Generalization in Mean Field Games by Learning Master Policies

Sarah Perrin[1, ⋆], **Mathieu Laurière**[2, ⋆], **Julien Pérolat**[3], **Romuald Élie**[3], **Matthieu Geist**[2, †], **Olivier Pietquin**[1, †*]

[1] Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL
[2] Google Research, Brain Team
[3] DeepMind Paris
sarah.perrin@inria.fr, {lauriere, mfgeist, pietquin}@google.com, {perolat, relie}@deepmind.com

Abstract

Mean Field Games (MFGs) can potentially scale multi-agent systems to extremely large populations of agents. Yet, most of the literature assumes a single initial distribution for the agents, which limits the practical applications of MFGs. Machine Learning has the potential to solve a wider diversity of MFG problems thanks to generalizations capacities. We study how to leverage these generalization properties to learn policies enabling a typical agent to behave optimally against any population distribution. In reference to the Master equation in MFGs, we coin the term "Master policies" to describe them and we prove that a single Master policy provides a Nash equilibrium, whatever the initial distribution. We propose a method to learn such Master policies. Our approach relies on three ingredients: adding the current population distribution as part of the observation, approximating Master policies with neural networks, and training via Reinforcement Learning and Fictitious Play. We illustrate on numerical examples not only the efficiency of the learned Master policy but also its generalization capabilities beyond the distributions used for training.

Introduction

Although learning in games has a long history (Shannon 1959), most of recent breakthroughs remain limited to a small number of players, *e.g.*, for chess (Campbell, Hoane Jr, and Hsu 2002), Go (Silver et al. 2016), poker (Brown and Sandholm 2018; Moravčík et al. 2017) or even video games such as Starcraft (Vinyals et al. 2019) with a large number of agents but only a handful of competing players. Learning in games involving a large number of players remains one of the challenges of modern game theory. Recently, Mean Field Games (MFGs), introduced concurrently by Lasry and Lions (2007) and Huang et al. (2006), have been considered as a promising approach to address this problem. They indeed model games with an infinite number of players. Instead of taking into account interactions between individuals, MFGs model the interaction between a so-called representative agent (sampled from the population distribution) and the full population itself. As in many multi-player games, solving an MFG boils down to finding a Nash equilibrium. Intuitively, it corresponds to a situation where no player can increase their reward (or decrease their cost) by changing their strategy, given that other players keep their current behavior. MFGs are classically described with a forward-backward system of partial differential equations (PDEs) or stochastic differential equations (SDEs) and can only be solved analytically in some specific cases. When an analytical solution is not available, numerical methods such as finite differences can be called to solve the PDE system. However, these techniques do not scale well with the dimensions of the state and action spaces. Another issue with PDE methods is that they are very sensitive to initial conditions. Especially, the policy obtained is only valid for a single initial distribution μ_0 for the population over the state space. This is a strong limitation for practical applications. For example, in an evacuation or traffic-flow scenario, the solution found by a PDE solver could potentially lead to an unforeseen congestion if the agents are not initially distributed as the model expected. This could have dramatic consequences. On the other hand, solving for every possible initial distribution is of course infeasible. Following the traditional trend in the literature, even solutions to MFGs that use most recent Machine Learning methods consider that the initial distribution is fixed and thus compute policies that are agnostic to the current population. A sensible idea to alleviate the sensitivity issue is to incorporate the population as part of the observation for the representative agent, such that it can behave optimally against the population, and not only w.r.t. its current state. Yet, using such a modification of the observation cannot be done seamlessly as the uniqueness of the initial distribution is a core assumption of existing methods, including very recent ones based on Machine Learning.

Here we do a first crucial step in this direction using Deep Reinforcement Learning (Deep RL), which sounds particularly well fitted to overcome the aforementioned difficulty. Our core contribution is to propose the first Deep RL algorithm that calculates an optimal policy independently of the initial population distribution.

Main contributions. First, we extend the basic framework of MFGs by introducing a class of *population-dependent policies* enabling agents to react to any population distribution. Within this class, we identify a *Master policy* and establish its connection with standard population-agnostic policies arising in MFG Nash equilibria (Thm. 1).

[*] equal contribution, [†] equal contribution

Second, we propose an algorithm, based on Fictitious Play and Deep RL, to learn a Master policy. We analyze a continuous time version of Fictitious Play and prove convergence at a linear rate (Thm. 2). Last, we provide empirical evidence that not only this method learns the Master policy on a training set of distributions, but that the learned policy *generalizes* to unseen distributions. Our approach is the first to tackle this question in the literature on MFGs.

Background and Related Works

We consider a finite state space X and finite action space A. The set of probability distributions on X and A are denoted by Δ_X and Δ_A. Let $p : X \times A \times \Delta_X \to \Delta_X$ be a transition probability function and $r : X \times A \times \Delta_X \to \mathbb{R}$ be a reward function. Let $\gamma \in (0, 1)$ be a discount parameter. In this section, we introduce the key concepts needed to explain our main contributions. Although there is no prior work tackling explicitly the question of generalization in MFG, we review along the way several related studies.

Mean Field Games

In the usual MFG setup (Lasry and Lions 2007; Huang et al. 2006), a stationary policy is a function $\pi : X \to \Delta_A$ and a non-stationary policy $\boldsymbol{\pi}$ is an infinite sequence of stationary policies. Let Π and $\boldsymbol{\Pi} = \Pi^{\mathbb{N}}$ be the sets of stationary and non-stationary policies respectively. Unless otherwise specified, by policy we mean a non-stationary policy. A mean-field (MF) state is a $\mu \in \Delta_X$. It represents the state of the population at one time step. An MF flow $\boldsymbol{\mu}$ is an infinite sequence of MF states. We denote by $M = \Delta_X$ and $\mathbf{M} = M^{\mathbb{N}}$ the sets of MF states and MF flows. For $\mu \in M$, $\pi \in \Pi$, let

$$\phi(\mu, \pi) : x \mapsto \sum_{x' \in X} p(x|x', \pi(x'), \mu)\mu(x')$$

denote the next MF state. The MF flow starting from μ_0 and controlled by $\boldsymbol{\pi} \in \boldsymbol{\Pi}$ is denoted by $\Phi(\mu_0, \boldsymbol{\pi}) \in \mathbf{M}$:

$$\Phi(\mu_0, \boldsymbol{\pi})_0 = \mu_0, \ \Phi(\mu_0, \boldsymbol{\pi})_{n+1} = \phi(\Phi(\mu_0, \boldsymbol{\pi})_n, \boldsymbol{\pi}_n), n \geq 0.$$

Facing such a population behavior, an infinitesimal agent seeks to solve the following Markov Decision Process (MDP). Given an initial μ_0 and a flow $\boldsymbol{\mu}$, maximize:

$$\boldsymbol{\pi} \mapsto J(\mu_0, \boldsymbol{\pi}; \boldsymbol{\mu}) = \mathbb{E}\Big[\sum_{n=0}^{+\infty} \gamma^n r(x_n, a_n, \boldsymbol{\mu}_n)\Big],$$

subject to: $x_0 \sim \mu_0$, $x_{n+1} \sim p(.|x_n, a_n, \boldsymbol{\mu}_n)$, $a_n \sim \boldsymbol{\pi}_n(.|x_n)$. Note that, at time n, the reward and transition depend on the current MF state $\boldsymbol{\mu}_n$. So this MDP is non-stationary but since the MF flow $\boldsymbol{\mu}$ is fixed and given, it is an MDP in the classical sense. In an MFG, we look for an equilibrium situation, in which the population follows a policy from which no individual player is interested in deviating.

Definition 1 (MFG Nash equilibrium)**.** Given $\mu_0 \in M$, $(\hat{\boldsymbol{\pi}}^{\mu_0}, \hat{\boldsymbol{\mu}}^{\mu_0}) \in \boldsymbol{\Pi} \times \mathbf{M}$ is an MFG Nash equilibrium (MFG-NE) consistent with μ_0 if: (1) $\hat{\boldsymbol{\pi}}^{\mu_0}$ maximizes $J(\mu_0, \cdot; \hat{\boldsymbol{\mu}}^{\mu_0})$, and (2) $\hat{\boldsymbol{\mu}}^{\mu_0} = \Phi(\mu_0, \hat{\boldsymbol{\pi}}^{\mu_0})$.

Being an MFG-NE amounts to say that the *exploitability* $\mathcal{E}(\mu_0, \hat{\boldsymbol{\pi}}^{\mu_0})$ is 0, where the exploitability of a policy $\boldsymbol{\pi} \in \boldsymbol{\Pi}$ given the initial MF state μ_0 is defined as:

$$\mathcal{E}(\mu_0, \boldsymbol{\pi}) = \max_{\boldsymbol{\pi}'} J(\mu_0, \boldsymbol{\pi}'; \Phi(\mu_0, \boldsymbol{\pi})) - J(\mu_0, \boldsymbol{\pi}; \Phi(\mu_0, \boldsymbol{\pi})).$$

It quantifies how much a representative player can be better off by deciding to play another policy than $\boldsymbol{\pi}$ when the rest of the population uses $\boldsymbol{\pi}$ and the initial distribution is μ_0 for both the player and the population. Similar notions are widely used in computational game theory (Zinkevich et al. 2007; Lanctot et al. 2009).

In general, $\hat{\boldsymbol{\pi}}^{\mu_0}$ is not an MFG-NE policy consistent with $\mu_0' \neq \mu_0$. Imagine for example a game in which the agents need to spread uniformly throughout a one-dimensional domain (see the experimental section). Intuitively, the movement of an agent at the center depends on where the bulk of the population is. If μ_0 is concentrated on the left (resp. right) side, this agent should move towards the right (resp. left). Hence the optimal policy depends on the whole population distribution.

Equilibria in MFG are traditionally characterized by a forward-backward system of equations (Lasry and Lions 2007; Carmona and Delarue 2018). Indeed, the value function of an individual player facing an MF flow $\boldsymbol{\mu}$ is:

$$V_n(x; \boldsymbol{\mu}) = \sup_{\boldsymbol{\pi} \in \boldsymbol{\Pi}} \mathbb{E}_{x, \boldsymbol{\pi}}\Big[\sum_{n'=n}^{+\infty} \gamma^{n'-n} r(x_{n'}, a_{n'}, \boldsymbol{\mu}_{n'})\Big],$$

where $x_n = x$ and $a_{n'} \sim \boldsymbol{\pi}_{n'}(\cdot|x_{n'})$, $n' \geq n$. Dynamic programming yields:

$$V_n(x; \boldsymbol{\mu}) = \sup_{\pi \in \Pi} \mathbb{E}_{x, \pi}\Big[r(x_n, a_n, \boldsymbol{\mu}_n) + \gamma V_{n+1}(x'; \boldsymbol{\mu})\Big],$$

where $x_n = x$, $a_n \sim \pi(\cdot|x)$ and $x' \sim p(\cdot|x, a, \boldsymbol{\mu}_n)$. Taking the maximizer gives an optimal policy for a player facing $\boldsymbol{\mu}$. To find an equilibrium policy, we replace $\boldsymbol{\mu}$ by the equilibrium MF flow $\hat{\boldsymbol{\mu}}$: $\hat{V}_n(\cdot) = V_n(\cdot; \hat{\boldsymbol{\mu}})$. But $\hat{\boldsymbol{\mu}}$ is found by using the corresponding equilibrium policy. This induces a coupling between the backward equation for the representative player and the forward population dynamics.

The starting point of our Master policy approach is to notice that $V_n(\cdot; \boldsymbol{\mu})$ depends on n and $\boldsymbol{\mu}$ only through $(\boldsymbol{\mu}_{n'})_{n' \geq n}$ hence V_n depends on n only through $(\boldsymbol{\mu}_{n'})_{n' \geq n}$:

$$V_n(x; \boldsymbol{\mu}) = V(x; (\boldsymbol{\mu}_{n'})_{n' \geq n})$$

where, for $\boldsymbol{\mu} \in \mathbf{M}, x \in X$,

$$V(x; \boldsymbol{\mu}) = \sup_{\pi \in \Pi} \mathbb{E}_{x, \pi}\Big[r(x, a, \boldsymbol{\mu}_0) + \gamma V(x'; (\boldsymbol{\mu}_n)_{n \geq 1})\Big], \tag{1}$$

where $a \sim \pi(\cdot|x)$ and $x' \sim p(\cdot|x, a, \boldsymbol{\mu}_0)$.

From here, we will express the equilibrium policy $\hat{\boldsymbol{\pi}}_n$ as a stationary policy (independent of n) which takes $\hat{\boldsymbol{\mu}}_n$ as an extra input. Replacing n by $\hat{\boldsymbol{\mu}}_n$ increases the input size but it opens new possibilities in terms of *generalization in MFGs*.

Learning in Mean Field Games

We focus on methods involving Reinforcement Learning, or Dynamic Programming when the model is known. Learning

in MFGs can also involve methods that approximate directly the forward-backward system of equations with function approximations (such as neural networks), but we will not address them here; see, *e.g.*, (Al-Aradi et al. 2018; Carmona and Laurière 2021).

In the literature, *Learning* in MFGs indistinctly refers to the optimization algorithm (being most of the time the fixed point or variations of Fictitious Play), or to the subroutines involving learning that are used to compute the policy (Reinforcement Learning) or the distribution. We make here a distinction between these notions for the sake of clarity.

Optimization algorithm. From a general point of view, learning algorithms for MFGs approximate two types of objects: (1) a policy for the representative agent, and (2) a distribution of the population, resulting from everyone applying the policy. This directly leads to a simple fixed-point iteration approach, in which we alternatively update the policy and the mean-field term. This approach has been used, e.g., by Guo et al. (2019). However without strong hypothesis of regularity and a strict contraction property, this scheme does not converge to an MFG-NE. To stabilize the learning process and to ensure convergence in more general settings, recent papers have either added regularization (Anahtarci, Kariksiz, and Saldi 2020; Guo, Xu, and Zariphopoulou 2020; Cui and Koeppl 2021) or used Fictitious Play (Cardaliaguet and Hadikhanloo 2017; Cardaliaguet and Lehalle 2018; Mguni, Jennings, and Munoz de Cote 2018; Perrin et al. 2020; Delarue and Vasileiadis 2021), while Hadikhanloo (2017) and Perolat et al. (2022) have introduced and analyzed Online Mirror Descent.

Reinforcement learning subroutine. For a given population distribution, to update the representative player's policy or value function, we can rely on RL techniques. For instance Guo et al. (2019); Anahtarci, Kariksiz, and Saldi (2020) rely on Q-learning to approximate the Q-function in a tabular setting, Fu et al. (2019) study an actor-critic method in a linear-quadratic setting, and Elie et al. (2020); Perrin et al. (2021) solve continuous spaces problems by relying respectively on deep deterministic policy gradient (Lillicrap et al. 2016) or soft actor-critic (Haarnoja et al. 2018). Two time-scales combined with policy gradient has been studied by Subramanian and Mahajan (2019) for stationary MFGs. Policy iterations together with sequential decomposition has been proposed by Mishra, Vasal, and Vishwanath (2020) while Guo et al. (2020) proposes a method relying on Trust Region Policy Optimization (TRPO, Schulman et al. (2015)).

Distribution embedding. Another layer of complexity in MFGs is to take into consideration population distributions for large spaces or even continuous spaces. To compute MFG solutions through a PDE approach, Al-Aradi et al. (2018); Carmona and Laurière (2021) used deep neural networks to approximate the population density in high dimension. In the context of RL for MFGs, recently, Perrin et al. (2021) have used Normalizing Flows (Rezende and Mohamed 2015) to approximate probability measures over continuous state space in complex environments.

Generalization in MFGs through Master Policies

So far, learning approaches for MFGs have considered only two aspects: optimization algorithms (*e.g.*, Fictitious Play or Online Mirror Descent), or model-free learning of a representative player's best response based on samples (*e.g.*, Q-learning or actor-critic methods). Here, we build upon the aforementioned notions and add to this picture another dimension of learning: *generalization* over population distributions. We develop an approach to learn the representative player's best response as a function of any current population distribution and not only the ones corresponding to a fixed MFG-NE. This is tightly connected with the so-called Master equation in MFGs (Lions 2006-2012; Bensoussan, Frehse, and Yam 2015; Cardaliaguet et al. 2019). Introduced in the continuous setting (continuous time, state and action), this equation is a partial differential equation (PDE) which corresponds to the limit of systems of Hamilton-Jacobi-Bellman PDEs characterizing Nash equilibria in symmetric N-player games. In our discrete context, we introduce a notion of Master Bellman equation and associated Master policy, which we then aim to compute with a new learning algorithm based on Fictitious Play. To the best of our knowledge, the literature focuses Master equations on a finite horizon. Here, we consider infinite horizon discounted problems, which allows us to look for stationary solutions.

Master Policies for MFGs

We introduce the notion of Master policy and connect it to standard population-agnostic policies arising in MFG-NE.

Consider an MFG-NE $(\hat{\boldsymbol{\pi}}^{\mu_0}, \hat{\boldsymbol{\mu}}^{\mu_0})$ consistent with some μ_0. Let $\hat{V}(\cdot;\mu_0) = V(\cdot;\hat{\boldsymbol{\mu}}^{\mu_0})$, *i.e.*,

$$\hat{V}(x;\mu_0) = \sup_{\pi\in\Pi} \mathbb{E}_\pi\Big[r(x,a,\mu_0) + \gamma V(x';(\hat{\boldsymbol{\mu}}_n^{\mu_0})_{n\geq 1})\Big],$$

where $a\sim\pi(\cdot|x,\mu_0)$ and $x'\sim p(\cdot|x,a,\mu_0)$. By definition, $\hat{\boldsymbol{\pi}}_0^{\mu_0}$ is a maximizer in the sup above. Moreover, in the right-hand side,

$$V(x';(\hat{\boldsymbol{\mu}}_n^{\mu_0})_{n\geq 1}) = \hat{V}(x';\hat{\boldsymbol{\mu}}_1^{\mu_0}), \qquad \hat{\boldsymbol{\mu}}_1^{\mu_0} = \phi(\mu_0, \hat{\boldsymbol{\pi}}_0^{\mu_0}).$$

By induction, the equilibrium can be characterized as:

$$\begin{cases} \hat{\boldsymbol{\pi}}_n^{\mu_0} \in \operatorname{argmax}_{\pi\in\Pi} \mathbb{E}_\pi\Big[r(x,a,\hat{\boldsymbol{\mu}}_n^{\mu_0}) + \gamma\hat{V}(x';\hat{\boldsymbol{\mu}}_{n+1}^{\mu_0})\Big] \\ \hat{V}(x;\hat{\boldsymbol{\mu}}_n^{\mu_0}) = \mathbb{E}_{\hat{\boldsymbol{\pi}}_n^{\mu_0}}\Big[r(x,a,\hat{\boldsymbol{\mu}}_n^{\mu_0}) + \gamma\hat{V}(x';\hat{\boldsymbol{\mu}}_{n+1}^{\mu_0})\Big] \\ \hat{\boldsymbol{\mu}}_{n+1}^{\mu_0} = \phi(\hat{\boldsymbol{\mu}}_n^{\mu_0}, \hat{\boldsymbol{\pi}}_n^{\mu_0}). \end{cases}$$

Note that $\hat{\boldsymbol{\mu}}_{n+1}^{\mu_0}$ and $\hat{\boldsymbol{\pi}}_n^{\mu_0}$ depend on each other (and also on μ_0), which creates a forward-backward structure.

In the sequel, we will refer to this function V as the *Master value function*. Computing the value function $(x,\mu)\mapsto\hat{V}(x;\mu)$ would allow us to know the value of any individual state x facing an MFG-NE starting from any MF state μ. However, it would not allow to easily find the corresponding equilibrium policy, which still depends implicitly on the equilibrium MF flow. For this reason, we introduce the notion of population-dependent policy. The set of population-dependent policies $\widetilde{\pi}: X\times\Delta_X\to\Delta_A$ is denoted by $\widetilde{\Pi}$.

Definition 2. A population-dependent $\widetilde{\pi}^* \in \widetilde{\Pi}$ is a Master policy if for every μ_0, $(\boldsymbol{\pi}^{\mu_0,\widetilde{\pi}^*}, \boldsymbol{\mu}^{\mu_0,\widetilde{\pi}^*})$ is an MFG-NE, where: $\boldsymbol{\mu}_0^{\mu_0,\widetilde{\pi}^*} = \mu_0$ and for $n \geq 0$,

$$\begin{cases} \boldsymbol{\pi}_n^{\mu_0,\widetilde{\pi}^*}(x) = \widetilde{\pi}^*(x, \boldsymbol{\mu}_n^{\mu_0,\widetilde{\pi}^*}) \\ \boldsymbol{\mu}_{n+1}^{\mu_0,\widetilde{\pi}^*} = \phi(\boldsymbol{\mu}_n^{\mu_0,\widetilde{\pi}^*}, \boldsymbol{\pi}_n^{\mu_0,\widetilde{\pi}^*}). \end{cases} \tag{2}$$

A Master policy allows recovering the MFG-NE starting from any initial MF state. A core question is the existence of such a policy, which we prove in Theorem 1 below. Hence, if there is a unique Nash equilibrium MF flow (*e.g.*, thanks to monotonicity), the MF flow $\boldsymbol{\mu}^{\mu_0,\widetilde{\pi}^*}$ obtained with the Master policy $\widetilde{\pi}^*(a|x, \boldsymbol{\mu}_n^{\mu_0,\widetilde{\pi}^*})$ is the same as the one obtained with a best response policy $\hat{\boldsymbol{\pi}}_n^{\mu_0}(a|x)$ starting from μ_0.

Theorem 1. *Assume that, for all $\mu_0 \in M$, the MFG admits an equilibrium consistent with μ_0 and that the equilibrium MF flow is unique. Then there exists a Master policy $\widetilde{\pi}^*$.*

Existence and uniqueness of the MFG-NE for a given μ_0 can be proved under a mild monotonicity condition, see *e.g.* Perrin et al. (2020). Thm. 1 is proved by checking, step by step, that the MF flow generated by $\widetilde{\pi}^*$ and the associated population-agnostic policy as defined in (2) form a MFG-NE. The key idea is to use dynamic programming relying on the Master value function V and the uniqueness of the associated equilibrium MF flow. We omit the details for brevity.

Algorithm

We have demonstrated above that the Master policy is well-defined and allows to recover Nash equilibria. We now propose a method to compute such a policy.

Fictitious Play

We introduce an adaptation of the Fictitious Play algorithm to learn a Master policy. This extends to the case of population-dependent policies the algorithm introduced by Cardaliaguet and Hadikhanloo (2017). In the same fashion, at every iteration k, it alternates three steps: (1) computing a best response policy $\widetilde{\pi}_k$ against the current averaged MF flows $\bar{\mathcal{M}}_k$, (2) computing $(\boldsymbol{\mu}^{\mu_0,\widetilde{\pi}_k})_{\mu_0\in\mathcal{M}}$, the MF flows induced by $\widetilde{\pi}_k$, and (3) updating $\bar{\mathcal{M}}_{k+1}$ with $(\boldsymbol{\mu}^{\mu_0,\widetilde{\pi}_k})_{\mu_0\in\mathcal{M}}$.

We choose Fictitious Play rather than a simpler fixed-point approach because it is generally easier to check that an MFG model satisfies the assumptions used to prove convergence (monotonicity condition rather than contraction properties, as *e.g.* in Huang et al. (2006); Guo et al. (2019)).

Ideally, we would like to train the population-dependent policy on every possible distributions, but this is not feasible. Thus, we take a finite training set $\mathcal{M}$ of initial distributions. Each training distribution is used at each iteration of Fictitious Play. Another possibility would have been to swap these two loops, but we chose not to do this because of *catastrophic forgetting* (French 1999; Goodfellow et al. 2014), a well-know phenomenon in cognitive science that also occurs in neural networks, describing the tendency to forget previous information when learning new information. Our proposed algorithm is summarized in Alg. 1 and we refer to it as *Master Fictitious Play*.

Algorithm 1: Master Fictitious Play

input : Initial $\widetilde{\pi}_0 \in \widetilde{\Pi}$, training set of initial distributions $\mathcal{M}$, number of Fictitious Play steps K

Let $\bar{\pi} = \widetilde{\pi}_0$; let $\bar{\boldsymbol{\mu}}_{0,n}^{\mu_0} = \mu_0$ for all $\mu_0 \in M$, all $n \geq 0$
Let $\bar{\mathcal{M}}_0 = (\bar{\boldsymbol{\mu}}_0^{\mu_0})_{\mu_0\in\mathcal{M}}$
for $k = 1, \dots, K$ **do**
  Train $\widetilde{\pi}_k$ against $\bar{\mathcal{M}}_k = (\bar{\boldsymbol{\mu}}_k^{\mu_0})_{\mu_0\in\mathcal{M}}$, to maximize Eq. (4)
  for $\mu_0 \in \mathcal{M}$ **do**
    Compute $\boldsymbol{\mu}_k^{\mu_0}$, the MF flow starting from μ_0 induced by $\widetilde{\pi}_k$ against $\bar{\boldsymbol{\mu}}_k^{\mu_0}$
    Let $\bar{\boldsymbol{\mu}}_k^{\mu_0} = \frac{k}{k+1}\bar{\boldsymbol{\mu}}_{k-1}^{\mu_0} + \frac{1}{k+1}\boldsymbol{\mu}_k^{\mu_0}$
  Update $\bar{\widetilde{\pi}}_k = \text{UNIFORM}(\widetilde{\pi}_0, \dots, \widetilde{\pi}_k)$
return $\bar{\widetilde{\pi}}_K = \text{UNIFORM}(\widetilde{\pi}_0, \dots, \widetilde{\pi}_K)$

Alg. 1 returns $\bar{\widetilde{\pi}}_K$, which is the uniform distribution over past policies. We use it as follows. First, let: $\boldsymbol{\mu}_{k,0}^{\mu_0} = \mu_0$, $k = 1, \dots, K$, $\bar{\boldsymbol{\mu}}_{K,0}^{\mu_0} = \frac{1}{K}\sum_{k=1}^K \boldsymbol{\mu}_{k,0}^{\mu_0}$, and then, for $n \geq 0$,

$$\begin{cases} \boldsymbol{\mu}_{k,n+1}^{\mu_0} = \phi(\boldsymbol{\mu}_{k,n}^{\mu_0}, \widetilde{\pi}_k(\cdot|\cdot, \bar{\boldsymbol{\mu}}_{K,n}^{\mu_0})), & k = 1, \dots, K \\ \bar{\boldsymbol{\mu}}_{K,n+1}^{\mu_0} = \frac{1}{K}\sum_{k=1}^K \boldsymbol{\mu}_{k,n+1}^{\mu_0}. \end{cases}$$

Note that $\bar{\widetilde{\pi}}_K$ is used in the same way for every μ_0. We will show numerically that this average distribution and the associated average reward are close to the equilibrium ones.

Define the average exploitability as:

$$\bar{\mathcal{E}}_{\mathcal{M}}(\bar{\widetilde{\pi}}_K) = \mathbb{E}_{\mu_0\sim\text{UNIFORM}(\mathcal{M})}\left[\bar{\mathcal{E}}(\mu_0, \bar{\widetilde{\pi}}_K)\right], \tag{3}$$

where

$$\bar{\mathcal{E}}(\mu_0, \bar{\widetilde{\pi}}_K) = \max_{\boldsymbol{\pi}'} J(\mu_0, \boldsymbol{\pi}'; \bar{\boldsymbol{\mu}}_K^{\mu_0}) - \frac{1}{K}\sum_{k=1}^K J(\mu_0, \widetilde{\pi}_k; \bar{\boldsymbol{\mu}}_K^{\mu_0}).$$

We expect $\bar{\mathcal{E}}(\mu_0, \bar{\widetilde{\pi}}_K) \to 0$ as $K \to +\infty$. We show that this indeed holds under suitable conditions in the idealized setting with continuous time updates, where $\bar{\widetilde{\pi}}_k$, $k = 0, 1, 2, \dots$, is replaced by $\bar{\widetilde{\pi}}_t$, $t \in [0, +\infty)$.

Theorem 2. *Assume the reward is separable and monotone, i.e., $r(x, a, \mu) = r_A(x, a) + r_M(x, \mu)$ and $\sum_{x\in X}(r_M(x, \mu) - r_M(x, \mu'))(\mu - \mu')(x) < 0$ for every $\mu \neq \mu'$. Assume the transition depends only on x and a: $p(\cdot|x, a, \mu) = p(\cdot|x, a)$. Then $\bar{\mathcal{E}}_{\mathcal{M}}(\bar{\widetilde{\pi}}_t) = O(1/t)$, where $\bar{\widetilde{\pi}}_t$ is the average policy at time t in the continuous time version of Master Fictitious Play.*

The proof follows the lines of (Perrin et al. 2020) adapted to our setting and is omitted for the sake of brevity. Studying continuous time updates instead of discrete ones enables us to use calculus, which leads to a simple proof. To the best of our knowledge, there is no rate of convergence for discrete time Fictitious Play in the context of MFG except for potential or linear-quadratic structures, see (Geist et al. 2022) and (Delarue and Vasileiadis 2021).

Deep RL to Learn a Population-dependent Policy

In Alg. 1, a crucial step is to learn a population-dependent best response against the current averaged MF flows $\bar{\mathcal{M}}_k = (\bar{\boldsymbol{\mu}}_k^{\mu_0})_{\mu_0 \in \mathcal{M}}$, *i.e.*, $\widetilde{\pi}_k^*$ maximizing

$$\widetilde{\pi} \mapsto \tfrac{1}{|\mathcal{M}|} \textstyle\sum_{\mu_0 \in \mathcal{M}} J(\mu_0, \widetilde{\pi}; \bar{\boldsymbol{\mu}}_k^{\mu_0}). \tag{4}$$

Solving the optimization problem (4) can be reduced to solving a standard but non-stationary MDP. Since we aim at optimizing over population-dependent policies, the corresponding Q-function is a function of not only an agent's state-action pair (x, a) but also of the population distribution: $\widetilde{Q}(x, \mu, a)$. Adding the current mean field state μ to the Q-function allows us to recover a stationary MDP. As we know that the optimal policy is stationary, we now have a classical RL problem with state (x, μ) (instead of x only), and we can use Deep RL methods such as DQN (Mnih et al. 2013) to compute $\widetilde{Q}_k$. The policy $\widetilde{\pi}_k$ can then be recovered easily by applying the argmax operator to the Q-function.

Various algorithms could be used, but we choose DQN to solve our problem because it is sample-efficient. For the numerical results presented below, we used the default implementation of RLlib (Liang et al. 2017).

The neural network representing the Q-function takes as inputs the state x of the representative player and the current distribution μ of the population, which can simply be represented as a histogram (the proportion of agents in each state). In practice, μ is a mean-field state coming from one of the averaged MF flows $\bar{\boldsymbol{\mu}}_k^{\mu_0}$ and is computed in steps 7 and 8 of Alg. 1 with a Monte-Carlo method, *i.e.* by sampling a large number of agents that follow the last population-dependent best response $\widetilde{\pi}_k$ and averaging it with $\bar{\boldsymbol{\mu}}_{k-1}^{\mu_0}$. Then, the Q-function can be approximated by a feedforward fully connected neural network with these inputs. In the examples considered below, the finite state space comes from the discretization of a continuous state space in dimension 1 or 2. The aforementioned simple approximation gives good results in $1D$. However, in $2D$, the neural network did not manage to learn a good population-dependent policy. This is probably because passing a histogram as a flat vector ignores the geometric structure of the problem. We thus resort to a more sophisticated representation. We first create an *embedding* of the distribution by passing the histogram to a convolutional neural network (ConvNet). The output of this embedding network is then passed to a fully connected network which outputs probabilities for each action (see Fig.1). The use of a ConvNet is motivated by the fact that the state space in our examples has a clear geometric interpretation and that the population can be represented as an image.

On the Theoretical vs. Experimental Settings

Theoretically, we expect the algorithm Alg. 1 to converge perfectly to a Master policy. This intuition is supported by Thm. 2 and comes from the fact that Fictitious Play has been proved to converge to population-agnostic equilibrium policies when the initial distribution is fixed (Cardaliaguet and Hadikhanloo 2017; Perrin et al. 2020). However, from a practical viewpoint, here we need to make several approximations. The main one is related to the challenges of conditioning on a MF state. Even though the state space X is finite, the space of MF states $M = \Delta_X$ is infinite and of dimension equal to the number of states, which is potentially very large. This is why we need to rely on function approximation (*e.g.*, by neural networks as in our implementation) to learn an optimal population-dependent policy. Furthermore, the training procedure uses only a finite (and relatively small) set of training distributions. On top of this, other more standard approximations are to be taken into account, in particular due to the use of a Deep RL subroutine.

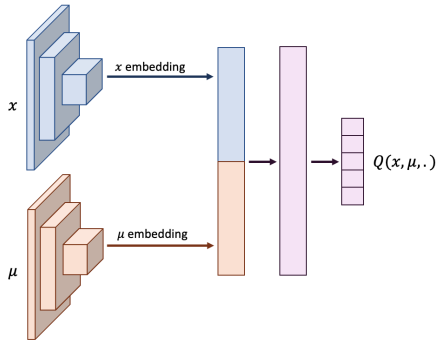

Figure 1: Neural network architecture of the Q-function for the 2D beach bar experience.

Numerical Experiments

Experimental Setup

We now illustrate the efficiency and generalization capabilities of the Master policy learned with our proposed method.

Procedure. To demonstrate experimentally the performance of the learned Master policy trained by Alg. 1, we consider: several initial distributions, several benchmark policies and several metrics. For each metric, we illustrate the performance of each policy on each initial distribution. The initial distributions come from two sets: the training set $\mathcal{M}$ used in Alg. 1 and a testing set. For the benchmark policies, in the absence of a population-dependent baseline of reference (since, to the best of our knowledge, our work is the first to deal with Master policies), we focus on natural candidates that are population-agnostic. The metrics are chosen to give different perspectives: the population distribution and the policy performance in terms of reward.

Training set of initial distributions. In our experiments, we consider a training set $\mathcal{M}$ composed of Gaussian distributions such that the union of all these distributions sufficiently covers the whole state space. This ensures that the policy learns to behave on any state $x \in X$. Furthermore, although we call "training set" the set of initial distributions, the policy actually sees more distributions during the training episodes. Each distribution visited could be considered as an initial distribution. Note however that it is very different from training the policy on all possible *population distributions* (which is a simplex with dimension equal to the number of states, *i.e.*, 32 or $16^2 = 256$ in our examples).

Testing set of initial distributions. The testing set is composed of two types of distributions. First, random distributions generated by sampling uniformly a number in $[0, 1]$ for each state independently, and then normalizing the distribution. Second, Gaussian distributions with means located between the means of the training set, and various variances.

Benchmark type 1: Specialized policies. For a given initial distribution μ_0^i with $i \in \{1, \dots, |\mathcal{M}|\}$, we consider a Nash equilibrium starting from this MF state, *i.e.*, a population-agnostic policy $\hat{\boldsymbol{\pi}}^i$ and a MF flow $\hat{\boldsymbol{\mu}}^i$ satisfying Def. 1 with μ_0 replaced by μ_0^i. In the absence of analytical formula, we compute such an equilibrium using Fictitious Play algorithm with backward induction (Perrin et al. 2020). We then compare our learned Master policy with each $\hat{\boldsymbol{\pi}}^i$, either on μ_0^i or on another μ_0^j. In the first case, it allows us to check the correctness of the learned Master policy, and in the second case, to show that it generalizes better than $\hat{\boldsymbol{\pi}}^i$.

Benchmark type 2: Mixture-reward policy. Each (population-agnostic) policy discussed above is specialized for a given μ_0^i but our Alg. 1 trains a (population-dependent) policy on various initial distributions. It is thus natural to see how the learned Master policy fares in comparison with a population-agnostic policy trained on various initial distributions. We thus consider another benchmark, called *mixture-reward policy*, which is a population-agnostic policy trained to optimize an average reward. It is computed as the specialized policies described above but we replace the reward definition with an average over the training distributions. For $1 \leq i \leq |\mathcal{M}|$, recall $\hat{\boldsymbol{\mu}}^i$ is a Nash equilibrium MF flow starting with MF state μ_0^i. We consider the average reward: $\bar{r}_n(x, a) = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} r(x, a, \hat{\boldsymbol{\mu}}_n^i)$. The mixture-reward policy is an optimal policy for the MDP with this reward function. In our experiments, we compute it as for the specialized policies described above.

Benchmark type 3: Unconditioned policy. Another meaningful comparison is to use the same algorithm while removing the population input. This amounts to running Alg. 1 where, in the DQN subroutine, the Q-function neural network is a function of x and a only. So in Fig. 1, we replace the μ input embedding by zeros. We call the resulting policy *unconditioned policy* because it illustrates the performance when removing the conditioning on the MF term. This benchmark will be used to illustrate that the success of our approach is not only due to combining Deep RL with training on various μ_0: conditioning the Q-function and the policy on the MF term plays a key role.

Metric 1: Wasserstein distance between MF flows. We first measure how similar the policies are in terms of induced behavior at the scale of the population. Based on the Wasserstein distance W between two distributions, we compute the following distance between MF flows truncated at some horizon N_T:

$$W_{i,j} := \frac{1}{N_T+1} \sum_{n=0}^{N_T} W(\boldsymbol{\mu}_n^{\boldsymbol{\pi}^i, \mu_0^j}, \boldsymbol{\mu}_n^{\boldsymbol{\pi}^j, \mu_0^j}).$$

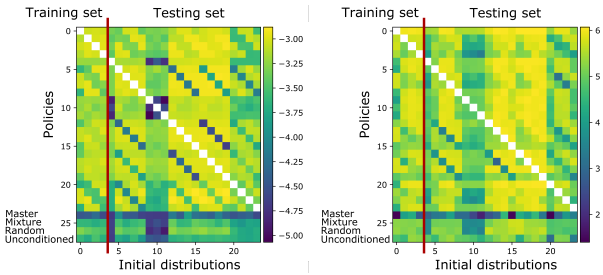

Figure 2: Exploration 1D: Performance matrices when the training set is made of Gaussian distributions. From left to right: **(a)** Log of Wasserstein distances to the exact solution (time average); **(b)** Log of exploitabilities. The x-axis is the initial distribution index: on the left (resp. right) of the vertical red line are the training (resp. testing) distributions.

Note that $W_{i,i} = 0$. The term $\boldsymbol{\mu}^{\boldsymbol{\pi}^j, \mu_0^j} = \hat{\boldsymbol{\mu}}^j$ is the equilibrium MF flow starting from μ_0^j, while $\boldsymbol{\mu}^{\boldsymbol{\pi}^i, \mu_0^j}$ is the MF flow generated by starting from μ_0^j and using policy $\boldsymbol{\pi}^i$.

Metric 2: Exploitability. We also assess the performance of a given policy by measuring how far from being a Nash it is. To this end, we use the exploitability. We compute for each i, j: $E_{i,j} = \mathcal{E}(\mu_0^j, \hat{\boldsymbol{\pi}}^i)$. When $i = j$, $E_{i,i} = 0$ because $(\hat{\boldsymbol{\pi}}^i, \hat{\boldsymbol{\mu}}^i)$ is a Nash equilibrium starting from μ_0^i. When $i \neq j$, $E_{i,j}$ measures how far from being optimal $\hat{\boldsymbol{\pi}}^i$ is when the population also uses $\hat{\boldsymbol{\pi}}^i$, but both the representative player and the population start with μ_0^j. If $E_{i,j} = 0$, then $\hat{\boldsymbol{\pi}}^i$ is a Nash equilibrium policy even when starting from μ_0^j.

Experiment 1: Pure Exploration in 1D

We consider a discrete 1D environment inspired by Geist et al. (2022). Transitions are deterministic, the state space is $X = \{1, \dots, |X| = 32\}$. The action space is $A = \{-1, 0, 1\}$: agents can go left, stay still or go right (as long as they stay in the state space). The reward penalizes the agent with the amount of people at their location, while discouraging them from moving too much: $r(x, a, \mu) = -\log(\mu(x)) - \frac{1}{|X|}|a|$. The training set of initial distributions $\mathcal{M}$ consists of four Gaussian distributions with the same variance but different means. The testing set is composed of random and Gaussian distributions with various variances. We can see that the Master policy is still performing well on these distributions, which highlights its generalization capacities. The diagonal is white since the Wasserstein distance and exploitability are zero for specialized baselines evaluated on their corresponding μ_0. We also observe that the random policy is performing well on random distributions, and that exact solutions trained on a randomly generated distribution seem to perform quite well on other randomly generated distributions. We believe this is due to this specific environment, because a policy that keeps enough entropy performs well.

Experiment 2: Beach Bar in 2D

We now consider the 2 dimensional beach bar problem, introduced by Perrin et al. (2020), to highlight that the method

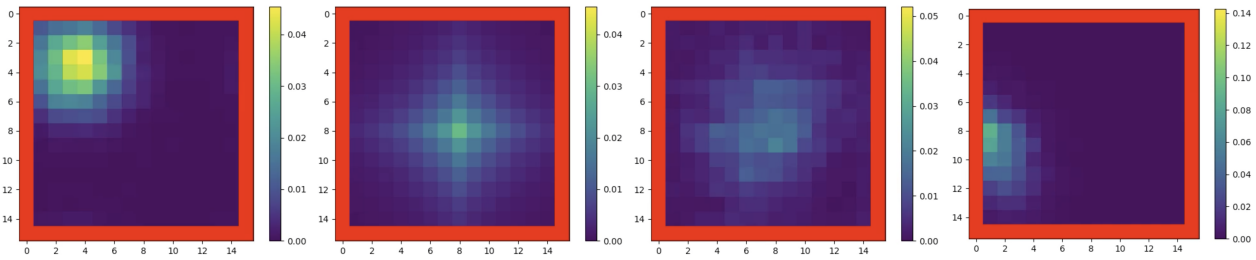

Figure 3: Beach bar 2D: Environment. From left to right: **(a)** an initial distribution $\mu_0 \in \mathcal{M}$; **(b)** MF state at equilibrium (specialized policy); **(c)** MF state at equilibrium (learned Master policy); **(d)** MF state at equilibrium (specialized policy of another initial distribution). Note that the scale is very different for the last figure.

can scale to larger environments. The state space is a discretization of a 2-dimensional square. The agents can move by one state in the four directions: up, down, left, right, but there are walls on the boundaries. The instantaneous reward is: $r(x, a, \mu) = d_{\text{bar}}(x) - \log(\mu(x)) - \frac{1}{|X|}\|a\|_1$, where d_{bar} is the distance to the bar, located at the center of the domain. Here again, the second term discourages the agent from being in a crowded state, while the last term discourages them from moving if it is not necessary. Starting from an initial distribution, we expect the agents to move towards the bar while spreading a bit to avoid suffering from congestion.

We use the aforementioned architecture (Fig. 1) with one fully connected network following two ConvNets: one for the agent's state, represented as a one-hot matrix, and one for the MF state, represented as a histogram. Having the same dimension (equal to the number $|X|$ of states) and architecture for the position and the distribution makes it easier for the deep neural network to give an equal importance to both of these features. Deep RL is crucial to cope with the high dimensionality of the input. Here $|X| = 16^2 = 256$.

Fig. 4 illustrates the performance of the learned Master policy. Once again, it outperforms the specialized policies as well as the random, mixture-reward, and unconditioned policies. An illustration of the environment and of the different policies involved is available in Fig. 3.

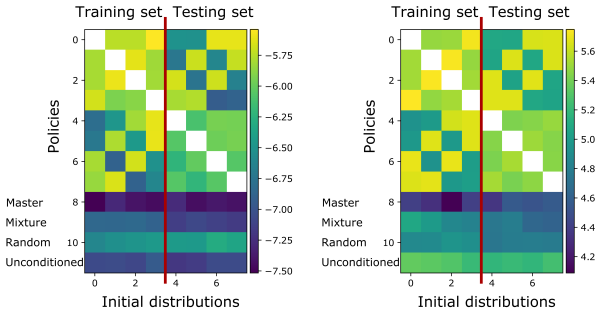

Figure 4: Beach bar 2D: Performance matrices with Gaussian distributions. From left to right: **(a)** Log of Wasserstein distances to the exact solution (average over time steps); **(b)** Log of exploitabilities. Each row is a policy. Top part: a row j gives the performance of the equilibrium policy for the j-th initial distribution. Bottom part: policies given in the text).

Conclusion

Motivated by the question of generalization in MFGs, we extended the notion of policies to let them depend explicitly on the population distribution. This allowed us to introduce the concept of Master policy, from which a representative player is able to play an optimal policy against any population distribution, as we proved in Thm. 1. We then proved that a continuous time adaptation of Fictitious Play can approximate the Master policy at a linear rate (Thm. 2). However, implementing this method is not straightforward because policies and value functions are now functions of the population distribution and, hence, out of reach for traditional computational methods. We thus proposed a Deep RL-based algorithm to compute an approximate Master policy. Although this algorithm trains the Master policy using a small training set of distributions, we demonstrated numerically that the learned policy is competitive on a variety of unknown distributions. In other words, for the first time in the RL for MFG literature, our approach allows the agents to generalize and react to many population distributions. This is in stark contrast with the existing literature, which focuses on learning population-agnostic policies, see *e.g.* (Guo et al. 2019; Anahtarci, Kariksiz, and Saldi 2020; Fu et al. 2019; Elie et al. 2020; Perrin et al. 2021). To the best of our knowledge, the only work considering policies that depend on the population is (Mishra, Vasal, and Vishwanath 2020), but their approach relies on solving a fixed point at each time step for every possible distribution, which is infeasible except for very small state space.

Our approach opens many directions for future work. First, the algorithm we proposed should be seen as a proof of concept and we plan to investigate other methods, such as Online Mirror Descent (Hadikhanloo 2017; Perolat et al. 2022). For high-dimensional examples, the question of distribution embedding deserves a special attention. Second, the generalization capabilities of the learned Master policy offers many new possibilities for applications. We plan to investigate how it can be used when the agent can only access a partial observation of the population. Last, the theoretical properties (such as the approximation and generalization theory) are also left for future work. An interesting question, from the point of view of learning is choosing the training set so as to optimize generalization capabilities of the learned Master policy.

References

Al-Aradi, A.; Correia, A.; Naiff, D.; Jardim, G.; and Saporito, Y. 2018. Solving nonlinear and high-dimensional partial differential equations via deep learning. *arXiv preprint arXiv:1811.08782*.

Anahtarci, B.; Kariksiz, C. D.; and Saldi, N. 2020. Q-learning in regularized mean-field games. *arXiv preprint arXiv:2003.12151*.

Bensoussan, A.; Frehse, J.; and Yam, S. C. P. 2015. The Master equation in mean field theory. *Journal de Mathématiques Pures et Appliquées*, 103(6): 1441–1474.

Brown, N.; and Sandholm, T. 2018. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374): 418–424.

Campbell, M.; Hoane Jr, A. J.; and Hsu, F.-h. 2002. Deep blue. *Artificial intelligence*, 134(1-2): 57–83.

Cardaliaguet, P.; Delarue, F.; Lasry, J.-M.; and Lions, P. L. 2019. *The master equation and the convergence problem in mean field games*, volume 381 of *AMS-201*.

Cardaliaguet, P.; and Hadikhanloo, S. 2017. Learning in mean field games: the fictitious play. *ESAIM Cont. Optim. Calc. Var.*

Cardaliaguet, P.; and Lehalle, C.-A. 2018. Mean field game of controls and an application to trade crowding. *Mathematics and Financial Economics*, 12(3): 335–363.

Carmona, R.; and Delarue, F. 2018. *Probabilistic theory of mean field games with applications. I*, volume 83 of *Probability Theory and Stochastic Modelling*. Springer, Cham. ISBN 978-3-319-56437-1; 978-3-319-58920-6. Mean field FBSDEs, control, and games.

Carmona, R.; and Laurière, M. 2021. Convergence Analysis of Machine Learning Algorithms for the Numerical Solution of Mean Field Control and Games I: The Ergodic Case. *SIAM Journal on Numerical Analysis*, 59(3): 1455–1485.

Cui, K.; and Koeppl, H. 2021. Approximately solving mean field games via entropy-regularized deep reinforcement learning. In *proc. of ICML*, 1909–1917. PMLR.

Delarue, F.; and Vasileiadis, A. 2021. Exploration noise for learning linear-quadratic mean field games. *arXiv preprint arXiv:2107.00839*.

Elie, R.; Perolat, J.; Laurière, M.; Geist, M.; and Pietquin, O. 2020. On the Convergence of Model Free Learning in Mean Field Games. In *proc. of AAAI*.

French, R. M. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4): 128–135.

Fu, Z.; Yang, Z.; Chen, Y.; and Wang, Z. 2019. Actor-Critic Provably Finds Nash Equilibria of Linear-Quadratic Mean-Field Games. In *proc. of ICLR*.

Geist, M.; Pérolat, J.; Laurière, M.; Elie, R.; Perrin, S.; Bachem, O.; Munos, R.; and Pietquin, O. 2022. Concave Utility Reinforcement Learning: the Mean-field Game viewpoint. In *proc. of AAMAS*.

Goodfellow, I. J.; Mirza, M.; Xiao, D.; Courville, A.; and Bengio, Y. 2014. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *proc. of ICLR*.

Guo, X.; Hu, A.; Xu, R.; and Zhang, J. 2019. Learning mean-field games. In *proc. of NeurIPS*.

Guo, X.; Hu, A.; Xu, R.; and Zhang, J. 2020. A General Framework for Learning Mean-Field Games. *CoRR*, abs/2003.06069.

Guo, X.; Xu, R.; and Zariphopoulou, T. 2020. Entropy Regularization for Mean Field Games with Learning. arXiv:2010.00145.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *proc. of ICML*.

Hadikhanloo, S. 2017. Learning in anonymous nonatomic games with applications to first-order mean field games. *arXiv preprint arXiv:1704.00378*.

Huang, M.; Malhamé, R. P.; Caines, P. E.; et al. 2006. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems*, 6(3): 221–252.

Lanctot, M.; Waugh, K.; Zinkevich, M.; and Bowling, M. 2009. Monte Carlo sampling for regret minimization in extensive games. volume 22, 1078–1086.

Lasry, J.-M.; and Lions, P.-L. 2007. Mean field games. *Japanese journal of mathematics*, 2(1): 229–260.

Liang, E.; Liaw, R.; Nishihara, R.; Moritz, P.; Fox, R.; Gonzalez, J.; Goldberg, K.; and Stoica, I. 2017. Ray RLLib: A Composable and Scalable Reinforcement Learning Library. *CoRR*, abs/1712.09381.

Lillicrap, T.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971.

Lions, P.-L. 2006-2012. Lecture at the Collège de France.

Mguni, D.; Jennings, J.; and Munoz de Cote, E. 2018. Decentralised Learning in Systems With Many, Many Strategic Agents. In *proc. of AAAI*.

Mishra, R. K.; Vasal, D.; and Vishwanath, S. 2020. Model-free reinforcement learning for non-stationary mean field games. In *2020 59th IEEE Conference on Decision and Control (CDC)*, 1032–1037. IEEE.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *In NIPS Workshop on Deep Learning and Unsupervised Feature Learning. arXiv preprint arXiv:1312.5602*.

Moravčík, M.; Schmid, M.; Burch, N.; Lisỳ, V.; Morrill, D.; Bard, N.; Davis, T.; Waugh, K.; Johanson, M.; and Bowling, M. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337): 508–513.

Perolat, J.; Perrin, S.; Elie, R.; Laurière, M.; Piliouras, G.; Geist, M.; Tuyls, K.; and Pietquin, O. 2022. Scaling up Mean Field Games with Online Mirror Descent. In *proc. of AAMAS*.

Perrin, S.; Laurière, M.; Pérolat, J.; Geist, M.; Élie, R.; and Pietquin, O. 2021. Mean Field Games Flock! The Reinforcement Learning Way. In *proc. of IJCAI*.

Perrin, S.; Pérolat, J.; Laurière, M.; Geist, M.; Elie, R.; and Pietquin, O. 2020. Fictitious play for mean field games: Continuous time analysis and applications. In *proc. of NeurIPS*.

Rezende, D.; and Mohamed, S. 2015. Variational Inference with Normalizing Flows. In *proc. of ICML*.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust Region Policy Optimization. In *proc. of ICML*.

Shannon, C. E. 1959. Programming a Computer Playing Chess. *Philosophical Magazine*, Ser.7, 41(312).

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587).

Subramanian, J.; and Mahajan, A. 2019. Reinforcement learning in stationary mean-field games. In *proc. of AAMAS*, 251–259.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Zinkevich, M.; Johanson, M.; Bowling, M.; and Piccione, C. 2007. Regret minimization in games with incomplete information. volume 20, 1729–1736.