Finding and Only Finding Differential Nash Equilibria by Both Pretending to be a Follower

Xuchan Bao *jennybao@cs.toronto.edu*
Department of Computer Science
University of Toronto, Vector Institute

Guodong Zhang *gdzhang@cs.toronto.edu*
Department of Computer Science
University of Toronto, Vector Institute

Reviewed on OpenReview: *https://openreview.net/forum?id=igdWKxK5RZ*

Abstract

Finding Nash equilibria in two-player differentiable games is a classical problem in game theory with important relevance in machine learning. We propose double Follow-the-Ridge (double-FTR), an algorithm that locally converges to and only to differential Nash equilibria in general-sum two-player differentiable games. To our knowledge, double-FTR is the first algorithm with such guarantees for general-sum games. Furthermore, we show that by varying its preconditioner, double-FTR leads to a broader family of algorithms with the same convergence guarantee. In addition, double-FTR avoids oscillation near equilibria due to the real-eigenvalues of its Jacobian at fixed points. Empirically, we validate the double-FTR algorithm on a range of simple zero-sum and general sum games, as well as simple Generative Adversarial Network (GAN) tasks.

1 Introduction

Much of the recent success in deep learning can be attributed to the effectiveness of gradient-based optimization. It is well-known that for a minimization problem, with appropriate choice of learning rates, gradient descent enjoys convergence guarantee to local minima (Lee et al., 2016; 2019). Based on this foundational result, an array of accelerated and higher-order methods have since been proposed and widely applied in training neural networks (Duchi et al., 2011; Kingma & Ba, 2014; Reddi et al., 2018; Zhang et al., 2019b).

However, once we leave the realm of minimization problems and consider the multi-agent setting, the optimization landscape becomes much more complicated. Multi-agent optimization problems arise in diverse fields such as robotics, economics and machine learning (Foerster et al., 2016; Von Neumann & Morgenstern, 2007; Goodfellow et al., 2014; Ben-Tal & Nemirovski, 2002; Gemp et al., 2020; Anil et al., 2021).

A classical abstraction that is especially relevant for machine learning is two-player differentiable games, where the objective is to find global or local Nash equilibria. The equivalent of gradient descent in such a game would be each agent applying gradient descent to minimize their own objective function. However, in stark contrast with gradient descent in solving minimization problems, this gradient-descent-style algorithm may converge to spurious critical points that are not Nash equilibria, and in the general-sum game case, Nash equilibria might not even be stable critical points for this algorithm (Mazumdar et al., 2020b)!

These negative results have driven a surge of recent interest in developing other gradient-based algorithms for finding Nash equilibria in differentiable games. Among them is Mazumdar et al. (2019), who proposed an update algorithm whose attracting critical points are only local Nash equilibria in the special case of zero-sum games. However, to the best of our knowledge, such guarantees have not been extended to general-sum games.

We propose double Follow-the-Ridge (double-FTR), a gradient-based algorithm for general-sum differentiable games that locally converges to and only to differential Nash equilibria.[1] Double-FTR is closely related to the Follow-the-Ridge (FTR) algorithm for Stackelberg games (Wang et al., 2019), which converges to and only to local Stackelberg equilibria (Fiez et al., 2019). Double-FTR can be viewed as its counterpart for simultaneous games, where each player adopts the "follower" strategy in FTR.

The rest of this paper is organized as follows. In Section 2, we give background on two-player differentiable games and equilibrium concepts. We also explain the issues with using gradient-descent-style algorithms on such games. In Section 3, we present the double-FTR algorithm and prove its local convergence to and only to differential Nash equilibria. We also identify a more general class of algorithms that share these properties. We discuss recent works directly relevant to double-FTR in Section 4 and other related work in Section 5. In Section 6, we show empirical evidence of double-FTR's convergence to and only to differential Nash equilibria.

2 Background

2.1 Two-player differentiable games and equilibrium concepts

In a general-sum two-player differentiable game, player 1 aims to minimize $f : \mathbb{R}^{n+m} \to \mathbb{R}$ with respect to $\boldsymbol{x} \in \mathbb{R}^n$, whereas player 2 aims to maximize $g : \mathbb{R}^{n+m} \to \mathbb{R}$ with respect to $\boldsymbol{y} \in \mathbb{R}^m$. Following the notation in Mazumdar et al. (2019), we denote such a game as $\{(f, -g), \mathbb{R}^{n+m}\}$. We also make the following assumption on the twice-differentiability of f and g.

Assumption 1. $\forall\, \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{y} \in \mathbb{R}^m$, f and g are twice-differentiable, and the second derivatives are continuous. Also, $\nabla^2_{\boldsymbol{xx}} f$ and $\nabla^2_{\boldsymbol{yy}} g$ are invertible.

For two rational, non-cooperative players, their optimal outcome is to achieve a local Nash equilibrium (Ratliff et al., 2013).[2]

Definition 2.1 (Local Nash equilibrium)**.** A point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is a local Nash equilibrium of $\{(f, -g), \mathbb{R}^{n+m}\}$ if there exists open sets $\mathcal{S}_{\boldsymbol{x}} \subset \mathbb{R}^n$, $\mathcal{S}_{\boldsymbol{y}} \subset \mathbb{R}^m$ such that $\boldsymbol{x}^* \in \mathcal{S}_{\boldsymbol{x}}$, $\boldsymbol{y}^* \in \mathcal{S}_{\boldsymbol{y}}$, and

$$f(\boldsymbol{x}^*, \boldsymbol{y}^*) \leq f(\boldsymbol{x}, \boldsymbol{y}^*), \quad g(\boldsymbol{x}^*, \boldsymbol{y}^*) \geq g(\boldsymbol{x}^*, \boldsymbol{y}), \quad \forall \boldsymbol{x} \in \mathcal{S}_{\boldsymbol{x}}, \quad \forall \boldsymbol{y} \in \mathcal{S}_{\boldsymbol{y}}.$$

For twice-differentiable f and g, a local Nash equilibrium at $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ implies (Ratliff et al., 2013, Proposition 2):

$$\begin{aligned} &\text{(First-order condition)} && \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) = 0 \text{ and } \nabla_{\boldsymbol{y}} g(\boldsymbol{x}^*, \boldsymbol{y}^*) = 0, \\ &\text{(Second-order necessary condition)} && \nabla^2_{\boldsymbol{xx}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) \succeq 0 \text{ and } \nabla^2_{\boldsymbol{yy}} g(\boldsymbol{x}^*, \boldsymbol{y}^*) \preceq 0. \end{aligned}$$

A closely related notion of equilibrium is the differential Nash equilibrium (DNE) (Ratliff et al., 2013), which satisfies a second-order *sufficient* condition for local Nash equilibrium.

Definition 2.2 (Differential Nash equilibrium)**.** $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is a differential Nash equilibrium of $\{(f, -g), \mathbb{R}^{n+m}\}$ if the following two conditions hold:

- $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) = 0$ and $\nabla_{\boldsymbol{y}} g(\boldsymbol{x}^*, \boldsymbol{y}^*) = 0$,
- $\nabla^2_{\boldsymbol{xx}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) \succ 0$ and $\nabla^2_{\boldsymbol{yy}} g(\boldsymbol{x}^*, \boldsymbol{y}^*) \prec 0$.

The conditions of DNE are slightly stronger than that of local Nash equilibria in that the second-order conditions are definite instead of semi-definite. In this paper, we focus on DNE, as they make up almost all local Nash equilibria in the mathematical sense, and are well-suited for the analysis of second-order algorithms.

[1] Slightly stronger than the local Nash equilibria, discussed in Section 2.1

[2] Note that local Nash equilibrium is not guaranteed to exist in nonconvex-nonconcave games ((Jin et al., 2020), Proposition 6), although the (non-)existence of local NE is out of the scope of this paper.

Discussion on Assumption 1 We assume twice-differentiability of f and g since we are concerned with finding DNE. As for the assumption that $\nabla^2_{\boldsymbol{xx}} f$ and $\nabla^2_{\boldsymbol{yy}} g$ are invertible, we need it for deriving our main algorithm in its original form. However, as discussed in Section 3.2, with a practical implementation of our main algorithm, the assumption on invertibility can be relaxed.

2.2 Issues with gradient-based algorithms

A natural strategy for agents to search for DNE in a differentiable game is to use gradient-based algorithms. The simplest gradient-based algorithm is the gradient descent-ascent (GDA) (Ryu & Boyd, 2016; Zhang et al., 2021b) (Algorithm 1) or its variants (Zhang et al., 2021a; Korpelevich, 1976; Mokhtari et al., 2020).

Algorithm 1 Gradient descent-ascent (GDA)

Require: Number of iterations T, learning rate γ
1: **for** $t = 1, \ldots, T$ **do**
2: $\quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_t, \boldsymbol{y}_t)$
3: $\quad \boldsymbol{y}_{t+1} = \boldsymbol{y}_t + \gamma \nabla_{\boldsymbol{y}} g(\boldsymbol{x}_t, \boldsymbol{y}_t)$
4: **end for**

Let $\boldsymbol{z} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}$ and $\gamma > 0$ be the learning rate, a gradient-based update algorithm can be written as:

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \gamma \boldsymbol{\omega}(\boldsymbol{z}_t). \tag{1}$$

The Jacobian of $\boldsymbol{\omega}(\boldsymbol{z})$ is defined as $\boldsymbol{J}(\boldsymbol{z}) := \frac{\partial \boldsymbol{\omega}(\boldsymbol{z})}{\partial \boldsymbol{z}}$. In the case of GDA, we have:

$$\boldsymbol{\omega}_{\mathrm{GDA}}(\boldsymbol{z}) = \begin{bmatrix} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{y}) \\ -\nabla_{\boldsymbol{y}} g(\boldsymbol{x}, \boldsymbol{y}) \end{bmatrix}, \quad \boldsymbol{J}_{\mathrm{GDA}} = \begin{bmatrix} \nabla^2_{\boldsymbol{xx}} f & \nabla^2_{\boldsymbol{xy}} f \\ -\nabla^2_{\boldsymbol{yx}} g & -\nabla^2_{\boldsymbol{yy}} g \end{bmatrix}.$$

Using the Jacobian matrix, we characterize the stability around fixed points of equation 1.

Definition 2.3 ((Strictly) stable fixed point)**.** $\boldsymbol{z}^*$ is a stable fixed point of the discrete-time dynamical system in equation 1 if

$$\boldsymbol{\omega}(\boldsymbol{z}^*) = \boldsymbol{0} \quad \text{and} \quad \rho(\boldsymbol{I} - \gamma \boldsymbol{J}(\boldsymbol{z}^*)) \leq 1,$$

where $\rho(\cdot)$ denotes the spectral radius of a matrix. If we additionally have $\rho(\boldsymbol{I} - \gamma \boldsymbol{J}(\boldsymbol{z}^*)) < 1$, then $\boldsymbol{z}^*$ is a strictly stable fixed point.

Strictly stable fixed points are important for analysis, as they are locally asymptotically convergent (Galor, 2007), i.e. there exists an open set $\mathcal{S}_{\boldsymbol{z}}$ such that $\boldsymbol{z}^* \in \mathcal{S}_{\boldsymbol{z}}$ and $\lim_{t \to \infty} \boldsymbol{z}_t = \boldsymbol{z}^* \ \forall \boldsymbol{z}_0 \in \mathcal{S}_{\boldsymbol{z}}$.

A closely related concept is the locally asymptotically stable equilibrium (LASE) for the continuous-time system $\dot{\boldsymbol{z}} = -\boldsymbol{\omega}(\boldsymbol{z})$. (Ratliff et al., 2013).

Definition 2.4 (Locally asymptotically stable equilibrium (LASE))**.** $\boldsymbol{z}^*$ is a locally asymptotically stable equilibrium of the continuous-time dynamics $\dot{\boldsymbol{z}} = -\boldsymbol{\omega}(\boldsymbol{z})$ if

$$\boldsymbol{\omega}(\boldsymbol{z}^*) = \boldsymbol{0} \quad \text{and} \quad \mathrm{Re}(\lambda) > 0 \text{ for } \forall \lambda \in \mathrm{spec}(\boldsymbol{J}(\boldsymbol{z}^*)),$$

where $\mathrm{Re}(\cdot)$ denotes the real part of a complex number, and $\mathrm{spec}(\cdot)$ returns the spectrum (i.e. the set of eigenvalues) of a matrix.

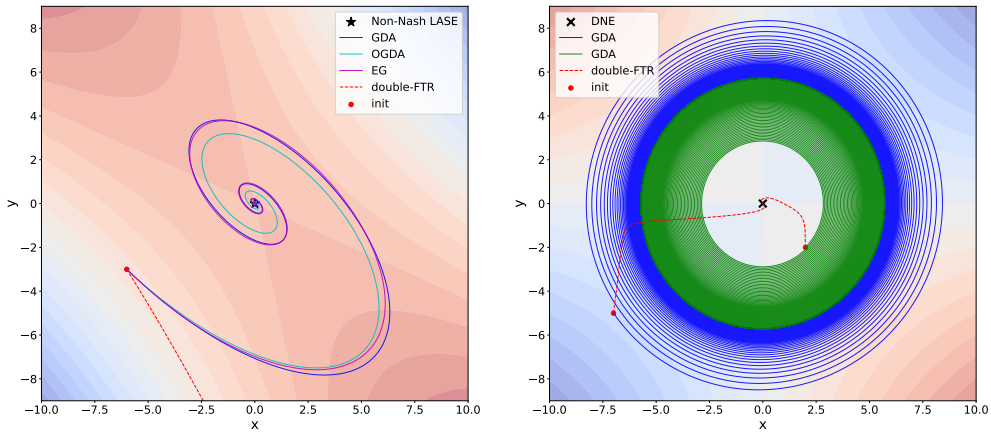
Note that in the limit $\gamma \to 0$, strictly stable fixed points of GDA are equivalent to LASE of $\dot{\boldsymbol{z}} = -\boldsymbol{\omega}_{\mathrm{GDA}}(\boldsymbol{z})$. In this paper, we prove convergence results in discrete-time (using Definition 2.3), but we often provide intuition using continuous-time concepts such as LASE.

Unfortunately, GDA is not guaranteed to converge to DNE, nor are DNE necessarily (strictly) stable fixed points of the GDA dynamics. The relationship between the LASE of $\dot{\boldsymbol{z}} = -\boldsymbol{\omega}_{\mathrm{GDA}}(\boldsymbol{z})$ and DNE is shown in the Venn diagrams in Figure 1. Below, we give a few examples to help build intuition.

(a) General-sum **(b)** Zero-sum

Figure 1: Venn diagrams showing the relationship between the set of locally asymptotically stable equilibria (LASE) of the GDA flow and the set of differential Nash equilibria (DNE) in two-player differentiable games. Note that for general-sum games, there exist DNE that are *unstable* for GDA flow.

Zero-sum games Even in the special case of zero-sum games ($g = f$), GDA dynamics can still have stable fixed points that are not DNE (Daskalakis & Panageas, 2018; Mazumdar et al., 2020b). In Figure 2, we demonstrate the failure modes of GDA in zero-sum games. In Figure 2a, GDA converges to a spurious strictly stable fixed point which is not DNE (corresponding to the pink areas in Figure 1). In 2b, GDA fails to converge to the unique DNE (Hsieh et al., 2020). Instead, it goes into a limit cycle, due to the strong rotation introduced by large complex parts in its Jacobian eigenvalues. We stress that these pathologies are not limited to GDA, but common for many other first-order algorithms (Wang et al., 2019).



(a) $f(x, y) = g(x, y) = 2x^2 + 5xy + y^2$ **(b)** $f(x, y) = g(x, y) = xy + \epsilon(x^2 - y^4)$

Figure 2: Two examples of GDA failure modes in finding DNE in zero-sum games. (a) GDA converges to the spurious strictly stable fixed point $(0, 0)$, which is not a DNE. Other first-order methods such as the optimistic GDA (OGDA) and extragradient (EG) converge to the spurious fixed point as well. (b) Instead of the unique DNE at $(0, 0)$, GDA converges to a limit cycle both when initialized "inside" (green) and "outside" (blue). We use $\epsilon = 0.0001$, $\gamma = 0.01$.

General-sum games In general-sum games, apart from non-DNE spurious fixed points for GDA, a DNE might not even be a stable fixed point of GDA (the cyan area in Figure 1). For example, let $n = m = 1$, $f(x, y) = x^2 + 2xy$ and $g(x, y) = -3xy - 0.5y^2$. It is easy to confirm that $(0, 0)$ is a DNE. However,

$$\boldsymbol{J}_{\text{GDA}} = \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix}, \quad \text{spec}(\boldsymbol{J}_{\text{GDA}}) = \{4, -1\}.$$

This means that for the GDA dynamics, $(0, 0)$ is a saddle point rather than a LASE.

3 Double Follow-the-Ridge

We propose double Follow-the-Ridge (double-FTR), an update rule for general-sum differential games that locally converges to and only to differential Nash equilibria. The double-FTR update is shown in Algorithm 2 (the arguments $\boldsymbol{x}_t$, $\boldsymbol{y}_t$ of f and g are dropped to avoid notational clutter).

Algorithm 2 Double Follow-the-Ridge

Require: Learning rate $\eta_{\boldsymbol{x}}$ and $\eta_{\boldsymbol{y}}$; number of iterations T.

1: **for** $t = 1, \dots, T$ **do**

2: $\quad \boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \eta_{\boldsymbol{x}} \nabla_{\boldsymbol{x}} f - \eta_{\boldsymbol{y}} (\nabla^2_{\boldsymbol{xx}} f)^{-1} \nabla^2_{\boldsymbol{xy}} g \nabla_{\boldsymbol{y}} g$

3: $\quad \boldsymbol{y}_{t+1} \leftarrow \boldsymbol{y}_t + \eta_{\boldsymbol{y}} \nabla_{\boldsymbol{y}} g + \eta_{\boldsymbol{x}} (\nabla^2_{\boldsymbol{yy}} g)^{-1} \nabla^2_{\boldsymbol{yx}} f \nabla_{\boldsymbol{x}} f$

4: **end for**

Let $\boldsymbol{z} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}$, $\gamma = \eta_{\boldsymbol{x}}$ and $c = \frac{\eta_{\boldsymbol{y}}}{\eta_{\boldsymbol{x}}}$, we can express Algorithm 2 in vectorized form (equation 2). To simplify the notation, we drop the subscript t for f and g.

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \gamma \boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z}_t), \quad \boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z}_t) = \begin{bmatrix} \boldsymbol{I} & -(\nabla^2_{\boldsymbol{xx}} f)^{-1} \nabla^2_{\boldsymbol{xy}} g \\ -(\nabla^2_{\boldsymbol{yy}} g)^{-1} \nabla^2_{\boldsymbol{yx}} f & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \nabla_{\boldsymbol{x}} f \\ -c \nabla_{\boldsymbol{y}} g \end{bmatrix}. \tag{2}$$

3.1 Local convergence of double-FTR

In this section, we give our main theoretical result. First, we introduce an additional assumption.

Assumption 2. At fixed points of equation 2, $\boldsymbol{J}_{\text{GDA}}(\boldsymbol{z})$ has full rank.

Assumption 2 ensures that in double-FTR, the additional terms in the update do not exactly cancel out the GDA terms, which in turn ensures that fixed points with double-FTR are also fixed points with double-GDA. Note a similar assumption is introduced in Mazumdar et al. (2019) Theorem 4.

Our main theoretical result is stated below.

Theorem 1. *Under Assumptions 1 and 2 and with an appropriate choice of learning rate γ, $\boldsymbol{z}^*$ is a strictly stable fixed point of the double-FTR update (equation 2) if and only if it is a differential Nash equilibrium of the game $\{(f, -g), \mathbb{R}^{n+m}\}$. Furthermore, at fixed points of equation 2, all eigenvalues of the Jacobian $\boldsymbol{J}_{\text{FTR}} := \frac{\partial \boldsymbol{\omega}_{\text{FTR}}}{\partial \boldsymbol{z}}$ are real.*

Intuitively, the first part of the theorem classifies the strictly stable fixed points of double-FTR, which leads to local convergence to and only to DNE (Corollary 1). The second part ensures that there is no rotation caused by complex eigenvalues in the neighbourhood of the DNEs. This is beneficial, as eigenvalues with large imaginary parts can often cause instability (such as the oscillation when training GANs) (Mescheder et al., 2017; Balduzzi et al., 2018). We defer the proof of Theorem 1 to Appendix A.

Corollary 1 (Local convergence)**.** Let $\boldsymbol{z}^*$ be a DNE of the game $\{(f, -g), \mathbb{R}^{n+m}\}$. Under Assumptions 1 and 2 and with an appropriate choice of learning rate γ, there exists an open set $\mathcal{S}_{\boldsymbol{z}} \subset \mathbb{R}^{n+m}$ where $\boldsymbol{z}^* \in \mathcal{S}_{\boldsymbol{z}}$, such that when following equation 2, $\forall\ \boldsymbol{z}_0 \in \mathcal{S}_{\boldsymbol{z}}$, $\lim_{t \to \infty} \boldsymbol{z}_t \to \boldsymbol{z}^*$.

Proof. The proof follows naturally by combining Theorem 1 with the local convergence of strictly stable fixed points (Galor (2007), Proposition 1.9). □

To the best of our knowledge, double FTR is the first algorithm with such local convergence result for general-sum games.

3.2 General preconditioners

In the following remark, we show that double-FTR can be generalized to include a whole family of algorithms.

Remark 1. Theorem 1 applies to a more general version of the double FTR algorithm. In particular, we can generalize equation 2 to allow a broader class of "preconditioners":

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \gamma \tilde{\boldsymbol{\omega}}_{\mathrm{FTR}}(\boldsymbol{z}_t), \quad \tilde{\boldsymbol{\omega}}_{\mathrm{FTR}}(\boldsymbol{z}) = \begin{bmatrix} \boldsymbol{P_x} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{P_y} \end{bmatrix} \boldsymbol{J}_{\mathrm{GDA}}^{\top}(\boldsymbol{z}_t) \begin{bmatrix} \nabla_{\boldsymbol{x}} f \\ -c\nabla_{\boldsymbol{y}} g \end{bmatrix}, \tag{3}$$

where $\boldsymbol{P_x}$, $\boldsymbol{P_y}$ are continuous functions of $\boldsymbol{x}, \boldsymbol{y}$ respectively, which satisfy $\boldsymbol{P_x} \succ 0 \iff \nabla^2_{\boldsymbol{xx}} f \succ 0$ and $\boldsymbol{P_y} \prec 0 \iff \nabla^2_{\boldsymbol{yy}} g \prec 0$.

Equation 2 corresponds to the special case of $\boldsymbol{P_x} = (\nabla^2_{\boldsymbol{xx}} f)^{-1}$, $\boldsymbol{P_y} = (\nabla^2_{\boldsymbol{yy}} g)^{-1}$. The proof for Theorem 1 directly applies to the case of general preconditioners in Remark 1.

Remark 1 provides intuition on the convergence properties of double-FTR. Without the preconditioner $\boldsymbol{P_x}$ and $\boldsymbol{P_y}$, double-FTR reduces to Hamiltonian gradient descent (Mescheder et al., 2017; Balduzzi et al., 2018; Loizou et al., 2020; Abernethy et al., 2021), which has spurious non-Nash equilibria. It is the introduction of the preconditioner that enables strictly stable fixed points to satisfy the second-order condition of DNE.

Remark 1 also sheds light on how to derive a more practical algorithm. Naively implementing Algorithm 2 might cause instability when $\nabla^2_{\boldsymbol{xx}} f$ and $\nabla^2_{\boldsymbol{yy}} g$ are near singular. In practice, we use $(\nabla^2_{\boldsymbol{xx}} f \nabla^2_{\boldsymbol{xx}} f + \lambda \boldsymbol{I})^{-1} \nabla^2_{\boldsymbol{xx}} f$ instead of $(\nabla^2_{\boldsymbol{xx}} f)^{-1}$ in Algorithm 2 (where a small $\lambda > 0$ is the damping parameter). Note that this also allows us to drop the assumption on the invertibility of $\nabla^2_{\boldsymbol{xx}} f$ and $\nabla^2_{\boldsymbol{yy}} g$ in Assumption 1.

3.3 N-player games

Algorithm 2 naturally extends to n-player games. The algorithm and theoretical results for n-player games are exactly analogous to the two-player setting. The n-player Follow-the-Ridge algorithm and its convergence properties are shown in Appendix B.

4 Connection with other algorithms

Mazumdar et al. (2019) proposed local symplectic surgery (LSS) – a gradient-based algorithm whose LASE are exactly DNE in two-player zero-sum games. LSS avoids oscillatory behaviour at DNE, similar to double-FTR. Compared to LSS, double-FTR appears to have a simpler form and enables a broader family of algorithms with such local convergence result in general-sum games.

The Follow-the-Ridge (FTR) algorithm (Wang et al., 2019) is closely related to our proposed double-FTR. FTR was proposed for two-player sequential games and is guaranteed to converge to and only to local minimax for zero-sum and Stackelberg equilibria for general-sum sequential games. FTR applies a gradient correction term on the follower in a sequential game, so that the agents approximately follow a ridge in the landscape of the objective function. The double-FTR can be viewed as a counterpart of FTR for simultaneous games. The update rule of double-FTR resembles that of FTR, with the gradient modification term applied on both players.

Another related algorithm is the Hamiltonian gradient descent (HGD) (Mescheder et al., 2017; Balduzzi et al., 2018; Loizou et al., 2020; Abernethy et al., 2021). HGD performs gradient-descent on the Hamiltonian, or the squared norm of the gradient. HGD is guaranteed to converge, as it is essentially a minimization problem. However, in general it may have spurious non-Nash equilibria points. Interestingly, our double-FTR can be viewed as a preconditioned HGD.

5 Related work

Mazumdar et al. (2020b) introduced a general framework for competitive gradient-based learning. They characterized DNE in terms of the critical points of the gradient algorithms. They showed the lack of convergence of the gradient algorithm in games, which motivated the development of the double-FTR algorithm.

Much work has focused on improving the dynamics in finding stable fixed points, which is crucial in applications such as GANs, where oscillation caused by eigenvalues with zero real parts and large imaginary parts in the gradient Jacobian can lead to training instability. Mescheder et al. (2017) proposes Consensus Optimization, which encourages agreement between the two players by introducing a regularization term in the objectives of both players. The regularization term results in a more negative real-part for the eigenvalues of the gradient Jacobian, therefore reduces oscillation and allows larger learning rates. Balduzzi et al. (2018); Gemp & Mahadevan (2018) proposes Symplectic Gradient Adjustment (SGA), which decomposes the gradient Jacobian into symmetric (potential) and asymmetric (Hamiltonian) parts and adds a gradient adjustment term for rapid convergence to stable fixed points. Schäfer & Anandkumar (2019) proposes Competitive Gradient Descent (CGD), whose update is given by the Nash equilibrium of a regularized bilinear approximation of the original game. Compared to other methods, CGD has the advantage of not needing to adapt step size when the interaction strength changes between players. Many other methods have been proposed with different strategies for predicting other agents' moves, such as Learning with Opponent Learning Awareness (LOLA) (Foerster et al., 2016) and optimistic gradient descent-ascent (OGDA) (Popov, 1980; Rakhlin & Sridharan, 2013; Daskalakis et al., 2018; Mertikopoulos et al., 2018). However, none of these existing methods address the problem of spurious (i.e. non-Nash) stable fixed points.

A separate but related line of research is on bilevel optimization. Bilevel optimization involves nested objectives: the upper-level objective depends on the solution of the lower-level problem. It has wide applications such as hyperparameter optimization and GAN training (Franceschi et al., 2018; Goodfellow et al., 2014). One approach to bilevel optimization is based on implicit differentiation (Ochs et al., 2015; Pedregosa, 2016; Lorraine et al., 2020). Implicit differentiation is closely related to the original Follow-the-Ridge algorithm (Wang et al., 2019), which tackles bilevel optimization from a game-theoretic perspective. Compared to simultaneous games (this paper), bilevel optimization has different notions of equilibrium, such as the local minimax (Jin et al., 2020) and Stackelberg equilibrium (Von Stackelberg, 2010).

6 Experiments

We conduct simple experiments to demonstrate the implications of our theoretical results. First, through a 2-D toy example, we show that double-FTR converges to DNE and successfully avoids spurious non-DNE fixed points, as predicted by Theorem 1. Then, we demonstrate that in a *general-sum* linear quadratic game, double-FTR is able to converge to DNE that naive policy gradient avoids. Lastly, we show that double-FTR can be scaled up and applied to train Generative Adversarial Networks (Goodfellow et al., 2014). Unlike GDA which suffers from severe mode collapse, double-FTR recovers all the modes and learns a distribution that closely matches the target.

6.1 2-D toy example

We consider the zero-sum game $\{f, -f\}, \mathbb{R}^2$ with the following 2-D function (same as in Mazumdar et al. (2019)):

$$f(x, y) = e^{-0.01(x^2+y^2)}\big((0.3x^2 + y)^2 + (0.5y^2 + x)^2\big).$$

This function has several strictly stable fixed points for the GDA dynamics, among which some are DNE and some are not. As shown in Figure 3, while GDA may converge to fixed points that are not DNE, double-FTR avoids such spurious fixed points. Also, in the neighbourhood of the DNE, GDA exhibits oscillatory behaviour due to complex eigenvalues of the Jacobian matrix. In contrast, the double-FTR does not have oscillatory behaviour near the DNE. For reference, we also show the trajectories of the Local Symplectic Surgery (LSS). In this experiment, LSS has similar convergence properties – it avoids spurious fixed points and does not have oscillatory behaviour near the DNE.

Observing the optimization trajectory, we note that double-FTR can approach a non-Nash spurious fixed point, then steer away without converging to it. In this paper, we do not discuss the convergence rate of double-FTR, because the dynamics of a multi-player game is much more complicated than that of a

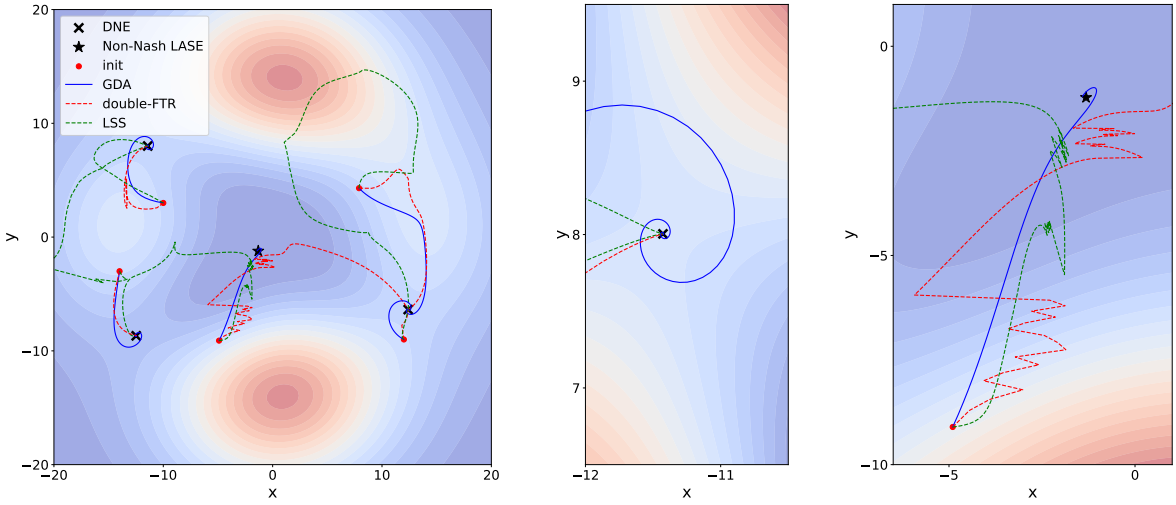


Figure 3: Left: evolution of GDA and double-FTR in the 2-D toy example from multiple initial points. Middle: zoom-in near a differential DNE point. Right: zoom-in near a non-Nash LASE for the GDA algorithm.

minimization problem, making the convergence rate a less meaningful metric. Instead, we focus on the local convergence itself and the characterization of the fixed points.

6.2 General-sum linear quadratic game

The linear quadratic (LQ) game is a classic problem in multi-agent learning. It is an extension of the famous linear quadratic regulator (LQR) problem of optimal control to the multi-agent setting. Just as LQR being a simple yet important benchmark problem for studying properties of reinforcement learning algorithms, the LQ game provides valuable insights to multi-agent RL algorithms (Fazel et al., 2018; Zhang et al., 2019a).

Consider the discrete-time linear dynamical system, where $\boldsymbol{z} \in \mathbb{R}^{d_z}$ is the state, and two players provide control inputs $\boldsymbol{u} \in \mathbb{R}^{d_u}$ and $\boldsymbol{v} \in \mathbb{R}^{d_v}$ respectively.

$$\boldsymbol{z}_{t+1} = \boldsymbol{A}\boldsymbol{z}_t + \boldsymbol{B}_u\boldsymbol{u}_t + \boldsymbol{B}_v\boldsymbol{v}_t, \quad \boldsymbol{z}_0 \sim p(\boldsymbol{z}_0)$$

Each player adopts a linear state-feedback policy: $\boldsymbol{u}_t = -\boldsymbol{K}_u\boldsymbol{z}_t$, $\boldsymbol{v}_t = -\boldsymbol{K}_v\boldsymbol{z}_t$, where the parameters $\boldsymbol{K}_u \in \mathbb{R}^{d_u \times d_z}$, $\boldsymbol{K}_v \in \mathbb{R}^{d_v \times d_z}$ are to be determined by optimization. In a general-sum LQ game, each player aims to find their corresponding policy parameters $\boldsymbol{K}$ that minimizes their individual quadratic loss function f (shown in equation 4, $f_v(\boldsymbol{K}_u, \boldsymbol{K}_v)$ defined analogously using $\boldsymbol{Q}_v$ and $\boldsymbol{R}_v$).

$$f_u(\boldsymbol{K}_u, \boldsymbol{K}_v) = \mathbb{E}_{\boldsymbol{z}_0 \sim p(\boldsymbol{z}_0)}\left[\sum_{t=0}^{\infty} \boldsymbol{z}_t^\top \boldsymbol{Q}_u \boldsymbol{z}_t + \boldsymbol{u}_t^\top \boldsymbol{R}_u \boldsymbol{u}_t\right] \quad (\boldsymbol{Q}_u \succ 0,\ \boldsymbol{R}_u \succ 0) \tag{4}$$

Despite their simplicity, LQ games are challenging to optimize, because even though the loss functions are quadratic in the states and actions, they are *not* convex with respect to the player parameters $\boldsymbol{K}_u$ and $\boldsymbol{K}_v$. One straightforward algorithm choice is GDA, which we refer to as the naive policy gradient method, as the gradients are computed using the policy gradient algorithm (Williams, 1992). Importantly, Mazumdar et al. (2020a) show that in general sum LQ games, naive policy gradient almost surely avoids some Nash equilibria.

We demonstrate in general-sum LQ games, double-FTR is able to find DNE that are avoided by naive policy gradient. We use a setting mentioned in Mazumdar et al. (2020a), where $d_z = 2$, $d_u = d_v = 1$, $\boldsymbol{R}_u = \boldsymbol{R}_v = 0.01$, and

$$\boldsymbol{A} = \begin{bmatrix} 0.511 & 0.064 \\ 0.533 & 0.993 \end{bmatrix}, \boldsymbol{B}_u = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \boldsymbol{B}_v = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \boldsymbol{Q}_u = \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{Q}_v = \begin{bmatrix} 1 & 0 \\ 0 & 0.147 \end{bmatrix}.$$

The initial state $\boldsymbol{z}_0$ is set to $\begin{bmatrix} 1 & 1 \end{bmatrix}^\top$ or $\begin{bmatrix} 1 & 1.1 \end{bmatrix}^\top$ with equal probability.

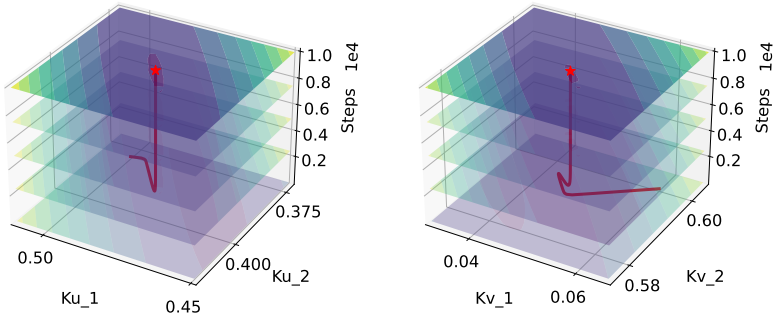
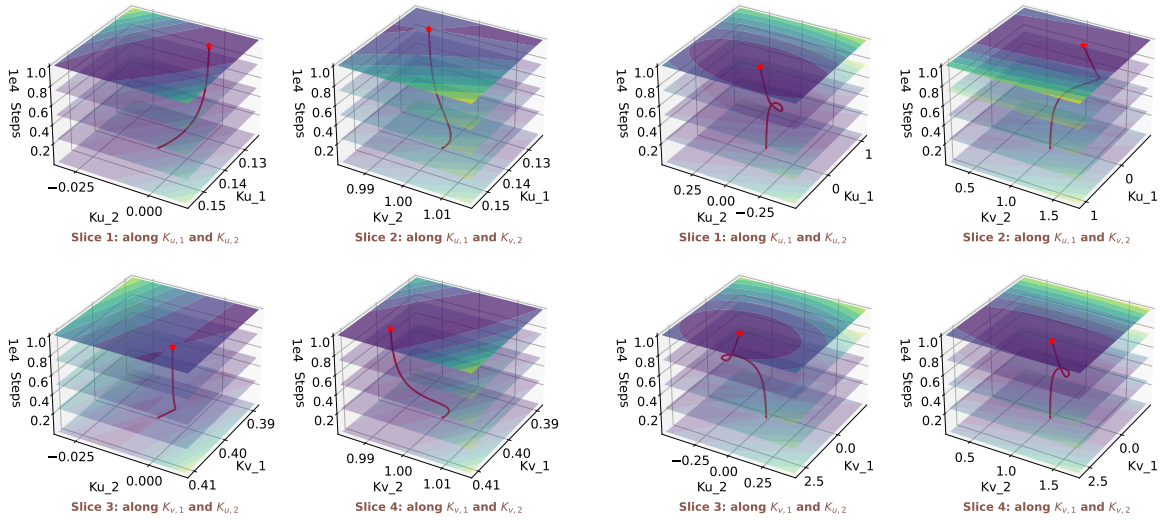


Figure 4: Evolution of the loss landscape of a general-sum linear quadratic game when optimized by double-FTR. We visualize two 2D slices $(K_{u,1}, K_{u,2})$ and $(K_{v,1}, K_{v,2})$ and the loss functions f_u and f_v respectively. Yellow represents higher value and purple represents lower value. As seen on the top levels of the illustration, the loss landscape is "bowl-shaped" at convergence, confirming that double-FTR solution satisfies the second-order conditions for DNE.



(a) Double-FTR **(b)** Naive policy gradient

Figure 5: For the general-sum linear quadratic game, we visualize how the Jacobian evolves during training. At each step, we visualize $(\boldsymbol{z} - \boldsymbol{z}_t)^\top \boldsymbol{J}_{\mathrm{GDA},t}(\boldsymbol{z} - \boldsymbol{z}_t)$, the quadratic function defined by the current $\boldsymbol{J}_{\mathrm{GDA}}$, centered at the current weight values $\boldsymbol{z}_t$. Due to the difficulty of directly visualizing a 4D function, we plot different slices of the quadratic function instead. Each contour plot shows an axis-aligned 2D slice, with the remaining 2 dimensions fixed at their corresponding values at $\boldsymbol{z}_t$. Yellow represents higher value and purple represents lower value. In both (a) and (b), the weights are initialized near the same DNE that is an *unstable* fixed point for GDA (naive policy gradient) dynamics, which is shown as saddle points in some 2D slices. (a): using double-FTR, the weights converge to the DNE. At convergence, the saddle points of $(\boldsymbol{z} - \boldsymbol{z}_t)^\top \boldsymbol{J}_{\mathrm{GDA},t}(\boldsymbol{z} - \boldsymbol{z}_t)$ still remain. (b): the gradient method avoids this unstable DNE, and converges to a stable, farther away fixed point instead.

Figure 4 and 5 shows an instance where the double-FTR is able to converge to a DNE, but naive policy gradient fails to. For both algorithms, we use the same initial policy parameters $\boldsymbol{K}_u$ and $\boldsymbol{K}_v$. Figure 4 visualizes the loss landscape for $f_u(\boldsymbol{K}_u, \boldsymbol{K}_v)$ and $f_v(\boldsymbol{K}_u, \boldsymbol{K}_v)$ when optimized by double-FTR. It confirms that the solution double-FTR converges to is indeed a DNE (the second-order condition in Definition 2.2). Figure 5a visualizes the local vector field Jacobian (i.e. $\boldsymbol{J}_{\mathrm{GDA}}$) and shows that the Jacobian contains negative eigenvalues, which makes it a saddle point for the naive policy gradient method. Indeed, naive policy gradient

(shown in Figure 5b) avoids this DNE. Instead, it eventually finds another DNE that is stable fixed point for naive policy gradient.

6.3 Generative Adversarial Networks

The Generative Adversarial Network (GAN) (Goodfellow et al., 2014) is a popular deep learning application for two-player games. The goal is to find the DNE where the generator perfectly matches the target distribution, and the discriminator is completely fooled by the generator.

In this experiment, we use the GAN framework to learn mixture of Gaussians (MoG). We use the original saturating loss function. Both the generator and the discriminator are multi-layer perceptrons with 2 hidden layers and 64 hidden units in each layer. With neural networks, directly implementing the Hessian would be computationally inefficient or infeasible. Instead, we use conjugate gradient to approximate vector products with the Hessian inverse. Details of the experiments can be found in Appendix C.

As shown in Figure 6 and 7, we apply GDA and double-FTR to learn MoG in 1D and 2D. In both cases, GDA gets stuck at a spurious equilibrium and suffers from mode collapse. In contrast, double-FTR recovers all the modes, and the generated distribution closely matches the target.

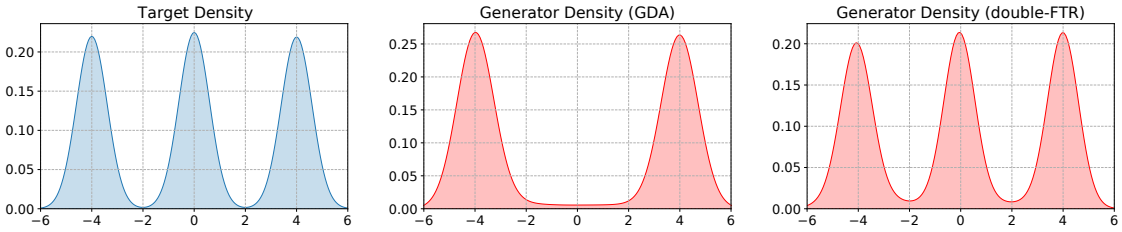


Figure 6: Mixture of Gaussians in 1D. Left: ground-truth. Middle: generator distribution learned by GDA. Right: generator distribution learned by double-FTR.

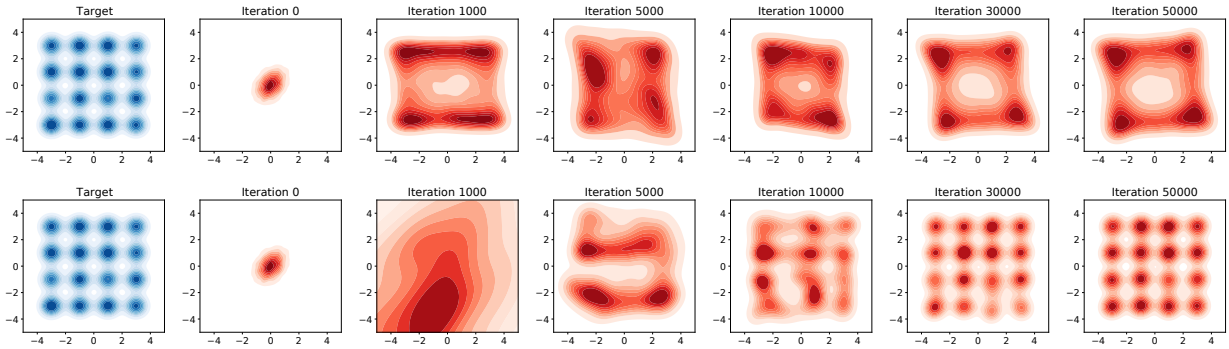


Figure 7: Mixure of Gaussians in 2D. Top: GDA suffers from mode collapse. Bottom: the generator distribution learned by double-FTR recovers all the modes.

7 Conclusion

We propose double Follow-the-Ridge (double-FTR), a gradient-based algorithm for finding differential Nash equilibria in differentiable games. We prove that under certain regularity assumptions, double-FTR locally converges to and only to differential Nash equilibria in the general-sum games, and avoids oscillation in the neighbourhood of fixed points. Furthermore, we remark that by varying the preconditioner, double-FTR leads to a broader family of algorithms that share the same convergence guarantee. Finally, we empirically verify the effectiveness of double-FTR in finding and only finding differential Nash equilibria across a broad range of problems.

Acknowledgments

The authors thank Roger Grosse and Jakob Foerster for their helpful discussions and valuable feedback.

References

Jacob Abernethy, Kevin A Lai, and Andre Wibisono. Last-iterate convergence rates for min-max optimization: Convergence of hamiltonian gradient descent and consensus optimization. In *Algorithmic Learning Theory*, pp. 3–47. PMLR, 2021.

Cem Anil, Guodong Zhang, Yuhuai Wu, and Roger Grosse. Learning to give checkable answers with prover-verifier games. *arXiv preprint arXiv:2108.12099*, 2021.

David Balduzzi, Sebastien Racaniere, James Martens, Jakob Foerster, Karl Tuyls, and Thore Graepel. The mechanics of n-player differentiable games. In *International Conference on Machine Learning*, pp. 354–363. PMLR, 2018.

Aharon Ben-Tal and Arkadi Nemirovski. Robust optimization–methodology and applications. *Mathematical programming*, 92(3):453–480, 2002.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Constantinos Daskalakis and Ioannis Panageas. The limit points of (optimistic) gradient descent in min-max optimization. *Advances in neural information processing systems*, 31, 2018.

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

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, pp. 1467–1476. PMLR, 2018.

Tanner Fiez, Benjamin Chasnov, and Lillian J Ratliff. Convergence of learning dynamics in stackelberg games. *arXiv preprint arXiv:1906.01217*, 2019.

Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pp. 1568–1577. PMLR, 2018.

Oded Galor. *Discrete dynamical systems*. Springer Science & Business Media, 2007.

Ian Gemp and Sridhar Mahadevan. Global convergence to the equilibrium of gans using variational inequalities. *arXiv preprint arXiv:1808.01531*, 2018.

Ian Gemp, Brian McWilliams, Claire Vernade, and Thore Graepel. Eigengame: Pca as a nash equilibrium. In *International Conference on Learning Representations*, 2020.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11): 139–144, 2020.

Ya-Ping Hsieh, Panayotis Mertikopoulos, and Volkan Cevher. The limits of min-max optimization algorithms: convergence to spurious non-critical sets. *arXiv preprint arXiv:2006.09065*, 2020.

Chi Jin, Praneeth Netrapalli, and Michael Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? In *International conference on machine learning*, pp. 4880–4889. PMLR, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257. PMLR, 2016.

Jason D Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I Jordan, and Benjamin Recht. First-order methods almost always avoid strict saddle points. *Mathematical programming*, 176(1):311–337, 2019.

Nicolas Loizou, Hugo Berard, Alexia Jolicoeur-Martineau, Pascal Vincent, Simon Lacoste-Julien, and Ioannis Mitliagkas. Stochastic hamiltonian gradient methods for smooth games. In *International Conference on Machine Learning*, pp. 6370–6381. PMLR, 2020.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics*, pp. 1540–1552. PMLR, 2020.

James Martens et al. Deep learning via hessian-free optimization. In *ICML*, volume 27, pp. 735–742, 2010.

Eric Mazumdar, Lillian J Ratliff, Michael I Jordan, and S Shankar Sastry. Policy-gradient algorithms have no guarantees of convergence in linear quadratic games. In *AAMAS*, 2020a.

Eric Mazumdar, Lillian J Ratliff, and S Shankar Sastry. On gradient-based learning in continuous games. *SIAM Journal on Mathematics of Data Science*, 2(1):103–131, 2020b.

Eric V Mazumdar, Michael I Jordan, and S Shankar Sastry. On finding local nash equilibria (and only local nash equilibria) in zero-sum games. *arXiv preprint arXiv:1901.00838*, 2019.

Panayotis Mertikopoulos, Bruno Lecouat, Houssam Zenati, Chuan-Sheng Foo, Vijay Chandrasekhar, and Georgios Piliouras. Optimistic mirror descent in saddle-point problems: Going the extra (gradient) mile. In *International Conference on Learning Representations*, 2018.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. *Advances in neural information processing systems*, 30, 2017.

Aryan Mokhtari, Asuman Ozdaglar, and Sarath Pattathil. A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. In *International Conference on Artificial Intelligence and Statistics*, pp. 1497–1507. PMLR, 2020.

Peter Ochs, René Ranftl, Thomas Brox, and Thomas Pock. Bilevel optimization with nonsmooth lower level problems. In *Scale Space and Variational Methods in Computer Vision: 5th International Conference, SSVM 2015, Lège-Cap Ferret, France, May 31-June 4, 2015, Proceedings 5*, pp. 654–665. Springer, 2015.

Fabian Pedregosa. Hyperparameter optimization with approximate gradient. In *International conference on machine learning*, pp. 737–746. PMLR, 2016.

Leonid Denisovich 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, 1980.

Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. In *Conference on Learning Theory*, pp. 993–1019. PMLR, 2013.

Lillian J Ratliff, Samuel A Burden, and S Shankar Sastry. Characterization and computation of local nash equilibria in continuous games. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 917–924. IEEE, 2013.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

Ernest K Ryu and Stephen Boyd. Primer on monotone operator methods. *Appl. Comput. Math*, 15(1):3–43, 2016.

Florian Schäfer and Anima Anandkumar. Competitive gradient descent. *Advances in Neural Information Processing Systems*, 32, 2019.

Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

John Von Neumann and Oskar Morgenstern. *Theory of games and economic behavior*. Princeton university press, 2007.

Heinrich Von Stackelberg. *Market structure and equilibrium*. Springer Science & Business Media, 2010.

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

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, pp. 5–32, 1992.

Guodong Zhang, Xuchan Bao, Laurent Lessard, and Roger Grosse. A unified analysis of first-order methods for smooth games via integral quadratic constraints. *Journal of Machine Learning Research*, 22:1–39, 2021a.

Guodong Zhang, Yuanhao Wang, Laurent Lessard, and Roger Grosse. Near-optimal local convergence of alternating gradient descent-ascent for minimax optimization. *arXiv preprint arXiv:2102.09468*, 2021b.

Kaiqing Zhang, Zhuoran Yang, and Tamer Basar. Policy optimization provably converges to nash equilibria in zero-sum linear quadratic games. *Advances in Neural Information Processing Systems*, 32, 2019a.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in Neural Information Processing Systems*, 32, 2019b.

A Deferred proofs

Theorem 1. *Under Assumptions 1 and 2 and with an appropriate choice of learning rate γ, $\boldsymbol{z}^*$ is a strictly stable fixed point of the double-FTR update (equation 2) if and only if it is a differential Nash equilibrium of the game $\{(f, -g), \mathbb{R}^{n+m}\}$. Furthermore, at fixed points of equation 2, all eigenvalues of the Jacobian $\boldsymbol{J}_{\text{FTR}} := \frac{\partial \boldsymbol{\omega}_{\text{FTR}}}{\partial \boldsymbol{z}}$ are real.*

Proof. To avoid notational clutter, we drop the subscripts for the Jacobian matrix of the GDA flow in this proof (i.e. we use $\boldsymbol{J}$ to refer to $\boldsymbol{J}_{\text{GDA}}$). Define $\boldsymbol{A} := \begin{bmatrix} (\nabla^2_{\boldsymbol{xx}} f)^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & -(\nabla^2_{\boldsymbol{yy}} g)^{-1} \end{bmatrix}$, we re-write equation 2 as,

$$\boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z}) = \boldsymbol{A}\boldsymbol{J}^\top \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & c\boldsymbol{I} \end{bmatrix} \boldsymbol{\omega}_{\text{GDA}}(\boldsymbol{z}).$$

Assumption 1 and 2 implies that $\boldsymbol{\omega}_{\text{FTR}} = \boldsymbol{0} \implies \boldsymbol{\omega}_{\text{GDA}} = \boldsymbol{0}$. Also, $\boldsymbol{\omega}_{\text{GDA}} = 0 \implies \boldsymbol{\omega}_{\text{FTR}} = 0$ trivially holds. Therefore, double-FTR and GDA share the same fixed points.

Next, we derive the Jacobian of the double-FTR dynamics. Define $\boldsymbol{M} := \boldsymbol{A}\boldsymbol{J}^\top \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & c\boldsymbol{I} \end{bmatrix}$ and $\tilde{\boldsymbol{J}} := \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \sqrt{c}\boldsymbol{I} \end{bmatrix} \boldsymbol{J}$. The Jacobian of $\boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z})$ is,

$$\begin{aligned} \boldsymbol{J}_{\text{FTR}} = \frac{\partial \boldsymbol{\omega}_{\text{FTR}}}{\partial \boldsymbol{z}} &= \boldsymbol{M}\frac{\partial \boldsymbol{\omega}_{\text{GDA}}}{\partial \boldsymbol{z}} + \left[\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{z}_1}\boldsymbol{\omega}_{\text{GDA}} \cdots \frac{\partial \boldsymbol{M}}{\partial \boldsymbol{z}_{m+n}}\boldsymbol{\omega}_{\text{GDA}}\right] \\ &= \boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}} + \left[\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{z}_1}\boldsymbol{\omega}_{\text{GDA}} \cdots \frac{\partial \boldsymbol{M}}{\partial \boldsymbol{z}_{m+n}}\boldsymbol{\omega}_{\text{GDA}}\right]. \end{aligned}$$

At fixed points, we have $\boldsymbol{\omega}_{\text{GDA}} = \boldsymbol{0}$, and only the first term of $\boldsymbol{J}_{\text{FTR}}$ remains.

Since $\boldsymbol{J}$ has full rank at fixed points (Assumption 2), so is $\tilde{\boldsymbol{J}}$, and $(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}$ exists. Then, notice that $\boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}}$ is similar to $(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}$. Hence, all eigenvalues of $\boldsymbol{J}_{\text{FTR}}$ are real at fixed points.

The rest of the proof consists of two directions: 1) all strictly stable fixed points of equation 2 are differential Nash equilibria (DNE); and 2) all DNE are strict stable fixed points of equation 2.

(1) Strictly stable fixed points $\implies$ DNE.

Let $\boldsymbol{z}^* = \begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{y}^* \end{bmatrix}$ be a strictly stable fixed point of equation 2, we have:

$$\boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z}^*) = \boldsymbol{0}, \quad \text{and} \quad \rho(\boldsymbol{I} - \gamma\boldsymbol{J}_{\text{FTR}}(\boldsymbol{z}^*)) < 1.$$

First-order condition for DNE Satisfied as double-FTR and GDA have the same fixed points.

Second-order condition for DNE At fixed points, $\boldsymbol{J}_{\text{FTR}} = \boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}}$. Notice that $\rho(\boldsymbol{I} - \gamma\boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}}) = \rho(\boldsymbol{I} - \gamma(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}})$. Then we have

$$-\boldsymbol{I} \prec \boldsymbol{I} - \gamma(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}} \prec \boldsymbol{I}.$$

Choose $\gamma < \frac{2}{\rho(\boldsymbol{J}_{\text{FTR}}(\boldsymbol{z}^*))}$, then the above inequality is satisfied if and only if $(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}} \succ 0$. Since positive-definiteness is preserved from congruence transformation, we have $\boldsymbol{A} \succ 0$.

Finally, $\boldsymbol{A} \succ 0 \implies (\nabla^2_{\boldsymbol{xx}} f)^{-1} \succ 0, (\nabla^2_{\boldsymbol{yy}} g)^{-1} \prec 0 \implies \nabla^2_{\boldsymbol{xx}} f \succ 0, \nabla^2_{\boldsymbol{yy}} g \prec 0$. The second-order condition for DNE is satisfied.

(2) DNE $\implies$ strictly stable fixed points.

Let $\boldsymbol{z}^* = \begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{y}^* \end{bmatrix}$ be a DNE of the game $\{(f, -g), \mathbb{R}^{n+m}\}$, we have:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) = \boldsymbol{0}, \quad \nabla_{\boldsymbol{y}} g(\boldsymbol{x}^*, \boldsymbol{y}^*) = \boldsymbol{0}, \quad \text{and} \quad \nabla^2_{\boldsymbol{xx}} f \succ 0, \quad \nabla^2_{\boldsymbol{yy}} g \prec 0.$$

First, from the update rule of double FTR, we immediately have $\boldsymbol{\omega}_{\text{FTR}}(\boldsymbol{z}^*) = \mathbf{0}$.
Then, we have shown that at fixed points, $\boldsymbol{J}_{\text{FTR}} = \boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}}$ and $(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}$ exists.
$\nabla^2_{\boldsymbol{xx}} f \succ 0,\ \nabla^2_{\boldsymbol{yy}} g \prec 0 \implies \boldsymbol{A} \succ 0 \implies (\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}} \succ 0$.
Let $0 < \gamma < \frac{2}{\rho((\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}})}$, we have

$$-\boldsymbol{I} \prec \boldsymbol{I} - \gamma(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}} \prec \boldsymbol{I}.$$

Since $\boldsymbol{J}_{\text{FTR}} = \boldsymbol{A}\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}}$ is similar to $(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}\boldsymbol{A}(\tilde{\boldsymbol{J}}^\top\tilde{\boldsymbol{J}})^{\frac{1}{2}}$, we have $\rho(\boldsymbol{I} - \gamma\boldsymbol{J}_{\text{FTR}}(\boldsymbol{z}^*)) < 1$.
Hence, $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is a fixed point of the double-FTR update (equation 2).

□

B Extension to n-player games

Our theoretical results can naturally be extended to n-player games.

Assume there are K players, each aims to minimize $f_1, \ldots, f_K$ with respect to $\boldsymbol{x}_1 \in \mathbb{R}^{n_1}, \ldots, \boldsymbol{x}_K \in \mathbb{R}^{n_K}$ respectively. We denote such an n-player game as $\{(f_1, \ldots, f_K), \mathbb{R}^{\sum_{i=1}^K n_i}\}$. To avoid notation clutter, we denote the first and second order partial derivatives as: $D_i f_k := \frac{\partial f_k}{\partial \boldsymbol{x}_i}$, $D^2_{ij} f_k := \frac{\partial^2 f_k}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}$.

Similar to Assumption 1, we make assumptions on the twice-differentiability of $f_1, \ldots, f_K$.

Assumption 3. $\forall\ \boldsymbol{x}_1 \in \mathbb{R}^{n_1}, \ldots, \boldsymbol{x}_K \in \mathbb{R}^{n_K}$, $f_1, \ldots, f_K$ are twice-differentiable, and the second derivatives are continuous. Also, $D^2_{11} f_1, \ldots, D^2_{KK} f_K$ are invertible.

We also extend the definition of differential Nash equilibrium to n-player games.

Definition B.1 (N-player Differential Nash equilibrium)**.** $(\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_K^*)$ is a differential Nash equilibrium of $\{(f_1, \ldots, f_K), \mathbb{R}^{\sum_{i=1}^K n_i}\}$ if:

- $\forall k = 1, \ldots, K,\ D_k f_k = 0$.
- $\forall k = 1, \ldots, K,\ D^2_{kk} f_k \succ 0$.

Let $\boldsymbol{z} = \begin{bmatrix}\boldsymbol{x}_1^\top & \cdots & \boldsymbol{x}_K^\top\end{bmatrix}^\top$. The gradient method for the n-player game is:

$$\boldsymbol{z}_{t+1} \leftarrow \boldsymbol{z}_t - \gamma_t \boldsymbol{\omega}_{\text{GD}}(\boldsymbol{z}_t), \quad \text{where } \boldsymbol{\omega}_{\text{GD}}(\boldsymbol{z}_t) = \begin{bmatrix} D_1 f_1(\boldsymbol{z}_t) \\ \vdots \\ D_K f_K(\boldsymbol{z}_t) \end{bmatrix}.$$

Correspondingly, the Jacobian matrix is:

$$\boldsymbol{J}_{\text{GD}} = \frac{\partial \boldsymbol{\omega}_{\text{GD}}}{\partial \boldsymbol{z}} = \begin{bmatrix} D^2_{11} f_1 & \cdots & D^2_{1K} f_1 \\ \vdots & \ddots & \vdots \\ D^2_{K1} f_K & \cdots & D^2_{KK} f_K \end{bmatrix}.$$

The n-player version of Algorithm 2 is given below.

Algorithm 3 N-player Follow-the-Ridge

Require: Number of players K, learning rate $\eta_1, \ldots, \eta_K$; number of iterations T.
1: **for** $t = 1, \ldots, T$ **do**
2: **for** $k = 1, \ldots, K$ **do**
3: $\boldsymbol{x}_{k,t+1} \leftarrow \boldsymbol{x}_{k,t} - \eta_k D_k f_k - \sum_{i=1, i\neq k}^K \eta_i (D^2_{kk} f_k)^{-1} D^2_{ki} f_i D_i f_i$
4: **end for**
5: **end for**

Writing Algorithm 3 in vector form, we have ($\gamma > 0$, $c_i := \frac{\eta_i}{\gamma}$):

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \gamma \boldsymbol{\omega}_{\mathrm{FTR}}(\boldsymbol{z}_t), \quad \boldsymbol{\omega}_{\mathrm{FTR}}(\boldsymbol{z}_t) \quad = \begin{bmatrix} (D^2_{11} f_1)^{-1} & & \\ & \ddots & \\ & & (D^2_{KK} f_K)^{-1} \end{bmatrix} \begin{bmatrix} c_1 & & \\ & \ddots & \\ & & c_K \end{bmatrix} \boldsymbol{J}^\top_{\mathrm{GD}}(\boldsymbol{z}_t) \boldsymbol{\omega}_{\mathrm{GD}}(\boldsymbol{z}_t) \tag{5}$$

Theorem 2. *Under Assumption 2 and 3 and with an appropriate choice of learning rate γ, $\boldsymbol{z}^*$ is a strictly stable fixed point of equation 5 if and only if it is a differential Nash equilibrium of the game $\{(f_1, \ldots, f_K), \mathbb{R}^{\sum_{i=1}^K n_i}\}$. Furthermore, at fixed points of equation 5, all eigenvalues of the Jacobian $\boldsymbol{J}_{\mathrm{FTR}} := \frac{\partial \boldsymbol{\omega}_{\mathrm{FTR}}}{\partial \boldsymbol{z}}$ are real.*

Proof of Theorem 2 is virtually identical with the proof of Theorem 1, with the two-player concepts swapped for their corresponding n-player counterparts.

C Experiment details

Our JAX (Bradbury et al., 2018) implementation of the experiments is attached to the submission. With GPU acceleration and just-in-time compilation supported by JAX, the experiments ran very efficiently. The total amount of compute used to produce the results is approximately 2 hours on a Tesla T4 GPU.

C.1 2-D toy example

As discussed in Section 3.2, we use a damping parameter $\lambda = 10^{-4}$ when computing the Hessian inverses in double-FTR. For LSS, since the toy example is low-dimensional, rather than implementing the two-timescale approximation, we directly discretize the following limiting ODE:

$$\dot{\boldsymbol{z}} = -\frac{1}{2}\Big(\boldsymbol{\omega}(\boldsymbol{z}) + \boldsymbol{J}^\top_{\mathrm{GDA}}\big(\boldsymbol{J}^\top_{\mathrm{GDA}}\boldsymbol{J}_{\mathrm{GDA}} + \lambda \boldsymbol{I}\big)^{-1}\boldsymbol{J}^\top_{\mathrm{GDA}}\boldsymbol{\omega}(\boldsymbol{z})\Big),$$

with a damping parameter $\lambda = 10^{-4}$. We tuned the learning rates for each curve, with values between 0.01 and 0.0005. For each curve, we optimize for $20,000$ steps.

C.2 General-sum linear quadratic game

For the general-sum LQ game, the gradient is computed analytically by solving the Algebraic Riccati Equations (ARE) (Mazumdar et al., 2020a). The gradient is given by:

$$\begin{aligned} \nabla_{\boldsymbol{K}_u} f_u(\boldsymbol{K}_u, \boldsymbol{K}_v) &= 2\big(\boldsymbol{R}_u \boldsymbol{K}_u - \boldsymbol{B}_u^\top \boldsymbol{P}_u(\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v)\big)\boldsymbol{\Sigma}_K, \\ \nabla_{\boldsymbol{K}_v} f_v(\boldsymbol{K}_u, \boldsymbol{K}_v) &= 2\big(\boldsymbol{R}_v \boldsymbol{K}_v - \boldsymbol{B}_v^\top \boldsymbol{P}_v(\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v)\big)\boldsymbol{\Sigma}_K, \end{aligned}$$

where the state covariance $\boldsymbol{\Sigma}_K = \mathbb{E}_{\boldsymbol{z}_0 \sim p(\boldsymbol{z}_0)}\big[\sum_{t=0}^{\infty} \boldsymbol{z}_t \boldsymbol{z}_t^\top\big]$, and $\boldsymbol{P}_u$, $\boldsymbol{P}_v$ are solutions of the AREs:

$$\begin{aligned} \boldsymbol{P}_u &= (\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v)^\top \boldsymbol{P}_u(\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v) + \boldsymbol{K}_u^\top \boldsymbol{R}_u \boldsymbol{K}_u + \boldsymbol{Q}_u, \\ \boldsymbol{P}_v &= (\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v)^\top \boldsymbol{P}_v(\boldsymbol{A} - \boldsymbol{B}_u\boldsymbol{K}_u - \boldsymbol{B}_v\boldsymbol{K}_v) + \boldsymbol{K}_v^\top \boldsymbol{R}_v \boldsymbol{K}_v + \boldsymbol{R}_v. \end{aligned}$$

We use fixed-point iteration to solve the ARE to a tolerance of 10^{-5}. We estimate the state covariance $\boldsymbol{\Sigma}_K$ by rolling out the dynamics for 40 steps. We differentiate through the whole gradient computation to compute the Jacobian. For Hessian inverse, we use a damping parameter $\lambda = 10^{-4}$. We use a learning rate of 0.01 and optimize for $50,000$ episodes.

C.3 Generative Adversarial Networks

For both the 1D and 2D mixture of Gaussians (MoG) experiments, we use the saturating loss originally proposed in Goodfellow et al. (2020). The generator and discriminators are both fully-connected neural networks with 2 hidden layers, 64 hidden units in each hidden layer, and `tanh` activation. A weight decay of 10^{-4} is applied on the discriminator. We optimize the neural networks using RMSProp (Tieleman et al., 2012) with learning rate 0.0002 and batch size 5000. Note that the gradient statistics computation and RMSProp preconditioning are applied before adding gradient correction terms of the double-FTR.

Similar to the discussion in Section D.4 of Wang et al. (2019), we evaluate the Hessians at "updated weight locations", i.e. $\nabla^2_{\boldsymbol{xx}} f$ and $\nabla^2_{\boldsymbol{yy}} g$ are evaluated at $\boldsymbol{x} - \eta_{\boldsymbol{x}} \nabla_{\boldsymbol{x}} f$ and $\boldsymbol{y} + \eta_{\boldsymbol{y}} \nabla_{\boldsymbol{y}} g$ respectively. We adjust the damping parameter λ using the Levenberg-Marquardt style heuristic (Martens et al., 2010):

$$\lambda \leftarrow \begin{cases} 10\lambda & \text{if } \rho \leq 0 \\ 1.5\lambda & \text{if } 0 < \rho \leq 0.5 \\ 0.9\lambda & \text{if } \rho > 0.95 \end{cases}$$

where ρ is the reduction ratio that measures how accurate the local quadratic approximation of the loss surface is, as defined in Wang et al. (2019), Section D.4.

For 1D MoG, the target distribution is a uniform mix of 3 Gaussians of standard deviation 0.1, and centered at -4, 0, 4 respectively. The latent samples are drawn from a 16-dimensional unit isotrophic Gaussian distribution. For 2D MoG, the target distribution is a uniform mix of 16 Gaussians that form a grid pattern, each with standard deviation 0.1. The latent samples are drawn from a 32-dimensional unit isotrophic Gaussian distribution.

We train the 1D MoG experiment for 25,000 steps, and the 2D MoG experiment for 50,000 steps. We observe that for the 1D MoG experiment, double-FTR experiences training instability very close to full convergence, so we apply early stopping at 25,000 steps.