ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240910

A Single Online Agent Can Efficiently Learn Mean Field Games

Chenyu Zhang[a], Xu Chen[b] and Xuan Di[b,*]

[a]Data Science Institute, Columbia University, New York, NY, USA
[b]Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, USA

Abstract. Mean field games (MFGs) are a promising framework for modeling the behavior of large-population systems. However, solving MFGs can be challenging due to the coupling of forward population evolution and backward agent dynamics. Typically, obtaining mean field Nash equilibria (MFNE) involves an iterative approach where the forward and backward processes are solved alternately, known as fixed-point iteration (FPI). This method requires fully observed population propagation and agent dynamics over the entire spatial domain, which could be impractical in some real-world scenarios. To overcome this limitation, this paper introduces a novel online single-agent model-free learning scheme, which enables a single agent to learn MFNE using online samples, without prior knowledge of the state-action space, reward function, or transition dynamics. Specifically, the agent updates its policy through the value function (Q), while simultaneously evaluating the mean field state (M), using the same batch of observations. We develop two variants of this learning scheme: off-policy and on-policy QM iteration. We prove that they efficiently approximate FPI, and a sample complexity guarantee is provided. The efficacy of our methods is confirmed by numerical experiments.

1 Introduction

Mean field games (MFGs) [18, 19] offer a tractable model to describe the population impact on individual agents in multi-agent systems with a large population. This work delves into the increasingly prominent field of applying reinforcement learning (RL) [33] to learn MFGs.

In an MFG, the influence of other agents is encapsulated by a *population mass* which provides a reliable approximation of real interactions between agents when the number of agents is large. A widely used method for learning MFGs is fixed-point iteration (FPI), which iteratively calculates the *best response* (BR) w.r.t. the current population, and the *induced population distribution* (IP) w.r.t. the current policy [16]. The FPI algorithm can be formally expressed as:

$$(\pi_k, \mu_k) = (\Gamma_{\mathrm{IP}} \circ \Gamma_{\mathrm{BR}})^k (\pi_0, \mu_0),$$

where operators Γ_{BR} calculates the best response and Γ_{IP} calculates the induced population distribution. We defer the full definitions of these operators to Section 2.

Although it is a prominent scheme for learning MFGs, current implementations of FPI and its variants face several limitations, especially in the IP calculation: 1) Γ_{BR} and Γ_{IP} are implemented separately and executed alternately, impeding parallel computing and potentially increasing the *computational complexity* of the entire algorithm. 2) The implementation of Γ_{IP} typically requires the knowledge of the transition dynamics of the environment [35, 27, 9, 10], limiting the use of *model-free* methods. 3) Despite some proposals of model-free strategies in existing literature, these methods demand direct observability of population dynamics [8, 23, 2]. In reality, fulfilling this requirement generally needs a central server capable of communication across the entire state space, restricting the feasibility of implementing such methods with a single online agent, i.e., an agent that interacts with the environment and collects local observations to learn and act on-the-go.

While these limitations paint part of the picture, we still need to answer the following question:

Why should we employ a single online agent to learn the equilibria of mean field games?

The reasons are multifold:

- In many real-world scenarios, a single online agent is often the most accessible, and sometimes the only available resource [30].
- Online single-agent model-free methods are more straightforward to implement, since they do not require prior knowledge of the data or the model.
- Once a single-agent model-free method is devised, this fundamental scheme can accommodate extensions such as multi-agent collaborative learning and model learning.

Motivated by answers to the "why" question, we ask:

Can a single online agent learn the equilibria of mean field games efficiently?

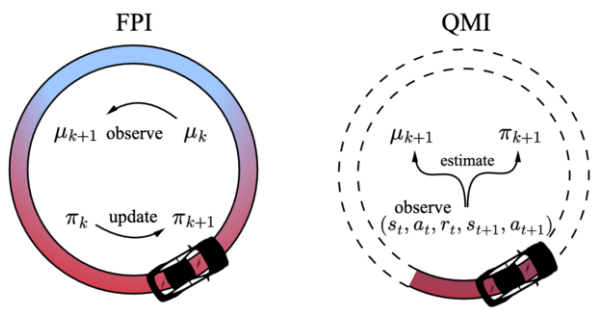


Figure 1: Illustration of learning processes of FPI and QMI for speed control on a ring road. The gradient color map signifies the varying population density on the ring road, with the dashed line indicating elements unobserved by the online agent. In FPI, the BR is calculated by a *representative* agent and the IP is directly observed. In QMI, a single online agent observes only *local* states and resultant rewards $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$, and uses these observations to estimate both the BR and IP.

Our work affirmatively answers this question by presenting QM it-

* Corresponding Author. Email: sharon.di@columbia.edu

eration (QMI), an efficient online single-agent model-free method for learning MFGs. QMI is strongly backed by the following theoretical premise. In an MFG, as all agents follow the same policy, we know that any agent's state is sampled from the population distribution. This fact reveals that a single agent encapsulates information about the entire population, suggesting that the induced population distribution can be learned through a single agent's state observations. More importantly, these observations are already collected during the phase where the agent updates its policy using an online RL method, suggesting that a single agent can learn both the BR and IP simultaneously using the same batch of online observations.

We present the example of speed control on a ring road, as illustrated in Figure 1, to concretize the above ideas and highlight the improvements of QMI over FPI. In this game, vehicles aim to maintain some desired speed while avoiding collisions. In FPI, a *representative* agent interacts with the population mass to learn the BR. Then, a dedicated forward process is needed to calculate the IP, either by leveraging knowledge of the transition dynamics or directly observing population dynamics across the entire state space. In contrast, QMI employs a single online agent with only local state and reward observations. Unlike FPI's representative agent, the online agent in QMI has no population information and thus no interaction with the population mass. Consequently, it maintains an *estimate* of the IP, and derives rewards according to this estimate. Equipped with this estimate, the online agent in QMI, similar to FPI's representative agent, can update its policy using local observations by online RL methods. As a distinctive feature, this agent also uses these local observations to update its population distribution estimate. Hence, QMI consolidates the two separate backward and forward processes in FPI into one and eliminates the need for prior environmental knowledge and global communication.

Contributions. Our primary contributions include:

- We propose an online single-agent model-free scheme for learning MFGs, termed as QM iteration (QMI). At each step of QMI, the agent updates its BR and IP estimates *simultaneously* using an online observation. More practical than FPI, QMI is applicable when no prior knowledge of the transition dynamics or the state space is available. We develop two variants of QMI, contingent on whether the agent selects actions following a fixed *behavior* policy, or adaptively updates its behavior policy within an outer iteration (Algorithm 1). An overview of the distinct features exhibited by the two variants is provided in Table 1.
- We prove that QMI efficiently approximates FPI and, therefore enjoys a similar convergence guarantee. The resemblance between the learning dynamics of QMI and FPI is illustrated in Figure 2. We provide sample complexity guarantees for our methods (Theorem 1). Our methods are the first provably efficient online single-agent model-free methods for learning MFGs. We validate our findings through numerical experiments on various MFGs (Section 6).

Related work. Huang et al. [18] introduced mean field games and suggested a forward-backward FPI scheme to solve them. To address the instability of FPI in discrete-time [11], researchers have proposed various stabilization techniques, including fictitious play [7, 26], online mirror descent [25, 22], and entropy regularization [11, 17, 2]. Yang et al. [35], Chen et al. [9] formulated MFGs as a population MDP, avoiding solving the forward-backward process in FPI, while requiring the knowledge of the entire state space and transition dynamics to update the state of the population.

A comprehensive survey on the application of RL in learning MFGs is presented by Laurière et al. [21], Cui et al. [12]. Existing work exclusively focuses on obtaining BRs in FPI using RL methods, including Q-learning [16, 27, 11], policy gradient [13], and actor critic [24, 32, 10]. For the population evolution (IP), most existing methods require either knowledge of the transition dynamics or direct observability. Recently, Angiuli et al. [3, 4] proposed an asynchronous Q-learning method for MFGs, removing the IP observability assumption, and proved its asymptotic convergence when population estimate updates occurs much slower than Q-value function updates ($\beta_t \ll \alpha_t$). Zaman et al. [36] extended this two-timescale model-free approach with model learning and proved its non-asymptotic convergence. In contrast, our methods employ the same timescale for population and policy estimates ($\beta_t \asymp \alpha_t$), substantially distinguishing our methodology.

2 Preliminaries

2.1 Mean Field Games

We consider an infinite-horizon discounted Markov decision process (MDP) denoted by $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the finite state and action spaces respectively, with their cardinality denoted by $S := |\mathcal{S}|$ and $A := |\mathcal{A}|$, r is the reward function, $\gamma \in (0,1)$ is the discount factor, and P is the transition kernel such that $P(s' \mid s, a)$ represents the probability that an agent transitions to state s' when it takes action a at state s. A policy (also referenced as a strategy or response) π maps a state to a distribution on the action space, guiding the action choices of an agent. When the policy π is fixed, we use P_π to denote the transition kernel and write $P_\pi(s, s') := \sum_{a\in\mathcal{A}} P(s' \mid s, a)\pi(a \mid s)$.

In MFGs, agents are considered indistinguishable with individually negligible influence. Thus, an MFG encapsulates the impact of all agents on a given one through the concept of *population*. In this work, we consider reward functions that depend on the population distribution over the state space $\mu \in \Delta(\mathcal{S}) := \{\text{distributions on } \mathcal{S}\}$. Specifically, a reward function $r : \mathcal{S} \times \mathcal{A} \times \Delta(\mathcal{S}) \to [0, R]$ signals a reward at each state-action pair based on the population distribution.

In MFGs, agents are rational and aim to maximize their expected cumulative reward. Our goal is to find an *optimal* policy—one that cannot be improved given that other agents' policies are fixed. We utilize a value-based approach to calculate policies. A Q-value function returns the expected cumulative reward starting from a state following the current policy π and population distribution μ:

$$Q_{\pi,\mu}(s,a) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, a_t, \mu\right) \,\middle|\, s_0 = s, a_0 = a\right], \tag{1}$$

where the expectation is taken w.r.t. the transition kernel P_π. Given a value function, we can choose the action accordingly, e.g., greedily select the action that maximizes the value function or use an ϵ-greedy selection. For broader adaptability, we presume access to a *policy operator* Γ_π that yields a policy based on a value function. Thus, the optimal policy can be characterized through a value function, translating our goal into discovering an optimal value function. We are now ready to present the optimality conditions.

Definition 1 (Mean field Nash equilibrium)**.** A value function-population distribution pair (Q, M) is a *mean field Nash equilibrium (MFNE)* if it satisfies

$$Q = \mathcal{T}_{Q,M} Q \quad \text{and} \quad M = \mathcal{P}_Q M, \tag{2}$$

where $\mathcal{T}_{Q,M}$ is the Bellman operator:

$$\mathcal{T}_{Q,M} Q(s,a) = \mathbb{E}_Q[r(s,a,M) + \gamma Q(s', a')], \tag{3}$$

where $\mathbb{E}_Q$ denotes the expectation over $a, a' \sim \Gamma_\pi(Q)$ and $s' \sim P(\cdot \mid s, a)$; and $\mathcal{P}_Q$ is the transition operator:

$$\mathcal{P}_Q M(s') = \textstyle\sum_{s\in\mathcal{S}} P_Q(s, s')M(s), \tag{4}$$

where we write $P_Q \coloneqq P_{\Gamma_\pi(Q)}$, as the policy is determined by the value function given a fixed policy operator.

In Definition 1, $Q \in \mathbb{R}^{S\times A}$ denotes a generic value function table, which is not necessarily an actual value function defined per (1). Similarly, $M \in \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is the probability simplex over $\mathcal{S}$, represents a generic population distribution which is not necessarily an actual policy-induced population distribution. Analogous to the Q-value function, we refer to this generic population distribution as the *M-value function*. We use subscripts, e.g., Q_M and μ_Q, to indicate actual BRs and IPs w.r.t. specific population distributions and value functions.

2.2 Fixed-Point Iteration for MFG

Fixed-point iteration (FPI), a classic method for learning MFGs, comprises two steps: evaluating the *best response* (BR) and the *induced population distribution* (IP). Fixing a population distribution M, the game reduces to a standard RL problem, which has a unique optimal value function [5], i.e., the BR w.r.t. the population distribution M. If the transition kernel P_Q yields a steady state distribution, this distribution is referred to as the IP w.r.t. the value function Q. Decomposing (2) gives formal definitions of these two operations.

Definition 2 (FPI operators)**.** The BR operator,

$$\Gamma_{\mathrm{BR}} : \Delta(\mathcal{S}) \to \mathbb{R}^{S\times A},\ M \mapsto Q_M,$$

returns the unique solution to the Bellman equation $Q_M = \mathcal{T}_{Q_M, M} Q_M$ for any population distribution M. The IP operator,

$$\Gamma_{\mathrm{IP}} : \mathbb{R}^{S\times A} \to \Delta(\mathcal{S}),\ Q \mapsto \mu_Q,$$

returns the unique fixed point of the transition operator $\mathcal{P}_Q$ defined in (4) for any value function Q. Then, the FPI operator is the composition of the above two operators: $\Gamma \coloneqq \Gamma_{\mathrm{IP}} \circ \Gamma_{\mathrm{BR}} : \Delta(\mathcal{S}) \to \Delta(\mathcal{S})$.

Notably, the optimality in the BR is determined by the policy operator Γ_π. For example, When Γ_π is the greedy selector: $\Gamma_\pi^{(\max)}(Q)[a|s] = \mathbb{1}(a = \operatorname{argmax}_a Q(s, a))$, (3) becomes the Bellman optimality operator:

$$\mathcal{T}_M Q(s, a) = \mathbb{E}[r(s, a, M) + \gamma \max_{a'} Q(s', a')],$$

making BRs deterministic optimal policies. When Γ_π is the softmax function: $\Gamma_\pi^{(\mathrm{softmax})}(Q)[a \mid s] = e^{LQ(s,a)} / \sum_{a'} e^{LQ(s,a')}$, where L is the inverse temperature parameter, the optimality corresponds to the MFG with entropy regularization [11, 17, 2].

To focus on the main ideas, we consider *contractive* MFGs in this paper, where FPI is guaranteed to converge to the unique MFNE. Then, in Section 5, we show that our methods approximate FPI, thus enjoying a similar convergence guarantee. Without the contraction condition, stabilization techniques like fictitious play and online mirror descent need to be applied to FPI. We envision that our algorithms can be extended to incorporate these techniques with our analysis applying with minimal adjustment.

Assumption 1 (Contractive MFG)**.** The FPI operator is $(1-\kappa)$-contractive ($\kappa \in (0, 1]$), i.e., for any $M_1, M_2 \in \Delta(\mathcal{S})$, it holds that

$$\|\Gamma M_1 - \Gamma M_2\|_2 \le (1-\kappa)\|M_1 - M_2\|_2.^1$$

3 Online Stochastic Updates

Without prior knowledge of the environment or the population, the online agent maintains two estimates—the Q-value function for the BR and the M-value function for the IP—which it updates using online stochastic observations. We first extend temporal difference (TD) control methods, a classic model-free RL framework covering Q-learning and SARSA [33], to learn BRs, and then derive an online stochastic update rule for the IPs in the same vein.

Q-value function update. Guided by the Bellman operator (3), TD control gives an online stochastic update for the Q-value function:

$$\begin{aligned} Q(s, a) \leftarrow& Q(s, a) - \alpha g_Q(s, a, s', a'),\\ \text{with}\quad g_Q =& Q(s, a) - r(s, a, M) - \gamma Q(s', a'), \end{aligned} \tag{5}$$

where α is the step size, $s' \sim P(\cdot \mid s, a)$, and $a' \sim \Gamma_\pi(Q)$. If the policy operator is greedy and the behavior policy is fixed, the above update rule gives rise to off-policy Q-learning [34]; for general policy operators, if the behavior policy updates in accordance to the value function, i.e., $\Gamma_\pi(Q)$ is the behavior policy and a' is the actual action, the above update rule gives rise to on-policy SARSA [29, 31]. We defer further discussion on these two TD control methods to Sections 4 and 5.

Population estimate. TD control replaces the expectation in the Bellman operator (3) with a stochastic approximation using online observations. Likewise, for the M-value function update, we first rewrite the transition operator using expectation:

$$\begin{aligned} \mathcal{P}_Q M(z) &= \sum_{s'\in\mathcal{S}} \delta_{s'}(z)\mathcal{P}_Q M(s')\\ &= \sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}} \delta_{s'}(z) P_Q(s, s')M(s)\\ &= \mathbb{E}_{Q,M}[\delta_{s'}(z)], \end{aligned}$$

where $\delta_{s'}$ is the indicator probability vector in $\Delta(\mathcal{S})$ such that $\delta_{s'}(z) = \mathbb{1}(z = s')$, and the expectation is taken over $s \sim M$ and $s' \sim P_Q(s, \cdot)$. Mimicking TD control and stochastic gradient descent, we remove the expectation and use the observed next successive state s' to stochastically approximate $\mathcal{P}_Q M$. This gives an online stochastic update for the M-value function:

$$M \leftarrow M - \beta g_M(s') = M + \beta(\delta_{s'} - M). \tag{6}$$

where β is the step size. Please refer to [37, Section J] for the full derivation of this update rule. Similar to TD control, we anticipate that this update rule drives the M-value function to converge to the population distribution induced by P_Q. Furthermore, selecting a step size of $\beta_t = 1/(t+1)$ simplifies it to a Markov chain Monte Carlo (MCMC) method, validating its correctness.

For online stochastic updates (5) and (6) and general online learning methods to yield optimal solutions, the environment must be readily *explorable*. Unlike offline methods which rely on pre-collected data, an online agent learns and acts based on its real-time observations. Hence, the efficient learning of optimal policy becomes unfeasible if certain states are inaccessible, leading to potential suboptimal solutions. To avoid this, we impose the following condition on the MDP.

[1] We consider L_1 and L_2 distances for probability measures in this work. For a finite state space with the trivial metric, the total variation distance equals the 1-Wasserstein distance [15], with the L_1 distance being twice as large as them. Without loss of generality, we redefine the total variation distance as twice its standard definition, and use it interchangeably with the L_1 distance.

Assumption 2 (Ergodic MDP)**.** For any $Q \in \mathbb{R}^{S\times A}$, the Markov chain induced by the transition kernel P_Q is ergodic with a uniform mixing rate. In other words, there exists a steady state distribution μ_Q for any policy $\Gamma_\pi(Q)$, with constants $m \geq 1$ and $\rho \in (0,1)$, such that

$$\sup_{s\in\mathcal{S}} \sup_{Q\in\mathbb{R}^{S\times A}} \|P_\pi\left(S_t = \cdot \,|\, S_0 = s\right) - \mu_Q\|_{\mathrm{TV}} \leq m\rho^t.$$

For future reference, we define an auxiliary constant $\sigma = \hat{n} + m\rho^{\hat{n}}/(1-\rho)$, where $\hat{n} = \lceil \log_\rho m^{-1} \rceil$. And the probability of visiting a state-action pair under a steady distribution is lower bounded:

$$\inf_{(s,a)\in\mathcal{S}\times\mathcal{A}} \inf_{Q\in\mathbb{R}^{S\times A}} \mu_Q(s)\cdot\Gamma_\pi(Q)[a\,|\,s] \geq \lambda_{\min} > 0.$$

Assumption 2 is a standard assumption for online methods [6, 38, 28].

4 Proposed Methods

4.1 Off-Policy QM Iteration

A significant advantage of our online stochastic formulation of the update rules, over the *iterative* BR and IP evaluation typical of FPI methods, is that it enables *simultaneous* updates of both the Q-value and M-value functions using the same batch of observations.

Taking one step in this direction, we first present an algorithm that simultaneously evaluates both the BR and IP, with the agent's behavior policy being fixed within each outer iteration. Since the behavior policy is not updated along with the Q-value function, we use off-policy Q-learning to learn the BR, and term this method *off-policy* QM iteration. The method is presented in Algorithm 1 with input option `off-policy` and the greedy policy operator $\Gamma_\pi^{(\max)}$.

Our algorithm showcases marked simplicity. At each time-step, the online agent observes a state transition and a reward; it then uses this information to update the Q-value and M-value function tables using (5) and (6), respectively, which only involves two elementary operations—scaling and addition. It is noteworthy that in the Q-value function update, a_{t+1}, which follows the fixed behavior policy, is not used. Instead, $a' \sim \Gamma_\pi^{(\max)}(Q_{k,t})$ is used according to (5). The discrepancy accounts for the naming of "off-policy" Q-learning.

The stationary nature of the transition kernel within each outer iteration directly gives the convergence guarantee of off-policy QMI and suggests its analogy with FPI (see Section 5). Nevertheless, fixed transition kernels make off-policy QMI learn BRs and IPs *parallelly*. That is, at kth iteration, the BR w.r.t. $M_{k,0}$ is approximated by $Q_{k,T} = Q_{k+1,0}$, whose corresponding population distribution is then approximated by $M_{k+1,T} = M_{k+2,0}$, rather than $M_{k+1,0}$. Let $Q_k \coloneqq Q_{k,0}$ and $M_k \coloneqq M_{k,0}$. Then, off-policy QMI generates two non-interacting parallel policy-population sequences: $\{(Q_{2k}, M_{2k+1})\}_{k=0}^{K/2}$ and $\{(Q_{2k+1}, M_{2k})\}_{k=0}^{K/2}$. This observation also implies that off-policy QMI is at least twice as data-efficient as FPI; see Figure 2 for an illustration.

To establish the convergence guarantee of off-policy QMI, we leverage the theoretical results of off-policy Q-learning. However, the greedy policy operator used is too *nonsmooth*: a slight difference in the Q-value function can lead to completely different action choices. As a result, the induced population distributions can drastically differ between outer iterations, leading to unstable convergence performance. Since we do not incorporate stabilization techniques, we make the following assumption.

Algorithm 1 QM Iteration

Input: initial value functions $Q_{-1,T} = Q_0$ and $M_{-1,T} = M_0$; initial state s_0; option `off-policy` or `on-policy`
for $k = 0, 1, ..., K$ **do**
  $Q_{k,0} = Q_{k-1,T}$, $M_{k,0} = M_{k-1,T}$
  $\pi_{k,0} = \Gamma_\pi(Q_{k,0})$
  for $t = 0, 1, \ldots, T$ **do**
    sample one Markovian observation tuple $(s_t, a_t, s_{t+1}, a_{t+1})$ following policy $\pi_{k,t}$
    observe the reward $r(s_t, a_t, M_{k,0})$
    $Q_{k,t+1}(s_t,a_t) = Q_{k,t}(s_t,a_t) - \alpha_t g_{Q_{k,t}}$
    $M_{k,t+1} = M_{k,t} - \beta_t g_{M_{k,t}}(s_{t+1})$
    if `off-policy` **then**
      $\pi_{k,t+1} = \pi_{k,0}$
    else if `on-policy` **then**
      $\pi_{k,t+1} = \Gamma_\pi\left(\mathrm{mix}\left(\{Q_{k,l}\}_{l=0}^{t+1}\right)\right)$
    end if
  end for
end for
return $Q_{K,T}, M_{K,T}$

Assumption 3 (Lipschitz continuous transition kernels for Q-learning)**.** For any $Q_1, Q_2 \in \mathbb{R}^{S\times A}$, it holds that

$$\|P_{Q_1} - P_{Q_2}\|_{\mathrm{TV}} \leq L\,\|Q_1 - Q_2\|_2\,,$$

where $\|P_Q\|_{\mathrm{TV}} \coloneqq \sup_{\|q\|_{\mathrm{TV}}=1} \left\|\sum_{s\in\mathcal{S}} q(s) P_Q(s,\cdot)\right\|_{\mathrm{TV}}$.

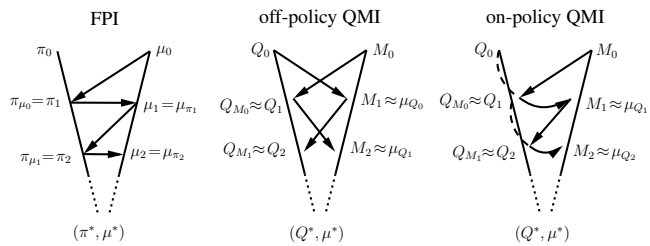


Figure 2: Illustration of learning processes. Each arrow represents one iteration in FPI or one outer iteration in QMI, matching the end BR or IP with the population distribution or value function at the start. The dashed line in on-policy QMI represents behavior policy updates, making M_k match the updated BR estimate Q_k.

4.2 On-Policy QM Iteration

Still, BR and IP evaluations are executed parallelly in off-policy QMI, and thus its efficiency boost indirectly attributes to parallel computing. This naturally raises a question: can we directly approximate the FPI operator Γ in one outer iteration? The *on-policy* variant of QM iteration provides a positive response. This time, we pass to Algorithm 1 the option `on-policy` and a general policy operator satisfying Assumption 4, facilitating dynamic updates of agent's behavior policy within each outer iteration. By syncing the behavior policy in accordance with the Q-value function, the policy learning process is governed by on-policy SARSA. Additionally, since the agent now observes the state transition induced by the updated policy, the M-value function is updated towards the population distribution induced by the updated policy. The learning process of on-policy QMI is illustrated in Figure 2.

On the other hand, constantly changing behavior policies in on-policy QMI yield nonstationary Markov chains. Such nonstationarity

renders the convergence guarantee of off-policy QMI not applicable here and complicates the convergence analysis of on-policy QMI. Nonetheless, we establish a similar convergence guarantee for on-policy QMI (Lemma 3). To achieve the sharp logarithmic dependency on T, we *mix* the Q-value functions obtained in an outer iteration: $\mathrm{mix}\left(\{Q_{k,l}\}_{l=0}^{t+1}\right) \coloneqq \sum_{l=0}^{t}(w_l/\sum_{l=0}^{t} w_l)Q_{k,l}$, where $w_l \asymp t$, and use this convex combination to determine the behavior policy.

Theoretical results of on-policy SARSA are used to establish the convergence guarantee of on-policy QMI. While the instability issue persists as in off-policy QMI, on-policy SARSA's adaptability and versatility—facilitated by its use of general policy operators—outstrip those of Q-learning, thus allowing us to directly impose the smoothness condition on Γ_π.

Assumption 4 (Lipschitz continuous policy operator for SARSA)**.** For any $Q_1, Q_2 \in \mathbb{R}^{S\times A}$ and $s \in \mathcal{S}$, it holds that

$$\|\Gamma_\pi(Q_1)[\cdot \mid s] - \Gamma_\pi(Q_1)[\cdot \mid s]\|_{\mathrm{TV}} \le L\|Q_1 - Q_2\|_2.$$

Furthermore, the Lipschitz constant satisfies $L \le \lambda_{\min}(1-\gamma)^2/(2R\sigma)$.

Remark 1. Assumption 3 is weaker than Assumption 4 as the latter implies the former (see [37, Lemma 4]) and requires a small Lipschitz constant. However, verifying Assumption 3 can be difficult as the dependence of P_Q on Q can be intricate and the model is unknown, whereas Assumption 4 is more achievable given the flexibility in choosing policy operators for SARSA. For instance, the softmax function with an apt temperature parameter satisfies Assumption 4 [14]. Actually, a softmax policy operator imposes entropy regularization to the greedy selection [14], a common technique used to stabilize the MFG learning process [11, 17, 2]. Absent such regularization, Assumption 3 ensures training stability. Other assumptions have been posited for this purpose, such as a strongly convex Bellman operator [1]. Notably, Assumption 1 and 3 or 4 are not mutually exclusive; either Assumption 3 or 4 with some conditions on the reward function's smoothness and Lipschitz constants can imply Assumption 1 [16, 1].

4.3 Comparison of Off- and On-Policy QMI

Table 1: Comparison of off- and on-policy QMI.

	Off-Policy	**On-Policy**
Behavior policy within an outer iteration	fixed	adaptive
Policy type	greedy	soft
MFNE	original	regularized
Sample efficiency boost mechanism	parallel	concurrent
Population-dependent transition kernels	✗	✓

Table 1 gives an overview of the differences between off- and on-policy variants of QMI. By utilizing Q-learning with a greedy policy operator, off-policy QMI can learn a deterministic optimal policy of the original MFG. On-policy QMI, on the other hand, utilizes SARSA with a *soft* (non-deterministic) policy operator, meaning that the learned MFNE depends on the policy operator and corresponds to an implicitly regularized MFG. Nevertheless, on-policy QMI affords flexibility in choosing a wider range of policy operators, and the soft policies it acquires exhibit greater robustness [33]. Furthermore, off-policy QMI boosts the sample efficiency by learning two policy-population sequences parallelly, while on-policy QMI directly boosts it by amalgamating the two steps in FPI into one. Last but not least, on-policy QMI and its convergence guarantee can directly accommodate transition kernels that are dependent not only on behavior policy but also on population distribution. However, such a dependence breaks the parallel procedure in off-policy QMI. See [37, Section M.1] for more discussion on population-dependent transition kernels.

5 Sample Complexity Analysis

In this section, we establish the sample complexity guarantee for both the off- and on-policy variants of Algorithm 1, given our assumptions on MDPs (Assumption 2), MFGs (Assumption 1), and smoothness (Assumption 3 or Assumption 4). To assist the analysis, we define the operators presented in Algorithm 1, which correspond to those in Definition 2.

Definition 3 (QMI operators)**.** For off-policy QMI, the Q- and M-value function operators,

$$\Gamma_Q(T) : \Delta(\mathcal{S}) \to \mathbb{R}^{S\times A},\ M_{k,0} \mapsto Q_{k,T} \quad \text{and}$$
$$\Gamma_M(T) : \mathbb{R}^{S\times A} \to \Delta(\mathcal{S}),\ Q_{k,0} \mapsto M_{k,T},$$

return the updated Q- and M-value function after an outer iteration of Algorithm 1, consisting of T online stochastic updates using Lines 8 and 9. Then, the off-policy QMI operator is the composition of the above two operators: $\hat{\Gamma}_{\mathrm{off}}(T) \coloneqq \Gamma_M(T) \circ \Gamma_Q(T)$.

The on-policy QMI operator,

$$\hat{\Gamma}_{\mathrm{on}}(T) : \Delta(\mathcal{S}) \to \Delta(\mathcal{S}),\ M_{k,0} \mapsto M_{k,T},$$

returns the updated M-value function after an outer iteration of Algorithm 1, consisting of T online stochastic updates using Lines 8 and 9, as well as the policy updates using Line 13.

As mentioned in previous sections, the equivalence between off-policy QMI and FPI comes from the convergence of off-policy Q-learning and MCMC. Specifically, we have the following two lemmas.

Lemma 1 (Sample complexity of Q-learning [28, Theorem 7])**.** *Suppose Assumptions 2 and 3 hold for the greedy policy operator. With a step size of $\alpha_t \asymp 1/(\lambda_{\min}(1-\gamma)t)$, for any $M\in\Delta(\mathcal{S})$, we have*

$$\mathbb{E}\,\|\Gamma_Q(T)M - \Gamma_{\mathrm{BR}}M\|_2^2 = O\left(\frac{SAR^2\log T}{\lambda_{\min}^2(1-\gamma)^5 T}\right),$$

where MDP components $\mathcal{S}$, $\mathcal{A}$, R, and γ are defined in Section 2.1, with S and A denote the cardinality of $\mathcal{S}$ and $\mathcal{A}$. σ and $\lambda_{\min}$ are defined in Assumption 2, and L is defined in Assumption 3.

Lemma 2 (Sample complexity of stationary MCMC [20, Theorem 3.1])**.** *Suppose Assumption 2 holds. With a step size of $\beta_t \asymp 1/t$, for any $Q \in \mathbb{R}^{S\times A}$, we have*

$$\mathbb{E}\,\|\Gamma_M(T)Q - \Gamma_{\mathrm{IP}}Q\|_2^2 = O\left(\frac{SA}{(1-\rho)^2 T}\right).$$

The preceding lemmas demonstrate that off-policy QMI efficiently approximates FPI, with the Q-value and M-value updates evaluating the BR and IP, respectively.

Lemmas 1 and 2 are not applicable to on-policy QMI, where transition kernels are nonstationary. Nonetheless, we can establish the following lemma.

Lemma 3 (Sample complexity of nonstationary MCMC with SARSA)**.** *Suppose Assumptions 2 and 4 hold for the chosen policy operator. With a step size of $\alpha_t \asymp 1/(\lambda_{\min}(1-\gamma)t)$ and $\beta_t \asymp 1/t$, for any $M \in \Delta(\mathcal{S})$, we have*

$$\mathbb{E}\left\|\hat{\Gamma}_{\text{on}}(T)M - \Gamma M\right\|_2^2 = O\left(\frac{SAR^2L^2\sigma^2\log T}{\lambda_{\min}^2(1-\gamma)^4 T}\right).$$

An outer iteration of on-policy QMI corresponds to a nonstationary MCMC. Thus, to prove Lemma 3, we employ a *backtracking* procedure, a technique developed in Zou et al. [38] to address nonstationarity in stochastic approximation methods. The key idea is that in order to exploit the mixing property of stationary Markov chains (Assumption 2), we virtually backtrack a period τ, and generate a virtual trajectory where the agent follows the fixed behavior policy $\pi_{t-\tau} := \Gamma_\pi(Q_{t-\tau})$ after time step $t-\tau$. This virtual trajectory is stationary after time step $t-\tau$ and rapidly converges to the steady distribution induced by $\pi_{t-\tau}$, denoted by $\mu_{t-\tau}$. Next, the convergence of SARSA confirms that $\mu_{t-\tau}$ converges to the steady distribution induced by the BR w.r.t. M, denoted by $\mu_M := \Gamma M$. Let s_t and $\tilde{s}_t$ be the state at time step t on the actual and virtual trajectories, respectively. Let π_t be the (actual) behavior policy at time step t, with its induced steady distribution denoted as μ_t. Then, the proof sketch for Lemma 3 can be succinctly portrayed as:

$$\underbrace{s_t \approx \tilde{s}_t}_{H_1,\text{backtrack}} \overbrace{\xrightarrow[\tau\to\infty]{d}}^{H_2,\text{mix}} s \sim \underbrace{\mu_{t-\tau} \approx \mu_t}_{H_3,\text{progress}} \overbrace{\xrightarrow[t\to\infty]{L_2}}^{H_4,\text{SARSA}} \mu_M,$$

where the backtracking discrepancy H_1 and the distribution progress H_3 are controlled by the virtual period τ ([37, Lemmas 8 and 10]), while the two convergence rates H_2 and H_4 are characterized by the geometric ergodicity of stationary MDPs and the sample complexity of SARSA ([37, Lemmas 7 and 9]), respectively. In brief, we show that the agent's state distribution, and thus its M-value function, rapidly converges to the IP μ_M.

Given the above lemmas, we are ready to compose the convergence guarantee and sample complexity of QMI.

Theorem 1 (Sample complexity of QMI)**.** *Suppose Assumptions 1 and 2 hold, and Assumptions 3 and 4 hold for off- and on-policy QMI, respectively. Let μ^* be the MFNE population distribution. Then Algorithm 1 returns an ϵ-approximate MFNE, that is,*

$$\mathbb{E}\left\|M_{K,T} - \mu^*\right\|_2^2 = \mathbb{E}\|\hat{\Gamma}(T)^K M_0 - \mu^*\|_2^2 \le \epsilon^2,$$

where $\hat{\Gamma}$ can be either $\hat{\Gamma}_{\text{off}}$ or $\hat{\Gamma}_{\text{on}}$, with the number of iterations being at most

$$K = O\left(\kappa^{-1}\log\epsilon^{-1}\right), \quad T = C\cdot O\left(\kappa^{-2}\epsilon^{-2}\log\epsilon^{-1}\right),$$

where

$$C \le \frac{SAR^2L^2\sigma^2}{\lambda_{\min}^2(1-\gamma)^5}.$$

Our complexity results match the prior work on learning MFGs [2] and are consistent with stochastic approximation methods [20, 38, 28].

6 Numerical Experiments

In this section, we present two experiments demonstrating the effectiveness of our methods. We compare our methods with model-based FPI using two key metrics: the mean squared error (MSE) of the population distribution and the exploitability of the policy. For a finite state space with the trivial metric, the total variation distance equals the 1-Wasserstein distance [15], and is equivalent to the Euclidean norms. Thus, we consider the L_2 MSE between the current M-value function and the MFNE population distribution:

$$\text{MSE}(M) := \|M_k - \mu^*\|_2^2 = \sum_{s\in\mathcal{S}}\left(M_k(s) - \mu^*(s)\right)^2.$$

Perrin et al. [26] defines the exploitability of a policy as follows:

$$\text{exploitability}(\pi) := \max_{\pi'} \mathbb{E}_{s\sim\mu_\pi} V(s;\pi',\mu_\pi) - \mathbb{E}_{s\sim\mu_\pi} V(s;\pi,\mu_\pi),$$

where $V(s;\pi,\mu)$ is the value function determined by policy π and population distribution μ. Given a policy operator, the exploitability quantifies the gap between the current value function Q and the BR w.r.t. the population distribution induced by Q. We denote this BR by Q_{μ_Q}, and calculate $\|Q_{\mu_Q} - Q\|$ as the exploitability in practice.

For model-based FPI, we use value iteration to calculate BRs [33]:

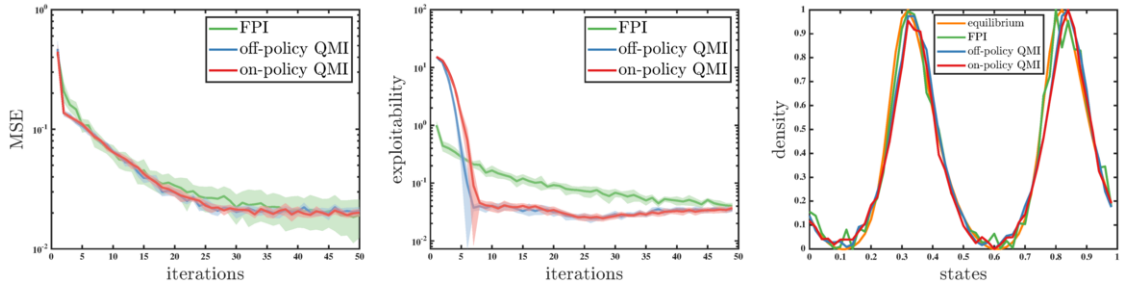
$$V_{t+1}(s) = \max_{a\in\mathcal{A}}\left\{r(s,a,\mu) + \gamma\sum_{s'} P(s'\mid s,a)V_t(s')\right\}, \quad (7)$$

and the induced population distributions are directly calculated using (4). Model-based FPI assumes full knowledge of the state-action space, reward function, as well as transition dynamics. During each iteration of value iteration and population update using (4), all S values are updated without any random sampling, and we refer to such an iteration as a *sweep*. It is expected that in order for online sampling to replicate the effects of a sweep, the number of samples should be at least S. Furthermore, since QMI assumes no knowledge of the action space and the reward function, it may require A samples to achieve the same effect as the $\max$ calculation in (7). The randomness in sampling can further impact efficiency. Therefore, we introduce a *sample compensation factor* η to relate the number of samples to the number of sweeps. Specifically, let T_{QMI} and T_{FPI} be the number of inner iterations of QMI (Algorithm 1) and the number of sweeps of value iteration and population update in FPI respectively; we let

$$T_{\text{QMI}} = \eta S T_{\text{FPI}}.$$

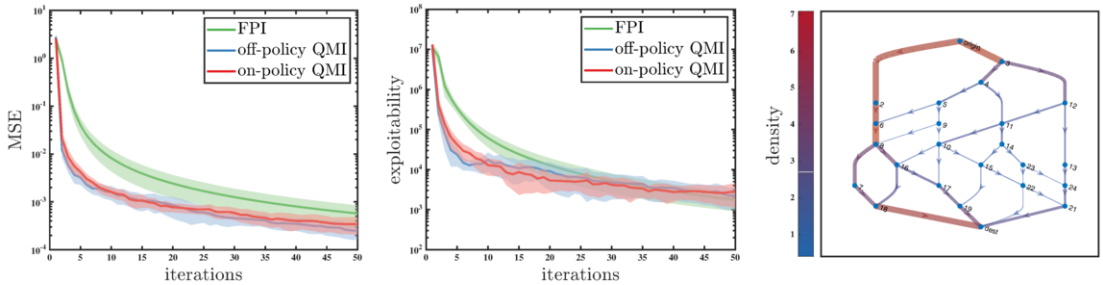
Please refer to [37, Section H] for additional experiments on different sample compensation factors, which suggest that a small sample compensation factor is sufficient for QMI. In this section, we fix η as an algorithmic parameter. Please note that we do not claim that our methods outperform FPI in all scenarios as they are different types of algorithms designed for different situations.

Speed control on a ring road. We consider a speed control game of autonomous vehicles on a ring road, i.e., the unit circle $\mathbb{S}^1 \cong [0,1)$, as illustrated in Figure 1. At location $s \in \mathbb{S}^1$, the representative vehicle selects a speed a, and then moves to the next location following transition $s' = s + a\Delta t \pmod 1$, where Δt is the time interval between two consecutive decisions. Without loss of generality, we assume that the speed is bounded by 1, i.e., the speed space is also $[0,1)$. Then we discretize both the location space and the speed space using a granularity of $\Delta s = \Delta a = 0.02$. Thus, both our discretized state (location) space and action (speed) space can be represented by $\mathcal{S} = \mathcal{A} = \{0, 0.02, \ldots, 0.98\} \cong [50]$. By the Courant-Friedrichs-Lewy condition, we choose the time interval to be $\Delta t = 0.02 \le \Delta s/\max a$. The objective of a vehicle is to maintain some desired speed while avoiding collisions with other vehicles. Thus, it needs to



(a) Mean squared error (b) Exploitability (c) Learned population distributions

Figure 3: Performance comparison of FPI, off-policy QMI, and on-policy QMI on ring road speed control. MSE represents the mean squared L_2 error between the current M-value function and the MFNE population distribution. Exploitability refers to the disparity between the current value function and the BR w.r.t. the current population distribution. Learned population distributions are scaled for better visualization.



(a) Mean squared error (b) Exploitability (c) Learned population distribution

Figure 4: Performance comparison of FPI, off-policy QMI, and on-policy QMI on Sioux Falls network routing. Only the population distribution learned by off-policy QMI is shown in (c); other methodsgive similar population distributions (please refer to [37, Section H.1]).

reduce the speed in areas with high population density. A classic cost function for this goal is the Lighthill-Whitham-Richards function:

$$r^{(\mathrm{LWR})}(s,a,\mu) = -\frac{1}{2}\left(\left(1-\frac{\mu(s)}{\mu_{\mathrm{jam}}}\right)-\frac{a}{a_{\max}}\right)^2\Delta s,$$

where μ_{jam} is the jam density, and $a_{\max}$ is the maximum speed. However, in an infinite horizon game, this cost function induces a *trivial* MFNE, where the equilibrium policy and population are both constant across the state space. Therefore, we introduce a stimulus term b that varies across different locations:

$$r(s,a,\mu) = -\frac{1}{2}\left(b(s)+\frac{1}{2}\left(1-\frac{\mu(s)}{\mu_{\mathrm{jam}}}\right)-\frac{a}{a_{\max}}\right)^2\Delta s,$$

where the factor of one-half before the population distribution term is included to account for the presence of the new stimulus term. This new cost function makes the MFNE more complex and corresponds to real-world situations where vehicles may have distinct desired speeds at different locations due to environmental variations. Specifically, we choose the stimulus term as $b(s) = 0.2(\sin(4\pi s)+1)$, and set $\mu_{\mathrm{jam}} = 3/S$ and $a_{\max} = 1$. The performance comparison is reported in Figure 3.

Routing game on a network. We consider a routing game on the Sioux Falls network, a graph with 24 nodes and 74 directed edges. We designate node 1 as the starting point and node 20 as the destination. To construct an infinite-horizon game, we add a *restart* edge e_{75} from the destination back to the starting point. On each edge, a vehicle selects its next edge to travel to. We consider a deterministic environment, meaning that the vehicle will follow the chosen edge without any randomness. Therefore, both the state space and the action space can be represented by the edge set, i.e., $\mathcal{S} = \mathcal{A} = \{e_1,\dots,e_{75}\} \cong [75]$, where e_{75} is the restart edge. It is worth noting that a vehicle can only select from the outgoing edges of its current location as its next edge.

The objective of a vehicle is to reach the destination as fast as possible. Due to congestion, a vehicle spends a longer time on an edge with higher population distribution. Specifically, the cost (time) on a non-restart edge is $r^{(\mathrm{cong.})}(s,a,\mu) = -c_1\mu(s)^2\mathbb{1}\{s \neq e_{75}\}$, where c_1 is a cost constant. To drive the vehicle to the destination, we impose a reward at the restart edge: $r^{(\mathrm{term.})}(s,a,\mu) = c_2\mathbb{1}\{s = e_{75}\}$. Together, we get the cost function:

$$r(s,a,\mu) = \underbrace{-c_1\mu(s)^2\mathbb{1}\{s \neq e_{75}\}}_{\text{congestion cost}} + \underbrace{c_2\mathbb{1}\{s = e_{75}\}}_{\text{terminal reward}}.$$

The performance comparison is reported in Figure 4.

All numerical results are averaged over 10 independent runs. They demonstrate that QMI efficiently approximates FPI and achieves comparable performance, validating our fully online model-free approach. Please refer to [37, Section H] for the full setups of two experiments and additional results.

7 Conclusion

This study introduces the QM iteration (QMI), a novel online single-agent model-free learning scheme for mean field games, offering a practical alternative to traditional fixed-point iteration methods. QMI provides both theoretically and numerically confirmation of the statement: *a single online agent can efficiently learn the equilibria of mean field games*, without any prior knowledge of the environment. We anticipate that our methods can provide a benchmark for online model-free learning in MFGs, and serve as a base scheme for further extensions, generalizations, and applications.

References

[1] B. Anahtarci, C. D. Kariksiz, and N. Saldi. Fitted Q-learning in mean-field games. *arXiv preprint arXiv:1912.13309*, 2019.

[2] B. Anahtarci, C. D. Kariksiz, and N. Saldi. Q-learning in regularized mean-field games. *Dynamic Games and Applications*, 13(1):89–117, 2023.

[3] A. Angiuli, J.-P. Fouque, and M. Laurière. Unified reinforcement Q-learning for mean field game and control problems. *Mathematics of Control, Signals, and Systems*, 34(2):217–271, 2022.

[4] A. Angiuli, J.-P. Fouque, M. Laurière, and M. Zhang. Convergence of multi-scale reinforcement Q-learning algorithms for mean field game and control problems. *arXiv preprint arXiv:2312.06659*, 2023.

[5] D. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[6] J. Bhandari, D. Russo, and R. Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Conference on learning theory*, pages 1691–1692. PMLR, 2018.

[7] P. Cardaliaguet and S. Hadikhanloo. Learning in mean field games: The fictitious play. *ESAIM: Control, Optimisation and Calculus of Variations*, 23(2):569–591, 2017.

[8] R. Carmona, M. Laurière, and Z. Tan. Model-free mean-field reinforcement learning: mean-field MDP and mean-field Q-learning. *arXiv preprint arXiv:1910.12802*, 2019.

[9] X. Chen, S. Liu, and X. Di. Learning dual mean field games on graphs. In *Proceedings of the 2023 European Conference on Artificial Intelligence*, pages 421–428, 2023.

[10] X. Chen, S. Liu, and X. Di. A hybrid framework of reinforcement learning and physics-informed deep learning for spatiotemporal mean field games. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1079–1087, 2023.

[11] K. Cui and H. Koeppl. Approximately solving mean field games via entropy-regularized deep reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1909–1917. PMLR, 2021.

[12] K. Cui, A. Tahir, G. Ekinci, A. Elshamanhory, Y. Eich, M. Li, and H. Koeppl. A survey on large-population systems and scalable multi-agent reinforcement learning. *arXiv preprint arXiv:2209.03859*, 2022.

[13] R. Elie, J. Perolat, M. Laurière, M. Geist, and O. Pietquin. On the convergence of model free learning in mean field games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7143–7150, 2020.

[14] B. Gao and L. Pavel. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv preprint arXiv:1704.00805*, 2017.

[15] A. L. Gibbs and F. E. Su. On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435, 2002.

[16] X. Guo, A. Hu, R. Xu, and J. Zhang. Learning mean-field games. *Advances in Neural Information Processing Systems*, 32, 2019.

[17] X. Guo, R. Xu, and T. Zariphopoulou. Entropy regularization for mean field games with learning. *Mathematics of Operations research*, 47(4): 3239–3260, 2022.

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

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

[20] K. Łatuszyński, B. Miasojedow, and W. Niemiro. Nonasymptotic bounds on the estimation error of MCMC algorithms. *Bernoulli*, 19(5A):2033–2066, 2013.

[21] M. Laurière, S. Perrin, M. Geist, and O. Pietquin. Learning mean field games: A survey. *arXiv preprint arXiv:2205.12944*, 2022.

[22] M. Laurière, S. Perrin, S. Girgin, P. Muller, A. Jain, T. Cabannes, G. Piliouras, J. Pérolat, R. Elie, O. Pietquin, et al. Scalable deep reinforcement learning algorithms for mean field games. In *International Conference on Machine Learning*, pages 12078–12095. PMLR, 2022.

[23] K. Lee, D. Rengarajan, D. Kalathil, and S. Shakkottai. Reinforcement learning for mean field games with strategic complementarities. In *International Conference on Artificial Intelligence and Statistics*, pages 2458–2466. PMLR, 2021.

[24] D. Mguni, J. Jennings, and E. M. de Cote. Decentralised learning in systems with many, many strategic agents. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[25] J. Perolat, S. Perrin, R. Elie, M. Laurière, G. Piliouras, M. Geist, K. Tuyls, and O. Pietquin. Scaling up mean field games with online mirror descent. *arXiv preprint arXiv:2103.00623*, 2021.

[26] S. Perrin, J. Pérolat, M. Laurière, M. Geist, R. Elie, and O. Pietquin. Fictitious play for mean field games: Continuous time analysis and applications. *Advances in Neural Information Processing Systems*, 33: 13199–13213, 2020.

[27] S. Perrin, M. Laurière, J. Pérolat, M. Geist, R. Élie, and O. Pietquin. Mean field games flock! The reinforcement learning way. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 356–362. International Joint Conferences on Artificial Intelligence Organization, 2021.

[28] G. Qu and A. Wierman. Finite-time analysis of asynchronous stochastic approximation and Q -learning. In *Conference on Learning Theory*, pages 3185–3205. PMLR, 2020.

[29] G. A. Rummery and M. Niranjan. *On-Line Q-learning Using Connectionist Systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[30] Z. Shou, X. Chen, Y. Fu, and X. Di. Multi-agent reinforcement learning for Markov routing games: A new modeling paradigm for dynamic traffic assignment. *Transportation Research Part C: Emerging Technologies*, 137:103560, 2022.

[31] S. P. Singh and R. S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158, 1996.

[32] S. Subramanian, M. Taylor, M. Crowley, and P. Poupart. Decentralized mean field games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[33] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[34] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

[35] J. Yang, X. Ye, R. Trivedi, H. Xu, and H. Zha. Learning deep mean field games for modeling large population behavior. *arXiv preprint arXiv:1711.03156*, 2017.

[36] M. A. U. Zaman, A. Koppel, S. Bhatt, and T. Basar. Oracle-free reinforcement learning in mean-field games along a single sample path. In *International Conference on Artificial Intelligence and Statistics*, pages 10178–10206. PMLR, 2023.

[37] C. Zhang, X. Chen, and X. Di. A single online agent can efficiently learn mean field games. *arXiv preprint arXiv:2405.03718*, 2024. Full version of this paper.

[38] S. Zou, T. Xu, and Y. Liang. Finite-sample analysis for SARSA with linear function approximation. *Advances in neural information processing systems*, 32, 2019.