Les Cahiers du GERAD ISSN: 0711–2440

Reinforcement learning in stationary mean-field games

J. Subramanian,
A. Mahajan

G–2019–18

March 2019

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Reinforcement learning in stationary mean-field games

Jayakumar Subramanian [a, b]

Aditya Mahajan [a, b]

[a] *GERAD HEC Montréal, Montréal (Québec), Canada, H3T 2A7*

[b] *Electrical and Computer Engineering Department, McGill University, Montréal (Québec), Canada, H3A 0E9*

jayakumar.subramanian@mail.mcgill.ca
aditya.mahajan@mcgill.ca

March 2019
Les Cahiers du GERAD
G–2019–18

Abstract: Multi-agent reinforcement learning has made significant progress in recent years, but it remains a hard problem. Hence, one often resorts to developing learning algorithms for specific classes of multi-agent systems. In this paper we study reinforcement learning in a specific class of multi-agent systems systems called mean-field games. In particular, we consider learning in stationary mean-field games. We identify two different solution concepts—stationary mean-field equilibrium and stationary mean-field social-welfare optimal policy—for such games based on whether the agents are non-cooperative or cooperative, respectively. We then generalize these solution concepts to their local variants using bounded rationality based arguments. For these two local solution concepts, we present two reinforcement learning algorithms. We show that the algorithms converge to the right solution under mild technical conditions and demonstrate this using two numerical examples.

Keywords: Multi-agent reinforcement learning, mean-field games, stationary mean-field games, bounded rationality

1 Introduction

Multi-agent reinforcement learning (MARL) refers to systems in which multiple agents are acting in a common and unknown environment. The presence of other agents makes MARL different from traditional single agent RL. When we view the MARL setup from the point of view of a particular agent, say agent i, all other agents are part of the environment. Since these agents are also learning and changing their policies, the environment faced by agent i changes with time. Due to this perception of non-stationary environment, traditional single agent RL algorithms cannot be used in MARL.

Another feature of MARL is that the agents may be strategic (i.e., selfish) and wish to maximize their individual reward or they might be cooperative and wish to maximize their team reward. Depending on the case, the learning process in MARL should converge to a variation of Nash equilibrium or of social-welfare optimal (or team optimal) solution.

There is a rich literature on MARL which models the multi-agent interaction using the framework of stochastic dynamic games starting with [30], where a Q-learning algorithm that converges to a minimax solution of a zero-sum game was proposed. This was extended to an algorithm that converges to the Nash equilibrium of a general sum game (under some conditions) in [14]. Several other variations have been proposed in the literature and we refer the reader to [13, 6] for a detailed survey.

In recent years, there has been considerable interest in using deep neural networks in MARL. Most papers adopt the paradigm of *centralized training with decentralized execution* in which a centralized critic estimates the Q-function and decentralized actors optimize the policy of the agents. Examples include BICNET [34], MADDPG [31], and COMA [10].

These approaches, in general, do not scale with the number of agents. In the literature on planning for multi-agent systems, various frameworks have been proposed which easily scale to thousands of homogeneous agents. These include swarm based models, mean-field games (MFG), mean-field teams, and cooperative multi-agent systems [19, 18, 28, 2]. The central theme in all these approaches is the idea of mean-field (MF) approximation from statistical physics [44].

Motivated by the success of the planning frameworks, there have been several approaches which use mean-field approximation for reinforcement learning. The earliest of these is [25], which proposed a model based adaptive control algorithm for mean-field games. A Q-learning based algorithm for MFG control of coupled oscillators is proposed in [48]. Model-free Q-learning and actor critic algorithms for mean-field games have been proposed in [47, 33]. A detailed description of these papers is presented in Section 5.3. Another related work is [46], which proposed a mean-field based solution for inverse RL. Mean-field games are related to the notion of anonymous games, which considers static games with large number of anonymous agents [21, 4]. A learning framework for such games was presented in [22].

In the last decade, mean-field models have been successfully used in many planning problems in control engineering, network economics, and finance, but these results haven't been translated to the learning setup. A remarkable feature of mean-field models is that as the number of agents becomes large, the non-stationarity problem has negligible impact on the solution. In a mean-field model, agents are homogeneous and coupled only through the mean-field. Agents impact each other only through the mean-field distribution and once this is fixed, the agents are decoupled. Thus, MF models circumvent the non-stationarity problem by changing the solution concept. It has been shown that under appropriate conditions, the mean-field equilibrium is also a ε-Nash equilibrium, where ε is $O(1/\sqrt{n})$.

In this paper, we present reinforcement learning algorithms for *stationary* mean-field games. In the game theory and stochastic control literature, there are two very closely related modeling frameworks that are referred to as mean-field games and *stationary* mean-field games. We highlight the difference between these two modeling frameworks in Section 5.2. The current literature on using mean-field ideas in MARL focuses on computing Nash equilibrium of mean-field games. We propose reinforcement learning algorithms that compute *stationary mean-field equilibrium* and *social-welfare optimal solution*

of stationary mean-field games. Both the modeling framework and the solution concepts are different from what has previously appeared in the MARL literature. Our main contribution is to obtain RL algorithms for stationary MF models. Most existing works for RL for MF assumes non-stationary solution concept.

2 Background

2.1 Mean-field games (MFG)

Consider a mean-field game with n homogeneous agents, indexed by the set $N = \{1, 2, \dots, n\}$. Each agent has the same state and action spaces, which we denote by $\mathcal{X}$ and $\mathcal{A}$ respectively. Both $\mathcal{X}$ and $\mathcal{A}$ are finite sets. At any time t, $X_t^i \in \mathcal{X}$ and $A_t^i \in \mathcal{A}$ denote the state and action of agent $i \in N$. In a MFG, the dynamical evolution and the reward of each agent are decoupled from the rest of the agents given the mean-field, where the mean-field or the empirical distribution of the system is given by:

$$Z_t(x) = \frac{1}{n} \sum_{i \in N} \mathbb{1}\{X_t^i = x\}, \quad \forall x \in \mathcal{X}. \tag{1}$$

Note that $Z_t \in \Delta(\mathcal{X})$, the space of probability mass functions on $\mathcal{X}$. The state of agent i evolves according to:

$$X_{t+1}^i \sim P(X_t^i, A_t^i, Z_t), \tag{2}$$

where $P(x, a, z) \in \Delta(\mathcal{X})$ is the transition probability distribution given the state x, action a and mean-field z. With a slight abuse of notation, we use $P(y|x, a, z)$ to denote the probability that the next state is y given that the current state, action and mean-field are x, a and z respectively. The per-step reward for each agent $i \in N$ is given by:

$$R_t^i = r(X_t^i, A_t^i, Z_t, X_{t+1}^i). \tag{3}$$

The utility or the expected total reward for agent $i \in N$ is given by:

$$U^i = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R_t^i\Big], \tag{4}$$

where $\gamma \in (0, 1)$ is the discount factor.

The main idea of mean-field games is to approximate the above finite population system by an infinite population system, where the empirical mean-field almost surely converges to the statistical mean-field due to the strong law of large numbers. Thus the agents assume that:

$$Z_t(x) \approx \frac{1}{n} \sum_{i \in N} \mathbb{P}(X_t^i = x). \tag{5}$$

In addition, it is assumed that agents use an identical time varying policy $(\pi_1, \pi_2, \dots)$, where $\pi_t \colon \mathcal{X} \to \Delta(\mathcal{A})$ is the stochastic policy at time t and $A_t^i \sim \pi_t(X_t^i)$. When all agents follow policy $(\pi_1, \pi_2, \dots)$, the statistical mean-field evolves according to the discrete time McKean Vlasov equation:

$$Z_{t+1}(y) = \sum_{x \in \mathcal{X}} \sum_{a \in \mathcal{A}} Z_t(x) \pi_t(a|x) P(y|x, a, Z_t), \quad \forall y \in \mathcal{X}, \tag{6}$$

which we denote as:

$$Z_{t+1} = \Phi(Z_t, \pi_t). \tag{7}$$

2.2 Stationary MFG

In *stationary* MFG, the following additional assumptions are made [41, 42, 1].

(A1) Time homogeneous policy: All agents follow a time-homogeneous, stochastic policy, $\pi_t = \pi \colon \mathcal{X} \to \Delta(\mathcal{A})$ for all t, i.e., each agent chooses an action given by $A_t^i \sim \pi(X_t^i)$. With a slight abuse of notation, we use $\pi(a|x)$ to denote the probability of choosing action a in state x under policy π. Let Π denote the space of all such policies.

(A2) Stationarity of mean-field: When all agents follow a policy $\pi \in \Pi$, the mean-field of states $\{Z_t\}_{t\geq 0}$ converges almost surely to a constant limit z, which we call the stationary mean-field. Note that the stationary mean-field satisfies:

$$z = \Phi(z, \pi). \tag{8}$$

(A3) Agent's performance evaluation: Agents evaluate their performance by assuming that the population is infinite and the corresponding mean-field takes its stationary value at all times. In particular, given a policy $\pi \in \Pi$ and a candidate stationary mean-field distribution $z \in \Delta(\mathcal{X})$, agent i evaluates its performance starting from initial state $x \in \mathcal{X}$ as:

$$V_{\pi,z}(x) = \mathbb{E}_{\substack{A_t^i \sim \pi(X_t^i) \\ X_{t+1}^i \sim P(X_t^i, A_t^i, z)}} \left[\sum_{t=0}^{\infty} \gamma^t r(X_t^i, A_t^i, z, X_{t+1}^i) \middle| X_0^i = x \right].$$

Such a *mean-field approximation* may be written as the solution of the following Bellman fixed-point equation.

$$V_{\pi,z}(x) = \sum_{a\in\mathcal{A}} \pi(a|x) \Big[\sum_{y\in\mathcal{X}} P(y|x,a,z) \times \big[r(x,a,z,y) + \gamma V_{\pi,z}(y) \big] \Big].$$

2.3 Solution concepts

When agents are strategic (non-cooperative), the following refinement of Markov perfect equilibrium (MPE) is used as a solution concept [1].

Definition 1 (Stationary mean-field equilibrium (SMFE)) *A stationary mean-field equilibrium (SMFE) is a pair of policy $\pi \in \Pi$ and mean-field $z \in \Delta(\mathcal{X})$ which satisfies the following two properties:*

1. *Sequential rationality: For any other policy π',*

$$V_{\pi,z}(x) \geq V_{\pi',z}(x), \quad \forall x \in \mathcal{X}.$$

2. *Consistency: The mean-field z is stationary under policy π, i.e.,*

$$z = \Phi(z, \pi).$$

When agents are cooperative, the following refinement of social welfare optimal solution is used as a solution concept.

Definition 2 (Stationary mean-field social-welfare optimal policy (SMF-SO)) *A policy $\pi \in \Pi$ is stationary mean-field social welfare optimal (SMF-SO) if it satisfies the following property:*

- *Optimality: For any other policy $\pi' \in \Pi$,*

$$V_{\pi,z}(x) \geq V_{\pi',z'}(x), \quad \forall x \in \mathcal{X},$$

where z and z' are the stationary mean-field distributions corresponding to π and π', respectively, i.e., satisfy

$$z = \Phi(z, \pi) \quad \text{and} \quad z' = \Phi(z', \pi').$$

2.3.1 Comparison of the two solution concepts

The definitions of sequential rationality and optimality are different. In particular, sequential rationality is defined with respect to the mean-field z; while considering the performance of an alternative policy $\pi' \in \Pi$ it is assumed that the mean-field does not change. In contrast, optimality is a property of a policy; while considering the performance of an alternative policy $\pi' \in \Pi$, the mean-field approximation of the performance is with respect to the stationary mean-field corresponding to π'. Thus, in general, SMFE and SMF-SO are different.

2.4 Local solution concepts

Both the solution concepts described in Section 2.3 are global concepts, i.e., they are defined over all possible policies $\pi \in \Pi$. They are difficult to verify by agents with bounded rationality or limited computational resources. So, we define local variations of these solution concepts that are easier to verify. It is worth highlighting that when these local solution concepts are unique (as is the case in many examples), they coincide with the the global ones. To define these local solution concepts, we make two assumptions:

1. The initial states of all agents are independent and identically distributed according to $\xi_0 \in \Delta(\mathcal{X})$. Thus, the performance of any policy $\pi \in \Pi$ is given by:
$$J_{\pi,z} = \mathbb{E}_{X\sim\xi_0}[V_{\pi,z}(X)] = \sum_{x\in\mathcal{X}} V_{\pi,z}(x)\xi_0(x).$$
2. The policy $\pi \in \Pi$ is parametrized by $\theta \in \Theta$, where Θ is a convex and closed subset of a Euclidean space. We denote the policy parametrized by $\theta \in \Theta$ as π_θ. Examples of such parametrizations include Gibbs/Boltzmann distribution and neural networks.

Both these assumptions are standard in the reinforcement learning literature on policy gradient methods [40]. Based on these assumptions, we define the following local variants of SMFE and SMF-SO.

Definition 3 (Local stationary mean-field equilibrium (LSMFE)) *A local stationary mean-field equilibrium (LSMFE) is a pair of policy $\pi_\theta \in \Pi$ and mean-field $z \in \Delta(\mathcal{X})$ which satisfies the following two properties:*

1. *Local sequential rationality: $\partial J_{\pi_\theta,z}/\partial\theta = 0$.*
2. *Consistency: $z = \Phi(z, \pi_\theta)$.*

Definition 4 (Local stationary mean-field social welfare optimal policy (LSMF-SO)) *A policy $\pi_\theta \in \Pi$ is local stationary mean-field social welfare optimal (LSMF-SO) if it satisfies the following property:*

- *Local optimality: $dJ_{\pi_\theta,z_\theta}/d\theta = 0$, where z_θ is the stationary mean-field distribution corresponding to π_θ, i.e., satisfies $z_\theta = \Phi(z_\theta, \pi_\theta)$.*

2.4.1 Comparison of the two local solution concepts

From the chain rule of derivatives, we have
$$\frac{dJ_{\pi,z}(x)}{d\theta} = \frac{\partial J_{\pi,z}(x)}{\partial\pi}\frac{\partial\pi}{\partial\theta} + \frac{\partial J_{\pi,z}(x)}{\partial z}\frac{\partial z}{\partial\theta}.$$

The first term is equal to $\partial J_{\pi_\theta,z}(x)/\partial\theta$. In general, $\partial J_{\pi,z}(x)/\partial z \neq 0$ and $\partial z/\partial\theta \neq 0$. Thus, local optimality is not the same as local sequential rationality. This is also illustrated by the numerical results presented in Section 4.

2.4.2 Comparison of global and local solution concepts

Local variants of Nash equilibrium have been studied in the literature [35]. An interesting feature for MFG is that uniqueness of SMFE does not imply that LSMFE is same as SMFE. This is because unlike standard Nash equilibrium, SMFE and LSMFE are a collection of a strategy profile and stationary distribution. Sufficient conditions for LSMFE to be unique (and agree with the SMFE) are:

1. SMFE is unique.
2. The value function is concave in the policy parameters for every value of mean-field.

Conditions for unique local equilibrium are satisfied for the malware spread model presented in Section 4 [15, 16, 17].

3 RL for stationary MFG

In this section we propose two RL algorithms corresponding to each of the local solution concepts defined in Section 2.4. For both cases we assume that the agent has access to a simulator that yields the next state and the per-step reward for an agent, given the agent's current local state, current action and the current mean-field.

3.1 RL algorithm for learning LSMFE

The key idea behind the RL algorithm to learn an LSMFE is as follows. Suppose $G_{\theta,z}$ is an unbiased estimator of $\partial J_{\pi_\theta,z}/\partial\theta$. Then, we can start with an initial guess $\theta_0 \in \Theta$ and $z_0 \in \Delta(\mathcal{X})$ and at each step of the iteration, update the guess (θ_k, z_k) using two-timescale stochastic gradient ascent [8]:

$$z_{k+1} = z_k + \beta_k\big[\hat{\Phi}(z_k, \pi_{\theta_k}) - z_k\big], \tag{9a}$$

$$\theta_{k+1} = \big[\theta_k + \alpha_k G_{\theta_k,z_k}\big]_\Theta, \tag{9b}$$

where $[\,\cdot\,]_\Theta$ denotes projection on to the set Θ and $\hat{\Phi}(z,\pi)$ is an unbiased approximation of $\Phi(z,\pi)$ which is generated as follows: generate a mini-batch of m samples $(X^j, A^j, Y^j)_{j=1}^m$ where $X^j \sim z$, $A^j \sim \pi(\cdot|X^j)$, and $Y^j \sim P(X^j, A^j, z)$ and set

$$\hat{\Phi}(z,\pi)(y) = \frac{1}{m}\sum_{j=1}^{m} \mathbb{1}\{Y^j = y\}.$$

The learning rates $\{\alpha_k, \beta_k\}_{k\geq 0}$ are chosen according to the standard conditions for two-timescale algorithms: $\sum \alpha_k = \infty$, $\sum \beta_k = \infty$, $\sum(\alpha_k^2 + \beta_k^2) < \infty$, $\lim_{k\to\infty} \alpha_k = 0$, $\lim_{k\to\infty} \beta_k = 0$ and $\lim_{k\to\infty} \alpha_k/\beta_k = 0$. Then, we have the following:

Proposition 1 *If the following conditions are satisfied:*

1. *$\Phi(z,\pi_\theta)$, $\partial J_{\pi_\theta,z}/\partial\theta$ are Lipschitz in θ, z.*
2. *$\hat{\Phi}(z,\pi)$ and $G_{\theta,z}$ are unbiased estimators of of $\Phi(z,\pi)$ and $\partial J_{\pi_\theta,z}/\partial\theta$. Moreover, the estimation error $G_{\theta,z} - \partial J_{\pi_\theta,z}/\partial\theta$ has bounded variance.*
3. *For all $\theta \in \Theta$, the ODE corresponding to (9a), i.e.,*

$$\dot{z} = \Phi(z,\pi_\theta) - z$$

has a unique globally asymptotically stable equilibrium point, which we denote by $f(\theta)$.
4. *$f(\theta)$ is Lipschitz in θ.*

Then, almost surely:

1. $\|z_n - f(\theta_n)\| \to 0$ *as* $n \to \infty$.
2. *Suitable continuous time interpolation of* $\{\theta_n\}$ *is an asymptotic pseudotrajectory of the semiflow induced by the ODE corresponding to* (9b) *for* θ, *i.e,*

$$\dot{\theta} = \partial J_{\pi_\theta, z}/\partial\theta.$$

3. *The iteration* (9) *converges to a LSMFE.*

Proof. Note that, because the image space of Φ is bounded, the estimation error $\hat{\Phi}(z,\pi) - \Phi(z,\pi)$ is uniformly bounded. Thus, the conditions stated in the proposition along with the learning rate conditions specified for iteration (9) satisfy all the conditions stated in [29, page 35], [8, Theorem 23]. The result then follows from the application of the theorem given in [29, page 35], [8, Theorem 23]. Consequently, iteration (9) almost surely converges to a limit (θ^*, z^*) such that [27]:

$$\partial J_{\pi_{\theta^*}, z^*}/\partial\theta = 0 \quad \textit{and} \quad z^* = f(\theta^*),$$

which implies (π_{θ^*}, z^*) is a LSMFE. □

In theory, two-timescale algorithms are nice because they are amenable to a proof of convergence. However, in practice, two-time scale algorithms converge slowly because there are no good methods to adapt the learning rates. So, rather than running a two-timescale algorithm, it is often better to run a large but fixed number of iterations of variable running at the faster timescale for every iteration of variable running at the slower timescale. For iteration (9) this is equivalent to running multiple iterations of (9a) (with a fixed learning rate β) for every iteration of (9b). In the sequel, we run B iterations of (9b) with $\beta_k = 1$, which is shown in Algorithm 1 and is equivalent to a particle based Monte Carlo computation of the generated mean-field of the system.

Algorithm 1: `Stationary_MF`

input : θ : Policy parameter, ξ_0 : Initial state distribution
B : Iteration count, m : Batch size
output : z : Final mean-field
for $j = 1 : m$ **do**
  for $i \in N$ **do**
    Sample $X_0^{i,j} \sim \xi_0$
  $z_0^j = \xi_0$
  for $t = 0 : B$ **do**
    for $i \in N$ **do**
      Sample $A_t^{i,j} \sim \pi(X_t^{i,j})$
    Sample $X_{t+1}^{i,j} \sim P(X_t^{i,j}, A_t^{i,j}, z_t^j)$
    for $x \in \mathcal{X}$ **do**
      $z_{t+1}^j(x) = \frac{1}{n}\sum_{i\in N} \mathbb{1}\{X_{t+1}^{i,j} = x\}$
$z = \frac{1}{m}\sum_{j=1}^{m} z_{B+1}^j$
return z

To convert iteration (9) to a complete algorithm, we need an algorithm that computes an unbiased estimator $G_{\theta,z}$ for $\partial J_{\pi_\theta,z}/\partial\theta$ for a given z. Since z is fixed, $\partial J_{\pi_\theta,z}/\partial\theta$ may be computed using any of the standard policy gradient based approaches for reinforcement learning: likelihood ratio based gradient estimators [40, 26] or simultaneous perturbation based gradient estimators [37, 32, 23, 5].

3.1.1 Likelihood ratio based gradient estimation

One approach to estimate the performance gradient is to use likelihood ratio based estimates [36, 12, 45]. Suppose the policy $\pi_\theta(X)$ is differentiable with respect to θ. For any time t, define the likelihood

function $\Lambda_\theta^t = \nabla_\theta \log[\pi_\theta(A_t \mid X_t)]$. Then from [45, 40, 3] we know that:

$$\frac{\partial V_{\theta,z}(x)}{\partial \theta} = \mathbb{E}_{A_t \sim \pi_\theta(X_t)}\Big[\sum_{t=0}^{\infty} \gamma^t \Lambda_\theta^t V_{\pi_\theta,z}(X_t) \;\Big|\; X_0 = x\Big].$$

Thus,

$$\frac{\partial J_{\theta,z}}{\partial \theta} = \mathbb{E}_{X\sim\xi_0}\left[\frac{\partial V_{\theta,z}(X)}{\partial \theta}\right].$$

An algorithm to compute LSMFE based on the likelihood ratio approach is given in Algorithm 2. The `PolicyGradient` function in Algorithm 2 can be obtained by an actor only method such as Monte Carlo [39] or Renewal Monte Carlo [38] or using an actor critic method such as SARSA [39]. Additionally, variance reduction techniques such as subtracting a baseline or using mini-batch averaging may also be used.

Algorithm 2: Likelihood ratio based algorithm to compute LSMFE

input : θ_0 : Initial policy, z_0 : Initial mean-field
ξ_0 : Initial state distribution
K : Iteration count
B : Iterations for mean-field update
m : Batch size for mean-field update
output : (θ^*, z^*) : Estimated LSMFE solution
for *iterations* $k = 0 : K$ **do**
$z_{k+1} = \texttt{Stationary\_MF}(\theta_k, \xi_0, B, m)$
$G_{\theta_k, z_{k+1}} = \texttt{PolicyGradient}(\theta_k, \xi_0, z_{k+1})$
$\theta_{k+1} = [\theta_k + \alpha_k G_{\theta_k, z_{k+1}}]_\Theta$
return θ_{K+1}

3.1.2 Simultaneous perturbation based gradient estimation

Another approach to estimate the performance gradient is to use simultaneous perturbation based methods [37, 32, 23, 5]. This approach is useful when the policy π_θ is not differentiable with respect to its parameters θ. Now, given any distribution ξ_0, we can estimate $J_{\pi_\theta,z}$ using $V_{\pi_\theta,z}$ as:

$$J_{\pi_\theta,z} = \mathbb{E}_{X\sim\xi_0}[V_{\pi_\theta,z}(X)].$$

To generate the two-sided form of simultaneous perturbation based estimate, we generate two random parameters $\theta^+ = \theta + c\,\eta$ and $\theta^- = \theta - c\,\eta$, where η is a random variable with the same dimension as θ and c is a small constant. Let $\pi^+ = \pi_{\theta^+}$ and $\pi^- = \pi_{\theta^-}$. Then, the two-sided simultaneous perturbation estimate is given by

$$G_{\theta,z} = \frac{\eta}{2c}(J_{\pi^+,z} - J_{\pi^-,z}).$$

When $\eta_i \sim$ Rademacher(± 1), the above method is called simultaneous perturbation stochastic approximation (SPSA) [37, 32]; when $\eta_i \sim$ Normal$(0, I)$ it is called smoothed functional stochastic approximation (SFSA) [23, 5].

An algorithm to compute LSMFE using the simultaneous perturbation approach is given in Algorithm 3. As in the case of the likelihood ratio based approach, the `PolicyEvaluation` function in Algorithm 3 may be obtained by an actor only method such as Monte Carlo [39] or Renewal Monte Carlo [38] or using an actor critic method such as SARSA [39].

Algorithm 3: Simultaneous perturbation based algorithm to compute LSMFE

input : θ_0 : Initial policy, z_0 : Initial mean-field
ξ_0 : Initial state distribution
K : Iteration count, c : Perturbation size
B : Iterations for mean-field update
m : Batch size for mean-field update
output : (θ^*, z^*) : Estimated LSMFE solution
for *iterations* $k = 1 : K$ **do**
$z_{k+1} = \texttt{Stationary\_MF}(\theta_k, \xi_0, B, m)$
Let $\eta \sim \texttt{Rademacher}(\pm 1)$ or $\eta \sim \mathcal{N}(0,1)$
$\theta_k^+ = \theta_k + \eta\beta$ and $\theta_k^- = \theta_k - \eta\beta$.
$\widehat{J}_k^+ = \texttt{PolicyEvaluation}(\theta_k^+, \xi_0, z_{k+1})$
$\widehat{J}_k^- = \texttt{PolicyEvaluation}(\theta_k^-, \xi_0, z_{k+1})$
$G_{\theta_k, z_{k+1}} = \frac{\eta}{2c}(\widehat{J}_k^+ - \widehat{J}_k^-)$
$\theta_{k+1} = [\theta_k + \alpha_k G_{\theta_k, z_{k+1}}]_\Theta$
return θ_{K+1}

3.2 RL algorithm for learning LSMF-SO

The key idea behind the RL algorithm to learn an LSMF-SO is as follows. Suppose T_θ is an unbiased estimator for $dJ_{\pi_\theta, z_\theta}/d\theta$, where z_θ is the fixed point of $z = \Phi(z, \pi_\theta)$. Then, we start with an initial guess $\theta_0 \in \Theta$, and at each step of the iteration, update the guess using stochastic gradient ascent:

$$\theta_{k+1} = \left[\theta_k + \alpha_k T_{\theta_k}\right]_\Theta, \tag{10}$$

where $\{\alpha_k\}_{k\geq 0}$ is a sequence of learning rates that satisfies the standard conditions: $\sum \alpha_k = \infty$ and $\sum \alpha_k^2 < \infty$. Then, we have the following:

Proposition 2 *If the following conditions are satisfied:*

1. $dJ_{\pi_\theta, z_\theta}/d\theta$ *is continuous in* θ.
2. T_θ *is an unbiased estimator of* $dJ_{\pi_\theta, z_\theta}/d\theta$ *and the error* $T_\theta - dJ_{\pi_\theta, z_\theta}/d\theta$ *has bounded variance.*
3. *The ODE for* θ, *i.e,*
$$\dot{\theta} = dJ_{\pi_\theta, z_\theta}/d\theta,$$
has isolated limit points that are locally asymptotically stable.

Then, almost surely:

1. *Suitable continuous time interpolation of* $\{\theta_n\}$ *is an asymptotic pseudotrajectory of the semiflow induced by the ODE for* θ.
2. *The iteration converges to a LSMF-SO.*

Proof. The conditions stated above and the learning rate conditions satisfy the standard stochastic approximation convergence conditions as given in [7, 27]. Hence, the iteration (10) converges almost surely to a limit θ^* such that:

$$dJ_{\pi_\theta, z_\theta}/d\theta = 0,$$

which implies (π_{θ^*}, z^*) is a LSMF-SO, where $z^* = \hat{\Phi}(z^*, \pi_{\theta^*})$. □

To convert iteration (10) to a complete algorithm, we need an algorithm that computes an unbiased estimator $T_{\theta,z}$ of $dJ_{\pi_\theta, z_\theta}/d\theta$. Likelihood ratio based gradient estimators do not work in this case because, in order to compute $d\mathbb{E}[r(X_t^i, A_t^i, z_\theta)]/d\theta$, we need to compute $dz_\theta/d\theta$ and there are no good methods to do so. There are some results in the literature on the sensitivity of the stationary distribution of a Markov chain to its transition probability (e.g., [11] and references therein), but these results only provide loose bounds on $dz_\theta/d\theta$. However, it is possible to adapt simultaneous perturbation based methods to generate estimators of $dJ_{\pi_\theta, z_\theta}/d\theta$. We present one such estimator in the next section.

3.3 Simultaneous perturbation based gradient estimation

We first consider estimating z_θ for a given π_θ. Under (A2), when each agent follows policy π_θ, the mean-field converges to the stationary distribution z_θ. Then, we can estimate z_θ by simply running the system for a sufficiently long time. An algorithm based on this idea is shown in Algorithm 1.

Then, to generate the two-sided simultaneous perturbation based estimate of $dJ_{\pi_\theta,z_\theta}/d\theta$, we generate two random parameters $\theta^+ = \theta + c\,\eta$ and $\theta^- = \theta - c\,\eta$, where η and c are as in Section 3.1.2. Let $\pi^+ = \pi_{\theta^+}$ and $\pi^- = \pi_{\theta^-}$. Generate $z^+ = z_{\pi^+}$ and $z^- = z_{\pi^-}$ using Algorithm 1. Then, the two-sided simultaneous perturbation estimate is given by

$$T_\theta = \frac{\eta}{2c}\big(J_{\pi^+,z^+} - J_{\pi^-,z^-}\big).$$

An algorithm to compute LSMF-SO using simultaneous perturbation approach is given in Algorithm 4. As was the case for Algorithm 3, the `PolicyEvaluation` function in Algorithm 4 may be obtained by an actor only method such as Monte Carlo [39] or Renewal Monte Carlo [38] or using an actor critic method such as SARSA [39].

Algorithm 4: Simultaneous perturbation based algorithm to compute LSMF-SO

input : θ_0 : Initial policy
ξ_0 : Initial state distribution
K : Iteration count, c : Perturbation size
B : Iterations for mean-field update
m : Batch size for mean-field update
output : θ^* : Estimated LSMF-SO solution
for *iterations* $k = 1 : K$ **do**
Let $\eta \sim$ `Rademacher`(± 1) or $\eta \sim \mathcal{N}(0,1)$
$\theta_k^+ = \theta_k + \eta\beta$ and $\theta_k^- = \theta_k - \eta\beta$.
$z_k^+ =$ `Stationary_MF`(θ^+, ξ_0, B, m)
$z_k^- =$ `Stationary_MF`(θ^-, ξ_0, B, m)
$\widehat{J}_k^+ =$ `PolicyEvaluation`$(\theta_k^+, \xi_0, z_k^+)$
$\widehat{J}_k^- =$ `PolicyEvaluation`$(\theta_k^-, \xi_0, z_k^-)$
$T_{\theta_k} = \frac{\eta}{2c}(\widehat{J}_k^+ - \widehat{J}_k^-)$
$\theta_{k+1} = [\theta_k + \alpha_k T_{\theta_k}]_\Theta$
return θ_{K+1}

4 Numerical experiment

4.1 Example 1: Malware spread

4.1.1 Environment

We consider the malware spread model presented in [15, 16, 17, 20]. This model is representative of several problems with positive externalities. Examples of such models include flue vaccination, economic models involving entry and exit of firms, collusion among firms, mergers, advertising, investment, network effects, durable goods, consumer learning etc. Hence, we consider the malware spread problem as a representative problem where an analytical solution is available. In this model, let $X \in [0,1]$ denote the state (level of infection) of agent i, where where $X_t^i = 0$ is the most healthy state and $X_t^i = 1$ is the least healthy state. The action space $\mathcal{A} = \{0,1\}$, where $A_t^i = 0$ implies DO NOTHING and $A_t^i = 1$ implies REPAIR. The dynamics are given by

$$X_{t+1}^i = \begin{cases} X_t^i + (1 - X_t^i)\omega_t, & \text{for } A_t^i = 0, \\ 0, & \text{for } A_t^i = 1, \end{cases}$$

where $\{\omega_t\}_{t\geq 1}$ is a $[0,1]$-valued i.i.d. process with probability density f. The above dynamics imply that if the agent takes the DO NOTHING action, then its state deteriorates to a worse condition in the interval $[1-X_t^i, 1]$; if the agent takes the REPAIR action, then its state resets to the most healthy state.

The rewards are coupled through the mean $\langle Z_t \rangle$ of the mean field Z_t (i.e., $\langle Z_t \rangle >= \int_0^1 x Z_t(x) dx$). Each agent incurs a cost $(k + \langle Z_t \rangle) X_t^i$, which captures the risk of getting infected, and an additional cost of λ for taking the REPAIR action, i.e.,

$$r(X_t^i, A_t^i, Z_t) = -(k + \langle Z_t \rangle) X_t^i - \lambda A_t^i.$$

4.1.2 Model and policy parameters

We consider $n = 1000$ agents, $f = \text{Uniform}[0,1]$, $k = 0.2$, $\lambda = 0.5$ and $\gamma = 0.9$. The continuous state space $\mathcal{X} = [0,1]$ is discretized into 101 uniformly sized cells $\{0, 0.01, \ldots, 1\}$. We consider two different policy parametrizations:

1. **Threshold based policy**: We consider threshold-based policies parametrized by $\theta \in [0,1]$ such that:[1]
$$\pi_\theta(x) = \begin{cases} 0, & \text{if } x < \theta, \\ 1, & \text{if } x \geq \theta. \end{cases} \tag{11}$$
We use this policy parametrization to estimate both LSMFE and LSMF-SO. The parameterized policies of the form (11) are not differentiable with respect to θ, so we estimate the gradient using simultaneous perturbation methods (Algorithms 3 and 4) with $c = 0.1$, $\eta \sim \text{Rademacher}(\pm 1)$, initial value of the threshold chosen uniformly at random, i.e., $\theta_0 \sim \text{Uniform}[0,1]$. In both algorithms, policy evaluation is done using Monte Carlo with $m = 1000$ trajectories of length $H = 200$.
2. **Neural network (NN) based policy**: We consider a neural network policy with two hidden layers with 5 neurons and tanh activation. We estimate the gradient using the likelihood ratio method. We use REINFORCE [45] to compute the performance gradient and backpropagate this gradient over the NN to compute $G_{\theta,z}$. Policy gradient estimation is done using Monte Carlo (actor only) with $m = 10$ and $H = 200$. Since we have a likelihood ratio based gradient estimation approach only for the RL algorithm for LSMFE (Section 3.1.1), we use this policy parametrization only to estimate LSMFE (Algorithm 2).

For both the policy parametrizations, $z_0 = \xi_0 = \text{Uniform}(\mathcal{X})$, $B = 200$ and $K = 200$. We choose the learning rate using ADAM [24].[2] We repeat the experiment 100 times for both the policy parametrizations.

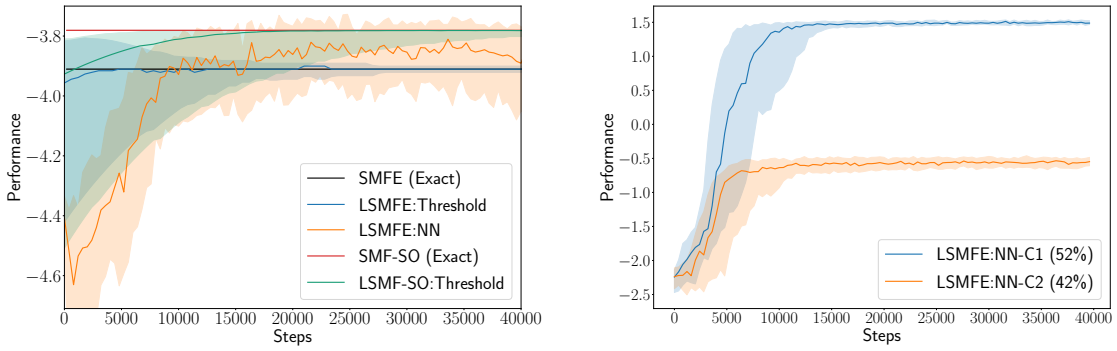
4.1.3 Results

The performance J for LSMFE (using both parameterizations) and LSMF-SO (using threshold based policies) are shown in Figure 1a. For the threshold based policy, J and $\langle z^* \rangle$ were evaluated using exact policy evaluation. For the neural network policy, they were estimated using 10 Monte Carlo evaluation runs. Hyperparameter tuning for selecting the NN parameters is given in the Appendix in Section 6.1.

For comparison, the exact SMF-SO and SMFE solutions are also plotted. The SMF-SO solution is computed by a brute force search over all $\theta \in [0,1]$. The SMFE solution is computed using the method described in [15]. These exact solutions are also shown in Figure 1a. The plots show that the convergence of the SPSA based RL algorithm is fairly fast and the variation across multiple runs is small. It is worth highlighting that LSMFE and LSMF-SO are different.

[1] It is shown in [15, 17] that such a parametrization is without loss of optimality.

[2] The α parameter of ADAM is set equal to 0.01 for the threshold based policy and 0.1 for the NN policy. All other ADAM parameters are equal to their default values.



(a) Performance versus steps for malware spread example **(b) Performance versus steps for investments example**

Figure 1: RL algorithm converging to LSMFE or LSMF-SO for the malware spread and product quality investments examples. The solid line shows the median value and the shaded region shows the region between the first and third quartiles over 100 runs

4.2 Example 2: Investments in product quality

4.2.1 Environment

We consider the investment decisions of firms in a fragmented market with a large number of firms. This model is adapted from [43]. In this model, each firm produces n_p products. The state of each firm X_t^i is represented by a n_p vector with each element $X_t^{i,j} \in [0,1]$, $j \in \{1,\dots,n_p\}$ denoting the normalized product quality for product j for firm i. At each time step, each firm $i \in N$ has to choose whether or not to invest in improving the quality of each of its products $j \in \{1,\dots,n_p\}$. Investment decisions are binary for each product. Thus the action space for firm i is $\mathcal{A}^i = \{0,1\}^{n_p}$, with $|\mathcal{A}^i| = 2^{n_p}$. When agent i decides to invest in product j, the quality of product j manufactured by i increases uniformly at random from its current value to the maximum value of 1, if the average mean-field for that product is below a particular threshold q. If this average mean-field value is above q, then the agent gets only half of the product quality improvement as compared to the former case. This implies that when the average quality of product j in the economy is below q, it is easier for each agent to improve its quality for product j. When the agent does not invest any amount in product j, its product quality for product j remains unchanged. This is given as:

$$X_{t+1}^{i,j} = \begin{cases} \omega_t(1 - X_t^{i,j}), & \text{if } \langle Z^j \rangle < q \text{ and } A_t^{i,j} = 1, \\ 0.5\omega_t(1 - X_t^{i,j}), & \text{if } \langle Z^j \rangle \geq q \text{ and } A_t^{i,j} = 1, \\ X_t^{i,j}, & \text{if } A_t^{i,j} = 0, \end{cases} \tag{12}$$

where ω_t is a $[0,1]$-valued i.i.d. process with probability density f and $\langle Z_t^j \rangle$ is the mean of Z_t^j (i.e., equal to $\int_0^1 x Z_t^j(x)dx$, $j \in \{1,\dots,n_p\}$).

At each step, each agent i incurs a cost due to its investment and earns a positive reward due to its own product quality for each product $j \in \{1,\dots,n_p\}$ and a negative reward due to the average product quality for product i, i.e., $\langle Z_t^j \rangle$. This per-step reward accumulated over all products is given as:

$$r(X_t^i, A_t^i, Z_t^i) = \sum_{j=1}^{n_p} \left[d^j X_t^{i,j} - c^j \langle Z_t^j \rangle - \lambda A_t^{i,j} \right] \tag{13}$$

4.2.2 Model and policy parameters

We consider $n = 100$ agents, $f = \text{Uniform}[0,1]$, $n_p = 3$, $q = 0.4$, $c = [0.21, 0.22, 0.23]$, $d = [0.31, 0.32, 0.33]$, $\lambda = [0.2, 0.21, 0.22]$, $B = 200$, $K = 200$, $X_0^{i,j} \sim \text{Uniform}[0,1]$ and $\gamma = 0.9$. The

policy is parametrizd using a two layer neural network with 8 and 16 neurons respectively with a tanh activation function for all hidden units. We use ADAM with learning rate of 0.1. Hyperparameter tuning for selecting these parameters is given in the Appendix in Section 6.2.

4.2.3 Results

In this example, we only demonstrate the computation of LSMFE using a neural network policy. We performed 100 independent runs for this example. We then clustered the tails (last 10 iterations) of these 100 trajectories and found that there are multiple LSMFE for this example. In Figure 1b, we plot the median, and the region between the first and third quantiles for the trajectories corresponding to the two most populated clusters. These two clusters, named C1 and C2 in Figure 1b, comprise of 52% and 42% of the total number of trajectories respectively.

5 Discussion

5.1 Finite vs. infinite populations

In both MFG and stationary MFG, the finite population system is approximated by an infinite population system. The infinite population system has two features: (i) each agent has an infinitesimal impact on the evolution of the mean-field which can be ignored; and (ii) the empirical mean-field can be approximated by the statistical mean-field, which evolves in a deterministic manner for a given policy. Thus, the strategic interactions between agents in a general n-player game is replaced by two consistency requirements: the policy is a best-response to the mean-field and the mean-field is consistent with the policy. As a result, the n-agent learning problem is reduced to optimality and consistency between a single generic (or canonical) agent and the mean-field.

However, since we are approximating the finite population system by an infinite population system, the approximation is meaningful only if the corresponding approximation error is small. There are several results in the mean-field games literature that show that under various (generally mild) technical conditions, the infinite population result is a $O(1/\sqrt{n})$ or a $O(1/n)$ approximation of the corresponding finite population result [41, 18]. These conditions are often model specific, so we don't list them here. What is important to note from the point of view of learning is that under these conditions, the learning algorithms proposed in this paper converge to $O(1/\sqrt{n})$ or $O(1/n)$ of the corresponding finite population solution.

5.2 Difference between MFG and stationary MFG models

MFG and stationary MFG are closely related but there is a fundamental difference between them. In MFG, assumptions (A1)–(A3) are not imposed. Thus the policy $\tilde{\boldsymbol{\pi}} = (\tilde{\pi}_1, \tilde{\pi}_2, \dots)$, $\tilde{\pi}_t : \mathcal{X} \to \Delta(\mathcal{A})$, is, in general, a time-varying policy (we denote the space of all such policies as $\tilde{\boldsymbol{\Pi}}$) and it is not assumed that the mean-field trajectory $\boldsymbol{z} = (z_1, z_2, \dots)$ converges to a limit. Thus, given a mean-field trajectory $\boldsymbol{z}$, the performance of a policy $\tilde{\boldsymbol{\pi}} \in \tilde{\boldsymbol{\Pi}}$ is given by:

$$\tilde{V}_{\tilde{\boldsymbol{\pi}},\boldsymbol{z}}(x) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(X_t^i, A_t^i, z_t, X_{t+1}^i) \;\Big|\; X_0^i = x\Big].$$

Note that even though the mean-field trajectory is fixed, the environment and rewards perceived by a generic agent are time-varying. Therefore, one cannot write a fixed-point Bellman equation for $\tilde{V}_{\tilde{\boldsymbol{\pi}},\boldsymbol{z}}$. Nonetheless, a time-varying Bellman equation can be written and, it is for this reason that, most of the literature on MFG apart from the special case of linear dynamics and quadratic cost considers finite horizon systems.

The commonly used solution concept for MFG is the following:

Definition 5 (NE-MFG) *A Nash equilibrium for MFG is a pair of time-varying policy* $\tilde{\boldsymbol{\pi}} = (\tilde{\pi}_1, \tilde{\pi}_2, \dots) \in \tilde{\boldsymbol{\Pi}}$ *and a trajectory of mean-fields* $\boldsymbol{z} = (z_1, z_2, \dots)$ *which satisfies the following two conditions:*

1. *Sequential rationality: For any other policy* $\tilde{\boldsymbol{\pi}}' = (\tilde{\pi}'_1, \tilde{\pi}'_2, \dots)$*, we have:*

$$V_{\tilde{\boldsymbol{\pi}}, \boldsymbol{z}}(x) \geq V_{\tilde{\boldsymbol{\pi}}', \boldsymbol{z}}(x), \quad \forall x \in \mathcal{X}.$$

2. *Consistency: The mean-field* z *evoles as:*

$$z_{t+1} = \Phi(z_t, \tilde{\pi}_t), \quad \forall t.$$

It is worth highlighting that NE-MFG is a pair of a trajectory of time-varying policy and time-varying mean-field. In contrast, SMFE is a pair of single policy and a single mean-field. Thus, SMFE is considerably easier to compute and implement as compared to NE-MFG. This simplicity comes at the cost of generality. The conditions for existence of SMFE are generally stricter than those for NE-MFG.

5.3 Related work

In view of the above discussion, we revisit the related work on mean-field approximation in MARL.

In [25], a model based adaptive control algorithm for computing NE-MFG of linear quadratic systems is considered. It is assumed that the dynamics takes one of finitely possible alternatives. Agents use maximum likelihood estimation to estimate the most likely dynamics and use certainty equivalent control laws corresponding to the estimated model. The results of [25] are difficult to generalize beyond the linear quadratic model.

In [48], a Q-learning algorithm for computing NE-MFG for a family of coupled oscillators is considered. The mean-field approximation is used to develop an approximate dynamic program (ADP) for the best-response equation and the ADP is solved using Q-learning. The approximation used for the ADP is specific for the model considered in [48] and does not apply to general models.

In [47], a Q-learning algorithm for computing NE-MFG for a stochastic game is presented. It is assumed that all agents observe the global state ($(x^1, x^2, \dots, x^n)$ in our model) and choose policies that map global state to local actions. The mean-field approximation is used to simplify the Q-function of the best-response and the simplified Q function is solved using Q-learning or DPG. When each agent has a local state (as is the case in the models presented in this paper), the global state is n-dimensional and it is impractical to assume that all agents know the global state. For example, in the malware example presented earlier, it will mean that all agents know the state of health of all agents in the system. Even if the global state were known, searching over policies $\pi : (x^1, x^2, \dots, x^n) \mapsto \Delta(A^i)$ will suffer from the curse of dimensionality.

In [33], a fictitious play based learning algorithm for computing NE-MFG of finite horizon common interest MFG is presented.[3] In this algorithm, one starts with a guess for the mean-field trajectory and the policy and improves them using actor critic functions. The proof of convergence relies on a technical property for NE-MFG for finite horizon MFGs proved in [9] and it is not immediately clear how that technical property can be extended to infinite horizon stationary MFG.

It is worth highlighting that all the previous work on mean-field based learning algorithms for MARL compute NE-MFG. As far as we are aware, this is the first paper to propose mean-field based learning algorithms to compute SMFE and SMF-SO for stationary MFGs.

[3] In [33], it is claimed that all MFGs are common interest games, but that is, in general, not the case.

5.4 Remarks on the generality of the model

For simplicity of exposition, we presented our results for the simplest model of stationary MFG. The results presented for this model continue to hold for the following generalizations.

- The coupling in the dynamics and reward is through the mean-field of states and actions rather than mean-field of just the states. In this case, the argument presented in the paper continues to work with minor changes because mean-field of states and actions is a function of the policy and the mean-field of states. In particular, if $\bar{z}$ denotes the mean-field of states and actions, then, in the infinite population limit

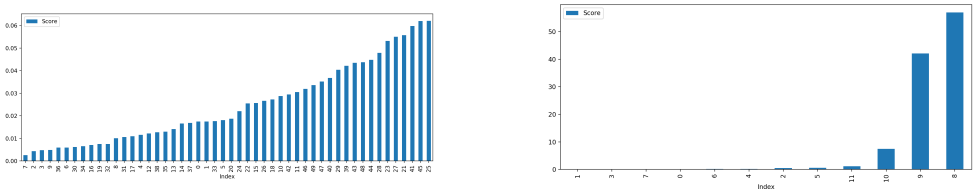
$$\bar{z}_t(x,a) = \bar{z}_t(x)\pi_t(a|x), \quad \forall x \in \mathcal{X}, a \in \mathcal{A}. \tag{14}$$

- The states and/or actions are continuous rather than discrete. In this case, the arguments hold under the standard conditions on measurability of dynamics, upper semi-continuity of the rewards, compactness of the action space, and the growth conditions on the rewards to ensure that value functions are well defined. An appropriate parametrization of the policy using a sufficiently rich family of function approximators such as radial basis functions or neural networks is also needed.
- There is a heterogeneous population consisting of multiple sub-populations of homogeneous agents. Such a model can be converted to a homogeneous population model by considering the type of the agent as a component of the state.

6 Appendix: Hyperparameter tuning

6.1 Example 1: Malware spread

For the NN based policy given in Section 4.1.2, we consider the following three hyperparameters—learning rate, architecture of hidden layers and activation function. We performed a grid search over a selected set of these three parameters to choose the best set of hyperparameters. For this grid search, we considered 5 learning rates—$0.1, 0.05, 0.01, 0.005, 0.001$, 5 hidden layer architectures—$\{4\}, \{4,4\}, \{5\}, \{5,5\}, \{10,10\}$ where $\{4\}$ represents a single hidden layer with 4 neurons and $\{5,5\}$ represents two hidden layers with 5 neurons each, and finally for the activation functions we considered ReLU and tanh. For each combination of parameters in the grid search, we collected results from 10 independent runs. For choosing the best hyperparameters we computed the mean of the NN loss ($G_{\pi_\theta,z}$) and mean of the absolute changes in the stationary mean field values averaged over the last 50 iterations for these 10 runs. The sum of these averaged quantities, called scores, are shown in Figure 2a. The hyperparameter combination corresponding to the X-axis indices are given in Table 1.



(a) Example 1: Malware spread

(b) Example 2: Investments in product quality

Figure 2: Score for each hyperparameter combination over 10 runs each

Table 1: Hyperparameter values and corresponding indices used in Figure 2a for Example 1

Index	Learning rate	Hidden layers	Activation function
0	0.1	{4}	ReLU
1	0.1	{4}	tanh
2	0.1	{4,4}	ReLU
3	0.1	{4,4}	tanh
4	0.1	{5}	ReLU
5	0.1	{5}	tanh
6	0.1	{5,5}	ReLU
7	0.1	{5,5}	tanh
8	0.1	{10,10}	ReLU
9	0.1	{10,10}	tanh
10	0.05	{4}	ReLU
11	0.05	{4}	tanh
12	0.05	{4,4}	ReLU
13	0.05	{4,4}	tanh
14	0.05	{5}	ReLU
15	0.05	{5}	tanh
16	0.05	{5,5}	ReLU
17	0.05	{5,5}	tanh
18	0.05	{10,10}	ReLU
19	0.05	{10,10}	tanh
20	0.005	{4}	ReLU
21	0.005	{4}	tanh
22	0.005	{4,4}	ReLU
23	0.005	{4,4}	tanh
24	0.005	{5}	ReLU
25	0.005	{5}	tanh
26	0.005	{5,5}	ReLU
27	0.005	{5,5}	tanh
28	0.005	{10,10}	ReLU
29	0.005	{10,10}	tanh
30	0.001	{4}	ReLU
31	0.001	{4}	tanh
32	0.001	{4,4}	ReLU
33	0.001	{4,4}	tanh
34	0.001	{5}	ReLU
35	0.001	{5}	tanh
36	0.001	{5,5}	ReLU
37	0.001	{5,5}	tanh
38	0.001	{10,10}	ReLU
39	0.001	{10,10}	tanh
40	0.01	{4}	ReLU
41	0.01	{4}	tanh
42	0.01	{4,4}	ReLU
43	0.01	{4,4}	tanh
44	0.01	{5}	ReLU
45	0.01	{5}	tanh
46	0.01	{5,5}	ReLU
47	0.01	{5,5}	tanh
48	0.01	{10,10}	ReLU
49	0.01	{10,10}	tanh

6.2 Example 2: Investments in product quality

For the investments in product quality example, we do hyperparameter tuning by performing a grid search over the same three parameters as the previous example—learning rate, architecture of hidden layers and activation function. Here, we considered 3 learning rates—0.1, 0.05 and 0.025, 2 hidden layer architectures—{8,16} and {16,32} and 2 activation functions—ReLU and tanh. Similar to the case of Example 1 in 6.1, for each combination of parameters in the grid search, we collected results from 10 independent runs. Here, for choosing the best hyperparameters we computed the mean of the NN loss ($G_{\pi_\theta,z}$) and mean of the absolute changes in the stationary mean field values averaged over

the last 25 iterations for these 10 runs. The corresponding scores as in 6.1, are shown in Figure 2b. The hyperparameter combination corresponding to the X-axis indices are given in Table 2.

Table 2: Hyperparameter values and corresponding indices used in Figure 2b for Example 2

Index	Learning rate	Hidden layers	Activation function
0	0.1	{8,16}	ReLU
1	0.1	{8,16}	tanh
2	0.1	{16,32}	ReLU
3	0.1	{16,32}	tanh
4	0.05	{8,16}	ReLU
5	0.05	{8,16}	tanh
6	0.05	{16,32}	ReLU
7	0.05	{16,32}	tanh
8	0.025	{8,16}	ReLU
9	0.025	{8,16}	tanh
10	0.025	{16,32}	ReLU
11	0.025	{16,32}	tanh

References

[1] Sachin Adlakha, Ramesh Johari, and Gabriel Y Weintraub. Equilibria of dynamic games with many players: Existence, approximation, and market structure. Journal of Economic Theory, 156:269–316, 2015.

[2] Jalal Arabneydi and Aditya Mahajan. Team optimal control of coupled subsystems with mean-field sharing. In IEEE Conference on Decision and Control, pages 1669–1674. IEEE, 2014.

[3] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. Journal of Artificial Intelligence Research, 15:319–350, 2001.

[4] James Bergin and Dan Bernhardt. Anonymous sequential games: existence and characterization of equilibria. Economic Theory, 5(3):461–489, 1995.

[5] Shalabh Bhatnagar, HL Prasad, and LA Prashanth. Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods, volume 434. Springer, 2013.

[6] Daan Bloembergen, Karl Tuyls, Daniel Hennes, and Michael Kaisers. Evolutionary dynamics of multi-agent learning: A survey. Journal of Artificial Intelligence Research, 53:659–697, 2015.

[7] Vivek S Borkar. Stochastic Approximation: A Dynamical Systems Viewpoint. Cambridge University Press, 2008.

[8] Vivek S Borkar. Stochastic approximation with two time scales. Systems & Control Letters, 29(5):291–294, 1997.

[9] Pierre Cardaliaguet and Saeed Hadikhanloo. Learning in mean field games: the fictitious play. ESAIM: Control, Optimisation and Calculus of Variations, 23(2):569–591, 2017.

[10] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, pages 2974–2982, 2018.

[11] R.E. Funderlic and C.D. Meyer. Sensitivity of the stationary distribution vector for an ergodic markov chain. Linear Algebra and its Applications, 76:1–17, 1986.

[12] Peter Glynn. Likelihood ratio gradient estimation for stochastic systems. Communications of the ACM, 33:75–84, 1990.

[13] P. Hernandez-Leal, B. Kartal, and M. E. Taylor. Is multiagent deep reinforcement learning the answer or the question? A brief survey. ArXiv e-prints, October 2018.

[14] Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. Journal of machine learning research, 4(Nov):1039–1069, 2003.

[15] M. Huang and Y. Ma. Mean field stochastic games: Monotone costs and threshold policies. In 2016 IEEE 55th Conference on Decision and Control (CDC), pages 7105–7110, Dec 2016.

[16] M. Huang and Y. Ma. Mean field stochastic games with binary action spaces and monotone costs. ArXiv e-prints, January 2017.

[17] M. Huang and Y. Ma. Mean field stochastic games with binary actions: Stationary threshold policies. In 2017 IEEE 56th Annual Conference on Decision and Control (CDC), pages 27–32, Dec 2017.

[18] Minyi Huang, Roland P Malhamé, and Peter E Caines. Large population stochastic dynamic games: closed-loop Mckean-Vlasov systems and the Nash certainty equivalence principle. Communications in Information & Systems, 6(3):221–252, 2006.

[19] M. Hüttenrauch, A. Šošić, and G. Neumann. Deep Reinforcement Learning for Swarm Systems. ArXiv e-prints, July 2018.

[20] Libin Jiang, Venkat Anantharam, and Jean Walrand. How bad are selfish investments in network security? IEEE/ACM Transactions on Networking (TON), 19(2):549–560, 2011.

[21] Boyan Jovanovic and Robert W. Rosenthal. Anonymous sequential games. Journal of Mathematical Economics, 17(1):77–87, 1988.

[22] Ian A Kash, Eric J Friedman, and Joseph Y Halpern. Multiagent learning in large anonymous games. Journal of Artificial Intelligence Research, 40:571–598, 2011.

[23] V. Katkovnik and Y. Kulchitsky. Convergence of a class of random search algorithms. Automation and Remote Control, 33(8):1321–1326, 1972.

[24] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[25] Arman C Kizilkale and Peter E Caines. Mean field stochastic adaptive control. IEEE Transactions on Automatic Control, 58(4):905–920, 2013.

[26] Vijay R Konda and John N Tsitsiklis. On actor-critic algorithms. SIAM Journal on Control and Optimization, 42(4):1143–1166, 2003.

[27] Harold Kushner and G George Yin. Stochastic approximation and recursive algorithms and applications, volume 35. Springer Science & Business Media, 2003.

[28] Jean-Michel Lasry and Pierre-Louis Lions. Mean field games. Japanese Journal of Mathematics, 2(1):229–260, 2007.

[29] D. S. Leslie. Reinforcement learning in games. PhD thesis, The University of Bristol, 2004.

[30] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In International Conference on Machine Learning., 1994.

[31] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. Neural Information Processing Systems (NIPS), 2017.

[32] John L Maryak and Daniel C Chin. Global random optimization by simultaneous perturbation stochastic approximation. 53(3):780–783, April 2008.

[33] David Mguni, Joel Jennings, and Enrique Munoz de Cote. Decentralised learning in systems with many, many strategic agents. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, pages 4686–4693, 2018.

[34] Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. arXiv preprint arXiv:1703.10069, 2017.

[35] Lillian J Ratliff, Samuel A Burden, and S Shankar Sastry. On the characterization of local nash equilibria in continuous games. IEEE Transactions on Automatic Control, 61(8):2301–2307, 2016.

[36] Reuven Y Rubinstein. Sensitivity analysis and performance extrapolation for computer simulation models. Operations Research, 37(1):72–81, 1989.

[37] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Transactions on Automatic Control, 37(3):332–341, 1992.

[38] Jayakumar Subramanian and Aditya Mahajan. Renewal monte carlo: Renewal theory based reinforcement learning. In 2018 IEEE Conference on Decision and Control (CDC), pages 5759–5764. IEEE, 2018.

[39] Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. MIT Press, 1998.

[40] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In Advances in Neural Information Processing Systems, pages 1057–1063, Nov. 2000.

[41] Gabriel Y Weintraub, C. Lanier Benkard, and Benjamin Van Roy. Oblivious Equilibrium: A Mean Field Approximation for Large-Scale Dynamic Games. In Advances in Neural Information Processing Systems, pages 1489–1496, December 2005.

[42] Gabriel Y Weintraub, C Lanier Benkard, and Benjamin Van Roy. Markov perfect industry dynamics with many firms. Econometrica, 76(6):1375–1411, 2008.

[43] Gabriel Y Weintraub, C Lanier Benkard, and Benjamin Van Roy. Computational methods for oblivious equilibrium. Operations research, 58(4–part–2):1247–1265, 2010.

[44] Pierre Weiss. L'hypothèse du champ moléculaire et la propriété ferromagnétique. J. Phys. Theor. Appl., 6(1):661–690, 1907.

[45] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3–4):229–256, 1992.

[46] Jiachen Yang, Xiaojing Ye, Rakshit Trivedi, Huan Xu, and Hongyuan Zha. Deep mean field games for learning optimal behavior policy of large populations. In International Conference on Learning Representations, 2018.

[47] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-agent reinforcement learning. In International Conference on Machine Learning, 2018.

[48] H. Yin, P. G. Mehta, S. P. Meyn, and U. V. Shanbhag. Learning in mean-field games. IEEE Transactions on Automatic Control, 59(3):629–644, March 2014.