)}_{\text{function of } \nu_n^\pi}\right]$$

- **Lifted problem:** population / social planner's optimization problem:
  - $\rightarrow$ state = population distribution μ_n^π
  - $\rightarrow$ value function = function of the distribution μ
- **Mean Field Markov Decision Process (MFMDP):** $(\bar{\mathcal{S}}, \bar{\mathcal{A}}, \bar{p}, \bar{r}, \gamma)$, where:
  - State space: $\bar{\mathcal{S}} = \mathcal{P}(\mathcal{X})$
  - Action space: $\bar{\mathcal{A}} = \mathcal{P}(\mathcal{X} \times \mathcal{U})$ with constraint: $pr_1(\bar{a}) = \mu$
  - Transition function: $\mu' = \bar{F}(\mu, \bar{a}, \epsilon^0) \sim \bar{p}(\mu, \bar{a})$
  - Reward function: $\bar{r}(\mu, \bar{a}) = -\int_{\mathcal{X}\times\mathcal{U}} f(x, a, \mu)\bar{a}(dx, da)$
- **Goal:** max. $\bar{J}^{\bar{\pi}}(\mu) = \mathbb{E}\left[\sum_{n=0}^{\infty} \gamma^n \bar{r}\left(\mu_n^{\bar{\pi}}, \bar{a}_n\right)\right]$, $\bar{a}_n \sim \bar{\pi}(\cdot|\mu_n^{\bar{\pi}})$, $\mu_{n+1}^{\bar{\pi}} \sim \bar{p}(\cdot|\mu_n^{\bar{\pi}}, \bar{a}_n)$, $\mu_0^{\bar{\pi}} = \mu$

Dynamic Programming Principle (DPP)

Theorem: DPP for MFMDP [Carmona, L., Tan'21]

Under suitable conditions,

$$\bar{J}^*(\mu) := \sup_{\bar{\pi}} \bar{J}^{\bar{\pi}}(\mu) = \sup_{\bar{\pi}} \left\{ \int_{\bar{\mathcal{A}}} \left[\bar{r}(\mu, \bar{a}) + \gamma \mathbb{E}\left[\bar{J}^*\left(\bar{F}(\mu, \bar{a}, \epsilon^0) \right) \right] \right] \bar{\pi}(d\bar{a}|\mu) \right\},$$

where the sup is over a subset of $\{\bar{\pi} : \bar{\mathcal{S}} \to \mathcal{P}(\bar{\mathcal{A}})\}$

Likewise for **mean field state-action value function** $\bar{Q}^*$

Dynamic Programming Principle (DPP)

Theorem: DPP for MFMDP [Carmona, L., Tan'21]

Under suitable conditions,

$$\bar{J}^*(\mu) := \sup_{\bar{\pi}} \bar{J}^{\bar{\pi}}(\mu) = \sup_{\bar{\pi}} \left\{ \int_{\bar{\mathcal{A}}} \left[\bar{r}(\mu, \bar{a}) + \gamma \mathbb{E}\left[\bar{J}^* \left(\bar{F}(\mu, \bar{a}, \epsilon^0) \right) \right] \right] \bar{\pi}(d\bar{a}|\mu) \right\},$$

where the sup is over a subset of $\{\bar{\pi} : \bar{\mathcal{S}} \to \mathcal{P}(\bar{\mathcal{A}})\}$

Likewise for **mean field state-action value function** $\bar{Q}^*$

Proof: based on "double lifting" [Bertsekas, Shreve'78]

Dynamic Programming Principle (DPP)

Theorem: DPP for MFMDP [Carmona, L., Tan'21]

Under suitable conditions,

$$\bar{J}^*(\mu) := \sup_{\bar{\pi}} \bar{J}^{\bar{\pi}}(\mu) = \sup_{\bar{\pi}} \left\{ \int_{\bar{\mathcal{A}}} \left[\bar{r}(\mu, \bar{a}) + \gamma \mathbb{E}\left[\bar{J}^*\left(\bar{F}(\mu, \bar{a}, \epsilon^0) \right) \right] \right] \bar{\pi}(d\bar{a}|\mu) \right\},$$

where the sup is over a subset of $\{\bar{\pi} : \bar{\mathcal{S}} \to \mathcal{P}(\bar{\mathcal{A}})\}$

Likewise for **mean field state-action value function** $\bar{Q}^*$

Proof: based on "double lifting" [Bertsekas, Shreve'78]

DPPs for MFC: [L., Pironneau; Pham, Wei; Gast *et al.*; Guo *et al.*; Motte, Pham;. . .]

Dynamic Programming Principle (DPP)

Theorem: DPP for MFMDP [Carmona, L., Tan'21]

Under suitable conditions,

$$\bar{J}^*(\mu) := \sup_{\bar{\pi}} \bar{J}^{\bar{\pi}}(\mu) = \sup_{\bar{\pi}} \left\{ \int_{\bar{\mathcal{A}}} \left[\bar{r}(\mu, \bar{a}) + \gamma \mathbb{E}\left[\bar{J}^*\left(\bar{F}(\mu, \bar{a}, \epsilon^0) \right) \right] \right] \bar{\pi}(d\bar{a}|\mu) \right\},$$

where the sup is over a subset of $\{\bar{\pi} : \bar{\mathcal{S}} \to \mathcal{P}(\bar{\mathcal{A}})\}$

Likewise for **mean field state-action value function** $\bar{Q}^*$

Proof: based on "double lifting" [Bertsekas, Shreve'78]

DPPs for MFC: [L., Pironneau; Pham, Wei; Gast *et al.*; Guo *et al.*; Motte, Pham;. . .]

Here: discrete time, infinite horizon, common noise, feedback controls, . . .

$\rightarrow$ well-suited for **RL**

$\rightarrow$ Mean-field Q-learning algorithm

Mean Field Learning Settings

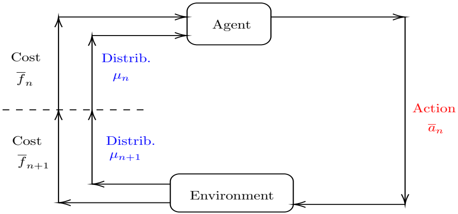
Hierarchy of settings:

- **Setting 1: known model**: computational method based on knowledge of MFMDP
  (a) Gradient based methods
  (b) Dynamic programming based methods

Mean Field Learning Settings

Hierarchy of settings:

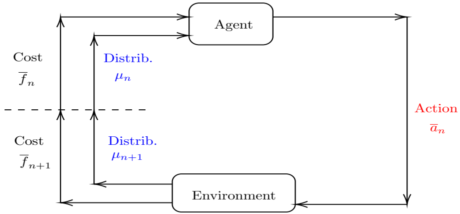
- **Setting 1: known model**: computational method based on knowledge of MFMDP
  - (a) Gradient based methods
  - (b) Dynamic programming based methods
- **Setting 2:** unknown model but **samples from MFMDP**: MF learning



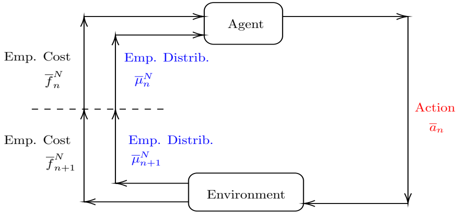
Mean Field Learning Settings

Hierarchy of settings:

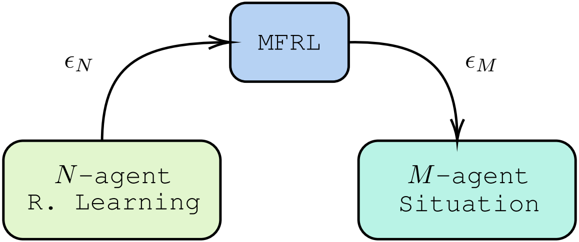
- **Setting 1: known model**: computational method based on knowledge of MFMDP
  - (a) Gradient based methods
  - (b) Dynamic programming based methods
- **Setting 2:** unknown model but **samples from MFMDP**: MF learning



- **Setting 3:** unknown model but **samples from N-agent MDP**: approx. MF learning



Mean Field Control: Finite Population Approximation



Outline

Algorithm

Idea 1: *Make the "policy gradient" approach model-free*

Policy Gradient (PG) to minimize $J(\theta)$

- Control $\approx$ parameterized function
- Look for the optimal parameter θ^*
- Perform gradient descent on the space of parameters

Algorithm

Idea 1: *Make the "policy gradient" approach model-free*

Policy Gradient (PG) to minimize $J(\theta)$

- Control $\approx$ parameterized function
- Look for the optimal parameter θ^*
- Perform gradient descent on the space of parameters

Hierarchy of three situations, more and more complex:

(1) access to the exact **(mean field) model**: $\theta^{(\mathtt{k}+1)} = \theta^{(\mathtt{k})} - \eta \nabla J(\theta^{(\mathtt{k})})$

Algorithm

Idea 1: *Make the "policy gradient" approach model-free*

Policy Gradient (PG) to minimize $J(\theta)$

- Control $\approx$ parameterized function
- Look for the optimal parameter θ^*
- Perform gradient descent on the space of parameters

Hierarchy of three situations, more and more complex:

(1) access to the exact **(mean field) model**: $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \nabla J(\theta^{(\mathtt{k})})$

(2) access to a **mean field simulator**:
$\rightarrow$ idem + gradient estimation (0^{th}-order opt.): $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \widetilde{\nabla} J(\theta^{(\mathtt{k})})$

Algorithm

Idea 1: *Make the "policy gradient" approach model-free*

Policy Gradient (PG) to minimize $J(\theta)$

- Control $\approx$ parameterized function
- Look for the optimal parameter θ^*
- Perform gradient descent on the space of parameters

Hierarchy of three situations, more and more complex:

(1) access to the exact **(mean field) model**: $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \nabla J(\theta^{(\mathtt{k})})$

(2) access to a **mean field simulator**:
$\rightarrow$ idem + gradient estimation (0^{th}-order opt.): $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \widetilde{\nabla} J(\theta^{(\mathtt{k})})$

(3) access to a N-agent **population simulator**:
$\rightarrow$ idem + error on mean $\approx$ empirical mean (LLN): $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \widetilde{\nabla}^N J(\theta^{(\mathtt{k})})$

Algorithm

Idea 1: *Make the "policy gradient" approach model-free*

Policy Gradient (PG) to minimize $J(\theta)$

- Control $\approx$ parameterized function
- Look for the optimal parameter θ^*
- Perform gradient descent on the space of parameters

Hierarchy of three situations, more and more complex:

(1) access to the exact **(mean field) model**: $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \nabla J(\theta^{(\mathtt{k})})$

(2) access to a **mean field simulator**:
$\rightarrow$ idem + gradient estimation (0^{th}-order opt.): $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \widetilde{\nabla} J(\theta^{(\mathtt{k})})$

(3) access to a N-agent **population simulator**:
$\rightarrow$ idem + error on mean $\approx$ empirical mean (LLN): $\theta^{(\mathtt{k+1})} = \theta^{(\mathtt{k})} - \eta \widetilde{\nabla}^N J(\theta^{(\mathtt{k})})$

Theorem: For **Linear-Quadratic** MFC [Carmona, L., Tan'19]

In each case, convergence holds at a linear rate:

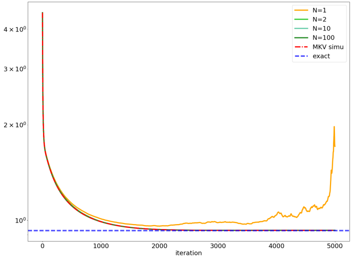
Taking $\mathtt{k} \approx \mathcal{O}\Big(\log(1/\epsilon)\Big)$ is sufficient to ensure $J(\theta^{(\mathtt{k})}) - J(\theta^*) < \epsilon$.

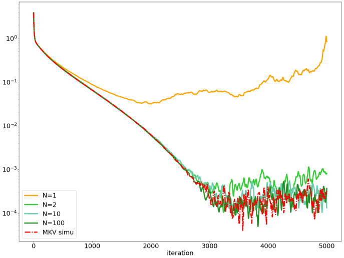
Proof: builds on [Fazel et al.'18], analysis of perturbation of Riccati equations

Numerical Illustration

Example: Linear dynamics, quadratic costs of the type:

$$f(x, \mu, v) = \underbrace{(\bar{\mu} - x)^2}_{\text{distance to mean position}} + \underbrace{v^2}_{\text{cost of moving}}, \qquad \bar{\mu} = \underbrace{\int \mu(\xi) d\xi}_{\text{mean position}},$$





Value of the MF cost

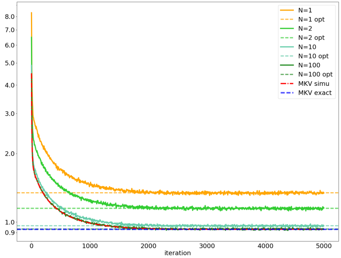
Rel. err. on MF cost

MF cost = cost in the mean field problem

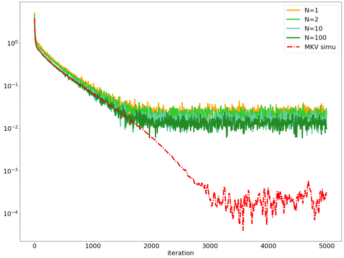
Numerical Illustration

Example: Linear dynamics, quadratic costs of the type:

$$f(x,\mu,v) = \underbrace{(\bar{\mu}-x)^2}_{\text{distance to mean position}} + \underbrace{v^2}_{\text{cost of moving}}, \qquad \bar{\mu} = \underbrace{\int \mu(\xi)d\xi}_{\text{mean position}},$$



Value of the social cost



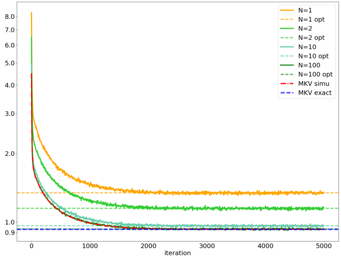
Rel. err. on social cost

Social cost = average over the N-agents

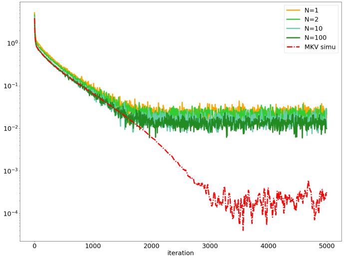
Numerical Illustration

Example: Linear dynamics, quadratic costs of the type:

$$f(x,\mu,v) = \underbrace{(\bar{\mu}-x)^2}_{\text{distance to mean position}} + \underbrace{v^2}_{\text{cost of moving}}, \qquad \bar{\mu} = \underbrace{\int \mu(\xi)d\xi}_{\text{mean position}},$$



Value of the social cost



Rel. err. on social cost

Social cost = average over the N-agents

Main take-away:

Trying to learn the mean-field regime solution can be efficient even for N small

Outline

Idea 2: *Generalize Q-learning to mean-field control*