Recursive Reasoning in Minimax Games: A Level k Gradient Play Method

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite the success of generative adversarial networks (GANs) in generating visually appealing images, they are notoriously challenging to train. In order to stabilize the learning dynamics in minimax games, we propose a novel recursive reasoning algorithm: Level k Gradient Play (Lv.k GP) algorithm. Our algorithm does not require sophisticated heuristics or second-order information, as do existing algorithms based on predictive updates. We show that as k increases, Lv.k GP converges asymptotically towards an accurate estimation of players' future strategy. Moreover, we justify that Lv.∞ GP naturally generalizes a line of provably convergent game dynamics which rely on predictive updates. Furthermore, we provide its local convergence property in nonconvex-nonconcave zero-sum games and global convergence in bilinear and quadratic games. By combining Lv.k GP with Adam optimizer, our algorithm shows a clear advantage in terms of performance and computational overhead compared to other methods. Using a single Nvidia RTX3090 GPU and 30 times fewer parameters than BigGAN on CIFAR-10, we achieve an FID of 10.17 for unconditional image generation within 24 hours, allowing GAN training on common computational resources to reach state-of-the-art performance.

1 Introduction

In recent years, there has been a surge of powerful models that require simultaneous optimization of several objectives. This increasing interest in multi-objective optimization arises in various domains - such as generative adversarial networks [21, 27], adversarial attacks and robust optimization [37, 7], and multi-agent reinforcement learning [36, 55] - where several agents aim at minimizing their objectives simultaneously. Games generalize this optimization framework by introducing different objectives for different learning agents, known as players. The generative adversarial network is a widely-used method of this type, which has demonstrated state-of-the-art performance in a variety of applications, including image generation [28, 52], image super-resolution [32], and image-to-image translation [10]. Despite their success at generating visually appealing images, GANs are notoriously challenging to train [40, 39]. Naive application of the gradient-based algorithm in GANs often leads to poor image diversity (sometimes manifesting as "mode collapse") [40], Poincare recurrence [39], and subtle dependency on hyperparameters [18]. An immense corpus of work is devoted to exploring and enhancing the stability of GANs, including techniques as diverse as the use of optimal transport distance [1, 22], critic gradient penalties [54], different neural network architectures [26, 6], feature matching [50], pre-trained feature space [51], and minibatch discrimination [50]. Nevertheless, architectural modifications (e.g., StyleGANs [27]) require extensive computational resources, and many theoretically appealing methods (Follow-the-ridge [56], CGD[53]) require Hessian inverse operations, which is infeasible for most GAN applications.

To stabilize the learning dynamics in GANs, many recent efforts rely on sophisticated heuristics that allow the agents to anticipate each other's next move [16, 53, 23]. This anticipation is an example of

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

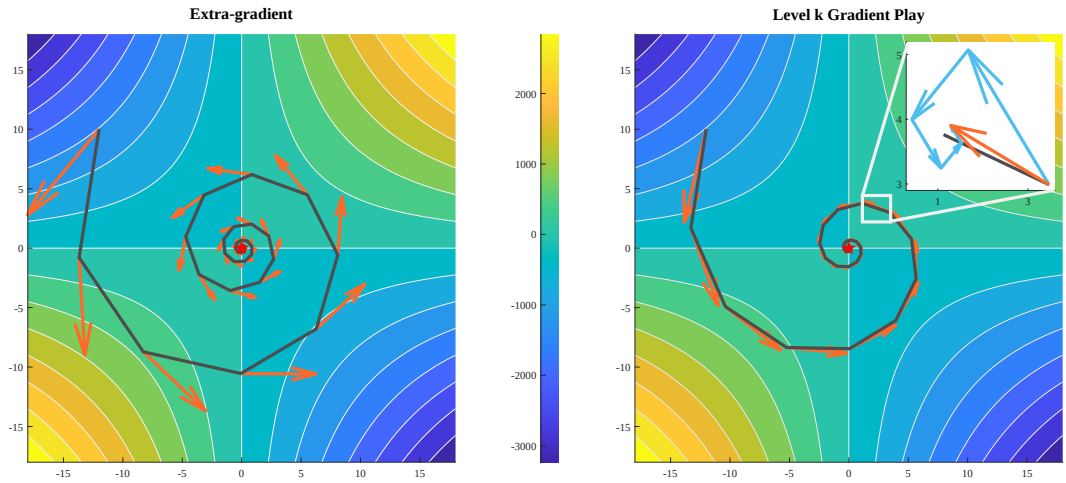

Figure 1: Illustration of predictive algorithms on: $\min_x \max_y 10xy$. Left: Extra-gradient algorithm. Right: Level k gradient play algorithm $(k = 6)$. The solution, trajectory $\{x_t, y_t\}_{t=1}^{T}$ and anticipated future state are shown with red star, black line and orange arrow, resp. The subplot in the right figure depicts how Lv.k GP predicts future states by showing its reasoning procedure with blue arrows. More steps in the reasoning process leading to better anticipations and faster convergence.

a recursive reasoning procedure in cognitive science [12]. Similar to how humans think, recursive reasoning represents the belief reasoning process where each agent considers the reasoning process of other agents, based on which it expects to make better decisions. Importantly, it enables the use of opponents that reason about the learning agent, rather than assuming fixed opponents; the process can therefore be nested in a form as 'I believe that you believe that I believe...'. Based on this intuition, we introduce a novel recursive reasoning algorithm that utilizes only gradient information to optimize GANs. Our contributions can be summarized as follows:

(i) We propose a novel algorithm: level k gradient play (Lv.k GP), which is capable of reasoning about players' future strategy. In a game, agents at Lv.k adjust their strategies in accordance with the strategies of Lv.$k-1$ agents. We justify that, while typical GANs optimizers, such as Learning with Opponent Learning Awareness (LOLA) and Symplectic Gradient Adjustment (SGA), approximate Lv.2 and Lv.3 GP, our algorithm permits higher levels of strategic reasoning. In addition, the proposed algorithm is amenable to neural network optimizers like Adam [29].

(ii) We show that, in smooth games, Lv.k GP converges asymptotically towards an accurate prediction of agents' next move. Under mutual opponent shaping, two Lv.∞ agents will naturally have a consistent view of one another if the Lv.k GP converges as k increases. Based on this idea, we provide a closed-form solution for Lv.∞ GP: the Semi-Proximal Point Method (SPPM).

(iii) We prove the local convergence property of Lv.∞ GP in nonconvex - nonconcave zero-sum games and its global convergence in bilinear and quadratic games. The theoretical analysis we present indicates that strong interactions between competing agents can increase the convergence rate of Lv.k GP agents in a zero-sum game.

(iv) By combining Lv.k GP with Adam optimizer, our algorithm shows a clear advantage in terms of performance and computational overhead compared to other methods. Using a single 3090 GPU with 30 times fewer parameters and 16 times smaller mini-batches than BigGAN [6] on CIFAR-10, we achieve an FID score [24] of 10.17 for unconditional image generation within 24 hours, allowing GAN training on common computing resources to reach state-of-the-art performance.

2 Related Works

In recent years, minimax problems have attracted considerable interest in machine learning in light of their connection with GANs. Gradient descent ascent (GDA), a generalization of gradient descent for minimax games, is the principal approach for training GANs in applications. GDA alternates

between a gradient descent step for the min-player and a gradient ascent step for the max-player. The convergence of GDA in games is far from as well understood as gradient descent in single-objective problems. Despite the impressive image quality generated by GANs, GDA fails to converge even in bilinear zero-sum games. Recent research on GDA has established a unified picture of its behavior in bilinear games in continuous and discrete-time [39, 46, 13, 41, 20]. First, [39] revealed that continuous-time GDA dynamics in zero-sum games result in Poincare recurrence, where agents return arbitrarily close to their initial state infinitely many times. Second, [3, 58] examined the discrete-time GDA dynamics, showing that simultaneous update of two players results in divergence while the agents' strategies remain bounded and cycle when agents take turns to update their strategies.

The majority of existing approaches to stabilizing GDA follow one of three lines of research. The essence of the first method is that the discriminator is trained until convergence while the generator parameters are frozen. As long as the generator changes slowly enough, the discriminator still converges in the presence of small generator perturbations. The two-timescale update rule proposed by [21, 24, 42] aims to keep the discriminator's optimality while updating the generator at an appropriate step size. [25, 14] proved that this two-timescale GDA with finite timescale separation converges towards the strict local minimax/Stackelberg equilibrium in differentiable games. [15, 56] explicitly find the local minimax equilibrium in games with secon-order optimization algorithms.

The second line of research overcomes the failure of GDA in games with predictive updates. Extra-gradient method (EG) [30] and optimistic gradient descent (OGD) [13] use the predictability of the agents' strategy to achieve better convergence property. Their variants are developed to improve the training performance of GANs [8, 19, 43]. [45] provided a unified analysis of EG and OGD, showing that they approximate the classical proximal point methods. Competitive gradient descent [53] models the agents' next move by solving a regularized bilinear approximation of the underlying game. Learning with opponent learning awareness (LOLA) and consistent opponent learning awareness (COLA) [16, 57] introduced opponent shaping to this problem by explicitly modeling the learning strategy of other agents in the game. LOLA models opponents as naive learners rather than LOLA agents, while COLA utilizes neural networks to predict opponents' next move. Lookahead-minimax [9] stabilizes GAN training by 'looking ahead' at the sequence of future states generated by an inner optimizer. In the game theory literature, recent work has proposed Clairvoyant Multiplicative Weights Update (CMWU) for regret minimization in general games [47]. Although CMWU is proposed to solve finite normal form games, which are different from unconstrained continuous games that Lv.k GP aims to solve, both CMWU and Lv.k GP share the same motivation of enabling the learning agent to update their strategy based on the opponent's future strategy. From this aspect, Lv.k GP can be viewed as a specialized variant of CMWU that is specific to the problem of two-player zero-sum games, but adapted for unconstrained continuous kernel games.

Other methods directly modify the GDA algorithm with ad-hoc modifications of game dynamics and introduction of additional regularizers. Consensus optimization (CO) [40] and gradient penalty [22, 54] improve convergence by directly minimizing the magnitude of players' gradients. Symplectic gradient adjustment (SGA) [33, 4, 17] improves convergence by disentangling convergent potential components from rotational Hamiltonian components of the vector field.

3 Preliminaries

3.1 Notation

In this paper, vectors are lower-case bold letters (e.g. $\boldsymbol{\theta}$), matrices are upper-case bold letters (e.g. $\boldsymbol{A}$). For a function $f : \mathbb{R}^d \to \mathbb{R}$, we denote its gradient by ∇f. For functions of two vector arguments $f(\boldsymbol{x}, \boldsymbol{y}) : \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \to \mathbb{R}$ we use $\nabla_x f, \nabla_y f$ to denote its partial gradients. We use $\nabla_{xx} f, \nabla_{yy} f, \nabla_{xy} f$ to denote its Hessian. A stationary point of f denotes the point where $\nabla_x f = \nabla_y f = \mathbf{0}$. We use $\|\boldsymbol{v}\|$ to denote the Euclidean norm of vector $\boldsymbol{v}$. We refer to the largest and smallest eigenvalues of a matrix $\boldsymbol{A}$ by $\lambda_{\max}(\boldsymbol{A})$ and $\lambda_{\min}(\boldsymbol{A})$, respectively. Moreover, we denote the spectral radius of matrix $\boldsymbol{A}$ by $\rho(\boldsymbol{A}) = \max\{|\lambda_1|, \ldots, |\lambda_n|\}$, i.e., the eigenvalue with largest absolute value.

3.2 Problem Definition

In order to justify the effectiveness of recursive reasoning procedure, in this paper, we consider the problem of training Generative Adversarial Networks (GANs)[21]. The GANs training strategy defines a two-player game between a generative neural network $G_{\boldsymbol{\theta}}(\cdot)$ and a discriminative neural

network $D_{\boldsymbol{\phi}}(\cdot)$. The generator takes as input random noise $\boldsymbol{z}$ sampled from a known distribution $\mathbf{P}_{\boldsymbol{z}}$, e.g., $\boldsymbol{z} \sim \mathbf{P}_{\boldsymbol{z}}$, and outputs a sample $G_{\boldsymbol{\theta}}(\boldsymbol{z})$. A discriminator takes as input a sample $\boldsymbol{x}$ (either sampled from the true distribution $\mathbf{P}_{\boldsymbol{x}}$ or from the generator) and attempts to classify it as real or fake. The goal of the generator is to fool the discriminator. The optimization of GAN is formulated as a two-player differentiable game where the generator $G_{\boldsymbol{\theta}}$ with parameter $\boldsymbol{\theta}$, and the discriminator $D_{\boldsymbol{\phi}}$ with parameters $\boldsymbol{\phi}$, aim at minimizing their own cost function $f(\boldsymbol{\theta}, \boldsymbol{\phi})$ and $g(\boldsymbol{\theta}, \boldsymbol{\phi})$ respectively, as follows:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^m} f(\boldsymbol{\theta}, \boldsymbol{\phi}) \text{ and } \min_{\boldsymbol{\phi} \in \mathbb{R}^n} g(\boldsymbol{\theta}, \boldsymbol{\phi}), \tag{1}$$

where the two function f and $g : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$. When $f = -g$ the corresponding optimization problem is called a two-player zero-sum game and it becomes a minimax problem:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^m} \max_{\boldsymbol{\phi} \in \mathbb{R}^n} f(\boldsymbol{\theta}, \boldsymbol{\phi}). \tag{Minimax}$$

In this work, we assume the cost functions have Lipschitz continuous gradients with respect to all model parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$:

Assumption 3.1. *The gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\phi})$, is $L_{\boldsymbol{\theta}\boldsymbol{\theta}}-$Lipschitz with respect to $\boldsymbol{\theta}$ and $L_{\boldsymbol{\theta}\boldsymbol{\phi}}-$Lipschitz with respect to $\boldsymbol{\phi}$ and the gradient $\nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}, \boldsymbol{\phi})$, is $L_{\boldsymbol{\phi}\boldsymbol{\phi}}-$Lipschitz with respect to $\boldsymbol{\phi}$ and $L_{\boldsymbol{\phi}\boldsymbol{\theta}}-$Lipschitz with respect to $\boldsymbol{\theta}$, i.e.,*

$$\begin{aligned}
\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_1, \boldsymbol{\phi}) - \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_2, \boldsymbol{\phi})\| &\leq L_{\boldsymbol{\theta}\boldsymbol{\theta}} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| \textit{ for all } \boldsymbol{\phi}, \\
\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\phi}_1) - \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{\phi}_2)\| &\leq L_{\boldsymbol{\theta}\boldsymbol{\phi}} \|\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2\| \textit{ for all } \boldsymbol{\theta}, \\
\|\nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}_1, \boldsymbol{\phi}) - \nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}_2, \boldsymbol{\phi})\| &\leq L_{\boldsymbol{\phi}\boldsymbol{\theta}} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| \textit{ for all } \boldsymbol{\phi}, \\
\|\nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}, \boldsymbol{\phi}_1) - \nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}, \boldsymbol{\phi}_2)\| &\leq L_{\boldsymbol{\phi}\boldsymbol{\phi}} \|\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2\| \textit{ for all } \boldsymbol{\theta}.
\end{aligned}$$

We define $L := \max\{L_{\boldsymbol{\theta}\boldsymbol{\theta}}, L_{\boldsymbol{\theta}\boldsymbol{\phi}}, L_{\boldsymbol{\phi}\boldsymbol{\theta}}, L_{\boldsymbol{\phi}\boldsymbol{\phi}}\}$.

4 Level k Gradient Play

In this section, we propose a novel recursive reasoning algorithm, Level k Gradient Play (Lv.k GP), that allows the agents to discover self-interested strategies while taking into account other agents' reasoning processes. In Lv.k GP, k steps of recursive reasoning is applied to obtain an anticipated future state $(\boldsymbol{\theta}_t^{(k)}, \boldsymbol{\phi}_t^{(k)})$, and the current states $(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$ are then updated as follows:

$$\text{Reasoning: } \begin{cases} \boldsymbol{\theta}_t^{(n)} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t^{(n-1)}) \\ \boldsymbol{\phi}_t^{(n)} = \boldsymbol{\phi}_t - \eta \nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}_t^{(n-1)}, \boldsymbol{\phi}_t) \end{cases} \quad \text{Update: } \begin{cases} \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t^{(k)} \\ \boldsymbol{\phi}_{t+1} = \boldsymbol{\phi}_t^{(k)} \end{cases} \tag{Lv.k GP}$$

We define the current state $(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$, to be the starting point $(\boldsymbol{\theta}_t^{(0)}, \boldsymbol{\phi}_t^{(0)})$, of the reasoning process. Learning agents that adopt Lv.k GP strategy are then called Lv.k agents. Lv.1 agents act naively in response to the current state using GDA dynamics and Lv.2 agents act in response to Lv.1 agents by assuming its opponent as a naive learner. Therefore, Lv.k GP allows for higher levels of strategic reasoning. The inspiration comes from how humans collaborate: humans are great at anticipating how their actions will affect others, so they frequently find out how to collaborate with other people to reach a "win-win" solution. The key to human collaboration is their ability to understand how other humans think which helps them develop strategies that benefit their collaborators. One of our main theoretical results is the following theorem, which demonstrates that agents adopting Lv.k GP can precisely predict other players' next move and reach a consensus on their future strategies:

Theorem 4.1. *Suppose Assumption 3.1 holds. Let us define $\boldsymbol{\omega}_t = [\boldsymbol{\theta}_t, \boldsymbol{\phi}_t]^T \in \mathbb{R}^{m+n}$, $\boldsymbol{\omega}_t^{(k)} = [\boldsymbol{\theta}_t^{(k)}, \boldsymbol{\phi}_t^{(k)}]^T \in \mathbb{R}^{m+n}$ and $\Delta_{\max} = 2 \times \max(\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)\|, \|\nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)\|)$. Assume $\boldsymbol{\omega}_t^{(k)}$ lie in a complete subspace of $\mathbb{R}^{m+n}$. Then for Lv.k GP we have:*

$$\|\boldsymbol{\omega}_t^{(k)} - \boldsymbol{\omega}_t^{(k-1)}\| \leq \eta \cdot (\eta L)^{(k-1)} \Delta_{\max}, \tag{2}$$

Suppose the learning rate satisfies: $\eta < (2L)^{-1}$, then the sequence $\{\boldsymbol{\omega}_t^{(k)}\}_{k=0}^{\infty}$ is a Cauchy sequence. That is, given $\epsilon > 0$, there exists N such that, if $a > b > N$ then:

$$\|\boldsymbol{\omega}_t^{(a)} - \boldsymbol{\omega}_t^{(b)}\| < \mathcal{O}(\eta^b) < \epsilon \tag{3}$$

Moreover, the sequence $\{\boldsymbol{\omega}_t^{(k)}\}_{k=0}^{\infty}$ converges to a limit $\boldsymbol{\omega}_t^*$: $\lim_{k\to\infty} \boldsymbol{\omega}_t^{(k)} = \boldsymbol{\omega}_t^*$.

In accordance with Theorem 4.1, if we define $\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t^*$, then Lv.$\infty$ GP is equivalent to the following implicit algorithm where we call it Semi-Proximal Point Method:

$$\begin{cases} \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_{t+1}) \\ \boldsymbol{\phi}_{t+1} = \boldsymbol{\phi}_t - \eta \nabla_{\boldsymbol{\phi}} g(\boldsymbol{\theta}_{t+1}, \boldsymbol{\phi}_t) \end{cases} \tag{SPPM}$$

4.1 Algorithms as an Approximation of SPPM

SPPM players arrive at a consensus by knowing precisely what their opponents' future strategies will be. Existing algorithms are not able to offer this kind of agreement. For instance, consensus optimization[40] forces the learning agents to cooperate regardless of their own benefits. Agents employ extra-gradient method[30], SGA[4], and LOLA[16] consider their opponents as naive learners, ignoring their strategic reasoning ability. CGD[53] takes into account the reasoning process of learning agents; however, it leads to an inaccurate prediction of agents in games that have cost functions with non-zero higher order derivatives ($n \geq 3$) [57]. In this section, we consider a subset of provably convergent variants of GDA in the Minimax setting, showing that, for specific choice of hyperparameters, the mentioned algorithms either approximate SPPM or approximate the approximations of SPPM:

Table 1: The update rules for the first player of SGA, LOLA, Lv.2 GP, LEAD, CGD, Lv.3 GP and SPPM in a Minimax problem and their precision as an approximation of SPPM. †The usage of zero or negative momentum has been suggested in recent works [20, 18]. For sake of comparison, we assume no momentum factor in LEAD's update, which corresponds to $\beta = 0$ in Equation 10 of [23].

Algorithm	Update Rule	Precision
SGA [4]	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta\gamma \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_\phi f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$	——
LOLA [16]	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta\delta \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_\phi f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$	——
Lv.2 GP	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t + \eta \nabla_\phi f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t))$	$\mathcal{O}(\eta^2)$
LEAD† [23]	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \alpha \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)(\boldsymbol{\phi}_t - \boldsymbol{\phi}_{t-1})$	——
CGD [53]	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)(\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}_t)$	$\mathcal{O}(\eta^3)$
Lv.3 GP	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t + \eta \nabla_\phi f(\boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t), \boldsymbol{\phi}_t))$	$\mathcal{O}(\eta^3)$
SPPM (Lv.∞ GP)	$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_{t+1})$	0

In Table 1, we compare the orders of precision of different algorithms as an approximation of SPPM in Minimax games with infinitely differentiable objective functions. In accordance with Equation (3) of Theorem 4.1, Lv.k GP is an $\mathcal{O}(\eta^k)$ approximation of SPPM. In order to analyze how well existing algorithms approximate SPPM, we consider the first-order Taylor approximation to SPPM:

$$\begin{cases} \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta^2 \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_\phi f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta^2 \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_{\phi\theta} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)(\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t) \\ \underbrace{\boldsymbol{\phi}_{t+1} = \boldsymbol{\phi}_t + \eta \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) - \eta^2 \nabla_{\phi\theta} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_\theta f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)}_{1^{st} \text{ order approximation of Lv.2 GP}} - \eta^2 \nabla_{\phi\theta} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) \nabla_{\theta\phi} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)(\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}_t) \end{cases}$$

Under-brace terms correspond to the first-order Taylor approximation of Lv.2 GP. For an appropriate choice of hyperparameters, SGA ($\gamma = \eta$) and LOLA ($\delta = \eta$) are identical to the first-order Taylor approximation of Lv.2 GP, where each agent models their opponent as a naive learner. Hence, we list them above the Lv.2 GP, which approximates SPPM up to $\mathcal{O}(\eta^2)$. CGD exactly recovers the first-order Taylor approximation of SPPM.[1] In games with cost functions that have non-negative higher order derivatives ($n \geq 3$), the remaining term in SPPM's first-order approximation is an error of magnitude $\mathcal{O}(\eta^3)$, which means that CGD's accuracy is in the same range as that of Lv.3 GP. In bilinear and quadratic games where the objective function is at most twice differentiable, CGD

[1]The CGD update for the max player $\boldsymbol{\phi}$ is $\boldsymbol{\phi}_{t+1} = \boldsymbol{\phi}_t + \eta \nabla_\phi f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t) + \eta \nabla_{\phi\theta} f(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t)(\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t)$. If we substitute $(\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}_t)$ into $\boldsymbol{\theta}$'s update (and substitute $(\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t)$ into $\boldsymbol{\phi}$'s update, respectively), we arrive at the first-order approximation of SPPM. See A.5 for derivation details.

is equivalent to SPPM. The distinction between LEAD ($\alpha = \eta$) and CGD can be understood by considering their update rules. LEAD is an explicit method where opponents' potential next strategies are anticipated based on their most recent move $(\boldsymbol{\phi}_t - \boldsymbol{\phi}_{t-1})$. On the contrary, CGD accounts for this anticipation in an implicit manner, $(\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}_t)$, where the future states appear in current states' update rules. Therefore, the computation of CGD updates requires solving an function involving additional Hessian inverse operations. A numerical justification is also provided in Table 2, showing that the approximation accuracy of Lv.k GP improves as k increases.

5 Convergence Property

In Theorem 4.1, we have analytically proved that Lv.k GP convergences asymptotically towards SPPM, we will use this result to study the convergence property of Lv.k GP in games based on our analysis of SPPM. The local convergence of SPPM in a non convex - non concave game can be analyzed via the spectral radius of the game Jacobian around a stationary point:

Theorem 5.1. *Consider the (Minimax) problem under Assumption 3.1 and Lv.k GP. Let $(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ be a stationary point. Suppose $\boldsymbol{\theta}_t - \boldsymbol{\theta}^*$ not in kernel of $\nabla_{\phi\theta} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$, $\boldsymbol{\phi}_t - \boldsymbol{\phi}^*$ not in kernel of $\nabla_{\theta\phi} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ and $\eta < (L)^{-1}$. There exists a neighborhood $\mathcal{U}$ of $(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ such that if SPPM started at $(\boldsymbol{\theta}_0, \boldsymbol{\phi}_0) \in \mathcal{U}$, the iterates $\{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t\}_{t\geq 0}$ generated by SPPM satisfy:*

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|^2 + \|\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}^*\|^2 \leq \frac{\rho^2(\boldsymbol{I} - \eta\nabla_{\theta\theta} f^*)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|^2 + \rho^2(\boldsymbol{I} + \eta\nabla_{\phi\phi} f^*)\|\boldsymbol{\phi}_t - \boldsymbol{\phi}^*\|^2}{1 + \eta^2\lambda_{\min}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*)}$$

where $f^* = f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$. Moreover,for any η satisfying:

$$\frac{\max(\rho^2(\boldsymbol{I} - \eta\nabla_{\theta\theta} f^*), \rho^2(\boldsymbol{I} + \eta\nabla_{\phi\phi} f^*))}{1 + \eta^2\lambda_{\min}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*)} < 1, \tag{4}$$

SPPM converges asymptotically to $(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$.

Remark 5.1. *Following the same condition as in 5.1, the iterates generated by Lv.$2k$ GP satisfies:*

$$\begin{aligned} &\|\boldsymbol{\theta}_t^{(2k)} - \boldsymbol{\theta}^*\|^2 + \|\boldsymbol{\phi}_t^{(2k)} - \boldsymbol{\phi}^*\|^2 \\ &\leq a\left(\frac{\rho^2(\boldsymbol{I} - \eta\nabla_{\theta\theta} f^*)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|^2 + \rho^2(\boldsymbol{I} + \eta\nabla_{\phi\phi} f^*)\|\boldsymbol{\phi}_t - \boldsymbol{\phi}^*\|^2}{1 + \eta^2\lambda_{\min}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*)}\right) + b \end{aligned} \tag{5}$$

where

$$a = \begin{cases} \frac{1+(\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k}{1-(\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k} & \textit{for odd } k, \\ \frac{1-(\eta^2\lambda_{\min}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k}{1-(\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k} & \textit{for even } k \end{cases} \quad \textit{and } b = \frac{(\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k(1 - (\eta^2\lambda_{\min}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k)}{1 - (\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*))^k}$$

We assume that the difference between the trajectory $\{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t\}_{t\geq 0}$ and the stationary point $(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ is not in the kernel of $\nabla_{\phi\theta} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ and $\nabla_{\theta\phi} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ for the sake of simplicity. Detailed proofs without this assumption are provided in Appendix. Following the same setting as Theorem 4.1, we have $\eta < L^{-1}$, which ensures that $\eta^2\lambda_{\max}(\nabla_{\theta\phi} f^* \nabla_{\phi\theta} f^*) < 1$. Therefore, as $k \to \infty$, $a \to 1$ and $b \to 0$ in Remark 5.1, and as such Lv.k GP has similar local convergence properties to SPPM. Figure 2 illustrates this property in a quadratic game. Lv.k GP may behave differently than SPPM at lower values of k, but as k increases, both max step size $\eta_{\max,k}$ and distance to equilibrium for a fixed number of iterations under $\eta_{\max,\infty}$ converges to that of SPPM. Thus, we observe that Lv.k GP is empirically similar to SPPM at higher values of k.

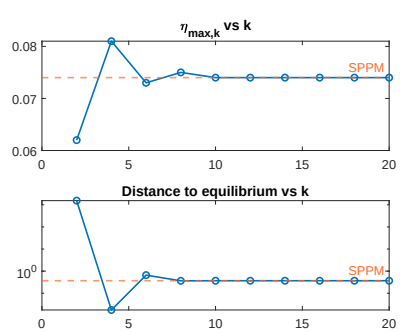

Figure 2: **Top**: max step size against k, **Right**: distance to equilibrium against k. In both figures the red dashed line represents the value for SPPM.

We study the global convergence property of SPPM by analyzing its behavior in bilinear and quadratic games. Consider the following bilinear game:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \max_{\boldsymbol{\phi}\in\mathbb{R}^n} \boldsymbol{\theta}^T \boldsymbol{M} \boldsymbol{\phi} \qquad \text{(Bilinear game)}$$

where $\boldsymbol{M}$ is a full rank matrix. The following theorem summarizes SPPM's convergence property:

Theorem 5.2. *Consider the Bilinear game and the SPPM method. Further, we define* $r_t = \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|^2 + \|\boldsymbol{\phi}_t - \boldsymbol{\phi}^*\|^2$. *Then, for any* $\eta > 0$, *the iterates* $\{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t\}_{t\geq 0}$ *generated by SPPM satisfy*

$$r_{t+1} \leq \frac{1}{1 + \eta^2 \lambda_{\min}(\boldsymbol{M}^T \boldsymbol{M})} r_t. \tag{6}$$

It is worth noting that the convergence property of Lv.k GP in bi-linear game has been studied in [2]. Furthermore, we study the convergence property of SPPM in the Quadratic game:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \max_{\boldsymbol{\phi}\in\mathbb{R}^n} \boldsymbol{\theta}^T \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{\phi}^T \boldsymbol{B}\boldsymbol{\phi} + \boldsymbol{\theta}^T \boldsymbol{C}\boldsymbol{\phi} \tag{Quadratic game}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ is symmetric and negative definite and the interaction term $\boldsymbol{C} \in \mathbb{R}^{n\times n}$ is full rank. SPPM in quadratic games converges with the following rate:

Theorem 5.3. *Consider the Quadratic game and the SPPM. Then, for any* $\eta > 0$, *the iterates* $\{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t\}_{t\geq 0}$ *generated by SPPM satisfy*

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|^2 + \|\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}^*\|^2 \leq \frac{\rho^2(1-\eta\boldsymbol{A})\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|^2 + \rho^2(1+\eta\boldsymbol{B})\|\boldsymbol{\phi}_t - \boldsymbol{\phi}^*\|^2}{1 + \eta^2\lambda_{\min}(\boldsymbol{C}^T\boldsymbol{C})} \tag{7}$$

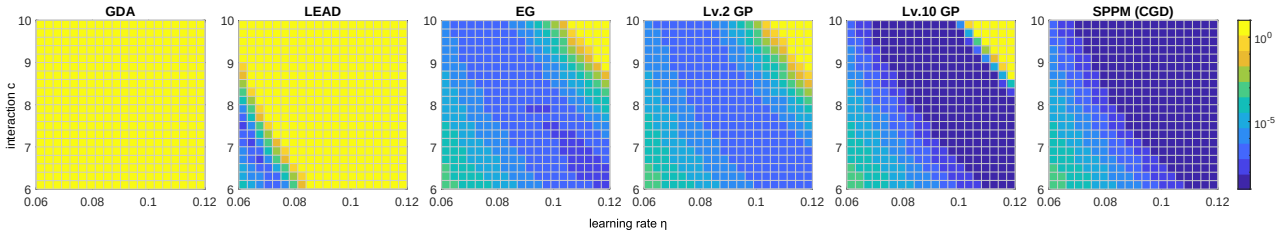

Figure 3: A grid of experiments for different algorithms with different values of interaction c and learning rates η. The color in each cell indicates the distance to the equilibrium after 100 iterations. Note that the CGD update is equivalent to SPPM in quadratic games.

Theorem 5.1,5.2 and 5.3 indicate that stronger interaction $\nabla_{\theta\phi} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ improve the convergence rate towards the stationary points. By contrast, in existing modifications of GDA, the step size is chosen in inversely proportional to the interaction term $\nabla_{\theta\phi} f(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ [40, 13, 34]. In Figure 3, we showcase the effect of interaction on different algorithms in the Quadratic game setup with dimension $n = 5$ and the interaction matrix is defined as $\boldsymbol{C} = c\boldsymbol{I}$. Stronger interaction corresponds to higher values of c. A key difference between SPPM and Lv.k GP in the experiments is that, SPPM converges with any step size - and so arbitrarily fast - while it is not the case for Lv.k GP. In order to approximate SPPM, one needs Lv.k GP to be a contraction and so have $\eta < L^{-1}$. This result implies an additional bound on step sizes for Lv.k GP. As long as the constraint on η is satisfied, stronger interaction only improves convergence for higher order Lv.k GP while all other algorithms quickly diverge as η and c increase.

6 Experimental Results

In this section, we discuss our implementation of Lv.k GP algorithm for training GANs. Its performance is evaluated on 8-Gaussians and two representative datasets CIFAR-10 and STL-10.

6.1 Level k Adam

We propose to combine Lv.k GP with the Adam optimizer [29]. Preliminary experiments find that Lv.k Adam to converge much faster than Lv.k GP, see A.8 for experiment results. A detailed pseudo-code for Level k Adam (Lv.k Adam) on GAN training with loss functions $\mathcal{L}_G$ and $\mathcal{L}_D$ is given in Algorithm 1. For the Adam optimizer, there are several possible choices on how to update the moments. This choice can lead to different algorithms in practice. Unlike [19] where the moments are updated on the fly, in Algorithm 1, we keep the moments fixed in the reasoning steps and update it together with model parameters. In Table 2, our experiment result suggests that the proposed Lv.k Adam algorithm converges asymptotically as the number of reasoning steps k increases.

Algorithm 1: Level k Adam: proposed Adam with recursive reasoning steps

Input: Stopping time T, reasoning steps k, learning rate $\eta_{\boldsymbol{\theta}}, \eta_{\boldsymbol{\phi}}$, decay rates for momentum estimates β_1, β_2, initial weight $(\boldsymbol{\theta}_0, \boldsymbol{\phi}_0)$, $\boldsymbol{P_x}$ and $\boldsymbol{P_z}$ real and noise-data distributions, losses $\mathcal{L}_G(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}, \boldsymbol{z})$ and $\mathcal{L}_D(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}, \boldsymbol{z})$, $\epsilon = 1e-8$.

Parameters : Initial parameters: $\boldsymbol{\theta}_0, \boldsymbol{\phi}_0$
Initialize first moments: $\boldsymbol{m}_{\theta,0} \leftarrow 0, \boldsymbol{m}_{\phi,0} \leftarrow 0$
Initialize second moments: $\boldsymbol{v}_{\theta,0} \leftarrow 0, \boldsymbol{v}_{\phi,0} \leftarrow 0$

for *t=0,...,T-1* **do**
  Sample new mini-batch: $\boldsymbol{x}, \boldsymbol{z} \sim \boldsymbol{P_x}, \boldsymbol{P_z}$,
  $\boldsymbol{\theta}_t^{(0)} \leftarrow \boldsymbol{\theta}_t, \boldsymbol{\phi}_t^{(0)} \leftarrow \boldsymbol{\phi}_t$,
  for *n=1,...,k* **do**
    Compute stochastic gradient:
    $\boldsymbol{g}_{\boldsymbol{\theta},t}^{(n)} = \nabla_\theta \mathcal{L}_G(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t^{(n-1)}, \boldsymbol{x}, \boldsymbol{z}); \boldsymbol{g}_{\boldsymbol{\phi},t}^{(n)} = \nabla_\phi \mathcal{L}_D(\boldsymbol{\theta}_t^{(n-1)}, \boldsymbol{\phi}_t, \boldsymbol{x}, \boldsymbol{z})$
    Update estimate of first moment:
    $\boldsymbol{m}_{\theta,t}^{(n)} = \beta_1 \boldsymbol{m}_{\theta,t-1} + (1-\beta_1)\boldsymbol{g}_{\theta,t}^{(n)}; \boldsymbol{m}_{\phi,t}^{(n)} = \beta_1 \boldsymbol{m}_{\phi,t-1} + (1-\beta_1)\boldsymbol{g}_{\phi,t}^{(n)}$
    Update estimate of second moment:
    $\boldsymbol{v}_{\theta,t}^{(n)} = \beta_2 \boldsymbol{v}_{\theta,t-1} + (1-\beta_2)(\boldsymbol{g}_{\theta,t}^{(n)})^2; \boldsymbol{v}_{\phi,t}^{(n)} = \beta_2 \boldsymbol{v}_{\phi,t-1} + (1-\beta_2)(\boldsymbol{g}_{\phi,t}^{(n)})^2$
    Correct the bias for the moments:
    $\hat{\boldsymbol{m}}_{\theta,t}^{(n)} = \frac{\boldsymbol{m}_{\theta,t}^{(n)}}{(1-\beta_1^t)}, \hat{\boldsymbol{m}}_{\phi,t}^{(n)} = \frac{\boldsymbol{m}_{\phi,t}^{(n)}}{(1-\beta_1^t)}; \hat{\boldsymbol{v}}_{\theta,t}^{(n)} = \frac{\boldsymbol{v}_{\theta,t}^{(n)}}{(1-\beta_2^t)}, \hat{\boldsymbol{v}}_{\phi,t}^{(n)} = \frac{\boldsymbol{v}_{\phi,t}^{(n)}}{(1-\beta_2^t)}$
    Perform Adam update: $\boldsymbol{\theta}_t^{(n)} = \boldsymbol{\theta}_t - \eta_\theta \frac{\hat{\boldsymbol{m}}_{\theta,t}^{(n)}}{\sqrt{\hat{\boldsymbol{v}}_{\theta,t}^{(n)}}+\epsilon}; \boldsymbol{\phi}_t^{(n)} = \boldsymbol{\phi}_t - \eta_\phi \frac{\hat{\boldsymbol{m}}_{\phi,t}^{(n)}}{\sqrt{\hat{\boldsymbol{v}}_{\phi,t}^{(n)}}+\epsilon}$
  $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t^{(k)}, \boldsymbol{\phi}_{t+1} \leftarrow \boldsymbol{\phi}_t^{(k)}$;
  $\boldsymbol{m}_{\theta,t} \leftarrow \boldsymbol{m}_{\theta,t}^{(k)}, \boldsymbol{m}_{\phi,t} \leftarrow \boldsymbol{m}_{\phi,t}^{(k)}$;
  $\boldsymbol{v}_{\theta,t} \leftarrow \boldsymbol{v}_{\theta,t}^{(k)}, \boldsymbol{v}_{\phi,t} \leftarrow \boldsymbol{v}_{\phi,t}^{(k)}$

6.2 8-Gaussians

In our first experiment, we evaluate Lv.k Adam on generating a mixture of 8-Gaussians with standard deviations equal to 0.05 and modes uniformly distributed around the unit circle. We use a two layer multi-layer perceptron with ReLU activations, latent dimension of 64 and batch size of 128. The generated distribution is presented in Figure 4.

Table 2: The difference between two states generated by Lv.k Adam and Lv.k GP averaged over 100 steps. †The difference is smaller than machine precision.

$\frac{1}{100}\sum_{t=1}^{100} r_t^{(k)}$	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$
Lv.k Adam	1.04×10^{-1}	1.60×10^{-2}	2.91×10^{-3}	6.08×10^{-4}	1.68×10^{-4}
Lv.k GP	9.79×10^{-9}	3.84×10^{-15}	1.31×10^{-17}	1.68×10^{-19}	$\approx 0^{\dagger}$

In addition to presenting the mode coverage of the generated distribution after training, we also study the convergence of the reasoning steps of Lv.k Adam and Lv.k GP. Let us define the difference between the states of Lv.k and Lv.$k-1$ agents at time t as:

$$r_t^{(k)} = \|\boldsymbol{\theta}_t^{(k)} - \boldsymbol{\theta}_t^{(k-1)}\|^2 + \|\boldsymbol{\phi}_t^{(k)} - \boldsymbol{\phi}_t^{(k-1)}\|^2. \quad (8)$$

We measure the difference averaged over the first 100 iterations, $\frac{1}{100}\sum_{t=1}^{100} r_t^k$, for Lv.10 Adam ($\eta = 10^{-4}$) and Lv.10 GP ($\eta = 10^{-2}$). The result presented in Table2 demonstrates that both Lv.k GP and Lv.k Adam are con-

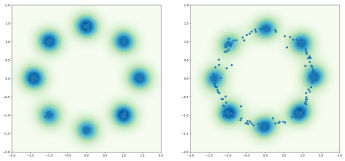

Figure 4: **Left**: real distribution, **Right**: generated distribution

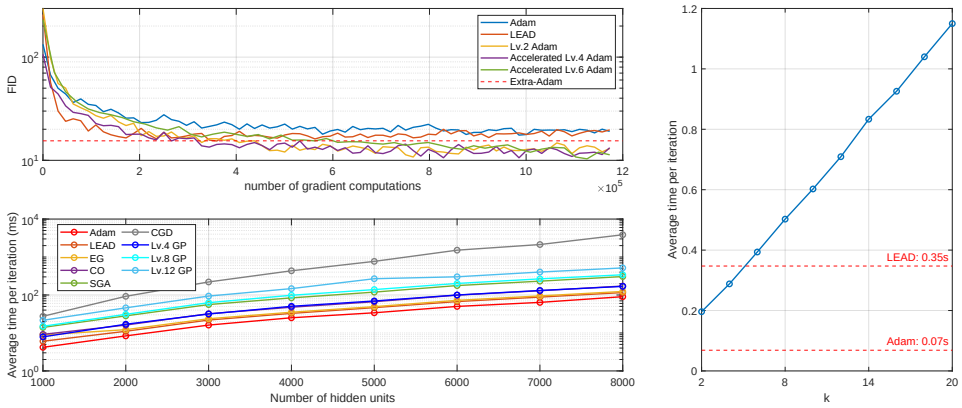

Figure 5: **Top Left**: Change of FID scores over 1.2 million gradient computations for Adam, LEAD, Lv.2, Lv.4 and Lv.6 Adam on CIFAR-10 with SNGAN. Note for Lv.4 and Lv.6 Adam, we use the accelerated implementation introduced in Appendix A.10. We also use dashed line to highlight the value obtained with Extra-Adam as a reference. **Bottom Left**: Average computational cost per iteration on 8-Gaussians experiment for MLPs with different widths. **Right**: Average computational cost per iteration on CIFAR-10 for Lv.k-Adam with different k. The values for LEAD (2.88) and Adam (14.65) are highlighted by dashed line.

verging as k increases. Moreover, the estimation precision of Lv.k GP improves rapidly and converges to 0 within finite steps, making it an accurate estimation for SPPM.

Table 3: FID and Inception scores of different algorithms and architectures on CIFAR-10. Results are averaged over 3 runs. †We re-evaluate its performance on the official implementation of FID.

Algorithm	Architecture	FID ↓	IS ↑
Adam [29]	BigGAN [6]	14.73	**9.22**
Adam [29]	StyleGAN2 [59]	11.07	9.18
Adam [29]	SN-GAN [44]	21.70 ± 0.21	7.60 ± 0.06
Unrolled GAN [42, 9]	SN-GAN [44]	17.51 ± 1.08	——
Extra-Adam [19, 38, 9]	SN-GAN [44]	15.47 ± 1.82	——
LEAD† [23]	SN-GAN [44]	14.45 ± 0.45	——
LA-AltGAN [9]	SN-GAN [44]	12.67 ± 0.57	8.55 ± 0.04
ODE-GAN(RK4) [48]	SN-GAN [44]	11.85 ± 0.21	8.61 ± 0.06
Lv.6 Adam	SN-GAN [44]	$\mathbf{10.17 \pm 0.16}$	$\mathbf{8.78 \pm 0.06}$

6.3 Image Generation Experiments

We evaluate the effectiveness of our Lv.k Adam algorithm on unconditional generation of CIFAR-10 [31]. We use the Inception score (the higher the better) [50] and the Fréchet Inception distance (the lower the better) [24] as performance metrics for image synthesis. For architecture, we use the SN-GAN architecture based on [44]. For baselines, we compare the performance of Lv.6 Adam to that of other first-order and second-order optimization algorithms with the same SN-GAN architecture, and to state-of-the-art models trained with Adam. For Lv.k Adam, we use $\beta_1 = 0$ and $\beta_2 = 0.9$ for all experiments. We use different learning rates for the generator ($\eta_\theta = 4e-5$) and the discriminator ($\eta_\phi = 2e-4$). We train Lv.k Adam with batch size 128 for 600 epochs. For testing, we use an exponential moving average of the generator's parameters with averaging factor $\beta = 0.999$.

Table 4: FID scores for the different loss functions with $k = \{2, 4, 6\}$.

	Lv.2 Adam	Lv.4 Adam	Lv.6 Adam
Non-saturated loss	11.33 ± 0.18	11.62 ± 0.25	10.93 ± 0.24
Hinge loss	10.68 ± 0.20	10.33 ± 0.22	10.17 ± 0.16

In table 3, we present the performance of our method and baselines. The best FID score and Inception score, 10.17 ± 0.16 and 8.78 ± 0.06, on SN-GAN are obtained with our Lv.6 Adam. We also outperform BigGAN and StyleGAN2 in terms of FID score. Notably, our model has 5.1M parameters in total, and is trained with a small batch size of 128, whereas BigGAN uses 158.3M parameters and a batch size of 2048.

The effect of k and losses: We evaluate values of $k = \{2, 4, 6\}$ on non-saturated loss [21] (non-zero-sum formulation) and hinge loss [35] (zero-sum formulation). The result is presented in Table 4. Remarkably, our experiments demonstrate that few steps of recursive reasoning can result in significant performance gain comparing to existing GAN optimizers. The gradual improvements in the FID scores justify the idea that better estimation of opponents' next move improves performance. Moreover, we observe performance gains in both zero-sum and non-zero-sum formulations which supplement our theoretical convergence guarantees in zero-sum games.

Experiment results on STL-10: To test whether the proposed Lv.k Adam optimizer works on higher resolution images, we evaluate its performance on the STL-10 dataset [11] with $3 \times 48 \times 48$ resolutions. In our experiments, Lv.6 Adam obtained an averaged FID of 25.43 ± 0.18 which outperforms that of the Adam optimizer, 30.25 ± 0.26, using the same SN-GAN architecture.

Memory and computation cost: Compared to SGD, Lv.k Adam requires the same extra memory as the EG method (one additional set of parameters per player). The relative cost of one iteration versus SGD is a factor of k and the computational cost increases linearly as k increases, we illustrate this relationship in Figure 5 (right). We provide an accelerated version of Lv.k Adam in A.8 which reduces the computation cost by half for $k > 2$. In Figure 5 (top-left), we compare the FID scores obtained by Lv.k Adam, Adam, and LEAD on CIFAR-10 over the same number of gradient computations. LEAD, Lv4 Adam, and Lv6 Adam all outperform Adam in this experiment. Lv4 Adam outperforms LEAD after 2×10^5 gradient computations. Our method is also compared with different algorithms on the 8-Gaussian problem in terms of its computational cost. On the same architecture with different widths, Figure 5 (bottom-left) illustrates the wall-clock time per computation for different algorithms. We observe that the computational cost of Lv.k Adam while being much lower than CGD, is similar to LEAD, SGA and CO which involve JVP operations. Each run on CIFAR-10 dataset takes $30 \sim 33$ hours on a Nvidia RTX3090 GPU. Each experiment on STL-10 takes $48 \sim 60$ hours on a Nvidia RTX3090 GPU.

7 Conclusion and Future Work

This paper proposes a novel algorithm: Level k gradient play, capable of reasoning about players' future strategies. We achieve an average FID score of 10.17 for unconditional image generation on CIFAR-10 dataset, allowing GAN training on common computing resources to reach state-of-the-art performance. Our results suggest that Lv.k GP is a flexible add-on that can be easily attached to existing GAN optimizers (e.g., Adam) and provides noticeable gains in performance and stability. In future work, we will examine the effectiveness of our approach on more complicated GAN designs, such as Progressive GANs [26] and StyleGANs [27], where optimization plays a more significant role. Additionally, we intend to examine the convergence property of Lv.k GP in games with more than two player in the future.

Broader Impact Our work introduces a novel optimizer that improves the performance of GANs and may reduce the amount of hyperparameter tuning required by practitioners of generative modeling. Generative models have been used to create illegal content(a.k.a. deepfakes [5]). There is risk of negative social impact resulting from malicious use of the proposed methods.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[2] Waïss Azizian, Ioannis Mitliagkas, Simon Lacoste-Julien, and Gauthier Gidel. A tight and unified analysis of gradient-based methods for a whole spectrum of differentiable games. In *International Conference on Artificial Intelligence and Statistics*, pages 2863–2873. PMLR, 2020.

[3] James P Bailey, Gauthier Gidel, and Georgios Piliouras. Finite regret and cycles with fixed step-size via alternating gradient descent-ascent. In *Conference on Learning Theory*, pages 391–407. PMLR, 2020.

[4] 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*, pages 354–363. PMLR, 2018.

[5] John Brandon. Terrifying high-tech porn: Creepy 'deepfake' videos are on the rise. *Fox News*, 2018.

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[8] Tatjana Chavdarova, Gauthier Gidel, François Fleuret, and Simon Lacoste-Julien. Reducing noise in GAN training with variance reduced extragradient. *Advances in Neural Information Processing Systems*, 32, 2019.

[9] Tatjana Chavdarova, Matteo Pagliardini, Sebastian U Stich, François Fleuret, and Martin Jaggi. Taming GANs with lookahead-minmax. *arXiv preprint arXiv:2006.14567*, 2020.

[10] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[11] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[12] Michael C Corballis. The uniqueness of human recursive thinking: the ability to think about thinking may be the critical attribute that distinguishes us from all other species. *American Scientist*, 95(3):240–248, 2007.

[13] Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training GANs with optimism. *arXiv preprint arXiv:1711.00141*, 2017.

[14] Tanner Fiez and Lillian J Ratliff. Local convergence analysis of gradient descent ascent with finite timescale separation. In *Proceedings of the International Conference on Learning Representation*, 2021.

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

[16] Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017.

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

[18] Ian Gemp and Brian McWilliams. The Unreasonable Effectiveness of Adam on Cycles. In *NeurIPS Workshop on Bridging Game Theory and Deep Learning*, 2019.

[19] Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent, and Simon Lacoste-Julien. A variational inequality perspective on generative adversarial networks. *arXiv preprint arXiv:1802.10551*, 2018.

[20] 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*, pages 1802–1811. PMLR, 2019.

[21] 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.

[22] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. *Advances in neural information processing systems*, 30, 2017.

[23] Reyhane Askari Hemmat, Amartya Mitra, Guillaume Lajoie, and Ioannis Mitliagkas. Lead: Least-action dynamics for min-max optimization. *arXiv preprint arXiv:2010.13846*, 2020.

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[25] Chi Jin, Praneeth Netrapalli, and Michael Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? In *International Conference on Machine Learning*, pages 4880–4889. PMLR, 2020.

[26] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[27] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[28] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

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

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

[31] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[32] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[33] Alistair Letcher, David Balduzzi, Sébastien Racaniere, James Martens, Jakob Foerster, Karl Tuyls, and Thore Graepel. Differentiable game mechanics. *The Journal of Machine Learning Research*, 20(1):3032–3071, 2019.

[34] 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*, pages 907–915. PMLR, 2019.

[35] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[36] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[37] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[38] 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. *arXiv preprint arXiv:1807.02629*, 2018.

[39] Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. Cycles in adversarial regularized learning. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2703–2717. SIAM, 2018.

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

[41] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

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

[43] Konstantin Mishchenko, Dmitry Kovalev, Egor Shulgin, Peter Richtárik, and Yura Malitsky. Revisiting stochastic extragradient. In *International Conference on Artificial Intelligence and Statistics*, pages 4573–4582. PMLR, 2020.

[44] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[45] 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*, pages 1497–1507. PMLR, 2020.

[46] Christos Papadimitriou and Georgios Piliouras. From Nash equilibria to chain recurrent sets: An algorithmic solution concept for game theory. *Entropy*, 20(10):782, 2018.

[47] Georgios Piliouras, Ryann Sim, and Stratis Skoulakis. Optimal no-regret learning in general games: Bounded regret with unbounded step-sizes via clairvoyant mwu. *arXiv preprint arXiv:2111.14737*, 2021.

[48] Chongli Qin, Yan Wu, Jost Tobias Springenberg, Andy Brock, Jeff Donahue, Timothy Lillicrap, and Pushmeet Kohli. Training generative adversarial networks by solving ordinary differential equations. *Advances in Neural Information Processing Systems*, 33:5599–5609, 2020.

[49] R Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.

[50] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in neural information processing systems*, 29, 2016.

[51] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected GANs converge faster. *Advances in Neural Information Processing Systems*, 34, 2021.

[52] Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets. *arXiv preprint arXiv:2202.00273*, 2022.

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

[54] Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. *arXiv preprint arXiv:1902.03984*, 2019.

[55] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[56] Yuanhao Wang, Guodong Zhang, and Jimmy Ba. On solving minimax optimization locally: A follow-the-ridge approach. *arXiv preprint arXiv:1910.07512*, 2019.

[57] Timon Willi, Johannes Treutlein, Alistair Letcher, and Jakob Foerster. COLA: Consistent Learning with Opponent-Learning Awareness. *arXiv preprint arXiv:2203.04098*, 2022.

[58] 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*, 2021.

[59] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *Advances in Neural Information Processing Systems*, 33:7559–7570, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 3 and 4
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3 and 4
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix 1-4
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We will provide the code needed to reproduce our main experimental results.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix 5-8
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 6, all of our experiment results are averaged over multiple runs with different initializations.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 6
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite CIFAR-10 and STL-10 in section 6, and describe libraries that we use in appendix 5-7.
   (b) Did you mention the license of the assets? [Yes] In appendix 7.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] We did not publish any new asset.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We did not curate any new datasets. Existing dataset are discussed in appendix 7
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] In appendix 7.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]