# STAY-ON-THE-RIDGE: GUARANTEED CONVERGENCE TO LOCAL MINIMAX EQUILIBRIUM IN NONCONVEX-NONCONCAVE GAMES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Min-max optimization problems involving nonconvex-nonconcave objectives have found important applications in adversarial training and other multi-agent learning settings. Yet, no known gradient descent-based method is guaranteed to converge to (even local notions of) min-max equilibrium in the nonconvex-nonconcave setting. For all known methods, there exist relatively simple objectives for which they cycle or exhibit other undesirable behavior different from converging to a point, let alone to some game-theoretically meaningful one (Vlatakis-Gkaragkounis et al., 2019; Hsieh et al., 2021). The only known convergence guarantees hold under the strong assumption that the initialization is very close to a local min-max equilibrium (Wang et al., 2019). Moreover, the afore-described challenges are not just theoretical curiosities. All known methods are unstable in practice, even in simple settings.

We propose the first method that is guaranteed to converge to a local min-max equilibrium for smooth nonconvex-nonconcave objectives. Our method is second-order and provably escapes limit cycles as long as it is initialized at an easy-to-find initial point. Both the definition of our method and its convergence analysis are motivated by the topological nature of the problem. In particular, our method is not designed to decrease some potential function, such as the distance of its iterate from the set of local min-max equilibria or the projected gradient of the objective, but is designed to satisfy a topological property that guarantees the avoidance of cycles and implies its convergence.

## 1 INTRODUCTION

Min-max optimization lies at the foundations of Game Theory (von Neumann, 1928), Convex Optimization (Dantzig, 1951; Adler, 2013) and Online Learning (Blackwell, 1956; Hannan, 1957; Cesa-Bianchi & Lugosi, 2006), and has found many applications in theoretical and applied fields including, more recently, in adversarial training and other multi-agent learning problems (Goodfellow et al., 2014; Madry et al., 2018; Zhang et al., 2019). In its general form, it can be written as

$$\min_{\theta\in\Theta}\max_{\omega\in\Omega} f(\theta,\omega), \tag{1}$$

where $\Theta$ and $\Omega$ are convex subsets of the Euclidean space, and $f$ is continuous.

Eq. (1) can be viewed as a model of a sequential-move game wherein a player who is interested in minimizing $f$ chooses $\theta$ first, and then a player who is interested in maximizing $f$ chooses $\omega$ after seeing $\theta$. A solution to (1) corresponds to a Nash equilibrium of this sequential-move game.

We may also study the simultaneous-move game with the same objective $f$ wherein the minimizing player and the maximizing player choose $\theta$ and $\omega$ simultaneously. The Nash equilibrium of the simultaneous-move game, also called a *min-max equilibrium*, is a pair $(\theta^\star,\omega^\star)\in\Theta\times\Omega$ such that

$$f(\theta^\star,\omega^\star)\le f(\theta,\omega^\star),\ \text{ for all } \theta\in\Theta \quad\text{and}\quad f(\theta^\star,\omega^\star)\ge f(\theta^\star,\omega),\ \text{ for all } \omega\in\Omega. \tag{2}$$

It is easy to see that a Nash equilibrium of the simultaneous-move game also constitutes a Nash equilibrium of the sequential-move game, but the converse need not be true. Here, we focus on

solving the (harder) simultaneous-move game. In particular, we study the existence of *dynamics* which converge to solutions of the simultaneous-move game, namely the existence of methods that make incremental updates to a pair $(\theta_t, \omega_t)$ so as the sequence $(\theta_t, \omega_t)$ converges, as $t \to \infty$, to some $(\theta^*, \omega^*)$ satisfying (2) or some relaxation of it.

This problem has been extensively studied in the special case where $\Theta$ and $\Omega$ are convex and compact and $f$ is convex-concave — i.e. convex in $\theta$ for all $\omega$ and concave in $\omega$ for all $\theta$. In this case, the set of Nash equilibria of the simultaneous-move game is equal to the set of Nash equilibria of the sequential-move game, and these sets are non-empty and convex (von Neumann, 1928). Even in this simple setting, however, many natural dynamics surprisingly fail to converge: *gradient descent-ascent*, as well as various continuous-time versions of *follow-the-regularized-leader*, not only fail to converge to a min-max equilibrium, even for very simple objectives, but may even exhibit chaotic behavior (Mertikopoulos et al., 2018; Vlatakis-Gkaragkounis et al., 2019; Hsieh et al., 2021). In order to circumvent these negative results, an extensive line of work has introduced other algorithms, such as *extragradient* (Korpelevich, 1976) and *optimistic gradient descent* (Popov, 1980), which exhibit last-iterate convergence to the set of min-max equilibria in this setting; see e.g. Daskalakis et al. (2018); Daskalakis & Panageas (2018); Mazumdar & Ratliff (2018); Rafique et al. (2018); Hamedani & Aybat (2018); Adolphs et al. (2019); Daskalakis & Panageas (2019); Liang & Stokes (2019); Gidel et al. (2019); Mokhtari et al. (2019); Abernethy et al. (2019); Golowich et al. (2020b;a). Alternatively, one may take advantage of the convexity of the problem, which implies that several no-regret learning procedures, such as online gradient descent, exhibit *average*-iterate convergence to the set of min-max equilibria (Cesa-Bianchi & Lugosi, 2006; Shalev-Shwartz, 2012; Bubeck & Cesa-Bianchi, 2012; Shalev-Shwartz & Ben-David, 2014; Hazan, 2016). Moreover, Lin et al. (2020); Kong & Monteiro (2021); Ostrovskii et al. (2021) show that convexity with respect to one of the two players is enough to design algorithms that exhibit average-iterate convergence to min-max equilibria.

Our focus in this paper is on the more general case where $f$ is not assumed to be convex-concave, i.e. it may fail to be convex in $\theta$ for all $\omega$, or may fail to be concave in $\omega$ for all $\theta$, or both. We call this general setting where neither convexity with respect to $\theta$ nor concavity with respect to $\omega$ is assumed, the *nonconvex-nonconcave* setting. This setting presents some substantial challenges. First, min-max equilibria are *not* guaranteed to exist, i.e. for general objectives there may be no $(\theta^\star, \omega^\star)$ satisfying (2); this happens even in very simple cases, e.g. when $\Theta = \Omega = [0, 1]$ and $f(\theta, \omega) = (\theta - \omega)^2$. Second, it is NP-hard to determine whether a min-max equilibrium exists (Daskalakis et al., 2021) and, as is easy to see, it is also NP-hard to compute Nash equilibria of the sequential-move game (which do exist under compactness of the constraint sets). For these reasons, the optimization literature has targeted the computation of local and/or approximate solutions in this setting (Daskalakis & Panageas, 2018; Mazumdar & Ratliff, 2018; Jin et al., 2019; Wang et al., 2019; Daskalakis et al., 2021; Mangoubi & Vishnoi, 2021). This is the approach we also take in this paper, targeting the computation of $(\varepsilon, \delta)$-*local min-max equilibria*, which were proposed in Daskalakis et al. (2021). These are approximate and local Nash equilibria of the simultaneous-move game, defined as feasible points $(\theta^\star, \omega^\star)$ which satisfy a relaxed and local version of (2), namely:

$$f(\theta^\star, \omega^\star) < f(\theta, \omega^\star) + \varepsilon, \text{ for all } \theta \in \Theta \text{ such that } \|\theta - \theta^\star\| \leq \delta; \tag{3}$$

$$f(\theta^\star, \omega^\star) > f(\theta^\star, \omega) - \varepsilon, \text{ for all } \omega \in \Omega \text{ such that } \|\omega - \omega^\star\| \leq \delta. \tag{4}$$

Besides being a natural concept of local, approximate min-max equilibrium, an attractive feature of $(\varepsilon, \delta)$-local min-max equilibria is that they are guaranteed to exist when $f$ is $\Lambda$-smooth and the locality parameter, $\delta$, is chosen small enough in terms of the smoothness, $\Lambda$, and the approximation parameter, $\varepsilon$, namely whenever $\delta \leq \sqrt{\frac{2\varepsilon}{\Lambda}}$. Indeed, in this regime of parameters the $(\varepsilon, \delta)$-local min-max equilibria are in correspondence with the approximate fixed points of the *Projected Gradient Descent/Ascent* dynamics. Thus, the existence of the former can be established by invoking Brouwer's fixed point theorem to establish the existence of the latter. (Theorem 5.1 of Daskalakis et al. (2020)).

There are a number of existing approaches which would be natural to use to find a solution $(\theta^\star, \omega^\star)$ satisfying (3) and (4), but all run into significant obstacles. First, the idea of averaging, which can be leveraged in the convex-concave setting to obtain provable guarantees for otherwise chaotic algorithms, such as online gradient descent, no longer works, as it critically uses Jensen's inequality which needs convexity/concavity. On the other hand, negative results abound for last-iterate convergence: Hsieh et al. (2021) show that a variety of zeroth, first, and second order methods may converge to a limit cycle, even in simple settings. Vlatakis-Gkaragkounis et al. (2019) study a particular class of nonconvex-nonconcave games and show that continuous-time gradient descent-ascent (GDA) exhibits

*recurrent* behavior. Furthermore, common variants of gradient descent-ascent, such as optmistic GDA (OGDA) or extra-gradient (EG), may be unstable even in the proximity of local min-max equilibria, or converge to fixed points that are not local min-max equilibria (Daskalakis & Panageas, 2018; Jin et al., 2019). While there do exist algorithms, such as FOLLOW-THE-RIDGE proposed by Wang et al. (2019), which provably exhibit *local convergence* to a (relaxation of) local min-max equilibrium, these algorithms do not enjoy global convergence guarantees, and no algorithm is known with guaranteed convergence to a local min-max equilibrium.

These negative theoretical results are consistent with the practical experience with min-maximization of nonconvex-nonconcave objectives, which is rife with frustration as well. A common experience is that the training dynamics of first-order methods are unstable, oscillatory or divergent, and the quality of the points encountered in the course of training can be poor; see e.g. Goodfellow (2016); Metz et al. (2016); Daskalakis et al. (2018); Mescheder et al. (2018); Daskalakis & Panageas (2018); Mazumdar & Ratliff (2018); Mertikopoulos et al. (2018); Adolphs et al. (2019). In light of the failure of essentially all known algorithms to guarantee convergence, even asymptotically, to local min-max equilibria, we ask the following question: *Is there an algorithm which is guaranteed to converge to a local min-max equilibrium in the nonconvex-nonconcave setting (Wang et al., 2019)?*

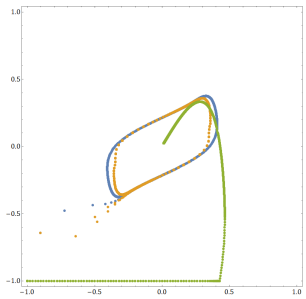


Figure 1: The **blue**, **orange** and **green** dots track the trajectories of **GDA**, **OGDA** and **STON'R** when $f(\theta,\omega) = (4\theta^2 - (\omega - 3\theta + \frac{\theta^3}{20})^2 - \frac{\omega^4}{10})\exp(-\frac{\theta^2+\omega^2}{100})$, $\theta,\omega \in [-1,1]$ and the dynamics are initialized at the left-most dots of the corresponding color. The **blue** and **orange** trajectories converge to a limit cycle while the **green** trajectory converges to an equilibrium.

**Our Contribution.** In this work we answer the above question in the affirmative: we propose a second-order method that is guaranteed to converge to a local min-max equilibrium (Theorem 1). Our algorithm, called STAY-ON-THE-RIDGE or STON'R, has some similarity to FOLLOW-THE-RIDGE or FTR, which only converges locally and to a relaxed notion of min-max equilibrium. Both the structure of our algorithm and its global convergence analysis are motivated by the topological nature of the problem, as established by Daskalakis et al. (2021) who showed that the problem is computationally (and mathematically) equivalent to Brouwer fixed point computation. In particular, the structure and analysis of STON'R are not based on a potential function argument but on a *parity argument*, akin to the combinatorial argument used to prove the existence of Brouwer fixed points.

In Figure 1, we give an example where Gradient Descent/Ascent and Optimistic GDA, whose trajectories are shown in **blue** and **orange** respectively, converge to a limit cycle while STON'R, whose trajectory is shown in **green**, successfully converges to a local min-max equilibrium of the objective. An example min-max optimization problem for which FTR fails to converge globally is when the objective is $f(\theta,\omega) = -\theta\omega - \frac{1}{20}\omega^2 + \frac{2}{20}S((\theta^2+\omega^2)/2)\cdot\omega^2$, where $\theta,\omega \in [-1,1]$ and $S$ is the smooth-step function $S(\theta) = 3\theta^2 - 2\theta^3$, for $\theta \in [0,1]$, $S(\theta) = 0$ for $\theta \le 0$ and $S(\theta) = 1$ for $\theta \ge 1$. In this case, STON'R converges to a local min-max equilibrium while FTR algorithm gets stuck at the point $\theta = -1, \omega = -1$. We refer to Appendix A for more simulations and detailed comparison of GDA, OGDA, EG, FTR, and STON'R in these two examples.

## 2 SOLUTION CONCEPT

First, a standard notation that we use is this: if $m$ is a natural number then $[m] = \{1,\dots,m\}$. Our main goal in this paper is to design optimization methods that have guaranteed convergence to local min-max equilibria of smooth objectives in the nonconvex-nonconcave setting. For exposition simplicity, we formulate this problem in the language of non-monotone variational inequalities.

**Variational Inequalities (VI).** For $K \subseteq \mathbb{R}^n$, consider a continuous map $V : K \to \mathbb{R}^n$. We say that $x \in K$ is a solution of the variational inequality $\mathrm{VI}(V,K)$ iff: $V(x)^\top \cdot (x-y) \le 0$ for all $y \in K$.

It is well known that finding local min-max equilibria of smooth objectives can be expressed as a non-monotone VI problem. Specifically, consider the min-max optimization problem (1), take

$K = \Theta \times \Omega$ and simplify notation by using $x \in K$ to denote points $(\theta, \omega) \in K$. Call the subset of coordinates of $x$ identified with $\theta$ the "*minimizing* coordinates" and the subset of coordinates of $x$ identified with $\omega$ the "*maximizing* coordinates." Then define $V : K \to \mathbb{R}^n$ as follows:

$$\text{For } j \in [n]\text{: set } V_j(x) := -\frac{\partial f(x)}{\partial x_j}, \text{ if } j \text{ is minimizing, and } V_j(x) := \frac{\partial f(x)}{\partial x_j}, \text{ otherwise.}$$

With these definitions, it is easy to see that computing $(\varepsilon, \delta)$-local min-max equilibria of smooth objectives, i.e. points satisfying (3) and (4), can be reduced to finding solutions to $\mathrm{VI}(V, K)$. (In fact, finding even an approximate VI solution $x$ satisfying $V(x)^\top (x - y) \geq -\alpha, \forall y \in K$, would suffice as long as $\alpha > 0$ is small enough. For more details see Theorem 5.1 of Daskalakis et al. (2020).)

In view of the above, for the remainder of the paper we focus on solving non-monotone variational inequality problems. For simplicity of exposition throughout we will take our constraint set to be $K = [0, 1]^n$. In this case there is a simple characterization of the solutions to $\mathrm{VI}(V, K)$.

**Definition 1.** *We call a coordinate $i$ at point $x \in [0, 1]^n$,*

1. ***zero-satisfied** if $V_i(x) = 0$,*
2. ***boundary-satisfied** if $(V_i(x) \leq 0$ and $x_i = 0)$ or $(V_i(x) \geq 0$ and $x_i = 1)$,*
3. ***satisfied** if $i$ is zero- or boundary- satisfied and **unsatisfied** if it is not satisfied.*

**Lemma 1** (Proof in Appendix D)**.** *$x$ is a solution of $\mathrm{VI}(V, [0, 1]^n)$ iff $j$ is satisfied at $x$, $\forall j \in [n]$.*

Finally, in the rest of the paper we make the following assumptions for $V$:

- ($\Lambda$-Lipschitz) $\|V(x) - V(y)\|_2 \leq \Lambda \cdot \|x - y\|_2$, for all $x, y \in [0, 1]^n$
- ($L$-smooth) $\|J(x) - J(y)\|_F \leq L \cdot \|x - y\|_2$, for all $x, y \in [0, 1]^n$

where $J$ is the Jacobian of V, and $\|A\|_F$ denotes the Frobenious norm of the matrix $A$.

## 3 STAY-ON-THE-RIDGE: HIGH-LEVEL DESCRIPTION

In this section we describe our algorithm and discuss the main design ideas leading to its convergence properties presented in Section 5. As explained in the previous section, our goal is to find a point $x$ such that every coordinate $i \in [n]$ is satisfied at $x$ according to the Definition 1.

Our algorithm is initialized at point $x(0) = (0, \ldots, 0)$ where a number of coordinates may be unsatisfied. The goal of the algorithm is to satisfy all unsatisfied coordinates one-by-one in lexicographic order (although, as we will see, coordinates may go from being satisfied to being unsatisfied in the course of the algorithm). We say that our algorithm "starts epoch $i$ at point $x$" iff all coordinates $\leq i - 1$ are satisfied at $x$ and the algorithm's immediate goal is to find a point $x' \neq x$ that satisfies all coordinates $\leq i$, namely:

$$\text{Goal of epoch } i\text{, starting at point } x\text{: find } x' \neq x \text{ satisfying all coordinates} \leq i.$$

We now describe how the algorithm tries to meet the afore-described goal. So let us assume that, at time $t$, our algorithm starts epoch $i$ at point $x(t)$. Postponing full details to Section 5.1, where we describe our algorithm in detail, for simplicity of exposition let us assume in this section that, at $x(t)$, all coordinates $\leq i - 1$ are zero-satisfied, i.e. $V_j(x(t)) = 0$ for all $j \leq i - 1$. To achieve the goal of epoch $i$ starting at $x(t)$, our algorithm will *try* to find a point $x' \neq x(t)$ where all coordinates $\leq i - 1$ remain zero-satisfied and coordinate $i$ is also satisfied as follows:

- First, it will try to find such a point in the connected subset $S^i(x(t)) \subseteq [0, 1]^n$ that contains $x(t)$ and all points $z$ satisfying the following: (a) all coordinates $\leq i - 1$ are zero-satisfied at $z$, and (b) for all $j \geq i + 1$, $z_j = x_j(t)$.
- Next, let us describe how our algorithm navigates $S^i(x(t))$ in the hopes of identifying a point $x' \neq x(t)$ where all coordinates $\leq i$ are satisfied. A natural approach is to run a continuous-time dynamics $\{z(\tau)\}_{\tau \geq 0}$ that is initialized at $z(0) = x(t)$ and moves inside $S^i(x(t))$. What are possible directions of movement for such dynamics so that it stays within $S^i(x(t))$? If the dynamics is at some point $z \in S^i(x(t))$, it will remain in this set if it moves, infinitessimally, in a unit direction $d$ satisfying the following constraints:

1. $d_j = 0$, for all $j \geq i+1$; */* this is so that (b) in the definition of $S^i(x(t))$ is maintained */*
2. $(\nabla V_j(z))^\top \cdot d = 0$, for all $j \in \{1, \ldots, j-1\}$. */* this is so that (a) is maintained */*

Notice that 1 and 2 specify $n-1$ constraints on $n$ variables. We will place mild assumptions on $f$ so that there is a unique, up to a sign flip, unit direction satisfying these constraints. (Specifically see Assumption 1 in Section 5.2, where our main result is formally stated.) Moreover, we will specify a way to break ties so that we choose one of the two unit directions satisfying our constraints. (Specifically this is done in part 3 of Definition 2 in Section 5.1.) Let us denote by $D^i(z)$ the unit direction that our tie-breaking rule selects at $z$.

- With the above choices, the continuous-time dynamics $\dot{z}(\tau) = D^i(z(\tau))$, initialized at $z(0) = x(t)$, is well-defined. We follow this dynamics until the earliest time that one of the following happens (if both events happen at the same time we will say that the good event happened):
  - (Good Event): the dynamics stops at a point $x' \neq x(t)$ where coordinate $i$ is satisfied;
  - (Bad Event): the dynamics stops at a point $x'$ lying on the boundary of $[0,1]^n$ (and if it were to continue it would violate the constraints).

So we have described what our algorithm does if, at time $t$, it starts epoch $i$ at some point $x(t)$. Suppose $x'$ is the point where the continuous-time dynamics executed during epoch $i$ terminates. If the good event happened, coordinate $i$ is satisfied at $x'$, and our algorithm starts epoch $i+1$ at $x'$. If the bad event happened, our algorithm will in fact *start epoch* $i-1$ at point $x'$. What does this mean? That it will run the continuous-time dynamics corresponding to epoch $i-1$ on the set $S^{i-1}(x')$ starting at $x'$ in order to find some point $x'' \neq x'$ where all coordinates $\leq i-1$ are satisfied. It may fail to do this, in which case it will start epoch $i-2$ next. Or it may succeed, in which case, it will start epoch $i$, and so on so forth until (as we will show!) all coordinates will be satisfied. The high-level pseudocode of our algorithm is given below.

**Dynamics 1** STay-ON-the-Ridge (STON'R) — High-Level Description

```
Initially x(0) ← (0, ..., 0), i ← 1, t ← 0.
while x(t) is not a VI solution do
    Initialize epoch i's continuous-time dynamics, ż(τ) = D^i(z(τ)), at z(0) = x(t).
    while exit condition of this dynamics has not been reached do
        Execute ż(τ) = D^i(z(τ)) forward in time.
    end while
    Set x(t + τ_exit) = z(τ_exit) (where τ_exit is time exit condition was met).
    if x(t + τ_exit) ≠ x(t) and coordinate i is satisfied at x(t + τ_exit) then
        Update the epoch i ← i + 1.
    else
        (Bad event happened so) move to the previous epoch i ← i − 1.
    end if
    Set t ← t + τ_exit.
end while
return x(t)
```

At this point we have described an algorithm that explores the space in a natural way in its effort to satisfy coordinates, but it is unclear why it would succeed in eventually satisfying all of them, how it would escape cycles, and how it would not get stuck at non-equilibrium points. Importantly, there is no quantity that seems to be consistently improving during the execution of the algorithm. For example, the number of satisfied coordinates might decrease during the algorithm's execution.

> How we can show this algorithm converges since no quantity seems to be consistently improving during its execution?

To show the convergence of our algorithm we need to use a different kind of argument than the classical arguments used in optimization which are based on some quantity improving. In particular, we use a topological argument that we describe in Section 4. Before that we give an example that illustrates the execution of our algorithm in a simple 2-dimensional setting.

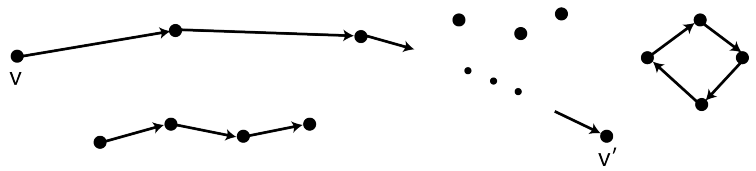


Figure 2: A directed graph whose nodes have in-degree and out-degree at most 1 is a collection of directed paths, directed cycles, and isolated nodes. Hence, if a node $v$ has in-degree 0 and out-degree 1 then it has to be the start of a directed path that must end at a node $v'$ after a finite number of steps.

## 4 A Topological Argument of Convergence

As discussed in Section 3, there seems to be no clear potential function that decreases in the course of our algorithm's execution, which we could track to show that it converges. Indeed, even the number of satisfied coordinates might decrease in the course the algorithm's execution as we explained in Section 3. So how we can show that our algorithm converges?

Our main idea is to use topological arguments that have been successfully employed to show the convergence of other equilibrium computation algorithms. In the celebrated Lemke & Howson (1964) algorithm, e.g., the following argument is used to prove the algorithm's convergence.

**Lemma 2.** *Let $G = (N, E)$ be a directed graph such that every node has in-degree at most 1 and out-degree at most 1. If there exists some node $v \in N$ with in-degree 0 and out-degree 1, then there is unique directed path starting at $v$ and ending at some $v' \in N$ that has in-degree 1 and out-degree 0.*

The proof of Lemma 2 is straightforward, as Figure 2 illustrates. The lemma suggests the following recipe for proving the convergence of some deterministic, iterative algorithm, with update rule $v_{t+1} \leftarrow F(v_t)$, whose iterates lie in a finite set $N$:

1. Define a directed graph $G$ whose vertex set is $V$ and edge set is $E = \{(u, v) \mid u \neq v \text{ and } v = F(u)\}$, i.e. there is a directed edge from $u$ to $v$ iff $v$ is different from $u$ and $v$ is reached after one iteration of the algorithm starting at $u$.
2. Argue that every vertex of $G$ has in-degree $\leq 1$. It is clear that every vertex has out-degree $\leq 1$.
3. Show that the algorithm can be initialized at some $v_0$ that has in-degree 0 and out-degree 1.
4. Employ Lemma 2 to argue that if the algorithm is initialized at $v_0$ it must, eventually, arrive at some node $v_{\text{end}}$ whose out-degree is 0. Out-degree 0 means that $v_{\text{end}} = F(v_{\text{end}})$.
5. The above prove that if the algorithm starts at $v_0$ it is guaranteed to converge.

Having this topological argument in place we are ready to formally describe our algorithm and argue its convergence. In the course of our description we will be sure to specify a finite set of points $V$ that will act as the nodes of the finite graph that we will construct to employ the above convergence argument. Intuitively, these are all the points at which our algorithm can possibly start an epoch. The map $F(\cdot)$ that we use to construct our graph is the outcome of the continuous-time process that our algorithm execute when it starts an epoch at such a point.

## 5 Detailed Description of STON'R and Main Result

We provide a formal description of our algorithm, state our main convergence theorem, and provide the main components of its proof building on the ideas from Section 4.

### 5.1 STON'R: Detailed Description

We provide a detailed description of our algorithm, building on the framework from Section 3. To simplify our exposition in that section, we only described the behavior of the algorithm when it starts an epoch at some $x$ where coordinates $\leq i-1$ are zero-satisfied and its goal is to identify some $x' \neq x$ at which coordinates $\leq i$ are satisfied. To achieve this goal the algorithm executed a continuous-time dynamics constrained by keeping all coordinates $\leq i - 1$ zero-satisfied. However, in the course

of its execution the algorithm might be hitting the boundary in its effort to satisfy coordinates. So, in the general case, when it starts a new epoch, some coordinates will be zero-satisfied and some will be boundary-satisfied. As such, what the algorithm will do in the general case during some epoch is execute a continuous-time dynamics constrained by keeping the zero-satisfied coordinates zero-satisfied and the boundary-satisfied coordinates at the right boundary.

More precisely, the epochs of our algorithm are, in fact, indexed not only by some coordinate $i$ but also by a subset of coordinates $S \subseteq [i-1]$ that are zero-satisfied at the point $x$ where the epoch starts. The goal of the epoch is the following.

> Goal of epoch $(i, S)$, starting at point $x$ (where $S \subseteq [i-1]$, coordinates in $S$ are zero-satisfied and coordinates in $[i-1] \setminus S$ are boundary-satisfied): find $x' \neq x$ where all coordinates $\leq i$ are satisfied, all coordinates in $S$ are zero-satisfied and all coordinates in $[i-1] \setminus S$ are boundary-satisfied.

As in our high-level description in Section 3, epoch $(i, S)$ starting at $x$ might achieve its goal or end before it achieves its goal. In both cases, a new epoch will start. Now, how does the algorithm try to achieve its goal in some epoch? Similar to the special case discussed in Section 3, in the general case considered here the algorithm will execute a continuous-time dynamics that maintains all the coordinates $j \in S$ zero-satisfied, all the coordinates $j \in [i-1] \setminus S$ boundary-satisfied, and leaves all coordinates $[n] \setminus (\{i\} \cup S)$ unchanged. The following definition captures the tangent unit vector of the curve that this continuous-time dynamics travels.

**Definition 2.** *Given $i \in [n]$, a set of coordinates $S = \{s_1, \ldots, s_m\} \subseteq [i-1]$, and some point $x$, we say that a unit vector $d \in \mathbb{R}^n$ is* admissible *iff it satisfies the following:*

1. $d_j = 0$, *for all* $j \notin S \cup \{i\}$.
2. $\nabla V_j(x)^\top \cdot d = 0$, *for all* $j \in S$.
3. *The sign of* $\begin{vmatrix} \frac{\partial V_{s_1}(x)}{\partial x_{s_1}} & \frac{\partial V_{s_2}(x)}{\partial x_{s_1}} & \cdots & \frac{\partial V_{s_m}(x)}{\partial x_{s_1}} & d_{s_1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial V_{s_1}(x)}{\partial x_{s_m}} & \frac{\partial V_{s_2}(x)}{\partial x_{s_m}} & \cdots & \frac{\partial V_{s_m}(x)}{\partial x_{s_m}} & d_{s_m} \\ \frac{\partial V_{s_1}(x)}{\partial x_i} & \frac{\partial V_{s_2}(x)}{\partial x_i} & \cdots & \frac{\partial V_{s_m}(x)}{\partial x_i} & d_i \end{vmatrix}$ *equals* $\operatorname{sign}\left((-1)^{|S|}\right)$.

*If there is a unique unit direction satisfying the above constraints, we denote that direction $D^i_S(x)$.*

We will place mild assumptions on $V$ so that $D^i_S(x)$ is (uniquely) defined for all $x \in [0,1]^n$ where coordinates $S$ are zero-satisfied. (Specifically see Assumption 1 in Section 5.2, where our main result is formally stated.) With this definition in place, when our algorithm starts epoch $(i, S)$ at point $x$, it will execute the continuous-time dynamics $\dot{z}(\tau) = D^i_S(z(\tau))$, initialized at $z(0) = x$, forward in time. The algorithm executes this dynamics until the earliest time $\tau_{\text{exit}}$ such that $z(\tau_{\text{exit}})$ is an exit point, as per the definition below.

**Definition 3.** *Suppose $i \in [n]$, $S \subseteq [i-1]$ and $x'$ is a point where coordinates in $S$ are zero-satisfied and coordinates in $[i-1] \setminus S$ are boundary-satisfied. Then $x'$ is an* exit point *for epoch $(i, S)$ iff it satisfies one of the following:*

- ***(Good Exit Point)**: Coordinate $i$ is satisfied at $x'$, i.e., $V_i(x') = 0$, or $x'_i = 0$ and $V_i(x') < 0$, or $x'_i = 1$ and $V_i(x') > 0$.*
- ***(Bad Exit Point)**: For some $j \in S \cup \{i\}$, it holds that $(D^i_S(x'))_j > 0$ and $x'_j = 1$, or $(D^i_S(x'))_j < 0$ and $x'_j = 0$; in other words, if the dynamics for epoch $(i, S)$ were to continue from $x'$ onward, they would violate the constraints.*
- ***(Middling Exit Point)**: For some $j \in [i-1] \setminus S$, it holds that $V_j(x') = 0$ and one of the following holds: $\nabla V_j(x')^\top D^i_S(x') > 0$ and $x'_j = 0$, or $\nabla V_j(x')^\top D^i_S(x') < 0$ and $x'_j = 1$; in other words, if the dynamics for epoch $(i, S)$ were to continue from $x'$ onward, some boundary-satisfied coordinate would become unsatisfied.*

We will place mild assumptions on $V$ so that there can be a unique $j$ triggering the condition of Bad Exit Point in Definition 3 and there can be a unique $j$ triggering the Middling Exit Point condition, when $x'$ is a point where coordinates in $S$ are zero-satisfied and coordinates in $[i-1] \setminus S$ are boundary-satisfied. (Specifically see Assumptions 2 and 3 in Section 5.2.)

Here are the actions that we need to take if one of the above exit conditions has been reached happen.

**Action at Good Events.** In case of a good event, we start epoch $(i+1, S')$ at $x'$, where $S' = S \cup \{i\}$, if $i$ is zero-satisfied at $x'$, and $S' = S$, if $i$ is boundary-satisfied at $x'$.

**Action at Bad Events.** In case of a bad event, note that the coordinate $j$ responsible for the condition in the bad event to trigger must lie in $S \cup \{i\}$ because in all other coordinates $(D_S^i(x'))_j = 0$ by definition. Depending on which $j$ triggers the condition of the bad event we do one of the following:

(1) if the triggering $j = i$, then we start epoch $(i-1, S \setminus \{i-1\})$ at $x'$.

(2) if the triggering $j \neq i$, then we start epoch $(i, S \setminus \{j\})$ at $x'$.

**Action at Middling Events.** In the case of a middling event, we start epoch $(i, S \cup \{j\})$ at $x'$ (because the coordinate $j$ that trigger this event is both zero- and boundary-satisfied at $x'$ so we add it to $S$ to constrain the dynamics to zero-satisfy it next.).

With all the above, our algorithm is easy to describe precisely as follows. In Section B we do a step-by-step analysis of what the algorithm would do for a simple min-max optimization problem.

**Dynamics 2** STay-ON-the-Ridge (STON'R)

Initially $x(0) \leftarrow (0, \ldots, 0)$, $i \leftarrow 1$, $S \leftarrow \emptyset$, $t \leftarrow 0$.
**while** $x(t)$ is not a VI solution **do**
  Initialize epoch $(i, S)$'s continuous-time dynamics, $\dot{z}(\tau) = D_S^i(z(\tau))$, at $z(0) = x(t)$.
  **while** $z(\tau)$ is not an exit point as per Definition 3 **do**
    Execute $\dot{z}(\tau) = D_S^i(z(\tau))$ forward in time.
  **end while**
  Set $x(t + \tau_{\text{exit}}) = z(\tau_{\text{exit}})$ *(where $\tau_{\text{exit}}$ is earliest time $z(\tau)$ became an exit point).*
  **if** $x(t + \tau_{\text{exit}})$ is (Good Exit Point) as in Definition 3 **then**
    **if** $i$ is zero-satisfied at $x(t + \tau_{\text{exit}})$ **then**
      Update $S \leftarrow S \cup \{i\}$.
    **end if**
    Update $i \leftarrow i + 1$.
  **else if** $x(t + \tau_{\text{exit}})$ is a (Bad Exit Point) as in Definition 3 for $j = i$ **then**
    Update $i \leftarrow i - 1$ and $S \leftarrow S \setminus \{i-1\}$.
  **else if** $x(t + \tau_{\text{exit}})$ is a (Bad Exit Point) as in Definition 3 for $j \neq i$ **then**
    Update $S \leftarrow S \setminus \{j\}$.
  **else if** $x(t + \tau_{\text{exit}})$ is a (Middling Exit Point) as in Definition 3 for $j < i$ **then**
    Update $S \leftarrow S \cup \{j\}$.
  **end if**
  Set $t \leftarrow t + \tau_{\text{exit}}$.
**end while**
**return** $x(t)$

### 5.2 Our Assumptions and Our Main Theorem

*Assumptions.* We next present the assumptions on $V$ that are needed for our convergence proof. We discuss these assumptions further in Appendix C explaining why they are mild.

**Assumption 1.** *There exist positive real numbers $0 < \sigma_{\min} < \sigma_{\max}$ so that the following holds: For all $x \in [0,1]^n$ and set of coordinates $S = \{s_1, \ldots, s_m\}$, if $V_\ell(x) = 0$ for all $\ell \in S$, then the singular values of the $m \times m$ matrix*

$$J_S^K(x) := \begin{pmatrix} \frac{\partial V_{s_1}(x)}{\partial x_{s_1}} & \cdots & \frac{\partial V_{s_1}(x)}{\partial x_{s_m}} \\ \vdots & & \vdots \\ \frac{\partial V_{s_m}(x)}{\partial x_{s_1}} & \cdots & \frac{\partial V_{s_m}(x)}{\partial x_{s_m}} \end{pmatrix}$$

*are greater than $\sigma_{min}$ and less than $\sigma_{max}$.*

**Assumption 2.** *For any $x \in [0,1]^n$, set of coordinates $S = \{s_1, \ldots, s_m\}$, and $i \notin S$: If ($V_\ell(x) = 0$ for all $\ell \in S$) and ($x_\ell \in \{0,1\}$ for all $\ell \notin S \cup \{i\}$) then there is at most one coordinate $j \in S \cup \{i\}$ such that $x_j = 0$ or $x_j = 1$.*

One may ensure Assumption 2 holds by restricting the domain of each variable $i$ in the subset $[\alpha_i, 1 - \beta_i]$ of $[0,1]$, where $\alpha_i, \beta_i$ are uniformly random in $[0, \epsilon]$. For details we refer to Section C.

**Assumption 3.** *For all: (i) collection of coordinates $S = (s_1, \ldots, s_m)$, (ii) coordinate $i \notin S$, (iii) point $x \in [0,1]^n$ such that ($V_\ell(x) = 0$ for all $\ell \in S$) and ($x_\ell \in \{0,1\}$ for all $\ell \notin S \cup \{i\}$), and (iv) vector $(d_{s_1}, \ldots, d_{s_m}, d_i)$ satisfying the equations,*

$$\nabla_{S \cup \{i\}} V_j(x)^\top \cdot (d_{s_1}, \ldots, d_{s_m}, d_i) = 0 \text{ for all } j \in S,$$

*we have that $d_j \neq 0$ if $x_j = 0$ or $x_j = 1$.*

We are now ready to state our main theorem. The full proof of Theorem 1 is presented in Appendix E.

**Theorem 1.** *Under Assumptions 1, 2, and 3, there exists some $\bar{T} = \bar{T}(\sigma_{\min}, \sigma_{\max}, n) > 0$ such that* STAY-ON-THE-RIDGE *(Dynamics 2) will stop, at some time $T \leq \bar{T}$, at some point $x(T) \in [0,1]^n$ that is a solution of* $\mathrm{VI}(V, [0,1]^n)$.

**Remark 1** (Discrete-time Algorithm)**.** *It is possible to combine the proof of Theorem 1 with standard numerical analysis techniques to show the convergence of a simple discrete version of the dynamics assuming that the step size is small enough. For more details about this we refer to Appendix K.*

### 5.3 SKETCH OF PROOF OF THEOREM 1

A sketch of our proof of Theorem 1 comes from the recipe that we described in Section 4.

1. We start with the definition of the set of nodes $N$. The set $N$ contains triples of the form $(i, S, x)$ where $i \in [n]$, $S$ is a subset of $[i-1]$ and $x \in [0,1]^n$ that satisfies the following:
   - (a) all coordinates in $S$ are zero-satisfied, (b) all coordinates in $[i-1] \setminus S$ are boundary-satisfied, (c) $x_j = 0$ for all $j \geq i+1$, and either (d1) $x_i = 0$ or (d2) $x$ is an exit point for epoch $(i, S)$ according to Definition 3 [1].

   We show in the Appendix the size of $N$ is finite (see Lemma 3).

   Next we describe a mapping $F : N \to N$ as in Section 4. Let $(i, S, x) \in N$ we use the dynamics $\dot{z} = D^i_S(z)$ with initial condition $z(0) = x$ and we find the minimum time $\tau_{\text{exit}}$ such that $z(\tau_{\text{exit}})$ is an exit point. We then update $i$, $S$ to $i'$, $S'$ according to the rules for actions on exit points of Section 5.1 and we define $F((i, S, x)) = (i', S', z(\tau_{\text{exit}}))$. As we show in the appendix the dynamics $\dot{z} = D^i_S(z)$ have a unique solution under our assumptions and hence $F$ is well defined (Lemma 9).

   The set $N$ and the mapping $F$ define the directed graph $G$ as we described in Section 4 that is guaranteed to have vertices with out-degree at most 1. We also show that any $v \in V$ with out-degree 0 is an equilibrium point (Lemma 5 and 6).

2. To show that the in-degree is at most 1 too we show that we can actually solve the dynamics backwards in time. In particular, if we specify $z(0)$ and there is the smallest time $\tau_{\text{exit}}$ such that $z(-\tau_{\text{exit}})$ is an exit point then $z(-\tau_{\text{exit}})$ is uniquely determined. This means that there exists $F^{-1} : N \to N$ such that if $v' = F(v)$ then $F^{-1}(v') = v$ which means that no vertex in $N$ can have in-degree more than 1 (Lemma 7).

3. We show that $v_0 = (1, \emptyset, (0, \ldots, 0)) \in N$. Also, if run the dynamics $\dot{z} = D^1_\emptyset(z)$ backwards in time starting at $z(0) = 0$ then we get outside $[0,1]^n$ and so $v_0$ has in-degree 0. We also show that the dynamics $\dot{z} = D^1_\emptyset(z)$ can move forward in time and stay inside $[0,1]^n$ so $v_0$ has out-degree 1 (Lemma 7).

4. The above show that our algorithm converges.

[1] The actual set of nodes that we used in the proof does not contain the information of $i$ and $S$ but we refer to the Appendix for the exact proof.

## REFERENCES


Jacob Abernethy, Kevin A Lai, and Andre Wibisono. Last-iterate convergence rates for min-max optimization. *arXiv preprint arXiv:1906.02027*, 2019.

Ilan Adler. The equivalence of linear programs and zero-sum games. *International Journal of Game Theory*, 42(1):165–177, 2013. doi: 10.1007/s00182-012-0328-8.

Leonard Adolphs, Hadi Daneshmand, Aurelien Lucchi, and Thomas Hofmann. Local saddle point optimization: A curvature exploitation approach. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 486–495, 2019.

David Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific J. Math.*, 6(1):1–8, 1956. doi: 10.2140/pjm.1956.6.1.

Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012. doi: 10.1561/2200000024.

Nikolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

George B. Dantzig. A proof of the equivalence of the programming problem and the game problem. In *Koopmans, T. C., editor(s), Activity Analysis of Production and Allocation*. Wiley, New York, 1951.

Constantinos Daskalakis and Ioannis Panageas. The limit points of (optimistic) gradient descent in min-max optimization. In *Advances in Neural Information Processing Systems*, pp. 9236–9246, 2018.

Constantinos Daskalakis and Ioannis Panageas. Last-iterate convergence: Zero-sum games and constrained min-max optimization. *Innovations in Theoretical Computer Science*, 2019.

Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training gans with optimism. In *International Conference on Learning Representations (ICLR 2018)*, 2018.

Constantinos Daskalakis, Stratis Skoulakis, and Manolis Zampetakis. The complexity of constrained min-max optimization. *CoRR*, abs/2009.09623, 2020. URL https://arxiv.org/abs/2009.09623.

Constantinos Daskalakis, Stratis Skoulakis, and Manolis Zampetakis. The Complexity of Constrained Min-Max Optimization. In *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC)*, 2021.

Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. Negative momentum for improved game dynamics. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1802–1811, 2019.

Noah Golowich, Sarath Pattathil, and Constantinos Daskalakis. Tight last-iterate convergence rates for no-regret learning in multi-player games. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020a.

Noah Golowich, Sarath Pattathil, Constantinos Daskalakis, and Asuman Ozdaglar. Last iterate is slower than averaged iterate in smooth convex-concave saddle point problems. In *Conference on Learning Theory*, pp. 1758–1784. PMLR, 2020b.

Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2014.

Erfan Yazdandoost Hamedani and Necdet Serhat Aybat. A primal-dual algorithm for general convex-concave saddle point problems. *arXiv preprint arXiv:1803.01401*, 2018.

J. Hannan. Approximation to Bayes risk in repeated play. *Contributions to the Theory of Games*, 3: 97–139, 1957.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Ya-Ping Hsieh, Panayotis Mertikopoulos, and Volkan Cevher. The limits of min-max optimization algorithms: Convergence to spurious non-critical sets. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

Arieh Iserles. *A first course in the numerical analysis of differential equations*. Cambridge university press, 2009.

Chi Jin, Praneeth Netrapalli, and Michael I Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? *arXiv preprint arXiv:1902.00618*, 2019.

Weiwei Kong and Renato DC Monteiro. An accelerated inexact proximal point method for solving nonconvex-concave min-max problems. *SIAM Journal on Optimization*, 31(4):2558–2585, 2021.

GM Korpelevich. The extragradient method for finding saddle points and other problems. *Matecon*, 12:747–756, 1976.

Carlton E Lemke and Joseph T Howson, Jr. Equilibrium points of bimatrix games. *Journal of the Society for industrial and Applied Mathematics*, 12(2):413–423, 1964.

Tengyuan Liang and James Stokes. Interaction matters: A note on non-asymptotic local convergence of generative adversarial networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 907–915, 2019.

Tianyi Lin, Chi Jin, and Michael Jordan. On gradient descent ascent for nonconvex-concave minimax problems. In *International Conference on Machine Learning*, pp. 6083–6093. PMLR, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Oren Mangoubi and Nisheeth K Vishnoi. Greedy adversarial equilibrium: An efficient alternative to nonconvex-nonconcave min-max optimization. In *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC)*, 2021.

Eric Mazumdar and Lillian J Ratliff. On the convergence of gradient-based learning in continuous games. *arXiv preprint arXiv:1804.05464*, 2018.

Panayotis Mertikopoulos, Christos H. Papadimitriou, and Georgios Piliouras. Cycles in adversarial regularized learning. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2018.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, pp. 3481–3490, 2018.

Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

Aryan Mokhtari, Asuman Ozdaglar, and Sarath Pattathil. A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. *arXiv preprint arXiv:1901.08511*, 2019.

Dmitrii M Ostrovskii, Andrew Lowy, and Meisam Razaviyayn. Efficient search of first-order nash equilibria in nonconvex-concave smooth min-max problems. *SIAM Journal on Optimization*, 31 (4):2508–2538, 2021.

L. D. Popov. A modification of the Arrow-Hurwicz method for search of saddle points. *Mathematical notes of the Academy of Sciences of the USSR*, 28(5):845–848, Nov 1980.

Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. Non-convex min-max optimization: Provable algorithms and applications in machine learning. *arXiv preprint arXiv:1810.02060*, 2018.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012. doi: 10.1561/2200000018.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Emmanouil-Vasileios Vlatakis-Gkaragkounis, Lampros Flokas, and Georgios Piliouras. Poincaré recurrence, cycles and spurious equilibria in gradient-descent-ascent for non-convex non-concave zero-sum games. *Advances in Neural Information Processing Systems*, 32, 2019.

John von Neumann. Zur Theorie der Gesellschaftsspiele. In *Math. Ann.*, pp. 295–320, 1928.

Yuanhao Wang, Guodong Zhang, and Jimmy Ba. On solving minimax optimization locally: A follow-the-ridge approach. In *International Conference on Learning Representations*, 2019.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*, 2019.