Article

Networked Communication in Mean-Field Games with Function Approximation and Empirical Mean-Field Estimation

Patrick Benjamin [1], Alessandro Abate [2]

[1] University of Oxford; patrick.benjamin@cs.ox.ac.uk
[2] University of Oxford; alessandro.abate@cs.ox.ac.uk

Abstract: Recent algorithms allow decentralised agents, possibly connected via a communication network, to learn equilibria in mean-field games from a single game with an empirical population and no episodic resets. However, these algorithms are for tabular settings: this computationally limits the size of agents' observation space, meaning the algorithms cannot handle anything but small state spaces, nor generalise beyond policies depending only on the agent's local state to so-called 'population-dependent' policies. We address this limitation by introducing function approximation to the existing setting, drawing on the Munchausen Online Mirror Descent method that has previously been employed only in finite-horizon, episodic, centralised settings. While this permits us to include the mean field in the observation for players' policies, it is restrictive to assume decentralised agents have access to this global information: we thus also provide new algorithms allowing agents to locally estimate the global empirical distribution, and to improve this estimate via inter-agent communication. We show theoretically that exchanging policy information helps networked agents outperform both independent and even centralised agents in function-approximation settings. Our experiments demonstrate this empirically, by an even greater margin than in tabular settings, and show that the communication network allows decentralised agents to estimate the mean field for population-dependent policies.

Keywords: mean-field games; deep reinforcement learning; networked communication

---

1. Introduction

The mean-field game (MFG) framework [1,2] can be used to address the difficulty faced by multi-agent reinforcement learning (MARL) regarding computational scalability as the number of agents grows [3,4]. It models a representative agent as interacting not with other individual agents in the population on a per-agent basis, but instead with a distribution over the other agents, called the *mean field*. The MFG framework analyses the limiting case when the population consists of an infinite number of symmetric and anonymous agents, that is, they have identical reward and transition functions which depend on the mean-field distribution rather than on the actions of specific other players. The solution to this game is the mean-field Nash equilibrium (MFNE), which can be used as an approximation for the Nash equilibrium (NE) in a finite-agent game (which is harder to solve in itself), with the error in the solution reducing as the number of agents N tends to infinity [5–10]. MFGs have thus been applied to a wide variety of real-world and game scenarios: see [11] for examples.

Recent works argue that classical algorithms for solving MFGs rely on assumptions and methods that are likely to be restrictive in realistic multi-agent games, such as large-scale strategy or simulation-based games, emphasising that desirable qualities for such MFG algorithms include: learning from the empirical distribution of N agents (i.e. this distribution is generated only by the policies of the in-game agents, rather than being updated by the algorithm itself or an external oracle); learning on the fly from a single, continued, non-episodic game (i.e. where the population/environment cannot be arbitrarily reset, as in persistent-world or real-time games); model-free learning; decentralisation; and fast practical convergence [12,13]. While these works address these desiderata, they do so only in settings in which the state and action spaces are small enough that the Q-function can be represented by a table, limiting scalability and applicability to complex games.

Moreover, in those works, as in many others on MFGs, agents only observe their local state as input to their Q-function (which defines their policy). This is sufficient when the solved MFG is expected to have a stationary distribution ('stationary MFGs') [6,11–15]. However, in reality there are numerous reasons for which agents may benefit from being able to respond to the current distribution. Recent work has thus increasingly focused on these more general settings where it is necessary for agents to have so-called 'master policies' (a.k.a. population-dependent policies) which depend on both the mean-field distribution and their local state [11,11,16–20].

The distribution is a large, high-dimensional observation object, taking a continuum of values. Therefore a population-dependent Q-function cannot be represented exactly in a table and must be approximated. To address these limitations while maintaining the desiderata for realistic multi-agent games given in recent works, we introduce function approximation to the MFG setting of decentralised agents learning on the fly from a single, continued run of the empirical game without episodic resets, allowing this setting to handle larger state spaces and to accept the mean-field distribution as an observation input.To overcome the difficulties of training non-linear approximators in this context, we use the so-called 'Munchausen' trick, introduced by [21] for single-agent RL, and extended to MFGs by [20], and to MFGs with population-dependent policies by [19].

We particularly explore this in the context of networked communication between decentralised agents [13]. We demonstrate that communication brings two specific benefits over the purely independent setting (while also removing the undesirable assumption of a centralised learner, which in realistic game settings may be restrictive, a computational bottleneck and a vulnerable single point of failure). Firstly, when the Q-function is approximated rather than exact, some updates lead to better performing policies than others. Allowing networked agents to propagate better performing policies through the population leads to faster learning than in the purely independent case and often even than in the centralised case, as we show both theoretically and empirically (this method is reminiscent of the use of fitness functions in distributed evolutionary algorithms [22,23]). Secondly, we argue that in realistic settings it is restrictive to assume that decentralised agents, endowed with local state observations and limited (if any) communication radius, would be able to observe the global mean-field distribution and use it as input to their Q-functions / policy. We therefore further contribute two setting-dependent algorithms by which decentralised agents can estimate the global distribution from local observations, and further improve their estimates by communication with neighbours.

We focus on 'coordination' games, where agents can increase their individual rewards by following the same strategy as others and therefore have an incentive to communicate policies, even if the MFG setting itself is technically non-cooperative.[1] In summary, our contributions are:

- We introduce function approximation to MFG settings with decentralised agents for the first time.
  - To do this, we use Munchausen RL for the first time in an infinite-horizon MFG context (cf. [19,20]). This also constitutes the first use of function approximation for solving MFGs from a single, non-episodic game with the empirical population (for tabular settings see [12,13]).
- Function approximation allows us to explore larger state spaces, and also settings where agents' policies depend on the mean-field distribution as well as their local state.
- Rather than assuming that agents have access to this global knowledge as in prior works, we present two additional novel algorithms allowing decentralised agents to locally estimate the empirical distribution and to improve these estimates by inter-agent communication.
- We show theoretically that networked agents may learn faster than both centralised and independent agents in the function-approximation setting.

[1] We further preempt concerns about the appropriateness of communication in competitive settings by wondering whether self-interested agents would be any more likely to want to obey a central learner as has usually been assumed. Moreover we show that self-interested communicating agents can obtain higher returns than independent agents even in non-coordination games (Fig. A3), indicating that they do have incentive to communicate.

- Our extensive experiments support the two benefits of the decentralised communication scheme, which significantly outperforms both the independent and centralised settings.

Paper structure: Preliminaries are in Sec. 2 and our core learning and policy improvement algorithm is in Sec. 3. We present our mean-field estimation and communication algorithms in Sec. 4, theoretical results in Sec. 5 and experiments in Sec. 6. Appx. G contains an extended 'Related work' section.

2. Preliminaries

2.1. Mean-field games

We use the following notation. N is the number of agents in a population, with $\mathcal{S}$ and $\mathcal{A}$ representing the finite state and common action spaces, respectively. The set of probability measures on a finite set $\mathcal{X}$ is denoted $\Delta_{\mathcal{X}}$, and $\mathbf{e}_x \in \Delta_{\mathcal{X}}$ for $x \in \mathcal{X}$ is a one-hot vector with only the entry corresponding to x set to 1, and all others set to 0. For time $t \geq 0$, $\hat{\mu}_t = \frac{1}{N}\sum_{i=1}^{N}\sum_{s\in\mathcal{S}} \mathbb{1}_{s_t^i=s}\mathbf{e}_s \in \Delta_{\mathcal{S}}$ is a vector of length $|\mathcal{S}|$ denoting the empirical categorical state distribution of the N agents at time t. For agent $i \in 1\ldots N$, i's policy at time t depends on its observation o_t^i. We explore three different forms that this observation object can take:

- In the conventional setting, the observation is simply i's local state s_t^i, such that $\pi^i(a|o_t^i) = \pi^i(a|s_t^i)$.
- When the policy is population-dependent, if we assume perfect observability of the global mean-field distribution then we have $o_t^i = (s_t^i, \hat{\mu}_t)$.
- It is restrictive to assume that decentralised agents with a possibly limited communication radius can observe the global mean field, so we allow agents to form a local estimate $\tilde{\hat{\mu}}_t^i$ which can be improved by communication with neighbours. Here we have $o_t^i = (s_t^i, \tilde{\hat{\mu}}_t^i)$.

In the following definitions we focus on the population-dependent case when $o_t^i = (s_t^i, \hat{\mu}_t)$, and clarify afterwards the connection to the other observation cases. Thus the set of policies is $\Pi = \{\pi : \mathcal{S} \times \Delta_{\mathcal{S}} \to \Delta_{\mathcal{A}}\}$, and the set of Q-functions is denoted $\mathcal{Q} = \{q : \mathcal{S} \times \Delta_{\mathcal{S}} \times \mathcal{A} \to \mathbb{R}\}$.

Definition 1 (N-player symmetric anonymous games)**.** *An N-player stochastic game with symmetric, anonymous agents is given by the tuple $\langle N, \mathcal{S}, \mathcal{A}, P, R, \gamma\rangle$, where $\mathcal{A}$ is the action space, identical for each agent; $\mathcal{S}$ is the identical state space of each agent, such that their initial states are $\{s_0^i\}_{i=1}^N \in \mathcal{S}^N$ and their policies are $\{\pi^i\}_{i=1}^N \in \Pi^N$. $P : \mathcal{S} \times \mathcal{A} \times \Delta_{\mathcal{S}} \to \Delta_{\mathcal{S}}$ is the transition function and $R : \mathcal{S} \times \mathcal{A} \times \Delta_{\mathcal{S}} \to [0,1]$ is the reward function, which map each agent's local state and action and the population's empirical distribution to transition probabilities and bounded rewards, respectively, i.e. $\forall i$: $s_{t+1}^i \sim P(\cdot|s_t^i, a_t^i, \hat{\mu}_t)$ and $r_t^i = R(s_t^i, a_t^i, \hat{\mu}_t)$.*

At the limit as $N \to \infty$, the infinite population of agents can be characterised as a limit distribution $\mu \in \Delta_{\mathcal{S}}$; the infinite-agent game is termed an MFG. The so-called 'mean-field flow' $\boldsymbol{\mu}$ is given by the infinite sequence of mean-field distributions s.t. $\boldsymbol{\mu} = (\mu_t)_{t\geq 0}$.

Definition 2 (Induced mean-field flow)**.** *We denote by $I(\pi)$ the mean-field flow $\boldsymbol{\mu}$ induced when all the agents follow π, where this is generated from π as follows: $\mu_{t+1}(s') = \sum_{s,a}\mu_t(s)\pi(a|s,\mu_t)P(s'|s,a,\mu_t)$.*

When the mean-field flow is stationary such that the distribution is the same for all t, i.e. $\mu_t = \mu_{t+1}$ $\forall t \geq 0$, the policy $\pi^i(a|s_t^i, \mu_t)$ need not depend on the distribution, such that $\pi^i(a|s_t^i, \mu_t) = \pi^i(a|s_t^i)$, i.e. we recover the classical population-independent policy. However, for such a population-independent policy the initial distribution μ_0 must be known and fixed in advance, whereas otherwise it need not be. We also give the following definitions.

Definition 3 (Mean-field discounted return)**.** *In a MFG where all agents follow policy π giving a mean-field flow $\boldsymbol{\mu} = (\mu_t)_{t\geq 0}$, the expected discounted return of the representative agent is given by*

$$V(\pi, \boldsymbol{\mu}) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t(R(s_t, a_t, \mu_t))\Bigg|_{\substack{s_0\sim\mu_0 \\ a_t\sim\pi(\cdot|s_t,\mu_t) \\ s_{t+1}\sim P(\cdot|s_t,a_t,\mu_t)}}\right].$$

Definition 4 (Best-response (BR) policy)**.** *A policy π^* is a best response (BR) against the mean-field flow $\boldsymbol{\mu}$ if it maximises the discounted return $V(\cdot, \boldsymbol{\mu})$; the set of these policies is denoted $BR(\boldsymbol{\mu})$, i.e.*

$$\pi^* \in BR(\boldsymbol{\mu}) := \arg\max_{\pi} V(\pi, \boldsymbol{\mu}).$$

Definition 5 (Mean-field Nash equilibrium (MFNE))**.** *A pair $(\pi^*, \boldsymbol{\mu}^*)$ is a mean-field Nash equilibrium (MFNE) if the following two conditions hold:*

- *π^* is a best response to $\boldsymbol{\mu}^*$, i.e. $\pi^* \in BR(\boldsymbol{\mu}^*)$;*
- *$\boldsymbol{\mu}^*$ is induced by π^*, i.e. $\boldsymbol{\mu}^* = I(\pi^*)$.*

π^* is thus a fixed point of the map $BR \circ I$, i.e. $\pi^* \in BR(I(\pi^*))$. If a population-dependent policy is a MFNE policy for any initial distribution μ_0, it is a 'master policy'.

Previous works have shown that, in tabular settings, it is possible for a finite population of decentralised agents (each of which is permitted to have a distinct population-independent policy π^i) to learn the MFNE using only the empirical distribution $\hat{\mu}_t$, rather than the exactly calculated infinite flow $\boldsymbol{\mu}$ [12,13]. This MFNE may be the goal in itself, or it can in turn serve as an approximate NE for the harder-to-solve game involving the finite population. In this work we provide algorithms to perform this process in non-tabular and population-dependent settings, and demonstrate them empirically.

2.2. (Munchausen) Online Mirror Descent

Instead of performing the computationally expensive process of finding a BR at each iteration, we can use a form of policy iteration for MFGs called Online Mirror Descent (OMD). This involves beginning with an initial policy π_0, and then at each iteration k, evaluating the current policy π_k with respect to its induced mean-field flow $\boldsymbol{\mu} = I(\pi_k)$ to compute its Q-function Q_{k+1}. To stabilise the learning process, we then use a weighted sum over this and past Q-functions, and set π_{k+1} to be the softmax over this weighted sum, i.e. $\pi_{k+1}(\cdot|s,\mu) = softmax\left(\frac{1}{\tau_q}\sum_{\kappa=0}^{k} Q_\kappa(s,\mu,\cdot)\right)$. τ_q is a temperature parameter that scales the entropy in Munchausen RL [21]; note that this is a different temperature to the one agents use when selecting which communicated parameters to adopt, denoted τ_k^{comm} (Sec. 3.2).

If the Q-function is approximated non-linearly, it is hard to compute this weighted sum. The 'Munchausen trick' addresses this by computing a single Q-function that mimics the weighted sum using implicit regularisation based on the Kullback-Leibler (KL) divergence between π_k and π_{k+1} [21]. This reparametrisation gives Munchausen OMD (MOMD), detailed further in Sec. 3.1 [19,20]. MOMD does not bias the MFNE and has the same convergence guarantees as OMD [19,24,25].

2.3. Networks

Definition 6 (Time-varying network)**.** *The network $\{\mathcal{G}_t\}_{t\geq 0}$ is given by $\mathcal{G}_t = (\mathcal{N}, \mathcal{E}_t)$, where $\mathcal{N}$ is the set of vertices each representing an agent i, and the edge set $\mathcal{E}_t \subseteq \{(i,j) : i,j \in \mathcal{N}\}$ is the set of undirected links present at time t. A network's diameter $d_{\mathcal{G}_t}$ is the maximum of the shortest path length between any pair of nodes.*

We conceive of the finite population as exhibiting two such networks. One of them, $\mathcal{G}_t^{comm}$, defines which agents can communicate information to each other at t. The second network $\mathcal{G}_t^{obs}$ is a graph defining which agents can observe each other's states, which we use in general settings for estimating the mean-field distribution from local information. The structure of the two networks may be identical (e.g. if agents that are located in physical space can both observe the position (state) of, and exchange information with, other agents within a certain physical distance from themselves), or different (e.g. if agents can observe the positions of nearby agents, but only exchange information with agents by which they are linked via radio/satellite, which may connect agents over long distances).

We also define an alternative version of the observation graph that is useful in a specific subclass of environments, which can most intuitively be thought of as those where agents' states are positions in physical space. When this is the case, we usually think of agents' ability to observe each other as depending more abstractly on whether states are visible to each other. We define this visibility graph:

Definition 7 (Time-varying state-visibility graph)**.** *The state-visibility graph* $\{\mathcal{G}_t^{vis}\}_{t\geq 0}$ *is given by* $\mathcal{G}_t^{vis} = (\mathcal{S}', \mathcal{E}_t^{vis})$*, where* $\mathcal{S}'$ *is the set of vertices representing the environment states* $\mathcal{S}$*, and the edge set* $\mathcal{E}_t^{vis} \subseteq \{(m,n) : m,n \in \mathcal{S}'\}$ *is the set of undirected links present at time* t*, indicating which states are visible to each other.*

We view an agent in s as able to obtain a count of the number of agents in s' if s' is visible to s. The benefit of this graph $\mathcal{G}_t^{vis}$ over $\mathcal{G}_t^{obs}$ is that there is mutual exclusivity: either an agent in state s is able to obtain a total count of all of the agents in state s' (if s' is visible to s), or it cannot obtain information about any agent in state s' (if those states are not visible to each other). Additionally, this graph permits an agent in state s to observe that there are *no* agents in state s' as long as s' is visible to s. These benefits are not available if the observability graph is defined strictly between agents as in $\mathcal{G}_t^{obs}$, such that using $\mathcal{G}_t^{vis}$ facilitates more efficient estimation of the global mean-field distribution from local information in settings where $\mathcal{G}_t^{vis}$ is applicable (see Sec. 4).

3. Learning and policy improvement

3.1. Q-network and update

Lines 1-14 of our novel Alg. A1 (Appx. A) contain the core Q-function/policy update method. Agent i has a neural network parametrised by θ_k^i to approximate its Q-function: $\check{Q}_{\theta_k^i}(o,\cdot)$. The agent's policy is given by $\pi_{\theta_k^i}(a|o) = \text{softmax}\Big(\frac{1}{\tau_q}\check{Q}_{\theta_k^i}(o,\cdot)\Big)(a)$. We denote the policy $\pi_k^i(a|o)$ for simplicity when appropriate. Each agent maintains a buffer (of size M) of collected transitions of the form $\big(o_t^i, a_t^i, r_t^i, o_{t+1}^i\big)$. At each iteration k, they empty their buffer (Line 3) before collecting M new transitions (Lines 4-7); each decentralised agent i then trains its Q-network $\check{Q}_{\theta_k^i}$ via L training updates as follows (Lines 8-12). For training purposes, i also maintains a target network $\check{Q}_{\theta_{k,l}^{i,\prime}}$ with the same architecture but parameters $\theta_{k,l}^{i,\prime}$ copied from $\theta_{k,l}^i$ less regularly than $\theta_{k,l}^i$ themselves are updated, i.e. only every ν learning iterations (Line 11). At each iteration l, agent i samples a random batch $B_{k,l}^i$ of $|B|$ transitions from its buffer (Line 9), and trains its neural network via stochastic gradient descent to minimise the empirical loss (Def. 8, Line 10). For $cl < 0$, $[\cdot]_{cl}^0$ is a clipping function used in Munchausen RL to prevent numerical issues if the policy is too close to deterministic [19,21]:

Definition 8 (Q-network empirical loss)**.** *This is given by* $\hat{\mathcal{L}}(\theta,\theta') = \frac{1}{|B|}\sum_{transition \in B_{k,l}^i}\Big|\check{Q}_{\theta_{k,l}^i}(o_t,a_t) - T\Big|^2$*, where the target is* $T = r_t + [\tau_q \ln \pi_{\theta_{k,l}^{i,\prime}}(a_t|o_t)]_{cl}^0 + \gamma\sum_{a\in\mathcal{A}}\pi_{\theta_{k,l}^{i,\prime}}(a|o_{t+1})\Big(\check{Q}_{\theta_{k,l}^{i,\prime}}(o_{t+1},a) - \tau_q \ln \pi_{\theta_{k,l}^{i,\prime}}(a|o_{t+1})\Big)$.

3.2. Communication and adoption of parameters

We use the communication network $\mathcal{G}_t^{comm}$ to share two types of information at different points in Alg A1. One is used to improve local estimates of the mean-field distribution (see Sec. 4). The other, described here, is used to privilege the spread of better performing policy updates through the population, allowing faster convergence in this networked case than in the independent and even centralised cases. We adapt the work in [13] for the function-approximation case, where in our work agents broadcast the parameters of the Q-network that defines their policy, rather than the Q-function table. At each iteration k, after independently updating their Q-network and policy (Lines 3-14), agents *approximate* the infinite discounted return (Def. 3) of their new policies by collecting rewards for E steps, and assign the finite-step discounted sum to σ_{k+1}^i (Lines 15-20). They then broadcast their Q-network parameters along with σ_{k+1}^i (Line 22). Receiving these from neighbours on the network, agents select which set of parameters to adopt by taking a softmax over their own and the received estimate values σ_{k+1}^j (Lines 23-25). They repeat the process for C_p rounds. This allows decentralised agents to adopt policy parameters estimated to perform better than their own, accelerating learning as shown in Sec. 5.

4. Mean-field estimation and communication

We describe here the most general version of our algorithm for decentralised estimation of the empirical categorical mean-field distribution, assuming the more general setting where $\mathcal{G}_t^{obs}$ applies (see discussion in Sec. 2.3). In Appx. C, we detail how the algorithm can be made more efficient in environments where the more abstract visibility graph $\mathcal{G}_t^{vis}$ applies, as in our experimental settings. In both cases, the algorithm runs to generate the observation object when a step is taken in the main Alg. A1, i.e. to produce $o_t^i = (s_t^i, \tilde{\hat{\mu}}_t^i)$ for the steps $a_t^i \sim \pi_k^i(\cdot|o_t^i)$ in Lines 5, 17 and 26. Both versions of the algorithm are subject to implicit assumptions, which we discuss methods for addressing in Appx. H.

In the general setting, our method (Alg. A2, Appx. B) assumes each agent is associated with a unique ID to avoid the same agents being counted multiple times. Each agent maintains a 'count' vector $\hat{v}_t^i$ of length $|\mathcal{S}|$ i.e. of the same shape as the vector denoting the true empirical categorical distribution of agents. Each state position in the vector can hold a list of IDs. Before any actions are taken at each time step t, each agent's count vector $\hat{v}_t^i$ is initialised as full of $\emptyset$ ('no count') markers for each state (Line 1). Then, for each agent j with which agent i is connected via the observation graph, i places j's unique ID in its count vector in the correct state position (Line 2). Next, for $C_e \geq 0$ communication rounds, agents exchange their local counts with neighbours on the communication network (Line 4), and merge these counts with their own count vector, filtering out the unique IDs of those that have already been counted (Line 6). If $C_e = 0$ then the local count will remain purely independent. By exchanging these partially filled vectors, agents are able to improve their local counts by adding the states of agents that they have not been able to observe directly themselves.

After the C_e communication rounds, each state position $\hat{v}_t^i[s]$ either still maintains the $\emptyset$ marker if no agents have been counted in this state, or contains $x_s > 0$ unique IDs. The local mean-field estimate $\tilde{\hat{\mu}}_t^i$ is then obtained from $\hat{v}_t^i$ as follows. All states that have a count x_s have this count converted into the categorical probability x_s/N (we assume that agents know the total number of agents in the finite population, even if they cannot observe them all at each t) (Line 11). The total number of agents counted in $\hat{v}_t^i$ is given by $counted\_agents = \sum_{s \in \mathcal{S}} x_s$, and the agents that have not been observed are $uncounted\_agents = N - counted\_agents$. In this general setting, the unobserved agents are assumed to be uniformly distributed across all the states, so $uncounted\_agents/(N \times |\mathcal{S}|)$ is added to all the values in $\tilde{\hat{\mu}}_t^i$, replacing the $\emptyset$ marker for states for which no agents have been observed (Line 10).

5. Theoretical results

To demonstrate the benefits of the networked architecture by comparison, we also consider (theoretically here and experimentally in Sec. 6) the results of modified versions of our algorithm for centralised and independent learners. In the centralised setting, the Q-network updates of arbitrary agent $i = 1$ are automatically pushed to all other agents, and the true global mean-field distribution is always used in place of the local estimate i.e. $\tilde{\hat{\mu}}_t^i = \hat{\mu}_t$. In the independent case, there are no links in $\mathcal{G}_t^{comm}$ or $\mathcal{G}_t^{vis}$, i.e. $\mathcal{E}_t^{comm} = \mathcal{E}_t^{vis} = \emptyset$. Networked agents often learn faster than centralised ones in our experiments; we justify theoretically this possibly counterintuitive result here. We first make two relatively strong assumptions that give conditions under which networked agents *do* outperform centralised ones. The fact that these strong assumptions do not always hold in practice explains why networked agents may not always outperform centralised ones.

Recall that at each iteration k of Alg. A1, after independently updating their policies in Line 14, the population has the policies $\{\pi_{k+1}^i\}_{i=1}^N$. There is randomness in these independent policy updates, stemming from the random sampling of each agent's independently collected buffer. In Lines 15-20, agents approximate the infinite discounted returns $\{V(\pi_{k+1}^i, I(\pi_{k+1}^i))\}_{i=1}^N$ (Def. 3) of their updated policies by computing $\{\sigma_{k+1}^i\}_{i=1}^N$: the E-step discounted return with respect to the *empirical* mean field generated when agents follow policies $\{\pi_{k+1}^i\}_{i=1}^N$ (i.e. they do not at this stage all follow a single identical policy). We can characterise the approximation as $\{\sigma_{k+1}^i\}_{i=1}^N = \{\widehat{V}(\pi_{k+1}^i, I(\pi_{k+1}^i))\}_{i=1}^N$.

Assumption 1. *Assume that* $\{\sigma^i_{k+1}\}^N_{i=1}$ *are sufficiently good approximations so as to respect the ordering of the true values* $\{V(\pi^i_{k+1}, I(\pi^i_{k+1}))\}^N_{i=1}$, *i.e.* $\forall i, j \in \{1, \dots, N\}$: $\sigma^i_{k+1} > \sigma^j_{k+1} \iff V(\pi^i_{k+1}, I(\pi^i_{k+1})) > V(\pi^j_{k+1}, I(\pi^j_{k+1}))$.

Assumption 2. *Assume that after the* C_p *rounds in Lines 21-27 in which agents exchange and adopt policies from neighbours, the population is left with a single policy such that* $\forall i, j \in \{1, \dots, N\}$ $\pi^i_{k+1} = \pi^j_{k+1}$.[2]

Call the network consensus policy π^{net}_{k+1} and its finitely approximated return $\sigma^{\text{net}}_{k+1}$. Recall that the centralised case is where the Q-network update of arbitrary agent $i = 1$ is automatically pushed to all the others instead of the policy evaluation and exchange in Lines 15-27; this is equivalent to a networked case where policy consensus is reached on a *random* one of the policies $\{\pi^i_{k+1}\}^N_{i=1}$. Call this policy *arbitrarily* given to the whole population π^{cent}_{k+1}, and its finitely approximated return $\sigma^{\text{cent}}_{k+1}$.

Theorem 1. *Given Ass. 1 and 2,* $\mathbb{E}[V(\pi^{\text{net}}_{k+1}, I(\pi^{\text{net}}_{k+1}))] > \mathbb{E}[V(\pi^{\text{cent}}_{k+1}, I(\pi^{\text{cent}}_{k+1}))]$. *Thus in expectation networked agents will increase their returns faster than centralised ones. Full proof in Appx. E.*

Proof intuition. The adoption scheme in Line 24 biases the spread of policies towards those estimated to be better, which, given sufficiently good approximations (Ass. 1), results in higher discounted returns in practice. By choosing updates in a more principled way, networked agents learn faster than the centralised case that adopts updates regardless of quality. This intuition applies even if we loosen Ass. 2 that the networked population converges on a single consensus policy within the C_p communication rounds. Similar logic can also be applied to understand why networked agents outperform entirely independent ones, combined with the fact that divergence between policies in the independent case worsens sample complexity over the networked and centralised cases by biasing approximations of the Q-function [12,13]. Significantly, the communication scheme not only allows us to avoid the undesirable assumption of a central learner, but even to outperform it.

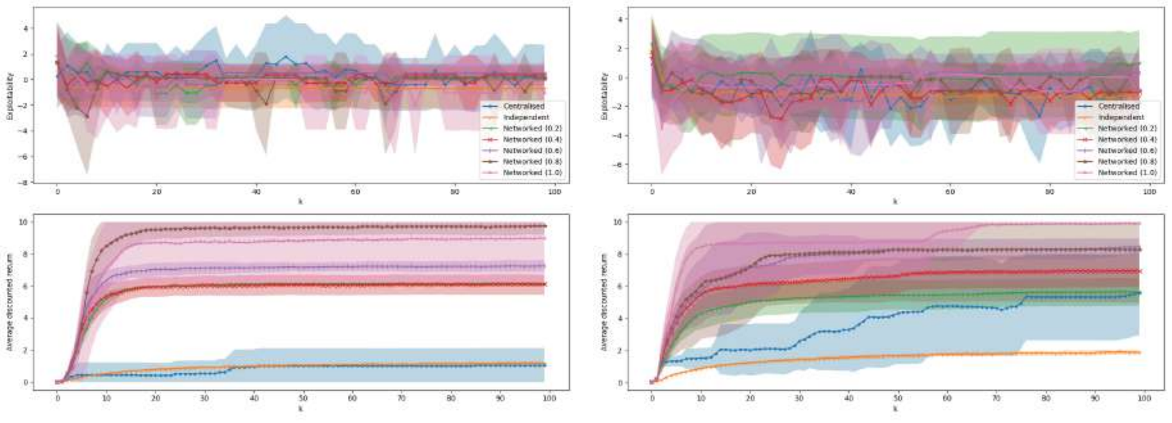
6. Experiments

We provide two sets of experiments. The first set showcases that our function-approximation algorithm (Alg. A1) can scale to large state spaces for population-independent policies, and that in such settings networked, communicating agents can outperform purely-independent and even centralised agents, and do so by an even greater margin than in the tabular settings from [13]. The second set (given in Appx. F.4) demonstrates that Alg. A1 can handle population-dependent policies, as well as the ability of Alg. A3 to practically estimate the mean-field distribution locally.

For the types of game used in our demonstrations we follow the gold standard in prior MFG works, i.e. grid-world environments where agents can move in the four cardinal directions or remain in place [13,15,19,20,27–29]. In these spatial environments, both the communication network $\mathcal{G}^{comm}_t$ and the visibility graph $\mathcal{G}^{vis}_t$ are determined by the physical distance from agent i; we show plots for various radii, expressed as fractions of the maximum possible distance (the grid's diagonal length).

We present results from five games defined by the agents' reward/transition functions, all but one of which are *coordination* games. The first two are those used with population-independent policies in [13], but while they show results for an 8x8 and a 'larger' 16x16 grid, our results are for 100x100 and 50x50 grids: **Cluster.** Agents are rewarded for gathering but given no indication where to do so, agreeing it over time. **Target agreement.** Agents are rewarded for visiting any of a given number of targets, but the reward is proportional to the number of other agents co-located at the target; agents

[2] Most simply we can think of Ass. 2 holding if 1) $\tau^{comm}_k \to 0$ $\forall k$ such that the softmax essentially becomes a max function, and 2) the communication network $\mathcal{G}^{comm}_t$ is static and connected during the C_p communication rounds, where C_p is larger than the network diameter $d_{\mathcal{G}^{comm}_t}$. Under these conditions, previous results on max-consensus algorithms show that all agents in the network will converge on the highest σ^{max}_{k+1} value (and hence the unique associated π^{max}_{k+1} within a number of rounds equal to the diameter $d_{\mathcal{G}^{comm}_t}$ [13,26]. However, policy consensus as in Ass. 2 might be achieved even outside of these conditions, including if the network is dynamic and not connected at every step, given appropriate values for C_p and $\tau^{comm}_{k+1} \in \mathbb{R}_{>0}$.



(**a**) 'Target agreement' game. (**b**) 'Cluster' game.

Figure 1. Population-independent policies, 100x100 grid.

must coordinate on which single target they will all meet at to maximise their individual rewards. See Appx. F.1 for a full technical description of these two games and of additional more complex ones.

We evaluate our experiments via two metrics. *Exploitability* is the most common metric in works on MFGs, and is a measure of proximity to the MFNE. It quantifies how much a best-responding agent can benefit by deviating from the set of policies that generate the current mean-field distribution, with a decreasing exploitability meaning the population is closer to the MFNE. However, there are several issues with this metric in our setting, particularly for our coordination games where competitive agents benefit from aligning behaviours, such that it may give limited or noisy information (discussed further in Appx. F.2.1). We thus also give a second metric, as in [13]: the population's *average discounted return*. This allows us to compare how quickly agents are learning to increase their returns, even when 'exploitability' gives us limited ability to distinguish between the desirability of the MFNEs to which populations converge. We discuss hyperparameters in Appx. F.3.

6.1. Results and discussion

Fig. 1 illustrates that introducing function approximation to algorithms in this setting allows them to converge within a practical number of iterations ($k \ll 100$), even for large state spaces (100x100 grids). By contrast, the tabular algorithms in [13] appear only just to converge by $k = 200$ for the same games for the larger of their two grids, which is only 16x16. In Fig. 1, networked agents all significantly outperform both centralised and independent agents in term of average return, despite the centralised agents appearing to have similar exploitability, and the independent agents having similar or slightly lower exploitability. This is because independent agents (and also the centralised ones in Fig. 1(**a**)) hardly improve their policies *at all*, so there is little a deviating agent can do to increase its return in these coordination games, meaning exploitability appears low, despite this being an undesirable equilibrium (see Appx. F.2 for further discussion on the limited information provided by the exploitation metric). The fact that the networked agents nevertheless significantly outperform the other architectures in terms of average return indicates that communication helps agents to find substantially 'preferable' equilibria. See Appx. F.4 for further experiments.

7. Conclusion

We novelly contributed function approximation to the setting of learning on the fly in empirical MFGs, and also contributed two novel algorithms for locally estimating the empirical mean field for population-dependent policies. We have justified theoretically why our networked communication algorithm is able to learn faster than both centralised and independent agents in this function approximation setting, and demonstrated empirically the ability of our algorithms to handle large state spaces and estimate the mean field. Limitations and ways to extend our algorithms are in Appx. H.

References

1. Lasry, J.M.; Lions, P.L. Mean Field Games. *Japanese Journal of Mathematics* **2007**, *2*, 229–260.
2. Huang, M.; Malhamé, R.P.; Caines, P.E. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems* **2006**, *6*, 221 – 252.
3. Yardim, B.; He, N. Exploiting Approximate Symmetry for Efficient Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2408.15173* **2024**.
4. Zeng, S.; Bhatt, S.; Koppel, A.; Ganesh, S. A Single-Loop Finite-Time Convergent Policy Optimization Algorithm for Mean Field Games (and Average-Reward Markov Decision Processes). *arXiv e-prints* **2024**, pp. arXiv–2408.
5. Saldi, N.; Başar, T.; Raginsky, M. Markov–Nash Equilibria in Mean-Field Games with Discounted Cost. *SIAM Journal on Control and Optimization* **2018**, *56*, 4256–4287, [https://doi.org/10.1137/17M1112583]. https://doi.org/10.1137/17M1112583.
6. Anahtarci, B.; Kariksiz, C.D.; Saldi, N. Q-learning in regularized mean-field games. *Dynamic Games and Applications* **2023**, *13*, 89–117.
7. Yardim, B.; Goldman, A.; He, N. When is Mean-Field Reinforcement Learning Tractable and Relevant? *arXiv preprint arXiv:2402.05757* **2024**.
8. Toumi, N.; Malhame, R.; Le Ny, J. A mean field game approach for a class of linear quadratic discrete choice problems with congestion avoidance. *Automatica* **2024**, *160*, 111420. https://doi.org/https://doi.org/10.1016/j.automatica.2023.111420.
9. Hu, A.; Zhang, J. MF-OML: Online Mean-Field Reinforcement Learning with Occupation Measures for Large Population Games. *arXiv preprint arXiv:2405.00282* **2024**.
10. Chen, Y.; Wu, L.; Xu, R.; Zhang, R. Periodic Trading Activities in Financial Markets: Mean-field Liquidation Game with Major-Minor Players. *arXiv preprint arXiv:2408.09505* **2024**.
11. Laurière, M.; Perrin, S.; Geist, M.; Pietquin, O. Learning Mean Field Games: A Survey. *arXiv preprint arXiv:2205.12944* **2022**.
12. Yardim, B.; Cayci, S.; Geist, M.; He, N. Policy Mirror Ascent for Efficient and Independent Learning in Mean Field Games. In Proceedings of the International Conference on Machine Learning. PMLR, 2023, pp. 39722–39754.
13. Benjamin, P.; Abate, A. Networked Communication for Decentralised Agents in Mean-Field Games. *arXiv preprint arXiv:2306.02766* **2023**.
14. Xie, Q.; Yang, Z.; Wang, Z.; Minca, A. Learning While Playing in Mean-Field Games: Convergence and Optimality. In Proceedings of the Proceedings of the 38th International Conference on Machine Learning; Meila, M.; Zhang, T., Eds. PMLR, 18–24 Jul 2021, Vol. 139, *Proceedings of Machine Learning Research*, pp. 11436–11447.
15. Zaman, M.A.U.; Koppel, A.; Bhatt, S.; Basar, T. Oracle-free Reinforcement Learning in Mean-Field Games along a Single Sample Path. In Proceedings of the International Conference on Artificial Intelligence and Statistics. PMLR, 2023, pp. 10178–10206.
16. Cardaliaguet, P.; Delarue, F.; Lasry, J.M.; Lions, P.L. The master equation and the convergence problem in mean field games, 2015, [arXiv:math.AP/1509.02505].
17. Carmona, R.; Delarue, F.; Lacker, D. Mean Field Games with Common Noise. *The Annals of Probability* **2016**, *44*, 3740–3803.
18. Perrin, S.; Laurière, M.; Pérolat, J.; Élie, R.; Geist, M.; Pietquin, O. Generalization in mean field games by learning master policies. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2022, Vol. 36, pp. 9413–9421.
19. Wu, Z.; Laurière, M.; Chua, S.J.C.; Geist, M.; Pietquin, O.; Mehta, A. Population-aware Online Mirror Descent for Mean-Field Games by Deep Reinforcement Learning. *arXiv preprint arXiv:2403.03552* **2024**.
20. Laurière, M.; Perrin, S.; Girgin, S.; Muller, P.; Jain, A.; Cabannes, T.; Piliouras, G.; Perolat, J.; Elie, R.; Pietquin, O.; et al. Scalable Deep Reinforcement Learning Algorithms for Mean Field Games. In Proceedings of the Proceedings of the 39th International Conference on Machine Learning; Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; Sabato, S., Eds. PMLR, 17–23 Jul 2022, Vol. 162, *Proceedings of Machine Learning Research*, pp. 12078–12095.
21. Vieillard, N.; Pietquin, O.; Geist, M. Munchausen Reinforcement Learning. In Proceedings of the Advances in Neural Information Processing Systems; Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; Lin, H., Eds. Curran Associates, Inc., 2020, Vol. 33, pp. 4235–4246.

22. Eiben, A.E.; Smith, J.E., What Is an Evolutionary Algorithm? In *Introduction to Evolutionary Computing*; Springer Berlin Heidelberg: Berlin, Heidelberg, 2015; pp. 25–48. https://doi.org/10.1007/978-3-662-44874-8_3.
23. Hart, E.; Steyven, A.; Paechter, B. Improving Survivability in Environment-Driven Distributed Evolutionary Algorithms through Explicit Relative Fitness and Fitness Proportionate Communication. In Proceedings of the Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, New York, NY, USA, 2015; GECCO '15, p. 169–176. https://doi.org/10.1145/2739480.2754688.
24. Hadikhanloo, S. Learning in anonymous nonatomic games with applications to first-order mean field games. *arXiv preprint arXiv:1704.00378* **2017**.
25. Perolat, J.; Perrin, S.; Elie, R.; Laurière, M.; Piliouras, G.; Geist, M.; Tuyls, K.; Pietquin, O. Scaling up Mean Field Games with Online Mirror Descent. *arXiv preprint arXiv:2103.00623* **2021**.
26. Nejad, B.M.; Attia, S.A.; Raisch, J. Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies. In Proceedings of the 2009 XXII International Symposium on Information, Communication and Automation Technologies, 2009, pp. 1–7. https://doi.org/10.1109/ICAT.2009.5348437.
27. Laurière, M. Numerical Methods for Mean Field Games and Mean Field Type Control. *Mean field games* **2021**, *78*.
28. Algumaei, T.; Solozabal, R.; Alami, R.; Hacid, H.; Debbah, M.; Takáč, M. Regularization of the policy updates for stabilizing Mean Field Games. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 2023, pp. 361–372.
29. Cui, K.; Fabian, C.; Koeppl, H. Multi-Agent Reinforcement Learning via Mean Field Control: Common Noise, Major Agents and Approximation Properties. *arXiv preprint arXiv:2303.10665* **2023**.
30. Cunha Queiroz, B.; MacRae, D. Occlusion-based object transportation around obstacles with a swarm of miniature robots. *Swarm Intelligence* **2024**, pp. 1–29.
31. Perrin, S.; Pérolat, J.; Laurière, M.; Geist, M.; Elie, R.; Pietquin, O. Fictitious Play for Mean Field Games: Continuous Time Analysis and Applications. In Proceedings of the Proceedings of the 34th International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 2020; NIPS'20.
32. Pérolat, J.; Perrin, S.; Elie, R.; Laurière, M.; Piliouras, G.; Geist, M.; Tuyls, K.; Pietquin, O. Scaling Mean Field Games by Online Mirror Descent. In Proceedings of the Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, Richland, SC, 2022; AAMAS '22, p. 1028–1037.
33. Perrin, S.; Laurière, M.; Pérolat, J.; Geist, M.; Élie, R.; Pietquin, O. Mean Field Games Flock! The Reinforcement Learning Way. In Proceedings of the IJCAI, 2021.
34. Yongacoglu, B.; Arslan, G.; Yüksel, S. Independent Learning in Mean-Field Games: Satisficing Paths and Convergence to Subjective Equilibria. *arXiv preprint arXiv:2209.05703* **2022**.
35. Cui, K.; Koeppl, H. Approximately Solving Mean Field Games via Entropy-Regularized Deep Reinforcement Learning. In Proceedings of the International Conference on Artificial Intelligence and Statistics. PMLR, 2021, pp. 1909–1917.
36. Guo, X.; Hu, A.; Xu, R.; Zhang, J. A General Framework for Learning Mean-Field Games. *Mathematics of Operations Research* **2023**, *48*, 656–686.
37. Subramanian, J.; Mahajan, A. Reinforcement Learning in Stationary Mean-Field Games. In Proceedings of the Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, Richland, SC, 2019; AAMAS '19, p. 251–259.
38. Yang, Y.; Luo, R.; Li, M.; Zhou, M.; Zhang, W.; Wang, J. Mean Field Multi-Agent Reinforcement Learning. In Proceedings of the Proceedings of the 35th International Conference on Machine Learning; Dy, J.; Krause, A., Eds. PMLR, 10–15 Jul 2018, Vol. 80, *Proceedings of Machine Learning Research*, pp. 5571–5580.
39. Ganapathi Subramanian, S.; Taylor, M.E.; Crowley, M.; Poupart, P. Partially Observable Mean Field Reinforcement Learning. In Proceedings of the Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems, 2021, pp. 537–545.
40. Ganapathi Subramanian, S.; Poupart, P.; Taylor, M.E.; Hegde, N. Multi Type Mean Field Reinforcement Learning. In Proceedings of the Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, 2020, pp. 411–419.
41. Subramanian, S.G.; Taylor, M.E.; Crowley, M.; Poupart, P. Decentralized Mean Field Games. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2022, Vol. 36, pp. 9439–9447.
42. Zhang, C.; Chen, X.; Di, X. Stochastic Semi-Gradient Descent for Learning Mean Field Games with Population-Aware Function Approximation. *arXiv preprint arXiv:2408.08192* **2024**.

Appendix A. Learning and communication algorithm

See Alg. A1.

Algorithm A1 Networked learning with non-linear function approximation

```
Require: loop parameters $K, M, L, E, C_p$, learning parameters $\gamma, \tau_q, |B|, cl, \nu, \{\tau_k^{comm}\}_{k\in\{0,\dots,K-1\}}$
Require: initial states $\{s_0^i\}_{i=1}^N$; $t \leftarrow 0$
 1: $\forall i$ : Randomly initialise parameters $\theta_0^i$ of Q-networks $\check{Q}_{\theta_0^i}(o,\cdot)$, and set $\pi_0^i(a|o) = \mathrm{softmax}\left(\frac{1}{\tau_q}\check{Q}_{\theta_0^i}(o,\cdot)\right)(a)$
 2: for $k = 0,\dots,K-1$ do
 3:   $\forall i$: Empty $i$'s buffer
 4:   for $m = 0,\dots,M-1$ do
 5:     Take step $\forall i : a_t^i \sim \pi_k^i(\cdot|o_t^i), r_t^i = R(s_t^i, a_t^i, \hat{\mu}_t), s_{t+1}^i \sim P(\cdot|s_t^i, a_t^i, \hat{\mu}_t)$; $t \leftarrow t+1$
 6:     $\forall i$: Add $\zeta_t^i$ to $i$'s buffer
 7:   end for
 8:   for $l = 0,\dots,L-1$ do
 9:     $\forall i$ : Sample batch $B_{k,l}^i$ from $i$'s buffer
10:     Update $\theta$ to minimise $\hat{\mathcal{L}}(\theta,\theta')$ as in Def. 8
11:     If $l \mod \nu = 0$, set $\theta' \leftarrow \theta$
12:   end for
13:   $\check{Q}_{\theta_{k+1}^i}(o,\cdot) \leftarrow \check{Q}_{\theta_{k,L}^i}(o,\cdot)$
14:   $\forall i : \pi_{k+1}^i(a|o) \leftarrow \mathrm{softmax}\left(\frac{1}{\tau_q}\check{Q}_{\theta_{k+1}^i}(o,\cdot)\right)(a)$
15:   $\forall i : \sigma_{k+1}^i \leftarrow 0$
16:   for $e = 0,\dots,E-1$ evaluation steps do
17:     Take step $\forall i : a_t^i \sim \pi_k^i(\cdot|o_t^i), r_t^i = R(s_t^i, a_t^i, \hat{\mu}_t), s_{t+1}^i \sim P(\cdot|s_t^i, a_t^i, \hat{\mu}_t)$
18:     $\forall i : \sigma_{k+1}^i = \sigma_{k+1}^i + \gamma^e \cdot r_t^i$
19:     $t \leftarrow t+1$
20:   end for
21:   for $C_p$ rounds do
22:     $\forall i$ : Broadcast $\sigma_{k+1}^i, \pi_{k+1}^i$
23:     $\forall i : J_t^i \leftarrow \{j \in \mathcal{N} : (i,j) \in \mathcal{E}_t^{comm}\}$
24:     $\forall i$ : Select adopted$^i \sim \Pr\left(\text{adopted}^i = j\right) = \frac{\exp(\sigma_{k+1}^j/\tau_k^{comm})}{\sum_{x\in J_t^i}\exp(\sigma_{k+1}^x/\tau_k^{comm})}\ \forall j \in J_t^i$
25:     $\forall i : \sigma_{k+1}^i \leftarrow \sigma_{k+1}^{\text{adopted}^i}, \pi_{k+1}^i \leftarrow \pi_{k+1}^{\text{adopted}^i}$
26:     Take step $\forall i : a_t^i \sim \pi_k^i(\cdot|o_t^i), r_t^i = R(s_t^i, a_t^i, \hat{\mu}_t), s_{t+1}^i \sim P(\cdot|s_t^i, a_t^i, \hat{\mu}_t)$; $t \leftarrow t+1$
27:   end for
28: end for
29: return policies $\{\pi_K^i\}_{i=1}^N$
```

Appendix B. Mean-field estimation algorithm for general environments

See Alg. A2.

Appendix C. Mean-field estimation algorithm for visibility-based environments

We give here the differences in our estimation algorithm (Alg. A3) for the subclass of environments where $\mathcal{G}_t^{vis}$ applies in place of $\mathcal{G}_t^{obs}$, i.e. the mutual observability of agents depends in turn on the mutual visibility of states. The benefit of $\mathcal{G}_t^{vis}$ over $\mathcal{G}_t^{obs}$ is that the former allows an agent in state s to obtain a correct, complete count $x_{s'} \geq 0$ of all the agents in state s', for any state s' that is visible to s (note the count may be zero). Unique IDs are thus not required as there is no risk of counting the same agent twice when receiving communicated counts: either *all* agents in s' have been counted, or no count has yet been obtained for s'. This simplifies the algorithm and helps preserve agent anonymity and privacy.

Algorithm A2 Mean-field estimation and communication in general settings

```
Require: Time-dependent observation graph $\mathcal{G}_t^{obs}$, time-dependent communication graph $\mathcal{G}_t^{comm}$, states $\{s_t^i\}_{i=1}^N$, number of communication rounds $C_e$
 1: $\forall i, s$ : Initialise count vector $\hat{v}_t^i[s]$ with $\emptyset$
 2: $\forall i : \hat{v}_t^i[s_t^j] \leftarrow \{ID^j\}_{j \in \mathcal{N}:(i,j) \in \mathcal{E}_t^{obs}}$
 3: for $c_e$ in $1, \dots, C_e$ do
 4:     $\forall i$ : Broadcast $\hat{v}_{t,c_e}^i$
 5:     $\forall i : J_t^i \leftarrow \{j \in \mathcal{N} : (i,j) \in \mathcal{E}_t^{comm}\}$
 6:     $\forall i, s : \hat{v}_{t,(c_e+1)}^i[s] \leftarrow \hat{v}_{t,c_e}^i[s] \cup \{\hat{v}_{t,c_e}^j[s]\}_{j \in J_t^i}$
 7: end for
 8: $\forall i : counted\_agents_t^i \leftarrow \sum_{s \in \mathcal{S}:\hat{v}_t^i[s] \neq \emptyset} |\hat{v}_t^i[s]|$
 9: $\forall i : uncounted\_agents_t^i \leftarrow N - counted\_agents_t^i$
10: $\forall i, s : \tilde{\mu}_t^i[s] \leftarrow \frac{uncounted\_agents_t^i}{N \times |\mathcal{S}|}$
11: $\forall i, s$ where $\hat{v}_t^i[s]$ is not $\emptyset$ : $\tilde{\mu}_t^i[s] \leftarrow \tilde{\mu}_t^i[s] + \frac{|\hat{v}_t^i[s]|}{N}$
12: return mean-field estimates $\{\tilde{\mu}_t^i\}_{i=1}^N$
```

Algorithm A3 Mean-field estimation and communication for environments with $\mathcal{G}_t^{vis}$

```
Require: Time-dependent visibility graph $\mathcal{G}_t^{vis}$, time-dependent communication graph $\mathcal{G}_t^{comm}$, states $\{s_t^i\}_{i=1}^N$, number of communication rounds $C_e$
 1: $\forall i, s$ : Initialise count vector $\hat{v}_t^i[s]$ with $\emptyset$
 2: $\forall i$ and $\forall s' \in \mathcal{S}' : (s_t^i, s') \in \mathcal{E}_t^{vis} : \hat{v}_t^i[s'] \leftarrow \sum_{j \in 1,\dots,N:s_t^j = s'} 1$
 3: for $c_e$ in $1, \dots, C_e$ do
 4:     $\forall i$ : Broadcast $\hat{v}_{t,c_e}^i$
 5:     $\forall i : J_t^i = i \cup \{j \in \mathcal{N} : (i,j) \in \mathcal{E}_t^{comm}\}$
 6:     $\forall i, s$ and $\forall j \in J_t^i : \hat{v}_{t,(c_e+1)}^i[s] \leftarrow \hat{v}_{t,c_e}^j[s]$ if $\hat{v}_{t,c_e}^j[s] \neq \emptyset$
 7: end for
 8: $\forall i : counted\_agents_t^i \leftarrow \sum_{s \in \mathcal{S}:\hat{v}_t^i[s] \neq \emptyset} \hat{v}_t^i[s]$
 9: $\forall i : uncounted\_agents_t^i \leftarrow N - counted\_agents_t^i$
10: $\forall i : unseen\_states_t^i \leftarrow \sum_{s \in \mathcal{S}:\hat{v}_t^i[s] = \emptyset} 1$
11: $\forall i, s$ where $\hat{v}_t^i[s]$ is not $\emptyset$ : $\tilde{\mu}_t^i[s] \leftarrow \frac{\hat{v}_t^i[s]}{N}$
12: $\forall i, s$ where $\hat{v}_t^i[s]$ is $\emptyset$ : $\tilde{\mu}_t^i[s] \leftarrow \frac{uncounted\_agents_t^i}{N \times unobserved\_states_t^i}$
13: return mean-field estimates $\{\tilde{\mu}_t^i\}_{i=1}^N$
```

Secondly, uncounted agents cannot be in states for which a count has already been obtained, since the count is complete and correct, even if the count is $x_{s'} = 0$. Therefore after the C_e communication rounds, the *uncounted_agents* proportion needs to be uniformly distributed only across the positions in the vector that still have the $\emptyset$ marker (Line 12), and not across all states as in the general setting. This makes the estimation more accurate in this special setting.

Appendix D. Additional remarks on mean-field estimation algorithms

In our Algs. A2 and A3, agents share their local *counts* with neighbours on the communication network $\mathcal{G}_t^{comm}$, and only after the C_e communication rounds do they complete their estimated distribution by distributing the uncounted agents along their vectors. An alternative would be for each agent to immediately form a local *estimate* from their local count obtained via $\mathcal{G}_t^{obs}$ or $\mathcal{G}_t^{vis}$, which is only then communicated and updated via the communication network. However, we take the former approach to avoid poor local estimations spreading through the network and leading to widespread inaccuracies. Information that is certain (the count) is spread as widely as possible, before being locally converted into an estimate of the total mean field. The same would be the case in our extension proposed in Sec.

H for averaging noisy counts, i.e. only the counts would be averaged, with the estimates completed by distributing the remaining agents after the C_e communication rounds.

Appendix E. Proof of Thm. 1

Proof. Recall that before the communication rounds in Line 21 (Alg. A1), the randomly updated policies $\{\pi^i_{k+1}\}^N_{i=1}$ have associated approximated returns $\{\sigma^i_{k+1}\}^N_{i=1}$. Denote the mean and maximum of this set $\sigma^{\text{mean}}_{k+1}$ and $\sigma^{\text{max}}_{k+1}$ respectively. Since π^{cent}_{k+1} is chosen arbitrarily from $\{\pi^i_{k+1}\}^N_{i=1}$, it will obey $\mathbb{E}[\sigma^{\text{cent}}_{k+1}] = \sigma^{\text{mean}}_{k+1}\ \forall k$, though there will be high variance. Conversely, the softmax adoption probability (Line 24, Alg. A1) for the networked case means by definition that policies with higher σ^i_{k+1} are more likely to be adopted at each communication round. Thus the π^{net}_{k+1} that gets adopted by the whole networked population will obey $\mathbb{E}[\sigma^{\text{net}}_{k+1}] > \sigma^{\text{mean}}_{k+1}$ (if $\tau^{comm}_{k+1} \to 0$, it will obey $\mathbb{E}[\sigma^{\text{net}}_{k+1}] = \sigma^{\text{max}}_{k+1}\ \forall k$). As such, $\mathbb{E}[\sigma^{\text{net}}_{k+1}] > \mathbb{E}[\sigma^{\text{cent}}_{k+1}]$, which by Ass. 1 implies the result. □

Appendix F. Experiments

Experiments were conducted on a Linux-based machine with 2 x Intel Xeon Gold 6248 CPUs (40 physical cores, 80 threads total, 55 MiB L3 cache). We use the JAX framework to accelerate and vectorise our code. Random seeds are set in our code in a fixed way dependent on the trial number to allow easy replication of experiments.

Appendix F.1. Games

We conduct numerical tests with five games. All are defined by the agents' reward/transition functions, and chosen for being particularly amenable to intuitive and visualisable understanding of whether the agents are learning behaviours that are appropriate and explainable for the respective objective functions. In all cases, rewards are normalised in [0,1] after they are computed.

Cluster.

This is the inverse of the 'exploration' game in [20], where in our case agents are encouraged to gather together by the reward function $R(s^i_t, a^i_t, \hat{\mu}_t) = \log(\hat{\mu}_t(s^i_t))$. That is, agent i receives a reward that is logarithmically proportional to the fraction of the population that is co-located with it at time t. We give the population no indication where they should cluster, agreeing this themselves over time.

Agree on a single target. Unlike in the above 'cluster' game, the agents are given options of locations at which to gather, and they must reach consensus among themselves. If the agents are co-located with one of a number of specified targets $\phi \in \Phi$ (in our experiments we place one target in each of the four corners of the grid), and other agents are also at that target, they get a reward proportional to the fraction of the population found there; otherwise they receive a penalty of -1. In other words, the agents must coordinate on which of a number of mutually beneficial points will be their single gathering place. Define the magnitude of the distances between x, y at t as $dist_t(x, y)$. The reward function is given by $R(s^i_t, a^i_t, \hat{\mu}_t) = r_{targ}(r_{collab}(\hat{\mu}_t(s^i_t)))$, where

$$r_{targ}(x) = \begin{cases} x & \text{if } \exists \phi \in \Phi \text{ s.t. } \text{dist}_t(s^i_t, \phi) = 0 \\ -1 & \text{otherwise,} \end{cases}$$

$$r_{collab}(x) = \begin{cases} x & \text{if } \hat{\mu}_t(s^i_t) > 1/N \\ -1 & \text{otherwise.} \end{cases}$$

Evade shark in shoal. Define the magnitude of the horizontal and vertical distances between x, y at t as $dist^h_t(x, y)$ and $dist^v_t(x, y)$ respectively. The state s^i_t now consists of agent i's position x^i_t and a 'shark's' position ϕ_t. At each time step, the shark steps towards the most populated grid point according to the empirical mean-field distribution i.e. $x^*_t = \arg\max_{x \in \mathcal{S}} \hat{\mu}_t(x)$. A horizontal step is taken if $dist^h_t(\phi_t, x^*_t) \geq dist^v_t(\phi_t, x^*_t)$, otherwise a vertical step is taken. As well as featuring a non-stationary distribution, we add 'common noise' to the environment, with the shark in a random direction with probability 0.01. Such noise that affects the local states of all agents in the same way, making the evolution of the distribution stochastic, makes population-independent policies sub-optimal [11]. Agents are rewarded more for being further from the shark, and also for clustering with other agents. The reward function is given by

$$R(s_t^i, a_t^i, \hat{\mu}_t) = dist_t^h(\phi_t, x_t^i) + \\ dist_t^v(\phi_t, x_t^i) + \text{norm}_{dist}(\log(\hat{\mu}_t(x_t^i))),$$

where $\text{norm}_{dist}(\cdot)$ indicates that the final term is normalised to have the same maximum and minimum values as the total combined vertical and horizontal distance.

Push object to edge. This is similar to the game presented in [30]. As before, define the magnitude of the horizontal and vertical distances between x, y at t as $dist_t^h(x,y)$ and $dist_t^v(x,y)$ respectively. The state s_t^i consists of agent i's position x_t^i and the object's position ϕ_t. The number of agents in the positions surrounding the object at time t generates a probability field around the object, such that the object is most likely to move in the direction away from the side with the most agents. As such, if agents are equally distributed around the object, it will be equally likely to move in any direction, but if they coordinate on choosing the same side, they can 'push' it in a certain direction. If Edges = {edge^1, . . . ,edge^4} are the grid edges, the closest edge to the object at time t is given by $\text{edge}_t^* = \arg\min_{\text{edge} \in \text{Edges}} \Big(\min(dist_t^h(\phi_t, \text{edge}), dist_t^h(\phi_t, \text{edge}) \Big)$. Agents are rewarded for how close they are to the object, and for how close the object is to the edge of the grid, i.e. they must coordinate on which side of the object from which to 'push' it, to ensure it moves to the grid's edge. The reward function is given by

$$R(s_t^i, a_t^i, \hat{\mu}_t) = dist_t^h(\phi_t, x_t^i) + dist_t^v(\phi_t, x_t^i) + \\ dist_t^h(\phi_t, \text{edge}_t^*) + dist_t^v(\phi_t, \text{edge}_t^*).$$

Disperse. This is similar to the 'exploration' games in [20], [19] and other MFG works. In our version agents are rewarded for being located in more sparsely populated areas but only if they are stationary. The reward function is given by $R(s_t^i, a_t^i, \hat{\mu}_t) = r_{stationary}(-\hat{\mu}_t(s_t^i))$, where

$$r_{stationary}(x) = \begin{cases} x & \text{if } a_t^i \text{ is 'remain stationary'} \\ -1 & \text{otherwise.} \end{cases}$$

Appendix F.2. Experimental metrics

To give as informative results as possible about both performance and proximity to the MFNE, we provide two metrics for each experiment. Both metrics are plotted with mean and standard deviation, computed over the ten trials (each with a random seed) of the system evolution in each setting.

Appendix F.2.1. Exploitability

Works on MFGs most commonly use the *exploitability* metric to evaluate how close a given policy π is to a NE policy π^* [11,13,19,20,28,31,32]. The metric usually assumes that all agents are following the same policy π, and quantifies how much an agent can benefit by deviating from π by measuring the difference between the return given by π and that of a BR policy with respect to the distribution generated by π:

Definition A1 (Exploitability of π)**.** *The exploitability Ex of policy π is given by:*

$$Ex(\pi) = V(BR(I(\pi)), I(\pi)) - V(\pi, I(\pi)).$$

If π has a large exploitability then an agent can significantly improve its return by deviating from π, meaning that π is far from π^*, whereas an exploitability of 0 implies that $\pi = \pi^*$. Prior works conducting empirical testing have generally focused on the centralised setting, so this classical definition, as well as most evaluations, only consider exploitability when all agents are following a single policy π_k. However, [13] notes that purely independent agents, as well as networked agents, may have divergent policies $\pi^i_k \neq \pi^j_k \; \forall i, k \in 1, \ldots, N$, as in our own setting. We therefore are interested in the 'exploitability' of the population's joint policy $\boldsymbol{\pi} := (\pi^1, \ldots, \pi^N) \in \Pi^N$.

Since we do not have access to the exact BR policy as in some related works [19,20], we must instead approximate the exploitability, similarly to [13,33]. We freeze the policy of all agents apart from a deviating agent, for which we store its current policy and then conduct 50 k loops of policy improvement. To approximate the expectations in Def. A1, we take the best return of the deviating agent across 10 additional k loops, as well as the mean of all the other agents' returns across these same 10 loops. (While the policies of all non-deviating agents is π_k in the centralised case, if the non-deviating agents do not share a single policy, then this method is in fact approximating the exploitability of their joint policy $\boldsymbol{\pi}^{-d}_k$, where d is the deviating agent.) We then revert the agent back to its stored policy, before learning continues for all agents as per the main algorithm. Due to the expensive computations required for this metric, we evaluate it every second k iteration of the main algorithm for Figs. 1(**a**), 1(**b**), A1, A2 and A3, and every fourth iteration for the population-dependent experiments.

The exploitability metric has a number of limitations in our setting. Our approximation takes place via MOMD policy improvement steps (as in the main algorithm) for an independent, deviating agent while the policies of the rest of the population are frozen. As such, the quality of our approximation is limited by the number of policy improvement/expectation rounds, which must be restricted for the sake of running speed of the experiments. Moreover, since one of the findings of our paper is that networked agents can improve their policies faster than independent or centralised agents, especially when non-linear function approximation is used, it is arguably unsurprising that approximating the BR by an independently deviating agent sometimes gives an unclear and noisy metric. This includes the exploitability going below zero, which should not be possible if the policies and distributions are computed exactly.

Moreover, in coordination games (the setting for all games apart from the 'disperse' game), agents benefit by following the same behaviour as others, and so a deviating agent generally stands to gain less from a BR policy than it might in the non-coordination games on which many other works focus. For example, the return of a best-responding agent in the 'push object' game still depends on the extent to which other agents coordinate on which direction in which to push the box, meaning it cannot significantly increase its return by deviating. This means that the downward trajectory of the exploitability metric is less clear in our plots than in other works. This is likely why the approximated exploitability gets lower in the non-coordination 'disperse' game in Fig. A3 than in the other games. Given the limitations presented by approximating exploitability, we also provide the second metric to indicate the progress of learning.

Appendix F.2.2. Average discounted return

We record the average discounted return of the agents' policies π^i_k during the M iterations - this allows us to observe that settings that converge to similar exploitability values may not have similar average agent returns, suggesting that some algorithms are better than others at finding not just NE, but preferable NE. See for example Figs. A1 and 1(**a**), where the networked agents converge to similar exploitability as the independent and centralised agents, but receive higher average returns.

Appendix F.3. Hyperparameters

See Table A1 for our hyperparameter choices. We can group our hyperparameters into those controlling the size of the experiment, those controlling the size of the Q-network, those controlling the

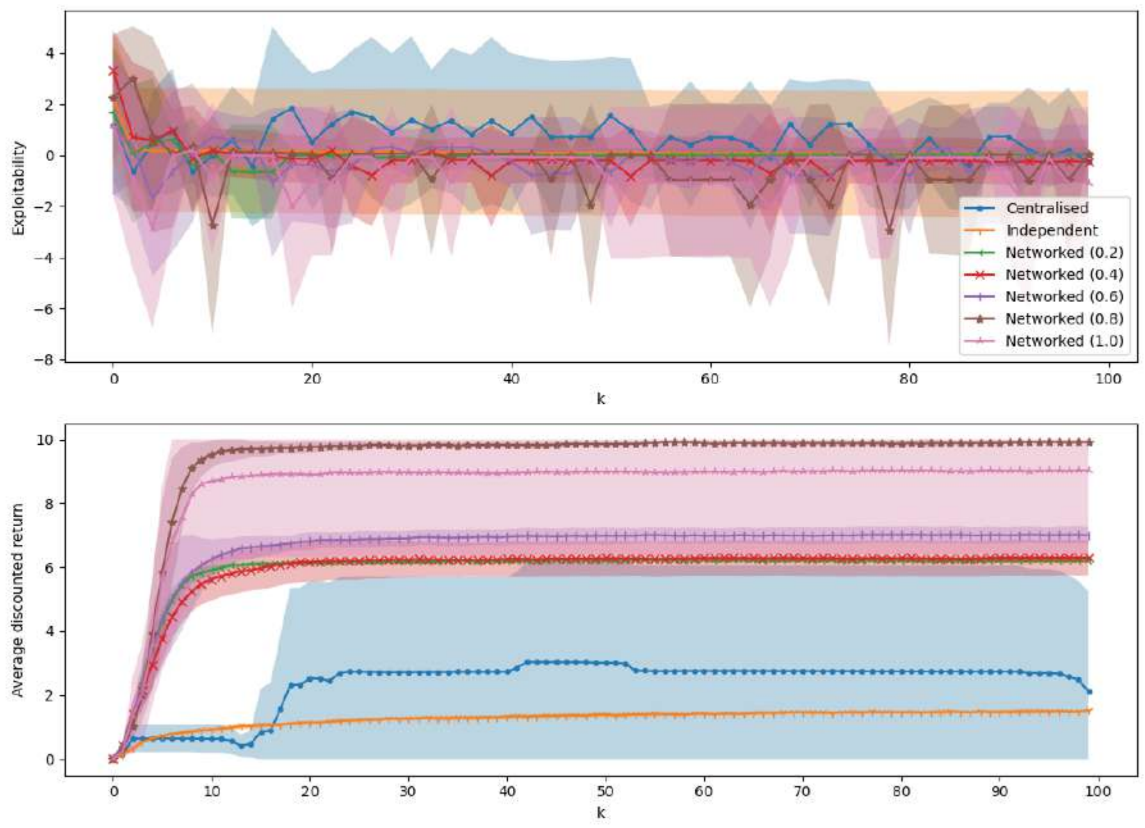


Figure A1. 'Target agreement' game, population-independent policies, 50x50 grid.

number of iterations of each loop in the algorithms and those affecting the learning/policy updates or policy adoption.

In our experiments we generally want to demonstrate that our communication-based algorithms outperform the centralised and independent architectures by allowing policies that are estimated to be better performing to proliferate through the population, such that convergence occurs within fewer iterations and computationally faster, even when the Q-function is poorly approximated and/or the mean field is poorly estimated, as is likely to be the case in on-the-fly, in-situ game learning. Moreover we want to show that there is a benefit even to a small amount of communication, so that communication rounds themselves do not excessively add to time complexity. As such, we generally select hyperparameters at the lowest end of those we tested during development, to show that our algorithms are particularly successful given what might otherwise be considered 'undesirable' hyperparameter choices.

Appendix F.4. Additional experiments

Appendix F.4.1. Additional experiments on large grids

We provide additional experiments on large grids in Figs. A1, A2 and A3.

In the 'target agreement' game in Fig. A1, the networked agents generally have lower exploitability than both centralised and independent agents, and significantly outperform the other architectures in terms of average return. As before, the margin by which the networked agents can outperform the centralised agents is much greater than in [13], showing that the benefits of the communication scheme are even greater in non-tabular settings.

In the 'cluster' game in Fig. A2, the networked agents obtain significantly higher return than the independent agents. While centralised agents have the lowest exploitability, networked agents of almost all communication radii outperform them in terms of average return, indicating that the communication scheme helps populations reach better performing equilibria.

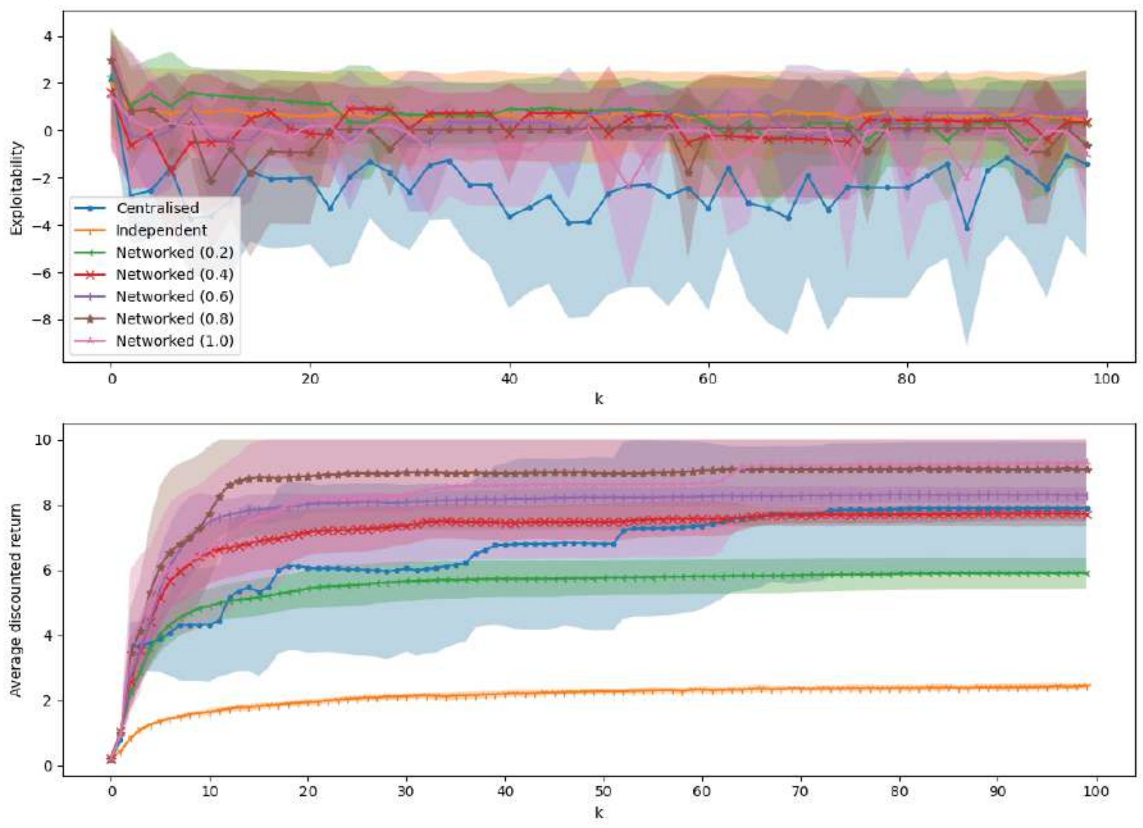


Figure A2. 'Cluster' game, population-independent policies, 50x50 grid.

In the 'disperse' game in Fig. A3, networked agents significantly outperform independent and centralised agents in terms of average return. They also outperform centralised agents in terms of exploitability, and significantly outperform independent agents in terms of exploitability. The fact that this happens in this non-coordination, competitive game shows that agents do have an incentive to communicate with each other even if they are self-interested.

Appendix F.4.2. Population-dependent policies in complex environments

We also showcase the ability of our algorithm to handle two more complex games, using population-dependent policies and estimated mean-field observations.

Figs. A4 and A5, where agents estimate the mean-field distribution via Alg. A3, differ minimally from Figs. A6 and A7, where agents directly receive the global mean-field distribution. This shows that our estimation algorithm allows agents to appropriately estimate the distribution, even with only one round of communication for agents to help each other improve their local counts. Only in the 'push object' game in Fig. A4, and there only with the smaller broadcast radii, do agents slightly underperform the returns of agents in the global observability case in Fig. A6, as is reasonable.

For the reasons given in Appx. F.2.1, the exploitability metric gives limited information in the 'push object' game in Fig. A4. In the 'evade' game in Fig. A5, exploitability suggests that centralised learners outperform the other cases. However, all of the networked cases significantly outperform the independent learners in terms of the average return to which they converge in both games. In the 'push object' game networked learners also significantly outperform centralised learners in all but the case with the smallest broad communication radius, while in the 'evade' game all networked cases perform similarly to the centralised case. Recall though that in realistic game settings a centralised architecture is a strong assumption, a computational bottleneck and single point of failure.

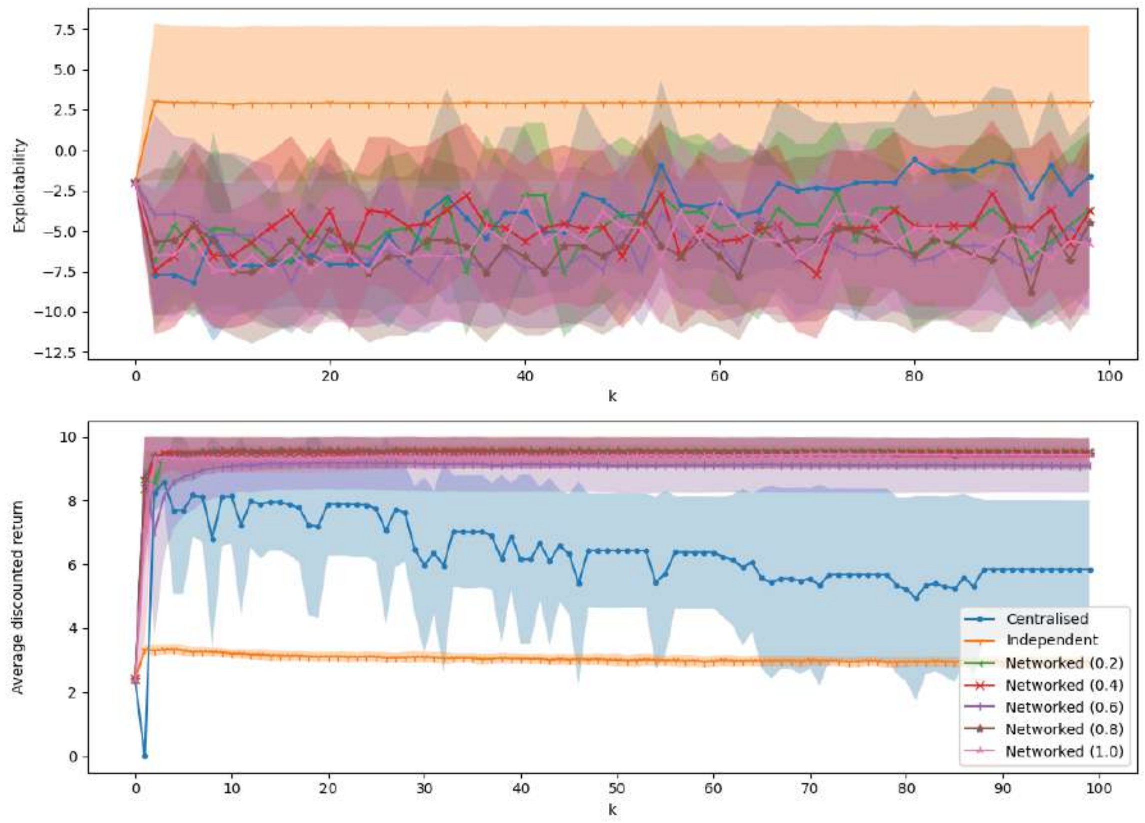


Figure A3. 'Disperse' game, population-independent policies, 100x100 grid.

Appendix G. Related work

MFGs are a quickly growing research area, so we only discuss the works most closely related to this present work, and instead refer the reader to [13] for detailed discussion around the setting of networked communication for MFGs, and to [11] for a broader survey of MFGs. Our work is most closely related to [13], which introduced networked communication to the infinite-horizon MFG setting. However, this work focuses only on tabular settings rather than using function approximation as in ours, and only addresses population-independent policies.

[20] uses Munchausen Online Mirror Descent (MOMD), similar to our method for learning with neural networks, but there are numerous differences to our setting: most relevantly, they study a finite-horizon episodic setting, where the mean-field distribution is updated in an exact way and an oracle supplies a centralised learner with rewards and transitions for it to learn a population-independent policy. [19] uses MOMD to learn population-dependent policies, albeit also with a centralised method that exactly updates the mean-field distribution in a finite-horizon episodic setting. [18] learns population-dependent policies with function approximation in infinite-horizon settings like our own, but does so in a centralised, two-timescale manner without using the empirical mean-field distribution. [42] also uses function approximation along a non-episodic trajectory, but involves a generic single agent learning only with an abstract estimate of the mean field rather than using an empirical population. Approaches that directly update an approximated mean field must be able to generate rewards from this arbitrary mean field, even if they otherwise claim to be oracle-free. They are thus inherently centralised algorithms and rely on strong assumptions that may not apply in realistic, complex games. Conversely, we are interested in practical convergence in on-the-fly, in-situ settings, where the reward is computed from the empirical finite population.

[34] addresses decentralised learning from a continuous, non-episodic run of the empirical system using either full or compressed information about the mean-field distribution, but agents are assumed to receive this information directly, rather than estimating it locally as in the algorithm we now present.

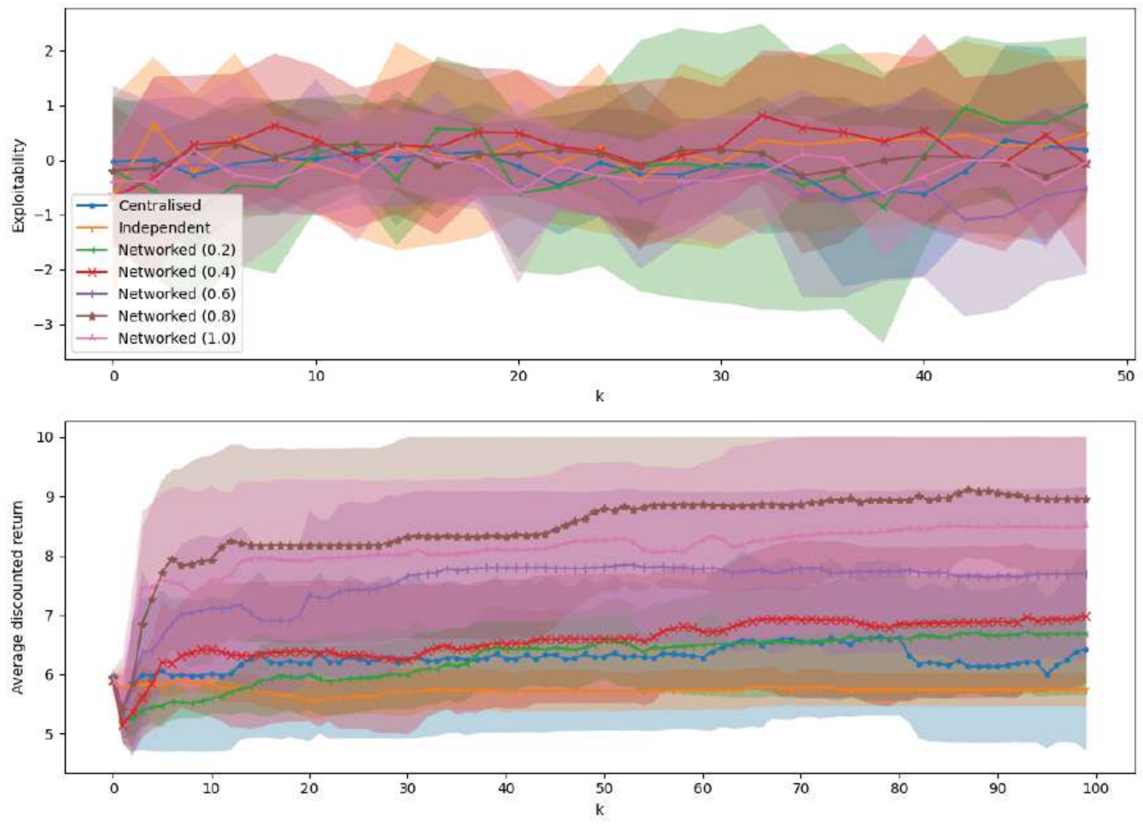


Figure A4. 'Push object' game, population-dependent policies with estimated mean-field distribution, 10x10 grid.

They also do not consider function approximation or inter-agent communication in their algorithms. In the closely related but distinct area of Mean-Field RL, [39] does estimate the empirical mean-field distribution from the local neighbourhood, however agents are seeking to estimate the mean action rather than the mean-field distribution over states as in our MFG setting. Their agents also do not have access to a communication network by which they can improve their estimates.

Appendix H. Limitations and future work

Our work follows the gold standard in MFGs by presenting experiments on grid world toy environments, albeit we show our algorithms are able to handle much larger and more complex games than prior work. Nevertheless future work lies in moving from these environments to more complex games. In Sec. 5 we give theoretical results showing that our networked algorithm can outperform a centralised alternative. We leave more general analysis, such proof of convergence and sample guarantees in the function approximation setting, for future work.

Alg. A3 assumes that if a state s' is connected to s on the visibility graph $\mathcal{G}_t^{vis}$, an agent in s is able to *accurately* count all the agents in s', i.e. it either counts the exact total or cannot observe the state at all. We assume this for simplicity but it is not inherently the case, since a realistic agent may have only noisy observations even of others located nearby, due to imperfect perception abilities. We suggest two ways to deal with this. Firstly, if agents share unique IDs as in Alg. A2, then when communicating their vectors of collected IDs with each other via $\mathcal{G}_t^{comm}$, agents would gain the most accurate picture possible of all the agents that have been observed in a given state. However, as we note above, there are various reasons why sharing IDs might be undesirable, including privacy and scalability. If instead only counts are taken, and if the noise on each agents' count is assumed to be independent and, for example, subject to a Gaussian distribution, the algorithm can easily be updated such that communicating agents compute averages of their local and received counts to improve their accuracy, rather than simply using communication to fill in counts for previously unobserved states.

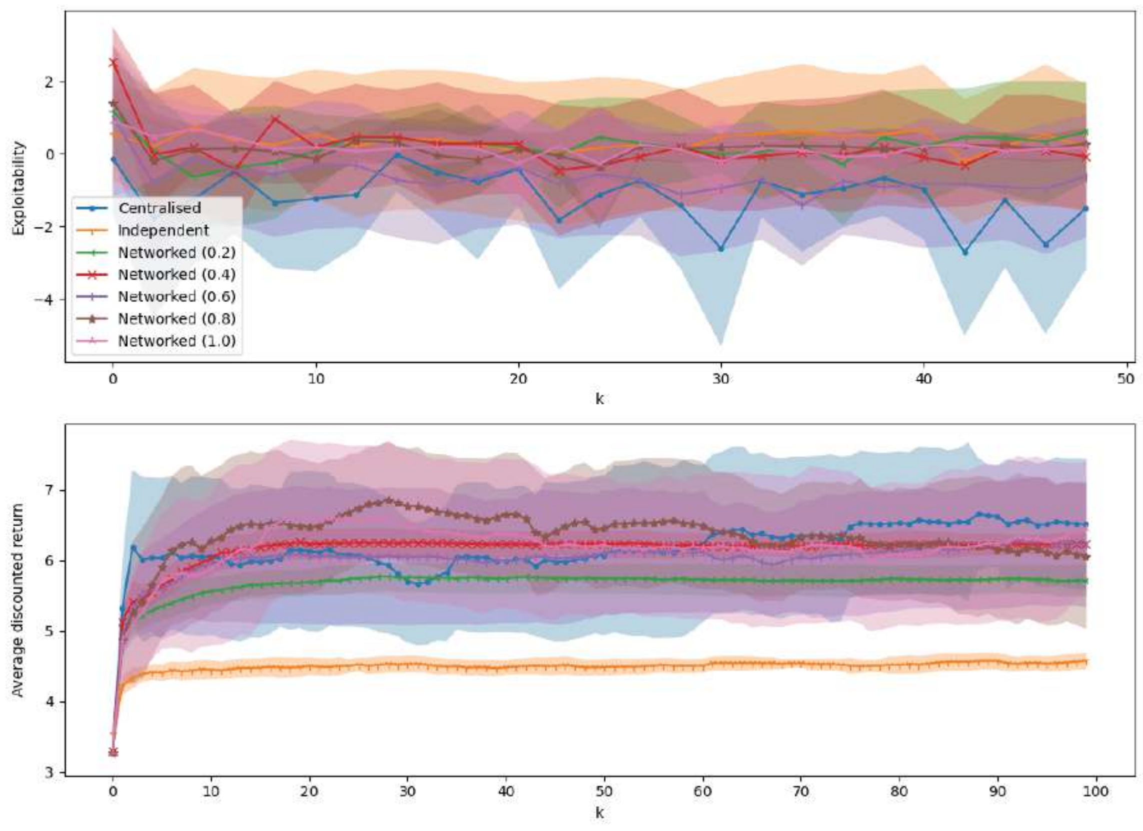


Figure A5. 'Evade' game, population-dependent policies with estimated mean-field distribution, 10x10 grid.

(Note that we can also consider the original case without noise to involve averaging, since averaging identical values equates to using the original value). Since the algorithm is intended to aid in local estimation of the mean-field distribution, which is inherently approximate due to the uniform method for distributing the uncounted agents, we are not concerned with reaching exact consensus between agents on the communicated counts, so we do not require repeated averaging to ensure asymptotic convergence.

We may wish to consider more sophisticated methods for distributing the uncounted agents across states, in place of the current uniform distribution. Such choices may be domain-specific based on knowledge of a particular environment. For example, one might use the counts to perform Bayesian updates on a specific prior, where this prior may relate to the estimated mean-field distribution at the previous time step $t-1$. If agents seek to learn to predict the *evolution* of the mean field based on their own policy or by learning a model, the Bayesian prior may also be based on forward prediction from the estimated mean-field distribution at $t-1$. Future work lies in conducting experiments in all of these more general settings.

[18] notes that in grid-world settings such as those in our experiments, passing the (estimated or true global) mean-field distribution as a flat vector to the Q-network ignores the geometric structure of the problem. They therefore propose to create an embedding of the distribution by first passing the vector to a convolutional neural network, essentially treating the categorical distribution as an image. This technique is also followed in [19] (for their additional experiments, but not in the main body of their paper). As future work, we can test whether such a method improves the performance of our algorithms.

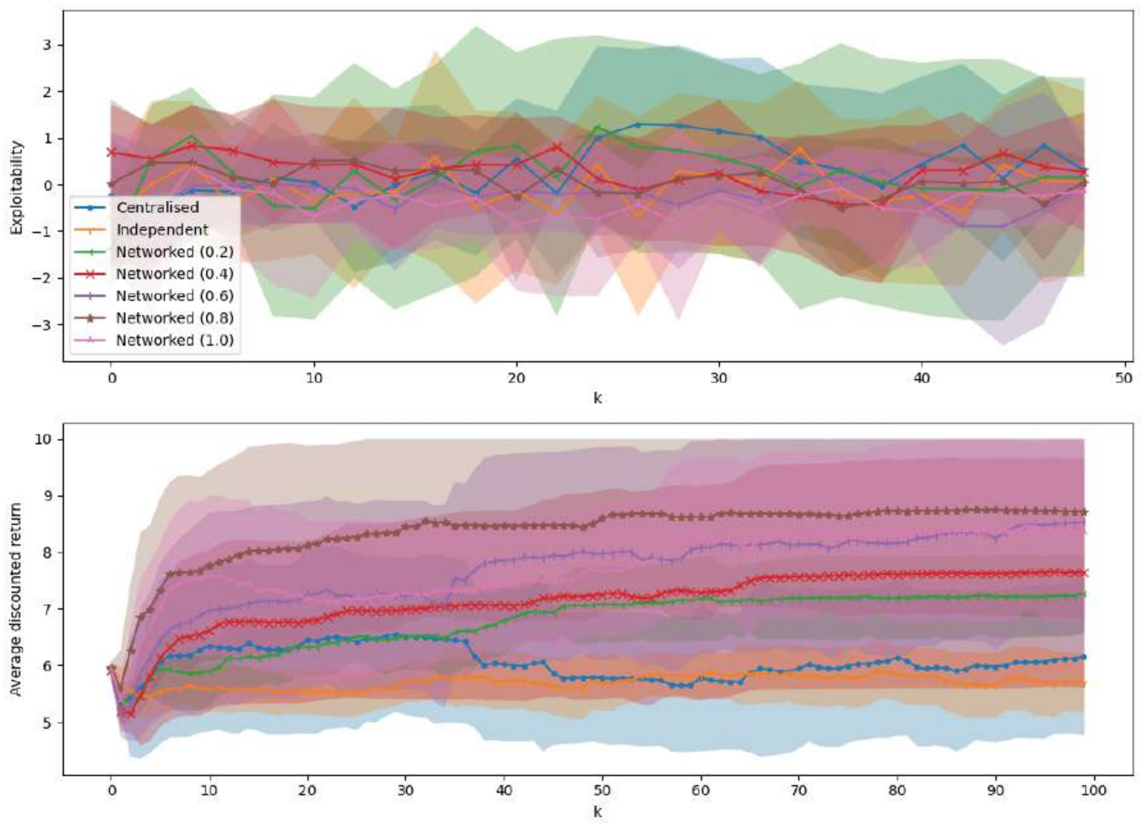


Figure A6. 'Push object' game, population-dependent policies with global observability of mean field, 10x10 grid.

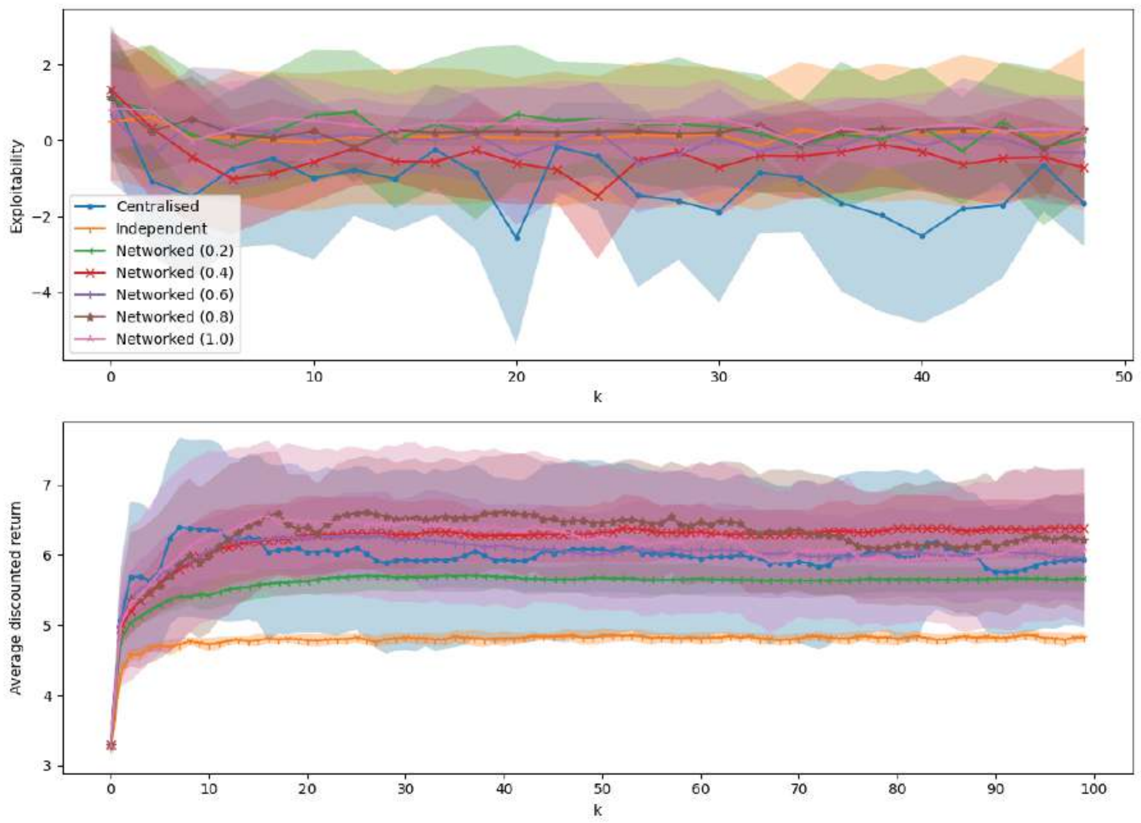


Figure A7. 'Evade' game, population-dependent policies with global observability of mean field, 10x10 grid.

Hyperparam.	Value	Comment
Trials	10	We run 10 trials with different random seeds for each experiment. We plot the mean and standard deviation for each metric across the trials.
Gridsize	10x10 / 50x50 / 100x100	Experiments with population-dependent policies are run on the 10x10 grid (Figs. A4, A5, A6 and A7), while experiments on large state spaces are run on 50x50 and 100x100 grids (Figs. 1(**a**), 1(**b**), A1, A2 and A3).
Population	500	We chose 500 for our demonstrations to show that our algorithm can handle large populations, indeed often larger than those demonstrated in other mean-field works, especially for grid-world environments, while also being feasible to simulate wrt. time and computation constraints [13,19,29,34–41].
Number of neurons in input layer	cf. comment	The agent's position is represented by two concatenated one-hot vectors indicating the agent's row and column. An additional two such vectors are added for the shark's/object's position in the 'evade' and 'push object' games. For population-dependent policies, the mean-field distribution is a flattened vector of the same size as the grid. As such, the input size in the 'evade' and 'push object' games is $[(4 \times \text{dimension}) + (\text{dimension}^2)]$; in the other settings it is $[2 \times \text{dimension}]$.
Neurons per hidden layer	cf. comment	We draw inspiration from common rules of thumb when selecting the number of neurons in hidden layers, e.g. it should be between the number of input neurons and output neurons / it should be 2/3 the size of the input layer plus the size of the output layer / it should be a power of 2 for computational efficiency. Using these rules of thumb as rough heuristics, we select the number of neurons per hidden layer by rounding the size of the input layer down to the nearest power of 2. The layers are all fully connected.
Hidden layers	2	We experimented with 2 and 3 hidden layers in the Q-networks. While 3 hidden layers gave similar or slighly better performance, we selected 2 for increased computational speed for conducting our experiments.
Activation function	ReLU	This is a common choice in deep RL.
K	100	K is chosen to be large enough to see at least one of the metrics converging.
M	50	We tested M in {50,100} and found that the lower value was sufficient to achieve convergence while minimising training time. It may be possible to converge with even smaller choices of M.
L	50	We tested L in {50,100} and found that the lower value was sufficient to achieve convergence while minimising training time. It may be possible to converge with even smaller choices of L.
E	20	We tested E in {20,50,100}, and choose the lowest value to show the benefit to convergence even from very few evaluation steps. It may be possible to reduce this value further and still achieve similar results.
C_p	1	As in [13], we choose this value to show the convergence benefits brought by even a single communication round, even in networks that may have limited connectivity; higher choices are likely to have even better performance.
C_e	1	Similar to C_p, we choose this value to show the ability of our algorithm to appropriately estimate the mean field even with only a single communication round, even in networks that may have limited connectivity.
γ	0.9	Standard choice across RL literature.
τ_q	0.03	We tested τ_q in {0.01,0.02,0.03,0.04,0.05}, as well as linearly decreasing τ_q from $0.05 \to 0$ as k increases. However, only 0.03 gave stable increase in return. Note that this is the value also chosen in [21].
$\|B\|$	32	This is a common choice of batch size that trades off noisy updates and computational efficiency.
cl	-1	We use the same value as in [21].
ν	$L-1$	We tested ν in $\{1,4,20,L-1\}$. We found that in our setting, updating $\theta' \leftarrow \theta$ once per k iteration s.t. $\theta'_{k+1,l} = \theta_{k,l}\ \forall l$ gave sufficient learning that was similar to the other potential choices of ν, so we do this for simplicity, rather than arbitrarily choosing a frequency to update θ' during each k loop. Setting the target to be the policy from the previous iteration is similar to the method in [20]. Whilst [19] updates the target within the L loops for stability, we do not find this to be a problem in our experiments.
Optimiser	Adam	As in [21], we use the Adam optimiser with initial learning rate 0.01.
τ_k^{comm}	cf. comment	τ_k^{comm} increases linearly from 0.001 to 1 across the K iterations. This is a simplification of the annealing scheme used in [13]. Further optimising the annealing process may lead to better results.

Table A1. Hyperparameters