Distributed gradient methods to reach a Nash equilibrium in potential games

Vineeth Varma, Jomphop Veetaseveera, Romain Postoyan, Irinel-Constantin Morârescu

Distributed gradient methods to reach a Nash equilibrium in potential games

Vineeth S. Varma, Jomphop Veetaseveera, Romain Postoyan and Irinel-Constantin Morărescu

***Abstract*— We study multi-agent optimization problems described by ordinal potential games. The objective is to reach a Nash equilibrium (NE) in a distributed manner. It is well known that an asynchronous best response dynamics (ABRD) will always converge to a pure NE in this case. However, computing the exact best response at every step of the algorithm may be computationally heavy, if not impossible. Therefore, instead of computing the exact best response, we propose an algorithm that performs a "better response", which decreases the local cost rather than minimizing it. The agents perform a distributed asynchronous gradient descent algorithm, in which only a finite number of iterations of the gradient descent are performed by each player. We prove that this algorithm always converges to a NE and demonstrate via simulations that the computational time to reach the NE can be much shorter than with the classical ABRD. Taking into account the time required for each agent to communicate, the proposed algorithm is shown to also outperform a distributed synchronous gradient descent in simulations.**

I. Introduction

Game theory has been widely applied in various fields like economics [1], wireless communication [2] and automatic control [3]. A special class of games called potential games was introduced and studied by Monderer in [4], and several conditions guaranteeing the existence of a pure Nash equilibrium (NE) were provided. Potential games find many applications, e.g., in traffic management, wireless design, model predictive control, see [5].

In [6], it was shown that in potential games, the asynchronous best response dynamics (ABRD), always converges to a pure NE. Recall that the ABRD involves players updating their actions to be one that minimizes their local cost asynchronously, while the other player actions are fixed. However, in many situations, the exact best responses which are obtained by minimizing a certain cost are hard or impossible to compute, and instead, algorithms may be used to find approximate best responses. Recent works have studied the convergence of approximate best response dynamics for special applications. In [7], a framework for iterative approximate best response algorithms is provided for distributed constraint optimization problems. In [8], the authors focus on approximate best response dynamics in interference games, which are a special class of games often studied in wireless communication. On the other hand, [9] looks at stochastic algorithms which are used to approximate the NE. Finally, in [10], the authors show that better reply dynamics can converge to a NE in aggregative games. However, these results do not study the best responses approximated by gradient methods. Gradient-based methods are extremely popular in a variety of multi-agent settings due to their ease of implementation, versatility, and dependence on purely local information. Using gradient methods, even when computing the exact best response as required in [6] is feasible, it may take a very large number of steps, resulting in poor computational performance. Thus, the development of distributed gradient-based algorithms to reach the NE is of great interest to various fields and applications [2], [5].

Distributed gradient descent (DGD) methods were introduced in [11] and have been well studied in the literature. This involves a multi-objective optimization problem in which the optimization variable is shared across all agents and a consensus type dynamic is proposed in order to achieve a common solution, see also [12]. These algorithms have been used to solve various optimization problems such as large-scale matrix factorization [13] and linear programs [14]. Recently, distributed *synchronous* gradient-based methods and convergence properties to differential NE were studied in [15], [16]. These results focus on convergence to differential NE, which are typically easier to reach by gradient methods, and do not always correspond to a pure NE. The authors also look at the standard gradient descent, which may not be suitable for optimization problems with constraints. Finally, they apply a synchronous algorithm, which might result in a shorter computation time, as all agents compute local solutions in parallel, but will require more frequent communications as agents must share information after each step of the gradient descent. When all the agents share a communication medium, this requirement can lead to poor communication performance and consequently, slower updates in the algorithm. These issues motivate the development of an asynchronous algorithm applying projected gradient descent.

In this context, we provide a *distributed asynchronous gradient descent* (DAGD) algorithm, in which each player iteratively and asynchronously applies an arbitrary number of tunable steps of the local *projected* gradient descent algorithm. The DAGD only requires one agent to communicate per iteration and allows to tune the gradient method in order to reduce the total running time of the algorithm. We consider ordinal potential games in which the cost function of each player is convex. In this framework, we prove the convergence of the DAGD to a NE, and when the potential function is convex, we also prove uniform global asymptotic stability of the NE set. Later, we demonstrate via simulations that the proposed algorithm has an shorter overall running

The authors are with Université de Lorraine, CNRS, CRAN, F-54000 Nancy, France, vineeth.satheeskumar-varma@univ-lorraine.fr
This work was partially funded by the ANR projects NICETWEET and HANDY, and by CEFIPRA under the project 6001-1/2019.

time when compared to existing algorithms while accounting for the communication time.

The rest of the paper is organized in the following manner. In Section II, we describe the game and provide the essential notions used to develop our algorithm. In Section III, we provide the DAGD algorithm and analyze it. In Section IV, we recall some of the state-of-the-art algorithms used to reach the NE, which we then compare with the DAGD, in terms of the computational and communication load on numerical simulations.

Notations. We use $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{R}_{>0}$ for the set of real numbers, non-negative real numbers and positive real numbers respectively. The notation $\mathbb{Z}_{\geq 0}, \mathbb{Z}_{>0}$ respectively stands for the set of non-negative and positive integers respectively. We use $||\cdot||_2$ to denote the induced 2-norm of a real matrix and $|x|_S := \inf\{|x-y| : y \in S\}$ to denote the distance of a point to a non-empty set S. Next, for a compact and convex set S, we denote the projection of a point x on S as $\mathbf{P}_S(x) := \arg\min_{y\in S}\{|x-y|\}$, which is unique for convex S. We say a continuous function $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is of class $\mathcal{KL}$ if i) $\beta(\cdot, s)$ is of class $\mathcal{K}$ for any fixed $s \in \mathbb{R}_{\geq 0}$, i.e., it is strictly increasing and $\beta(0, s) = 0$, and ii) $\beta(r, s)$ is decreasing in s for fixed $r \in \mathbb{R}_{\geq 0}$ such that $\lim_{s\to\infty} \beta(r, s) = 0$.

II. Problem statement

A. Game model

We consider a game expressed in strategic form as $\mathcal{G} := \{\mathcal{N}, \{\mathcal{X}_i\}_{i\in\mathcal{N}}, \{J_i\}_{i\in\mathcal{N}}\}$, where

- $\mathcal{N} := \{1, \dots, N\}$ is the set of players with $N \in \mathbb{Z}_{>0}$ players,
- the actions of all players are given by $x_i \in \mathcal{X}_i \subset \mathbb{R}^{M_i}$, with $M_i \in \mathbb{Z}_{>0}$, $\mathcal{X}_i$ being compact for $i \in \mathcal{N}$,
- $J_i : \mathcal{X}_i \to \mathbb{R}_{\geq 0}$ is the cost function of player $i \in \mathcal{N}$.

We use $x := (x_1, \dots, x_N) \in \mathcal{X}$ to denote the action profile of all agents and, for any $i \in \mathcal{N}$, $x_{-i} := (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N) \in \mathcal{X}_{-i}$ to denote the actions of all agents excluding agent i, where $\mathcal{X}_{-i} := \mathcal{X}_1 \times \dots \mathcal{X}_{i-1} \times \mathcal{X}_{i+1} \cdots \times \mathcal{X}_N$. We recall below the definition of ordinal potential games.

Definition 1 (Ordinal potential game): $\mathcal{G}$ is an *ordinal potential game* if there exists a *potential function* $\phi : \mathcal{X} \to \mathbb{R}_{>0}$ such that for all $i \in \mathcal{N}$, $x \in \mathcal{X}$, and $x_i' \in \mathcal{X}_i$ $J_i(x_i, x_{-i}) - J_i(x_i', x_{-i}) < 0 \iff \phi(x_i, x_{-i}) - \phi(x_i', x_{-i}) < 0$. □

Ordinal potential games cover both exact and weighted potential games as special cases [4]. For the remainder of the paper, we make the next assumption.

Assumption 1: Game $\mathcal{G}$ is an ordinal potential game with a continuous potential function ϕ. □

In game theory, the notion of pure NE plays a heavy role and is defined as follows [17].

Definition 2 (Pure NE): A strategy $x^* \in \mathcal{X}$ is said to be a pure NE if and only if

$$J_i(x_i^*, x_{-i}^*) \leq J_i(x_i, x_{-i}^*) \tag{1}$$

for all $x_i \in \mathcal{X}_i$ and all $i \in \mathcal{N}$. □

In a pure NE, assuming that each player has knowledge of the actions played by the other players, no player has anything to gain by changing only their own action. Since $\mathcal{G}$ is an ordinal potential game, $\mathcal{G}$ admits at least one pure NE [4].

B. Best and better responses

As explained in the introduction, best response dynamics are often used to reach the pure NE. For any $i \in \mathcal{N}$, we use $f_i : \mathcal{X}_{-i} \to \mathcal{P}(\mathcal{X}_i)$ to denote the best response of player i to the actions of the other players x_{-i}, i.e.

$$f_i(x_{-i}) := \arg\min_{x_i \in \mathcal{X}_i} J_i(x_i, x_{-i}) \tag{2}$$

for any $x_{-i} \in \mathcal{X}_{-i}$, where $\mathcal{P}$ is the power set function, i.e., $\mathcal{P}(S)$ is the set of all subsets of S, including the empty set and S itself.

Best response dynamics are often used to find the NE of a game [18] and it can be verified by looking at (1)-(2) that any equilibrium point of this dynamics corresponds to a NE. However, in many practical situations, computing the exact best response as in (2) may be challenging or computationally heavy, see examples in [2], [3], [5]. To overcome this issue, we may resort to applying better responses, like in [10], which are defined as follows.

Definition 3 (Better response): We say that $F_i : \mathcal{X} \to \mathcal{X}_i$ is a *better response* by agent $i \in \mathcal{N}$ if

$$J_i(F_i(x), x_{-i}) < J_i(x) \tag{3}$$

for all x such that $x_i \notin f_i(x_{-i})$. □

A better response by player i to actions x_{-i}, given a previous action x_i, is an action which will strictly decrease its cost, unless x_i is already the best response.

Our objective is to propose a distributed asynchronous algorithm, which applies better responses rather than the true best response and is guaranteed to converge to a NE for ordinal potential games.

III. Main results

In this section, we present the distributed asynchronous gradient descent (DAGD) which we introduce and then analyze. As we focus on gradient-based methods, we make the following assumption on the cost functions.

Assumption 2: The action space $\mathcal{X}$ is convex, and for all $i \in \mathcal{N}$, the cost functions $J_i : \mathcal{X}_i \times \mathcal{X}_{-i} \to \mathbb{R}_{\geq 0}$ are twice differentiable and convex with respect to $x_i \in \mathcal{X}_i$. □

We use $\nabla_i J_i(x_i; x_{-i})$ to denote the partial gradient of J_i with respect to the i-th player action evaluated at x_i, for a given x_{-i}, which exists in view of Assumption 2. This gradient can be evaluated at any $x_i \in \mathcal{X}_i$ for any $i \in \mathcal{X}_i$, and is also differentiable. Moreover, as $\mathcal{X}_i$ is compact, we can define

$$L_i := \sup_{x\in\mathcal{X}} ||\nabla_i^2 J_i(x_i, x_{-i})||_2, \tag{4}$$

which is finite. The algorithm we present in the following requires the knowledge of L_i by each player, see also Remark 1.

Assumption 3: Each player $i \in \mathcal{N}$ knows L_i. □

Assumption 2 allows for the best response in (2) to be a single valued set (rather than a general set) and be computed using a projected gradient descent algorithm. We denote one step of the projected gradient descent by $\hat{f}_i : \mathcal{X} \times \mathbb{R}_{>0} \to \mathcal{X}_i$, which is given by

$$\hat{f}_i(x_{-i}, x_i, \gamma_i) := \mathbf{P}_{\mathcal{X}_i}\left(x_i - \gamma_i \nabla_i J_i(x_i, x_{-i})\right), \quad (5)$$

where $\gamma_i \in (0, L_i^{-1}]$ is the chosen step size, which each player can select in view of Assumption 3. In contrast to the standard ABRD, which requires each agent to compute the best response, in the DAGD, each player i only applies a finite tunable number M_i of iterations of the gradient descent as described in Algorithm 1. This algorithm takes the initialization x_i of player i, actions of the other players x_{-i}, and i, M_i as inputs, and returns the new action of player i.

Algorithm 1: The M-step gradient descent

Data: Given $\gamma_i \in (0, L_i^{-1}]$ and J.
$\mathrm{GD}(x(k), i, M_i)$
$x_i^+ = x_i$;
if $|\hat{f}_i(x_{-i}(k), x_i(k), \gamma_i) - x_i(k)| > 0$ **then**
  for M_i *iterations* **do**
    $x_i^+ \leftarrow \hat{f}_i(x_{-i}, x_i^+, \gamma_i)$;
  end
end
return x_i^+;

In Algorithm 1, if $\hat{f}_i(x_{-i}(k), x_i(k), \gamma_i) = x_i(k)$ then the gradient is 0 or the algorithm has converged to the boundary of $\mathcal{X}_i$ and can therefore be stopped. However, as long as $\hat{f}_i(x_{-i}(k), x_i(k), \gamma_i) \neq x_i(k)$, the cost function can be strictly decreased due to the convexity of J_i imposed by Assumption 2.

Remark 1: Even if L_i is unknown, i.e., Assumption 3 does not hold, a suitable γ_i may be found such that the local cost J_i is decreased by iteratively searching for a step size in $\{\gamma_i, \frac{1}{2}\gamma_i, \dots\}$. We do not treat this case for ease of presentation. □

Remark 2: Although we describe the M-step gradient descent in terms of the full state x, each player i only needs to know partial information x which affects its cost. For example, if the agents interact over a graph and only the neighbors of an agent affect its cost function, then only neighbors need to exchange information. We write everything in terms of x for ease of notation and presentation. □

We describe the output of Algorithm 1 with the function $F_i^m(x_i, x_{-i})$ where x_i is the initialization of the gradient descent, defined as

$$F_i^0(x) := x_i \quad (6)$$

and then for $m \in \mathbb{Z}_{>0}$, we recursively define

$$F_i^m(x) := \hat{f}_i\left(x_{-i}, F_i^{m-1}(x), \gamma_i\right). \quad (7)$$

Next, we show that F_i^m is a better response as in Definition 3.

Lemma 1: Under Assumption 2, given $\gamma_i \in (0, L_i^{-1}]$, F_i^m is a better response for any $m \in \mathbb{Z}_{>0}$. □

Proof: From Theorem 10.6 in [19], we know that for any $x \in \mathcal{X}$, $i \in \mathcal{N}$, and $\gamma_i \leq L_i^{-1}$,

$$J_i(x) - J_i(\hat{f}_i(x_{-i}, x_i, \gamma_i), x_{-i}) \geq \frac{1}{2\gamma_i}|x_i - \hat{f}_i(x_{-i}, x_i, \gamma_i)|^2. \quad (8)$$

This means that there are the following two possibilities:

1) $\hat{f}_i(x_{-i}, x_i, \gamma_i) \neq x_i$. In this case, the cost is strictly smaller after one step of the projected gradient descent.
2) Otherwise, $\hat{f}_i(x_{-i}, x_i, \gamma_i) = x_i$ which means that x_i is the minimum of J_i for the given x_{-i} due to J_i being convex in x_i according to Assumption 2.

First of all, we note that in both cases, $J_i(\hat{f}_i(x_{-i}, x_i, \gamma_i), x_{-i}) \leq J_i(x)$. Therefore (for any $m \in \mathbb{Z}_{>0}$) if we have case 1) applied at least once in the recursion (7), $J_i(F_i^m(x), x_{-i}) < J_i(x)$ making F_i^m a better response. If case 1) is never applied, then $F_i^m(x) = x_i$ and $\hat{f}_i(x_{-i}, x_i, \gamma_i) = x_i$ which means that x_i is the minimum of J_i for the given x_{-i}. Hence, $x_i \in f_i(x_{-i})$, which still satisfies Definition 3. This concludes the proof that F_i^m is a better response for any $m \in \mathbb{Z}_{>0}$. ■

We next present the DAGD in Algorithm 2, in which, each agent i applies $M_i \in \mathbb{Z}_{>0}$ steps of the gradient descent described in Algorithm 1 iteratively in an asynchronous manner, where M_i is a tunable parameter. Note that the ABRD is recovered as a special case of the DAGD by setting $M_i \to \infty$ for all $i \in \mathcal{N}$.

Algorithm 2: The DAGD

Data: Given $\theta(\cdot), x(0) \in \mathcal{X}$, $M \in \mathbb{Z}_{>0}^N$.
Initialize $\text{flag}_i = 1$ for all $i \in \mathcal{N}$ and $k = 1$;
do
  $i \leftarrow \theta(k+1)$;
  $\text{flag}_i \leftarrow 1$;
  $x_i(k+1) \leftarrow \mathrm{GD}(x(k), i, M_i)$;
  $x_{-i}(k+1) \leftarrow x_{-i}(k)$;
  if $x_i(k+1) = x_i(k)$ **then**
    $\text{flag}_i \leftarrow 0$
  end
  Communicate $x_i(k+1)$ to neighbors;
  $k \leftarrow k+1$;
while $\sum_i \text{flag}_i \neq 0$;

In contrast with the standard ABRD, where at each iteration, the active player i must find its best response, which requires the player to seek the asymptotic value of a projected gradient, the proposed DAGD algorithm just requires the active player i to perform M_i steps of the projected gradient descent at each iteration with $M = (M_1, \dots, M_N) \in \mathbb{Z}_{>0}^N$. When an agent i has no change in its action after applying their projected gradient descent, we set its flag to 0, and this means that $x_i(k)$ is the best response to $x_{-i}(k)$. When all the flags are 0 simultaneously, we stop the algorithm, and $x(k)$ is a NE as will be proven in Theorem 1. We impose the next

assumption on the update order $\theta(\cdot)$ to ensure convergence of the DAGD.

Assumption 4: There exists $K \in \mathbb{Z}_{>0}$ with $N \leq K < \infty$ such that

$$\bigcup_{\ell=1}^{K} \{\theta(k+\ell)\} = \mathcal{N} \tag{9}$$

for any $k \in \mathbb{Z}_{>0}$. □

Assumption 4 implies that the DAGD applies the gradient descent for all players at least once during a period of K steps. A simple example of a θ satisfying this assumption would be a round-robin scheduler. Note that once the sequence θ is fixed, this process is fully distributed as at each step k, the agent $\theta(k)$ updates its action by using gradient descent. In practice, implementing the DAGD would require all agents to agree on the update order θ before starting the algorithm, or a centralized scheduler informs each player when to update. The DAGD resulting from Algorithm 2 can be written as follows, for any $k \in \mathbb{Z}_{\geq 0}$,

$$\begin{aligned} x_{\theta(k)}(k+1) &= F_{\theta(k)}^{M_{\theta(k)}}(x(k)) \\ x_{-\theta(k)}(k+1) &= x_{-\theta(k)}(k), \end{aligned} \tag{10}$$

with $M_i \in \mathbb{Z}_{>0}$ for all $i \in \mathcal{N}$ and

$$\theta(k+1) \in \mathcal{N}, \tag{11}$$

respecting Assumption 4. Next, we present Theorem 1 which ensures the convergence of (10) to a pure NE.

Theorem 1: Under Assumptions 1-3, for any $x(0) \in \mathcal{X}$, $M = (M_1, \ldots, M_N) \in \mathbb{Z}_{>0}^N$ and sequence $\theta(\cdot)$ ensuring Assumption 4, the DAGD described by (10) will converge to a pure NE given in Definition 2. □

Proof: For a given initial state $x(0) \in \mathcal{X}$, after $k \in \mathbb{Z}_{>0}$ steps of the DAGD (10), let the action profile be given by $x(k)$. We use $i = \theta(k)$ to denote the player active at step k, $m = M_i$ the number of GD steps applied by it. From Lemma 1, we have that F_i^m is a better response. Therefore, by construction of the algorithm, we have either

1) $J_i(x_i(k+1), x_{-i}(k+1)) < J_i(x_i(k), x_{-i}(k))$. In this case, using the definition of a potential game, we have $\phi(x(k+1)) < \phi(x(k))$.
2) $J_i(x_i(k+1), x_{-i}(k+1)) = J_i(x_i(k), x_{-i}(k))$. Similarly, using the definition of a potential game, we have $\phi(x(k+1)) = \phi(x(k))$.

By construction of the algorithm, we have $x_i(k+1) = x_i(k)$, only when $\hat{f}_i(x_{-i}(k), x_i(k), \gamma_i(k-1)) = x_i(k)$. Since $J_i(x_i, x_{-i})$ is convex in x_i for any i, the projected gradient descent is stationary if and only if $J_i(x_i(k), x_{-i}(k)) = \min_{x_i}\{J_i(x_i, x_{-i}(k))\}$. Note that if this holds true for all i, $x(k)$ is a NE by definition.

Recall that within K steps of the DAGD, all players are visited at least once. Therefore, during K iterations, as long as $x \notin \mathcal{X}^*$, we have at least one $j \in \mathcal{N}$, such that $\phi(F_j^{M_j}(x_{-j}), x_{-j}) - \phi(x) < 0$. Therefore, as long as $x(k) \neq x^*$, we will have $\phi(x(k+K)) - \phi(x(k)) < 0$. This ensures that ϕ is strictly decreasing over K steps as long as $x \notin \mathcal{X}^*$. Therefore $\phi(x(k))$ will asymptotically reach a NE or its minimum in $\mathcal{X}$. Note that any x minimizing ϕ, i.e., x such that $\phi(x) = \min_{x' \in \mathcal{X}}\{\phi(x')\}$ is also a NE [4]. We have thus proven convergence to a NE. ■

When multiple NE exist, the DAGD may not converge to the same NE as the ABRD even when using the same sequence $\theta(\cdot)$ and initial condition $x(0)$. In general, varying M, $\theta(\cdot)$ or $x(0)$ may result in convergence to a different NE. On the other hand, when ϕ is strictly convex, the NE is unique [4] and therefore all algorithms will converge to the unique NE.

When ϕ is convex, we are able to show that the set of pure NE is also uniformly globally asymptotically stable (UGAS) under (10)-(11), as formalized below. This case of great interest in the framework of distributed optimization as a pure NE in a convex potential game minimizes the potential function [4]. We introduce the attractor set $\mathcal{A} := \{(x, \theta) : x \in \mathcal{X}^*, \theta \in \mathcal{N}\}$ for this purpose.

Theorem 2: Consider the dynamical system described by (10)-(11), and suppose Assumptions 1-4 hold with ϕ convex. Then, for any $M = (M_1, \ldots, M_n) \in \mathbb{Z}_{>0}^N$, $\mathcal{A}$ is UGAS, i.e., for any $M \in \mathbb{Z}_{>0}^N$ there exists $\beta_M \in \mathcal{KL}$ such that any solution $(x(k), \theta(k))$ verifies $|(x(k), \theta(k))|_{\mathcal{A}} \leq \beta_M(|(x(0), \theta(0))|_{\mathcal{A}}, k)$ for any $k \in \mathbb{Z}_{>0}$.

Proof: Since ϕ in Assumption 1 is convex and $\mathcal{X}$ is compact, it admits at least one minimum value on $\mathcal{X}$, which we denote ϕ^*, i.e. $\phi^* := \min_{x \in \mathcal{X}}\{\phi(x)\}$. Moreover, the set $\mathcal{X}^*$ of pure NE is $\{x \in \mathcal{X} : \phi(x) = \phi^*\}$ as ϕ is convex [4].

To prove the desired result, we apply [20, Theorem 1], which applies to system (10) as it is a particular case of [20, (1)] with empty set C. We consider the Lyapunov function candidate

$$V(x, \theta) := \phi(x) - \phi^*, \tag{12}$$

for any $x \in \mathcal{X}$ and $\theta \in \mathcal{N}$. The continuity of ϕ on $\mathcal{X}$ under Assumption 1 implies that $V(x, \theta)$ is also continuous. Since ϕ^* is the minimum value that ϕ can take on $\mathcal{X}$, we have $\phi(x) > \phi(x^*)$ for any $x \notin \mathcal{X}^*$, and $V(x, \theta) = 0$ if and only if $(x, \theta) \in \mathcal{A}$. Therefore, V is positive definite with respect to the compact set $\mathcal{A}$. Moreover, it is trivially radially unbounded as we work on the compact set $\mathcal{X}$. Hence, items 1) and 2) in [20] hold; recall that $C = \emptyset$ in our case.

Let $M = (M_1, \ldots, M_N) \in \mathbb{Z}_{>0}^N$, $x \in \mathcal{X}$ and $\theta \in \mathcal{N}$, and denote $x^+ = (x_1, \ldots, F_\theta^{M_\theta}(x), \ldots, x_N)$ and $\theta^+ \in \mathcal{N}$. We have shown in Theorem 1 that $\phi(x^+) \leq \phi(x)$ for all $x \in \mathcal{X}$, and so we have $V(x^+, \theta^+) \leq V(x, \theta)$ proving that [20, (3)] holds.

We now proceed by contradiction and suppose that there is a (complete) solution (x, θ) to (10)-(11) which keeps V constant and non-zero. Hence, there exists $c > 0$ such that for all $k \in \{k', k'+1, \ldots\}$, $V(x(k), \theta(k)) = c$. Since V is positive definite with respect to $\mathcal{A}$, as established above, this means that $x(k) \notin \mathcal{X}^*$ in view of the definition of $\mathcal{A}$. Now, we have shown in Theorem 1 that $x(k) \notin \mathcal{X}^*$ implies $\phi(x(k+K) < \phi(x(k))$. Therefore, $V(x(k), \theta(k))$ cannot be equal to c for all $\{k', k'+1, \ldots\}$: we have attained a contradiction. Consequently, no (complete) solution keeps V constant and non-zero. We then apply [20, Theorem 1] to

conclude that A is UGAS for system (10)-(11). ■

IV. Performance evaluation

In order to evaluate the performance of the proposed DAGD algorithm in terms of the computational and communication loads, we compare it with two other algorithms found in the literature: the classical asynchronous best response dynamics (ABRD) used in game theory literature [18] and the distributed synchronous gradient descent (DSGD) recently studied in [15]. In this section, we briefly recall these two algorithms and the assumptions required for their convergence to a pure NE, and then provide simulations to compare the computational and communication costs associated with each of them.

A. The ABRD and DSGD

Recall from [18], that the ABRD is given by, for any $k \in \mathbb{Z}_{>0}$,

$$\begin{aligned} x_{\theta(k)}(k+1) &\in f_{\theta(k)}(x_{-\theta(k)}(k)) \\ x_{-\theta(k)}(k+1) &= x_{-\theta(k)}(k) \end{aligned} \tag{13}$$

where $\theta(k) \in \mathcal{N}$ denotes the agent updating its action at time step k. It is shown in [21] that the ABRD always converges to a pure NE $x^* \in \mathcal{X}^*$ under Assumptions 1-4.

The asynchronous nature of the best response dynamics is necessary for convergence to the NE in general. However, while applying a single step of the gradient descent, it has been shown in [15] that even a synchronous algorithm applying the basic gradient descent can converge under appropriate assumptions to a (differential) NE. On the other hand, we are interested in applying the projected gradient descent and in convergence to pure NE. Therefore, in this subsection, we propose a modified version of the algorithm presented in [15], which converges to a pure NE under the following assumption.

Assumption 5: We consider

1) $\mathcal{G}$ is a weighted potential game with weights $w_i \in \mathbb{R}_{>0}$, i.e., there exists $\phi : \mathcal{X} \to \mathbb{R}_{\geq 0}$ such that
$$w_i(J_i(x_i, x_{-i}) - J_i(x_i', x_{-i})) = \phi(x_i, x_{-i}) - \phi(x_i', x_{-i}) \tag{14}$$
for any $x_i, x_i' \in \mathcal{X}_i$, $x_{-i} \in \mathcal{X}_{-i}$.
2) the potential function ϕ is convex and twice differentiable. We define $L := \sup_x\{||\nabla^2 \phi(x)||_2\}$ and assume that this is known to all players. □

In item 1) of Assumption 5, we require that $\mathcal{G}$ is a weighted potential game, which is a stronger version of Assumption 1 as weighted potential games are a special case of ordinal potential games. Item 2) of Assumption 5 requires ϕ to be convex and L to be known to all players, which is a stronger version of Assumptions 2-3. In Algorithm 3, we describe the DSGD which is a modification of the algorithm presented in [15]. Specifically, we work with the projected gradient descent which is more general than the basic gradient descent.

Corollary 1: Under Assumption 5, Algorithm 3 converges to a NE for any $x(0) \in \mathcal{X}$.

Algorithm 3: The DSGD applied by player i

Given $\gamma \in (0, L^{-1}]$;
$k \leftarrow 1$;
do
  Communicate $x_i(k)$ and receive $x_{-i}(k)$;
  $x_i(k+1) \leftarrow \hat{f}_i(x(k), w_i\gamma)$;
  $k \leftarrow k+1$;
while $x(k) \neq x(k-1)$;

Proof: This result is a refinement of Theorem 4.1 in [15] for convex potential games and projected gradient descent. Given $x(0)$, at any step $k \in \mathbb{Z}_{\geq 0}$, we have

$$x_i(k+1) = \hat{f}_i(x(k), w_i\gamma) \tag{15}$$

for all $i \in \mathcal{N}$ and thus

$$x(k+1) = \mathbf{P}_{\mathcal{X}}(x(k) - \gamma \nabla_x \phi(x(k)) \tag{16}$$

from item 2) of Assumption 5. This is simply a projected gradient descent of ϕ on the full state space $\mathcal{X}$. Since, we take $\gamma \leq L$ and ϕ is assumed to be convex, convergence to the global minimum is ensured [19]. ■

B. Stopping criteria and running time

In practice, as the algorithms may take an infinite time to converge to a NE x^*, we stop Algortithms 2 and 3 when the better responses do not improve the costs more than a certain threshold. For the purpose of performance evaluation, we look at a convex ϕ and we stop the algorithm at some iteration k^* such that

$$\phi(x(k^*)) - \phi^* \leq \epsilon, \tag{17}$$

where $\epsilon \in \mathbb{R}_{>0}$ is taken to be sufficiently small and ϕ^* is the smallest value taken by ϕ as in the proof of Theorem 2. As we have proven that the DAGD is UGAS in Theorem 2, we have k^* finite for any $\epsilon > 0$. We call the total number of iterations before (17) is met as k_A^* for the DAGD described in Algorithm 2 and k_S^* for DSGD described in Algorithm 3, for a given common initialization $x(0)$.

One of the main features of the proposed algorithm is to evaluate and tune the algorithm parameters such that the total running time can be reduced with respect to the ABRD. We denote by $\tau_P \in \mathbb{R}_{>0}$, the time required by the processor to compute one step of the gradient descent. We say that it takes $\tau_C \in \mathbb{R}_{>0}$ units of time for one agent to communicate its actions to all of its neighbors.

In asynchronous algorithms, we count the total number of gradient descent steps applied to evaluate the total computational time per iteration. We fix $M_i = M_1 \in \mathbb{Z}_{>0}$ for all $i \in \mathcal{N}$. Then, we evaluate the total running time as $T = k_A^*(M_1\tau_P + \tau_C)$.

On the other hand, in synchronous algorithms, the computations by each agent are done in parallel and so each iteration of the synchronous algorithm will take τ_P computation time. However, as all agents must communicate at each iteration, this results in a total communication time

per iteration of $N\tau_C$ assuming that the agents can not communicate simultaneously. This results in a total running time of $T = k_S^*(\tau_P + N\tau_C)$.

C. Numerical example

We apply the results of Section III to the distributed formation planning of N agents, with each agent $i \in \mathcal{N}$ representing a mobile planar system. The initial position of each agent is given by $s_i \in [-10, 10]^2$. An agent updates its position based on local information obtained by communicating with its neighbors which belong to the set N_i. Each agent shares its target position $x_i \in \mathcal{X}_i$, $\mathcal{X}_i := [-10, 10]^2$ with all the neighboring agents in N_i. We consider an undirected graph, i.e., if $j \in N_i$ then $i \in N_j$ for any $i, j \in \mathcal{N}$. The agents want to make a formation such that $x_i - x_j$ is d_{ij} with $d_{ji} = -d_{ij}$ and $d_{ij} \in \mathcal{X}_i$ for all $j \in N_i$ and all $i \in \mathcal{N}$. However, each agent also wants to minimize the energy consumed by moving from s_i to x_i. The cost function of each agent is taken as

$$J_i(x) = r_i(x_i - s_i)^2 + \sum_{j \in N_i} (x_i - x_j - d_{ij})^2. \tag{18}$$

The first term can be seen as the control effort or energy term, with $r_i \geq 0$ being the weight allocated to this effort. The second term is the cost associated with consensus and this term becomes 0 when agent i is in perfect formation with its neighbors. Now, we look at the game $\mathcal{G}$ formed by the N agents, with x_i as actions and J_i as cost.

Proposition 1: The game $\mathcal{G}$ is an exact potential game and the associated potential function ϕ is twice differentiable and convex. □

Proof: We define, for any $x \in \mathcal{X}$ and any given $s_i, d_{ij} \in \mathbb{R}^2$ for all $i, j \in \mathcal{N}$,

$$\phi(x) := \sum_{i=1}^{N} \left(r(x_i - s_i)^2 + \frac{1}{2} \sum_{j \in N_i} (x_i - x_j - d_{ij})^2 \right). \tag{19}$$

We observe that ϕ is an exact potential function, i.e., for any $x \in \mathcal{X}, x_i' \in \mathcal{X}_i$,

$$\begin{aligned} \phi(x_i, x_{-i}) - \phi(x_i', x_{-i}) &= r(x_i - s_i)^2 - r(x_i' - s_i)^2 \\ &+ \textstyle\sum_{j \in N_i} \frac{(x_i - x_j - d_{ij})^2 - (x_i' - x_j - d_{ij})^2}{2} \\ &+ \textstyle\sum_{j \in N_i} \frac{(x_j - x_i - d_{ji})^2 - (x_j - x_i' - d_{ji})^2}{2}. \end{aligned} \tag{20}$$

This holds as all the terms without x_i get cancelled since $i \in N_j$ iff $j \in N_i$. Additionally, since $d_{ij} = -d_{ji}$, we have $(x_i - x_j - d_{ij})^2 = (x_j - x_i - d_{ji})^2$ and so

$$\phi(x_i, x_{-i}) - \phi(x_i', x_{-i}) = J_i(x_i, x_{-i}) - J_i(x_i', x_{-i}). \tag{21}$$

Note that all the terms are quadratic in x and so ϕ is convex in x and J_i is convex with respect to x_i. ■

Thus, we have ensured that Assumptions 1-2 hold. We pick $\gamma_i = 0.1$ for all $i \in \mathcal{N}$ and $\theta(k) := 1 + (k \mod N)$ satisfying Assumptions 3-4. Next, we fix $\tau_P = 0.001$ seconds, $\epsilon = 0.001$ and take two cases corresponding to slow and fast communications, Case A: $\tau_C = 0.01$ and Case B: $\tau_C = 0.0001$ seconds. In practice, this could correspond to the agents being connected over a wireless network or wired network respectively. We plot the running time T of the DAGD vs various values of M_1 for the two cases in Figure 1. When the communication time is high, picking a larger M_1 results in a smaller running time as each iteration costs more time (recall that we fix $M_i = M_1$ for all $i \in \mathcal{N}$).

We next compare the performance of the DAGD with the ABRD and the DSGD in Table I by fixing $M_1 = 6$ for the DAGD. The ABRD is implemented by selecting a large enough number of steps of the gradient descent, specifically $M_1 = 20$. Since the ABRD is a special case of the DAGD, a suitable choice of M will always outperform or match the ABRD in terms of the running time T. On the other hand, the DSGD outperforms the DAGD in terms of the computational steps (due to parallel computing) but takes more communication steps. Consequently, in Case A, we notice that the running time T of the DSGD is 5.8 times that of the DAGD. We also see that the DAGD is always faster than the ABRD.

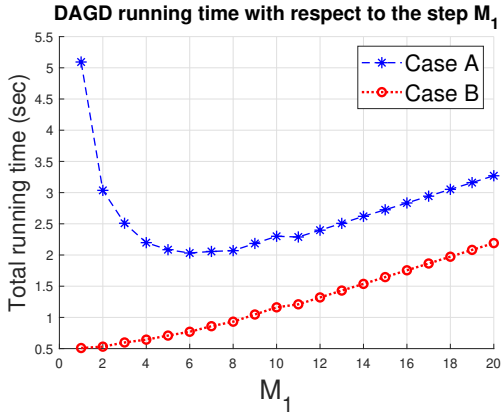

Fig. 1: Running time for the DAGD algorithm

Algorithm:	**DAGD**	**DSGD**	**ABRD**
Iterations k^*	127	193	109
Computation steps	762	193	2180
Communication count	127	1158	109
Running time T, Case A	2.03	11.77	3.27
Running time T, Case B	0.77	0.3	2.19

TABLE I: Comparison of DAGD using $M_1 = 6$ with the ABRD and DSGD.

V. Conclusion

In this work, we have presented a distributed asynchronous gradient descent (DAGD) algorithm in which each agent applies an arbitrary number of steps of the gradient descent algorithm with respect to a local cost function. We demonstrate convergence of this algorithm to a pure NE for ordinal potential games when the cost functions of each agent are quasi-convex with respect to its action. When the number of steps applied per iteration approaches infinity, this algorithm

corresponds to the asynchronous best response dynamics (ABRD) which is often applied in game theory. In our numerical simulations, we compare the performance, characterized by overall computation time and communication instants, of the proposed algorithm with the ABRD and a synchronous scheme (the DSGD) based on a recent publication [15]. We demonstrate that picking a suitable number of iterations can result in a suitable trade-off between the total computation steps and the number of communications required. This results in a smaller running time of the proposed algorithm when the communication network is slow.

REFERENCES

[1] R.S. Gibbons. *Game theory for applied economists*. Princeton University Press, 1992.

[2] S. Lasaulce and H. Tembine. *Game Theory and Learning for Wireless Networks : Fundamentals and Applications*. Academic Press, 2011.

[3] T. Başar and G.J. Olsder. *Dynamic noncooperative game theory*. SIAM, 1998.

[4] D. Monderer and L.S. Shapley. Potential games. *Games and economic behavior*, 14(1):124–143, 1996.

[5] R.R. Negenborn and J.M. Maestre. Distributed model predictive control: An overview and roadmap of future research opportunities. *IEEE Control Systems Magazine*, 34(4):87–97, 2014.

[6] H.P. Young. *Strategic learning and its limits*. OUP Oxford, 2004.

[7] A.C. Chapman, A. Rogers, D. Leslie, and N.R. Jennings. A unifying framework for iterative approximate best–response algorithms for distributed constraint optimisation problems. *The Knowledge Engineering Review*, 2011.

[8] I. Bistritz and A. Leshem. Approximate best-response dynamics in random interference games. *IEEE Transactions on Automatic Control*, 63(6):1549–1562, 2017.

[9] Y. Vorobeychik and M.P. Wellman. Stochastic search methods for nash equilibrium approximation in simulation-based games. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, pages 1055–1062. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[10] M. Dindoš and C. Mezzetti. Better-reply dynamics and global convergence to nash equilibrium in aggregative games. *Games and Economic Behavior*, 54(2):261–292, 2006.

[11] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[12] E. Ozfatura, D. Gündüz, and S. Ulukus. Speeding up distributed gradient descent by utilizing non-persistent stragglers. In *IEEE International Symposium on Information Theory (ISIT), Paris*, pages 2729–2733. IEEE, 2019.

[13] R. Gemulla, E. Nijkamp, P.J. Haas, and Y. Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, San Diego*, pages 69–77, 2011.

[14] B. Awerbuch and R. Khandekar. Stateless distributed gradient descent for positive linear programs. *SIAM Journal on Computing*, 38(6):2468–2486, 2009.

[15] E. Mazumdar, L.J. Ratliff, and S.S. Sastry. On gradient-based learning in continuous games. *SIAM Journal on Mathematics of Data Science*, 2(1):103–131, 2020.

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

[17] J. Nash. Non-cooperative games. *Annals of Mathematics*, pages 286–295, 1951.

[18] N. Nisan, M. Schapira, and A. Zohar. Asynchronous best-reply dynamics. In *International Workshop on Internet and Network Economics*, pages 531–538. Springer, 2008.

[19] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[20] A. Seuret, C. Prieur, S. Tarbouriech, A.R. Teel, and L. Zaccarian. A nonsmooth hybrid invariance principle applied to robust event-triggered design. *IEEE Trans. on Aut. Control*, 64(5):2061–2068, 2018.

[21] V. Boucher. Selecting equilibria using best-response dynamics. *Economics Bulletin*, 17(4):2728– 2734, 2017.