

AMERICAN UNIVERSITY OF BEIRUT

MOMENTUM BLOCK GDA: A BLOCK COORDINATE
ALGORITHM FOR SOLVING BILINEAR MIN-MAX GAMES

by
ISRAA ADEL ASAAD

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Engineering Management
of the Maroun Semaan Faculty of Engineering and Architecture
of the Department of Industrial and Engineering Management
at the American University of Beirut




Beirut, Lebanon
November 2022

AMERICAN UNIVERSITY OF BEIRUT

MOMENTUM BLOCK GDA: A BLOCK COORDINATE
ALGORITHM FOR SOLVING BILINEAR MIN-MAX GAMES

by
ISRAA ADEL ASAAD

Approved by:

	Signature
Dr. Maher Nouiehed, Assistant Professor Department of Industrial and Engineering Management	Advisor
	Signature
Prof. Bacel Maddah, Professor, Chairperson Department of Industrial and Engineering Management	Member of Committee
	Signature
Dr. Nadine Moacdieh, Assistant Professor Department of Industrial and Engineering Management	Member of Committee

Date of thesis defense: November 4, 2022

AMERICAN UNIVERSITY OF BEIRUT
THESIS RELEASE FORM

Student Name: Asaad Israa Adel
Last First Middle

I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of my thesis; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes:

- As of the date of submission
- One year from the date of submission of my thesis.
- Two years from the date of submission of my thesis.
- Three years from the date of submission of my thesis.

Israa 30-Mar-2023
Signature Date

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincerest gratitude for my thesis professor, Dr. Maher Nouiehed, for his unwavering support throughout the entire process. His guidance, patience, and encouragement were instrumental in helping me overcome the challenges that inevitably arose. His expertise and guidance helped me refine my ideas, iterate when needed, and improve my quality of work. I am truly grateful for his expertise and mentorship that have undoubtedly contributed to the success of this thesis.

I would also like to extend my heartfelt appreciation to the committee members, Dr. Bacel Maddah and Dr. Nadine Moacdieh. Their valuable feedback, insightful suggestions, and critical evaluation helped shape the direction and content of this thesis.

In addition, I would like to acknowledge the support of my friends and family, who made it possible to complete my thesis amidst the unfavorable circumstances. They provided me with the motivation and emotional support needed to make this thesis.

ABSTRACT OF THE THESIS OF

Israa Adel Asaad

for

Master of Engineering Management

Major: Engineering Management

Title: Momentum Block GDA: A Block Coordinate Algorithm for Solving Bilinear Min-Max Games

With growing applications in Machine Learning, Game Theory, and the training of Generative Adversarial Networks GANs, solving min-max problems and establishing their convergence properties have experienced significant attention. Gradient Descent Ascent (GDA) and Optimistic Gradient Descent Ascent (OGDA) algorithms are popular algorithms used to solve saddle point problems. In an effort to address the issue of oscillating convergence behavior of these algorithms, we propose a dynamic method for solving bilinear problems. Our proposed method is characterized by its novel mechanism that dynamically chooses the coordinates to be updated at every iteration to guarantee a more stable and efficient convergence. Motivated by OGDA, we propose an algorithm, denoted Momentum Block-based Gradient Descent Ascent (MBGDA), that utilizes the momentum at every iterate to determine the block of coordinates for which a gradient step is applied. We present several empirical results that demonstrate the superior performance of our proposed algorithm compared to existing first- order methods for solving bilinear saddle point problems. More specifically, MBGDA achieves more stable convergence properties and achieves a higher probability of convergence in non- convex non-concave settings.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	1
ABSTRACT	2
ILLUSTRATIONS	4
TABLES	5
INTRODUCTION.....	6
A. Related Work	9
B. Solving a Bilinear Problem.....	14
C. Contribution	16
METHODOLOGY	17
PROOF OF CONVERGENCE	23
EXPERIMENTS	27
A. Experiment I Convergence Dynamics	27
B. Experiment II Convergence Rate.....	29
C. Experiment III Non-Convex Non-Concave Cases.....	32
CONCLUSION	36
REFERENCES	37

ILLUSTRATIONS

Figure

1. Divergence of GDA algorithm for simple 2-Dimensional Bilinear Case.....	11
2. The position of θ at every iteration using MBGDA.	20
3. The position of θ for the first 15 iterations using MBGDA, OGDA and GDA methods.....	21
4. The value of the first elements of x and y at every iteration for the first 50,000 iterations.....	22
5. Quadrants defined by the momentum terms m_x and m_y	24
6. The distance to the optimal solution of MBGDA, GDA, OGDA, EG and PP at every iteration	28
7. The number of iterations as function of matrix size using MBGDA algorithm.	30
8. The number of iterations to converge using each of MBGDA, OGDA, EG and PP methods as function of matrix size.	30
9. The number of iterations to converge using each of MBGDA, OGDA, EG and PP methods as a function of vector initializations.	31

TABLES

Table

1. Conversion Rates of OGDA and EG in Bilinear and Strongly Convex Strongly Concave cases where ϵ is the error and κ is the kappa statistic.....	15
2. Convergence properties of MBGDA, GDA and OGDA on the critical points of f	34
3. Convergence properties of MBGDA, GDA and OGDA on the critical points of f	34

CHAPTER I

INTRODUCTION

In recent years, several methods and algorithms have been developed to solve min-max optimization problems; also commonly known as saddle point problems. This urge was fueled by the significant advancements in theoretical literature and computational power. These problems arise in several popular machine learning models; see Generative Adversarial Networks (GANs) (Goodfellow, et al., 2014), reinforcement learning (B. Dai, 2018), image reconstruction and registration (Modersitzki & Haber, 2007), fair machine learning (Joseph, Kearns, Morgenstern, & Roth, 2016), adversarial neural networks (Ajakan, Germain, Larochelle, Laviolette, & Marchand, 2015).

In its general settings, the min-max problem can be formulated as

$$\min_{\mathbf{x} \in X} \max_{\mathbf{y} \in Y} f(\mathbf{x}, \mathbf{y}), \quad (1)$$

where $f(\mathbf{x}, \mathbf{y})$ is the objective function $f: X \times Y \rightarrow \mathbb{R}$ and X and Y are the feasibility sets of \mathbf{x} and \mathbf{y} . By denoting the maximization value function as $g(\mathbf{x}) = \max_{\mathbf{y} \in Y} f(\mathbf{x}, \mathbf{y})$, problem (1) can be reformulated as

$$\min_{\mathbf{x} \in X} g(\mathbf{x}),$$

where $g(\mathbf{x})$ is the optimal objective value of a maximization problem.

Despite being expressed as a traditional optimization problem, this problem is hard to solve due to non-differentiability of $g(\mathbf{x})$ and the difficulty in evaluating the function. From a game-theory perspective, this problem can also be viewed as a zero-sum game where \mathbf{x} is the decision variable of the first player, \mathbf{y} is the decision variable of

the second player and $f(\mathbf{x}, \mathbf{y})$ is the objective function of the two players. The first player aims to minimize the objective function by choosing \mathbf{x} while the second aims to maximize it by choosing \mathbf{y} . The goal is to find a Nash equilibrium defined below:

Definition 1: We say a point $(\mathbf{x}^*, \mathbf{y}^*)$ is a Nash Equilibrium (NE) point if it satisfies the following inequality

$$f(\mathbf{x}^*, \mathbf{y}) \leq f(\mathbf{x}^*, \mathbf{y}^*) \leq f(\mathbf{x}, \mathbf{y}^*) \quad \forall \mathbf{x} \in \mathbf{X} \text{ and } \mathbf{y} \in \mathbf{Y}.$$

In applications of GANs, this translates to finding a zero-sum game between the Generator (G) and the Discriminator (D) (Goodfellow, et al., 2014). The Generator aims at generating data points similar to points in the provided dataset. The Discriminator, however, aims at discriminating between the generated sample and a true sample. In an attempt to reach equilibrium, the two deep neural networks are trained usually by stochastic methods applied on both players (Daskalakis, Ilyas, Syrgkanis, & Zeng, 2017).

In general, min-max problems arising in machine learning applications have complex structure and can potentially be nonconvex-nonconcave problems; i.e. $f(\cdot, \mathbf{y})$ is non-convex in \mathbf{x} and $f(\mathbf{x}, \cdot)$ is non-concave in \mathbf{y} . In these cases, finding a Nash Equilibria is NP-Hard. A modest goal in such cases would be to find a first-order stationary solution. Several first order methods have been recently proposed to solve saddle point problems. Many of these algorithms fail to converge even to local min-max optimal solutions (Daskalakis & Panageas, 2018).

In the non-convex concave settings, several papers have proposed gradient based algorithms like the stochastic sub-gradient descent method (Rafique, Liu, Lin, & Yang, 2019) and the multi-step gradient descent ascent algorithm (Nouiehed, Sanjabi, Huang, Lee, & Razaviyayn, 2019). The multi-step gradient method performs multiple gradient

ascent steps to estimate the solution for the inner maximization problem followed by a single gradient descent (Nouiehed, Sanjabi, Huang, Lee, & Razaviyayn, 2019). Under mild conditions, these methods are guaranteed to converge to a first-order solution of $g(\cdot)$.

In convex-concave settings, where $f(\cdot, \mathbf{y})$ is convex in \mathbf{x} for every $\mathbf{y} \in \mathbf{Y}$ and $f(\mathbf{x}, \cdot)$ concave in \mathbf{y} for every $\mathbf{x} \in \mathbf{X}$, finding a Nash equilibrium can be achieved using variants of the gradient based algorithms (Bubeck, 2015). These proposed first-order methods are local search iterative algorithms. A more special case is the min-max bilinear problem formulated below.

$$\min_{\mathbf{x} \in \mathbf{X}} \max_{\mathbf{y} \in \mathbf{Y}} \mathbf{x}^T \mathbf{A} \mathbf{y} + \mathbf{b}^T \mathbf{x} + \mathbf{c}^T \mathbf{y}, \quad (2)$$

where $\mathbf{A} \in R^{n \times m}$, $\mathbf{x} \in R^n$ is the decision variable to be minimized, and $\mathbf{y} \in R^m$ is the decision variable to be maximized.

Applications of bilinear programming extend to constrained bimatrix games, Markovian assignment, and complementarity problems (Konno, 1975; Nahapetyan, 2007). In this thesis, we focus on finding the Nash equilibrium for this bilinear problem. The min-max theorem by Von Neumann states that

$$\min_{\mathbf{x} \in \mathbf{X}} \max_{\mathbf{y} \in \mathbf{Y}} \mathbf{x}^T \mathbf{A} \mathbf{y} = \max_{\mathbf{y} \in \mathbf{Y}} \min_{\mathbf{x} \in \mathbf{X}} \mathbf{x}^T \mathbf{A} \mathbf{y}.$$

Thus, the solution of the right-hand side and the left-hand side are equivalent. This holds for any convex compact sets.

Most algorithms proposed for solving these problems are variants of GD. However, these variants might diverge for simple bilinear problems, cycle without the last iterate converging to the optimal solution, and converge to stationary points rather

than the optimal solution (Adolphs, Daneshmand, Lucchi, & Hofmann, 2019). To solve the min-max optimization problems and achieve this equilibrium, several algorithms have been developed. While being studied extensively, existing algorithms fail in scaling with the data size and are highly dependent on the vector initialization and underperform in non-convex non-concave settings.

Extensions of the gradient descent algorithms to min-max optimization problems, specifically the bilinear case, include: The Gradient Descent Ascent (GDA), Optimistic Gradient Descent Ascent (OGDA), Extra Gradient (EG), and the Proximal Point (PP) methods. These methods utilize the gradients at every iteration to solve the saddle point problem (Mokhtari, Ozdaglar, & Pattathil, 2020). The algorithms are further discussed in later sections.

We have conducted several experiments on these first order algorithms to monitor their performance. We have deduced that they cannot scale with the dimension of the problem. They are highly dependent on the problem initialization and the inter-correlation in the data. They also fail to converge in non-concave non-convex settings. In this favor, we proposed an algorithm Momentum based Block Gradient Descent Ascent (MBGDA) method that successfully converges to the optimal solution of bilinear problems. Compared to existing methods, MBGDA is less sensitive to the problem initializations and data size. It also guarantees a higher frequency of convergence in non-convex non-concave settings.

A. Related Work

Before discussing the methodology of the proposed algorithm, we discuss the existing first order algorithms used for solving the unconstrained min-max bilinear

problems. Assume that $\mathbf{x} \in R^n$, $\mathbf{y} \in R^m$, and $\mathbf{A} \in R^{n \times m}$. An iterative algorithm solves the saddle point problem by generating a sequence of iterates $\{(\mathbf{x}_t, \mathbf{y}_t)\}_t$ that guarantees convergence to an NE point. The most popular approach for solving bilinear problems is the vanilla **Gradient Descent Ascent (GDA)** method which is a natural extension of the Gradient Descent (GD) method. GDA performs a gradient descent step on \mathbf{x} followed by a gradient ascent step on \mathbf{y} . The next iterate thus depends on the current gradient only. Let the constant positive step size be $\alpha \in (0,1)$. The update procedure of GDA is the following:

$$\begin{cases} x_{t+1} = x_t - \alpha \nabla_x f(x_t, y_t) \\ y_{t+1} = y_t + \alpha \nabla_y f(x_t, y_t) \end{cases} \quad (3)$$

It is used in several applications and converges linearly to the optimal solution in strongly-convex strongly-concave problems (Yan, Xu, Lin, Liu, & Yang, 2020). While proven successful in several min-max settings, GDA fails to converge in many cases including the unconstrained bilinear framework; see example 1. Despite its successful employment in min-max problems, no-regret algorithm fails to show good performance in a wide variety of cases. The GDA method and other algorithms do not show convergence in the bilinear case due to cycling around the optimal solution.

Example 1: For instance, consider the simple two-dimensional bilinear example $\min_{x \in R^m} \max_{y \in R^n} f(x, y)$ where $f(x, y) = xy$, and apply the GDA method to optimize the problem. Let $\theta_t = (x_t, y_t)$. The update rule (3) then becomes the following.

$$\begin{cases} x_{t+1} = x_t - \alpha y_t \\ y_{t+1} = y_t + \alpha x_t \end{cases}$$

By noticing that

$$\|\theta_{t+1}\|^2 = (x_{t+1})^2 + (y_{t+1})^2 = (1 + \alpha^2)(x_t)^2 + (1 + \alpha^2)(y_t)^2 = (1 + \alpha^2)\|\theta_t\|^2 > \|\theta_t\|^2,$$

one can easily check that GDA diverges even for the simple bi-linear problem. Specifically, GDA fails to converge as the distance to the optimal solution increases at every iteration as illustrated in Figure 1.

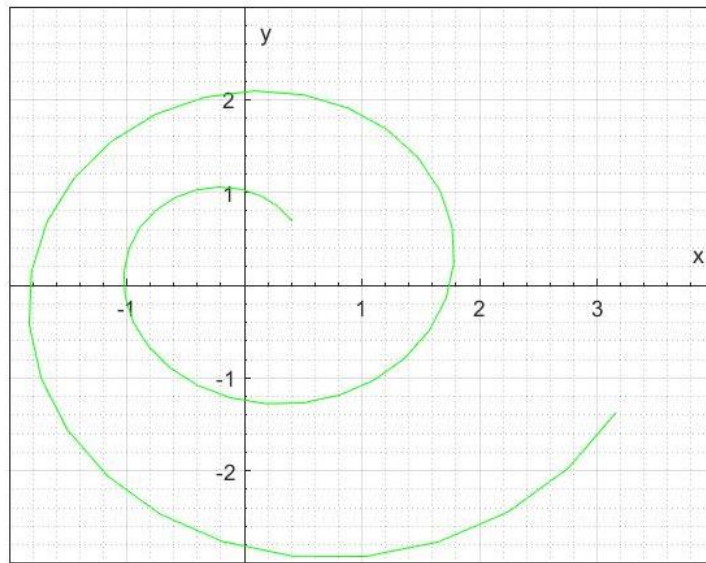


Figure 1 Divergence of GDA algorithm for simple 2-Dimensional Bilinear Case

Despite the divergence behavior the Gradient Descent Ascent algorithm achieves average convergence; i.e., that for the trajectory of $(\mathbf{x}_t, \mathbf{y}_t)$, the limit of $\frac{1}{t} \sum_{\tau \leq t} \mathbf{x}_\tau^T \mathbf{A} \mathbf{y}_\tau$ converges to the equilibrium as $t \rightarrow \infty$. This may occur without guaranteed convergence on the last iterate $(\mathbf{x}_t, \mathbf{y}_t)$ that might possibly diverge or cycle. We might also get closer to the optimal solution without necessarily reaching it (Lei, Nagarajan, Panageas, & Wang, 2018).

This motivated the development of a variant of the GDA method that can achieve last iterate convergence. To alleviate these issues, Daskalakis proposed the **Optimistic Gradient Descent Ascent (OGDA)** method to solve saddle point problems (Daskalakis,

Ilyas, Syrgkanis, & Zeng, 2017). The main concept behind OGDA lies in the addition of a negative momentum term to update at every iteration. It improves convergence to a local solution by acting as a “friction that can damp oscillations” and shifting the original eigenvalues towards the solution (Gidel, et al., 2020) The addition of this negative momentum also allows achieving an acceleration locally to enhance the convergence rate of GDA (Zhang & Wang, 2021). The generalized OGDA method considers that the step size of the descent and ascent steps of \mathbf{x} and \mathbf{y} respectively are not necessarily equal to the coefficient of the negative momentum. OGDA and generalized OGDA methods are proved to converge in bilinear objective functions and in strongly-concave strongly-convex problems (Daskalakis, Ilyas, Syrgkanis, & Zeng, 2017). The OGDA method uses the following updating equations.

$$\begin{cases} x_{t+1} = x_t - \alpha \nabla_x f(x_t, y_t) - \alpha (\nabla_x f(x_t, y_t) - \nabla_x f(x_{t-1}, y_{t-1})) \\ y_{t+1} = y_t + \alpha \nabla_y f(x_t, y_t) + \alpha (\nabla_y f(x_t, y_t) - \nabla_y f(x_{t-1}, y_{t-1})) \end{cases} \quad (4)$$

There exists a memory in its dynamics as the next iterate depend on the gradients of the current and previous points. However, this momentum allows moving in the negative gradient direction. Hence, if the momentum is negative, the next iterate updates the coordinates in an opposite direction. With that, a descent step might become an ascent and vice versa. This allows the coordinates to move further from the optimal solution before updating in the right direction and moving closer to it. This is further observed in Figure 3.

Another method for solving saddle point problems is the **Extra Gradient EG** method. It is a classical method introduced by (Korpelevich, 1976). As its name suggests, EG computes the gradient between the current iteration t and next iterations $t + 1$. More specifically, at every iteration, EG uses the current gradient to calculate an intermediate

point which is used in updating the iterates. At every iteration, EG uses the current gradient to calculate the intermediate point as follows.

$$\begin{cases} x_{t+1/2} = x_t - \alpha \nabla_x f(x_t, y_t) \\ y_{t+1/2} = y_t + \alpha \nabla_y f(x_t, y_t) \end{cases}$$

The next iterate then uses $x_{t+1/2}$ and $y_{t+1/2}$ as follows.

$$\begin{cases} x_{t+1} = x_t - \alpha \nabla_x f(x_{t+1/2}, y_{t+1/2}) \\ y_{t+1} = y_t + \alpha \nabla_y f(x_{t+1/2}, y_{t+1/2}) \end{cases} \quad (5)$$

The convergence of EG was established for the bilinear and strongly-convex strongly-concave saddle point problems. (Mokhtari, Ozdaglar, & Pattathil, 2020).

The EG and OGDA methods show similar behavior in solving min-max optimization problems. They both require using an added term to estimate the next gradient. Recently, they were proven as approximates of the **Proximal Point (PP)** method (Mokhtari, Ozdaglar, & Pattathil, 2020). The proximal point method was introduced by (Martinet, 1970) and was further studied for solving saddle point problems by (Rockafellar, 1976). The PP method is an implicit algorithm that requires computing the next gradient at every iteration. It also converges to the unique solution of the saddle point problem in the strongly convex strongly concave and bilinear settings (Mokhtari, Ozdaglar, & Pattathil, 2020). The PP method computes the iterates as the follows.

$$\begin{cases} x_{t+1} = \text{prox}_{\frac{1}{\alpha}f}(x_t) = \underset{x \in R^m}{\operatorname{argmin}} \left\{ f(x, y_{t+1}) + \frac{1}{2\alpha} \|x - x_t\|^2 \right\} \\ y_{t+1} = \text{prox}_{\frac{1}{\alpha}f}(y_t) = \underset{y \in R^n}{\operatorname{argmin}} \left\{ f(x_{t+1}, y) + \frac{1}{2\alpha} \|y - y_t\|^2 \right\} \end{cases}$$

Using the optimality conditions, its update method can be written as the following.

$$\begin{cases} x_{t+1} = x_t - \alpha \nabla_x f(x_{t+1}, y_{t+1}) \\ y_{t+1} = y_t + \alpha \nabla_y f(x_{t+1}, y_{t+1}) \end{cases} \quad (6)$$

The EG and OGDA can be seen as approximates of the Proximal Point method where the gradient at the next iterate being approximated (Mokhtari, Ozdaglar, & Pattathil, 2020).

B. Solving a Bilinear Problem

Our goal is to find a Nash equilibrium point $(\mathbf{x}^*, \mathbf{y}^*)$ as defined in Definition 1, for solving bilinear min-max optimization problem. We start by applying the **GDA** algorithm where we perform a gradient descent followed by a gradient ascent as shown in Equation (3). The algorithm does not converge; however, it converges in the strongly convex – strongly concave cases. As a counter example, we consider the following function.

$$\min_{x \in \mathbb{R}^m} \max_{y \in \mathbb{R}^n} xy$$

Using the **Proximal Point** method as per the algorithm shown in (6), the following update rule is performed.

$$\begin{cases} x_{t+1} = x_t + \alpha A y_{t+1} & (7) \\ y_{t+1} = y_t + \alpha A^T x_{t+1} & (8) \end{cases}$$

Replacing the expression in (7), we obtain

$$x_{t+1} = x_t + \alpha A (y_t + \alpha A^T x_{t+1}) = x_t + \alpha A y_t + \alpha^2 A A^T x_{t+1} \quad (9)$$

Solving for x_{t+1} , we get

$$x_{t+1} = (I - \alpha^2 A A^T)^{-1} (x_t - \alpha A y_t)$$

Starting from the PP method shown in Equation (6)

$$\begin{aligned}
x_{t+1} &= (I - \alpha^2 AA^T)^{-1}(x_t - \alpha Ay_t) = [I - \alpha^2 A^T A + O(\alpha^2)][x_t - \alpha Ay_t] \\
&= x_t - \alpha Ay_t - \alpha A[\alpha A^T x_t - \alpha^2 A^T Ay_t] + O(\alpha^2) \\
&= x_t - 2\alpha Ay_t - \alpha A[\alpha A^T x_t - (1 + \alpha^2 A^T A)y_t] + O(\alpha^2) \\
&= x_t - 2\alpha Ay_t - \alpha A[\alpha A^T x_t - (y_{t-1} + \alpha A^T x_{t-1})] + O(\alpha^2) \\
&= x_t - 2\alpha Ay_t + \alpha Ay_{t-1}
\end{aligned}$$

which is exactly the OGDA method. Similarly, EG is another linear approximation of PP method (Mokhtari, Ozdaglar, & Pattathil, 2020). Their convergence rate to ϵ optimal solution is summarized in Table 1 (Liang & Stokes, 2019; Gidel, Berard, Vincent, & Lacoste-Julien, 2018; Tseng, 1995).

Using linear approximations of the update term, the update term of **Optimistic Gradient Descent Ascent** algorithm shown in Equation (4) can be written as

$$\nabla_x f(x_{t+1}, y_{t+1}) \approx \nabla_x f(x_t, y_t) + \nabla_x f(x_t, y_t) - \nabla_x f(x_{t-1}, y_{t-1})$$

Table 1 Conversion Rates of OGDA and EG in Bilinear and Strongly Convex Strongly Concave cases where ϵ is the error and κ is the kappa statistic

	Rate (EG)	Rate (OGDA)
Bilinear	$O(\kappa^2 \log(1/\epsilon))$	$O(\kappa \log(1/\epsilon))$
Strongly Convex Strongly Concave	$O(\kappa \log(1/\epsilon))$	$O(\kappa \log(1/\epsilon))$

where $\kappa = \frac{\lambda_{\max}(A^T A)}{\lambda_{\min}(A^T A)}$ where λ_{\max} is the largest eigen value of the matrix and λ_{\min} is the lowest eigenvalue of the matrix (Mokhtari, Ozdaglar, & Pattathil, 2020)

Despite the convergence results, the experiments we have conducted showed that the results oscillate before reaching the optimal solution. In addition, they fail to scale with the size of the matrix A and are highly dependent on external factors like the

initial point and the kappa statistic. We propose an algorithm that dynamically updates the coordinates in the desired direction. We further empirically demonstrate faster converge the ability to scale with the size of matrix A .

C. Contribution

Motivated by the GDA and OGDA algorithms, we propose a method that solves the bilinear problem by dynamically updating the coordinates of \mathbf{x} and \mathbf{y} in favorable directions. Motivated by the multistep GDA algorithm, our method dynamically chooses the coordinates to be updated at every iteration. The updating method is motivated by the OGDA method as it tends to use the momentum term as an update metric. The update mechanism analyzes the momentum and computes a measure that determines the direction of the step at the next iterate. The strength of our algorithm lies in this dynamic selection as to when and how each update occurs. Our proposed algorithm is a gradient-based algorithm that dynamically updates the chosen coordinates of \mathbf{x} and \mathbf{y} at every iteration. We have shown its convergence to the Nash equilibrium solution in the bilinear case. It also achieves a higher probability of convergence in non-convex non-concave settings.

CHAPTER II METHODOLOGY

We study the following saddle point problem shown in equation (1)

$$\min_{x \in R^m} \max_{y \in R^n} f(x, y)$$

where the function f is a bilinear function and propose a first order method, referred to as Momentum Block Gradient Descent Ascent (MBGDA) that is able to compute the a Nash equilibrium solution defined by $(\mathbf{x}^*, \mathbf{y}^*) \in R^m \times R^n$ that $f(\mathbf{x}^*, \mathbf{y}) \leq f(\mathbf{x}^*, \mathbf{y}^*) \leq f(\mathbf{x}, \mathbf{y}^*)$ for all $\mathbf{x} \in R^m, \mathbf{y} \in R^n$.

In this section, we present the proposed algorithm. We first present the algorithm of MBGDA method then compare it with existing first-order methods like Gradient Descent Ascent GDA, Optimistic Gradient Descent Ascent OGDA, Extra Gradient EG and Proximal Point PP.

The update term of x following the OGDA method as per Equation (4) include the following momentum term

$$\begin{aligned} & -\alpha(\nabla_x f(\mathbf{x}_t, \mathbf{y}_t) - \nabla_x f(\mathbf{x}_{t-1}, \mathbf{y}_{t-1})) \\ & = -\alpha \left[\frac{[\nabla_x f(\mathbf{x}_t, \mathbf{y}_t)][\nabla_x f(\mathbf{x}_t, \mathbf{y}_t) - \nabla_x f(\mathbf{x}_{t-1}, \mathbf{y}_{t-1})]}{\nabla_x f(\mathbf{x}_t, \mathbf{y}_t)} \right] \end{aligned}$$

The direction of update thus depends on the gradients at the current and previous points. The main concept behind MBGDA is to dynamically choose what to update at a given iteration. Our decision matrix is motivated by OGDA method that utilizes a momentum term. We define the following update metrics

$$\begin{cases} m_x = \nabla_x f(x_t, y_t) \times (\nabla_x f(x_t, y_t) - \nabla_x f(x_{t-1}, y_{t-1})) \\ m_y = \nabla_y f(x_t, y_t) \times (\nabla_y f(x_t, y_t) - \nabla_y f(x_{t-1}, y_{t-1})) \end{cases} \quad (10)$$

that reflect the momentum at each coordinate point and the signs of each of m_x and m_y are used to determine the block we update in \mathbf{x} and \mathbf{y} respectively. These momentum terms determine favorable directions which guarantee converging to the optimal solution without any oscillations. More specifically, the signs of m_x and m_y select which decision variable to update and the direction of update. In particular, the updates follow the following rule

$$\begin{cases} \mathbf{x}_{t+1} = \begin{cases} \mathbf{x}_t - \alpha \nabla_x f(x_t, y_t) & \text{if } m_x \geq 0 \\ \mathbf{x}_t & \text{if } m_x < 0 \end{cases} \\ \mathbf{y}_{t+1} = \begin{cases} \mathbf{y}_t + \alpha \nabla_y f(x_t, y_t) & \text{if } m_y \geq 0 \\ \mathbf{y}_t & \text{if } m_y < 0 \end{cases} \end{cases} \quad (11)$$

In high dimensional problems, we perform the gradient steps updates along different coordinates of each vector based on their corresponding values of m_x and m_y .

We will apply MBGDA algorithm on the bilinear function $f(x, y) = \mathbf{x}^T \mathbf{A} \mathbf{y}$. Let $\mathbf{A} \in R^{n \times m}$ and assume that $\mathbf{x} \in R^n$, $\mathbf{y} \in R^m$, and the step size $\alpha > 0$. \mathbf{x}_0 and \mathbf{y}_0 are the initial vectors, \mathbf{x}_t and \mathbf{y}_t are the values of \mathbf{x} and \mathbf{y} at iteration t , and \otimes is the component wise product. Then, the momentum for x and y are as follows.

$$\begin{aligned} \mathbf{m}_x &= \nabla_x f(x_t, y_t) \otimes (\nabla_x f(x_t, y_t) - \nabla_x f(x_{t-1}, y_{t-1})) = \langle \mathbf{A} \mathbf{y}_t, (\mathbf{A} \mathbf{y}_t - \mathbf{A} \mathbf{y}_{t-1}) \rangle \\ &= \langle \mathbf{A} \mathbf{y}_t, \mathbf{A}(\mathbf{y}_t - \mathbf{y}_{t-1}) \rangle = \langle \alpha \mathbf{A} \mathbf{y}_t, \mathbf{A} \mathbf{A}^T \mathbf{x}_{t-1} \rangle \end{aligned}$$

$\mathbf{m}_x = \alpha \mathbf{A} \mathbf{y}_t \cdot \mathbf{A} \mathbf{A}^T \mathbf{x}_{t-1}$ and similarly, $\mathbf{m}_y = -\alpha \mathbf{A}^T \mathbf{x}_t \cdot \mathbf{A}^T \mathbf{A} \mathbf{y}_{t-1}$ with step size $\alpha > 0$, updating \mathbf{x} and \mathbf{y} can be determined by the signs of the following metrics

$$\begin{cases} m_x = \mathbf{A} \mathbf{y}_t \cdot \mathbf{A} \mathbf{A}^T \mathbf{x}_{t-1} \\ m_y = -\mathbf{A}^T \mathbf{x}_t \cdot \mathbf{A}^T \mathbf{A} \mathbf{y}_{t-1} \end{cases}$$

At every iteration t , the gradient step in (11) becomes the following

$$\begin{cases} (\mathbf{x}_{t+1})_i = \begin{cases} (\mathbf{x}_t)_i - \alpha (\mathbf{A}\mathbf{y}_t)_i & \text{if } (m_x)_i \geq 0 \\ (\mathbf{x}_t)_i & \text{if } (m_x)_i < 0 \end{cases} \\ (\mathbf{y}_{t+1})_i = \begin{cases} (\mathbf{y}_t)_i + \alpha (\mathbf{A}\mathbf{x}_t)_i & \text{if } (m_y)_i \geq 0 \\ (\mathbf{y}_t)_i & \text{if } (m_y)_i < 0 \end{cases} \end{cases} \quad (12)$$

To visualize the process, we solve the two-dimensional problem $f(x, y) = xy$. The optimal solution of this problem lies at the origin. We then calculate the momentum metrics. $m_x = y_t x_{t-1}$ and $m_y = -x_t y_{t-1}$. Updating x and y depends on the signs of these metrics. The update rule in (12) becomes the following

$$\begin{cases} x_{t+1} = \begin{cases} x_t - \alpha y_t & \text{if } y_t x_{t-1} \geq 0 \\ x_t & \text{if } y_t x_{t-1} < 0 \end{cases} \\ y_{t+1} = \begin{cases} y_t + \alpha x_t & \text{if } -x_t y_{t-1} \geq 0 \\ y_t & \text{if } -x_t y_{t-1} < 0 \end{cases} \end{cases} \quad .$$

We notice that at every iteration, we will be updating either one of the coordinates depending on the position of the point θ in the four quadrants. Assume the step size $\alpha=0.003$, maximum number of iterations is 120. We start with an initial point $\theta_0 = (0.01, 0.01)$. Figure 2 shows the position of the point θ at every iteration.

As shown in Figure 2, starting from the first quadrant, θ was only updated in the direction that guaranteed getting closer to the optimal solution by updating either one of its coordinates satisfying $\|x_{k+1} + y_{k+1}\| \geq \|x_k + y_k\| \forall k \in [0, 120]$. After 120 iterations, we reached the optimal solution with an accuracy of 10^{-6} .

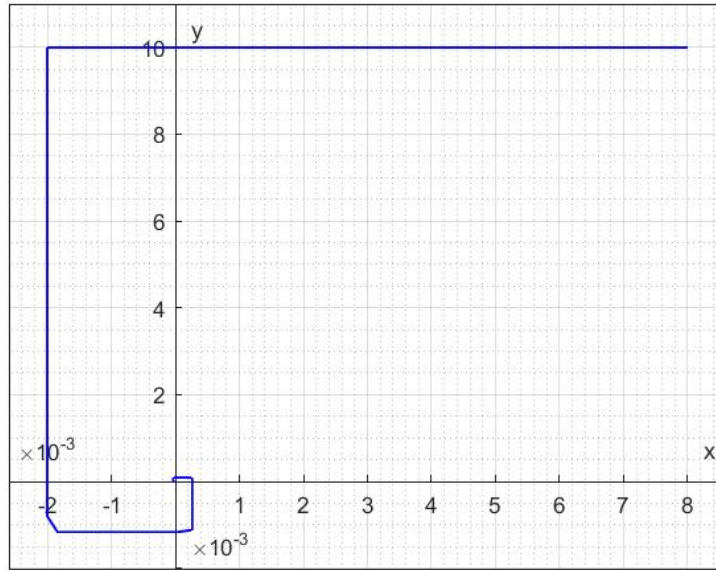


Figure 2 The position of θ at every iteration using MBGDA.

We can prove the convergence to the optimal solution by examining the position of point θ at every iterate and in every quadrant. In the first quadrant where $x_0 > 0$ and $y_0 > 0$, for some t , the coordinates of θ are the following.

$$\begin{cases} x_t = x_{t-1} - \alpha y_0 = x_{t-2} - \alpha y_0 - \alpha y_0 = \dots = x_0 - t\alpha y_0 < 0 < x_0 \\ y_t = y_0 \end{cases}$$

for sufficiently large t .

Thus, its position is closer to the optimal solution at the origin. Similarly, either one of the coordinates of θ are updated at every iteration depending on the quadrant which is evident in Figure 2.

We repeat the same example using other first order methods, GDA and OGDA methods, and monitor the position of θ . With GDA, we update the coordinates of θ at every step using the gradient. Similarly, OGDA updates both coordinates, however using a negative momentum term. Both methods managed to reach an optimal solution with an accuracy

of order 10^{-2} after 120 iterations. Figure 3 shows the path followed by θ in the first 15 iterations. GDA method has failed to converge to the optimal solution as shown in Figure 1. Using OGDA, the distance to the optimal solution increased widely before convergence which shows that unnecessary updates were performed. MBGDA managed to avoid updating in unfavorable directions by using the sign of the momentum to determine whether to update each of the coordinates of θ separately.

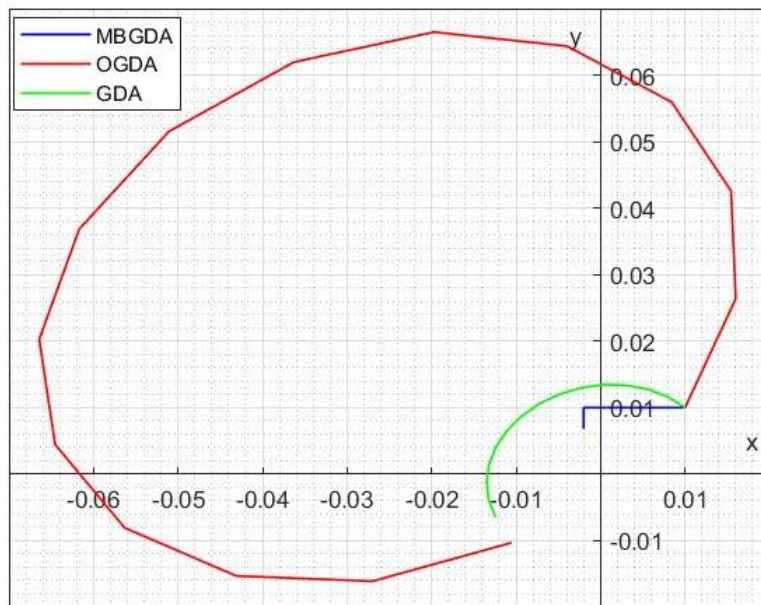


Figure 3 The position of θ for the first 15 iterations using MBGDA, OGDA and GDA methods

In higher dimensional problems, MBGDA allows updating certain elements of the vectors rather than the whole vector. We determine these elements using the same generalized momentum term referred to as \mathbf{m}_x and \mathbf{m}_y earlier. We conduct the experiment to solve the bilinear problem $\mathbf{x}^T \mathbf{A} \mathbf{y}$ where \mathbf{A} is a random matrix of size 5×5 , \mathbf{x} and \mathbf{y} are random vectors of size 5, \mathbf{x}_0 and \mathbf{y}_0 are random initializations, and the step size is 0.001.

For the first 50,000 iterations, we plot the values of the first elements of \mathbf{x} and \mathbf{y} respectively as shown in Figure 4. Using the OGDA method, oscillations around the

optimal solution occurred. Using MBGDA, this was not recorded, and the values of x and y consistently decreased with every iteration. The distance to the optimal solution was reduced.

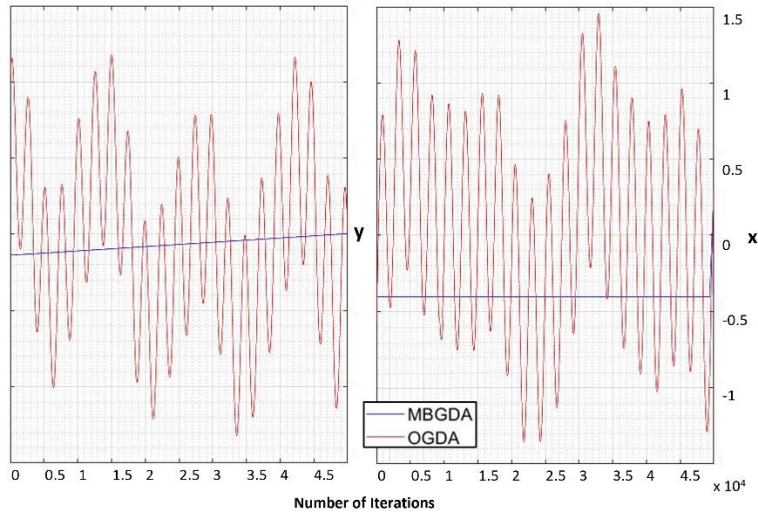


Figure 4 The value of the first elements of x and y at every iteration for the first 50,000 iterations

Note that MBGDA did not necessarily update both coordinates and every iteration. Figure 4 shows the update of the first coordinate of each of the vectors x and y at every iteration. It shows that the first coordinate of y was updated at every iteration in the direction that guaranteed converging. In the first iterations, the first coordinate of x remained constant after which they were updated in the favored direction reaching zero.

CHAPTER III

PROOF OF CONVERGENCE

We proved the convergence of MBGDA to Nash Equilibrium in the case of simple bilinear problems. Let

$$f(x, y) = axy + bx + cy$$

Then,

$$\nabla_x f(x, y) = ay + b \text{ and } \nabla_y f(x, y) = ax + c$$

The update metrics according to Equation (7) are calculated at each of x and y as follows.

$$m_x = \alpha(ay_t + b)(ax_{t-1} + c)$$

For $m_x > 0$, we either need to have

$$(ax_{t-1} + c) > 0 \text{ and } (ay_t + b) > 0 \text{ so } x_{t-1} > -c/a \text{ and } y_t > -b/a$$

or

$$(ax_{t-1} + c) < 0 \text{ and } (ay_t + b) < 0 \text{ so } x_{t-1} < -c/a \text{ and } y_t < -b/a$$

Similarly, $m_y = -(ax_t + c)(ay_{t-1} + b)$

For $m_y > 0$, we either need to have

$$(ax_t + c) > 0 \text{ and } (ay_{t-1} + b) > 0 \text{ so } x_t < -c/a \text{ and } y_{t-1} > -b/a$$

or

$$(ax_t + c) < 0 \text{ and } (ay_{t-1} + b) < 0 \text{ so } x_t > -c/a \text{ and } y_{t-1} < -b/a$$

The momentum terms then act as metrics to determine the terms to be updated.

$$\begin{cases} x_{t+1} = \begin{cases} x_t - \alpha(ay_t + b) & \text{if } (ay_t + b)(ax_{t-1} + c) \geq 0 \\ x_t & \text{if } (ay_t + b)(ax_{t-1} + c) < 0 \end{cases} \\ y_{t+1} = \begin{cases} y_t + \alpha(ax_t + c) & \text{if } -(ax_t + b)(ay_{t-1} + c) \geq 0 \\ y_t & \text{if } -(ax_t + b)(ay_{t-1} + c) < 0 \end{cases} \end{cases}, \quad (13)$$

where $\alpha > 0$ is the step size. We notice that these metrics define 4 quadrants where the coordinates of x and/or y are updated. Without loss of generality, assume we start in Q1.

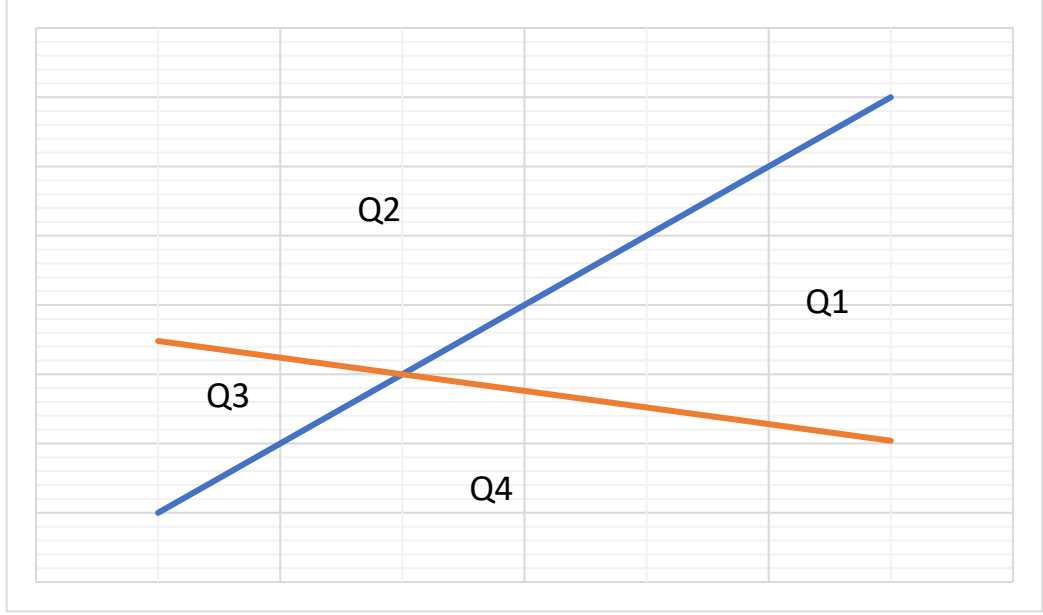


Figure 5 Quadrants defined by the momentum terms mx and my

- Quadrant 1 (Q1): $x > -c/a$ and $y > -b/a$

In this quadrant, y is not updated while x is updated as follows.

$$\begin{cases} x_{t_1} = x_{t_1-1} - \alpha(ay_{t_1-1} + b) = \dots = x_0 - t_1\alpha(ay_0 + b) = x_0 - t_1\alpha ay_0 - t_1\alpha b \\ y_{t_1} = y_0 \end{cases}$$

Since $(ay_{t_1-1} + b) > 0$, for a large enough t_1 , we get $x_{t_1} < -c/a$.

For a small enough α , $|x_{t_1}| < |x_0|$.

We continue in Q2

- Quadrant 2 (Q2): $x_{t_1} < -c/a$ and $y_{t_1} > -b/a$

In this quadrant, x is not updated while y is updated as follows.

$$\begin{cases} x_{t_2} = x_{t_1} \\ y_{t_2} = y_{t_2-1} + \alpha(ax_{t_1} + c) = \dots = y_0 + t_2\alpha(ax_0 + c) \end{cases}$$

Since $(ax_0 + c) < 0$, for a large enough t_2 , we get $y_{t_2} < -b/a$.

For a small enough α , $|y_{t_2}| < |y_{t_1}|$.

- Quadrant 3 (Q3): $x_{t_2} < -c/a$ and $y_{t_2} < -b/a$

$$\begin{cases} x_{t3} = x_{t2-1} - \alpha(ay_{t2} + b) = \dots = x_0 - t_3\alpha ay_0 + t_3ab > -c/a \\ y_{t3} = y_{t2} \end{cases}$$

- Quadrant 4 (Q4): $x_{t3} > -c/a$ and $y_{t3} < -b/a$

$$\begin{cases} x_{t4} = x_{t3} \\ y_{t4} = y_{t3} + t_4\alpha(ax_0 + c) \end{cases}$$

Assume we adopt is a diminishing step size. Thus $\alpha_t < \alpha_{t-1}$. At any iteration t_i , we aim to ensure that $|x_{t2}| \leq |x_{t1}|$ and $|y_{t3}| \leq |y_{t1}|$. Thus, step size should satisfy the following conditions. Otherwise update $\alpha = \alpha_t < \frac{\alpha_{t-1}}{2}$.

- Condition 1: $0 < \alpha < \left| \frac{2x_{t_{i+1}}}{(ay_{t_{i+1}} + b)} \right|$

- Condition 2: $0 < \alpha < \left| \frac{2y_{t_{i+1}}}{(ax_{t_{i+1}} + c)} \right|$

Assume $\alpha > 0$ is a fixed step size.

Assume at iteration t_i , the step size does not satisfy one of the conditions. This case will be resolved in the upcoming iteration t_{i+1} . Following Equation (13) updated.

$$\left\{ \begin{array}{l} \left(\begin{array}{l} \text{if } \alpha_x > \left| \frac{2x_{t_{i+1}}}{(ay_{t_{i+1}} + b)} \right|, \quad \alpha_x = \frac{\alpha_x}{2} \quad \text{and } x_{t+1} = x_t \\ \text{if } \alpha_x < \left| \frac{2x_{t_{i+1}}}{(ay_{t_{i+1}} + b)} \right|, \\ x_{t+1} = \begin{cases} x_t - \alpha_x (ay_t + b) & \text{if } (ay_t + b)(ax_{t-1} + c) \geq 0 \\ x_t & \text{if } (ay_t + b)(ax_{t-1} + c) < 0 \end{cases} \end{array} \right. \\ \left(\begin{array}{l} \text{if } \alpha_y > \left| \frac{2y_{t_{i+1}}}{(ax_{t_{i+1}} + c)} \right|, \quad \alpha_y = \frac{\alpha_y}{2} \quad \text{and } y_{t+1} = y_t \\ \text{if } \alpha_y < \left| \frac{2y_{t_{i+1}}}{(ax_{t_{i+1}} + c)} \right|, \\ y_{t+1} = \begin{cases} y_t + \alpha_y (ax_t + c) & \text{if } -(ax_t + b)(ay_{t-1} + c) \geq 0 \\ y_t & \text{if } -(ax_t + b)(ay_{t-1} + c) < 0 \end{cases} \end{array} \right. \end{array} \right. \quad (13)$$

Assume the following example. In Q1, if $\alpha_{ti} > \left| \frac{2x_{t_i}}{ay_{t_i}+b} \right|$

$$x_{t_{i+1}} = x_{t_i} - \alpha(ay_{t_i} + b), |ay_{t_i} + b| \gg |x_{t_i}|$$

$$\text{So, } \left| \frac{2x_{t_{i+1}}}{ay_{t_{i+1}}+b} \right| > \left| \frac{2x_{t_i}}{ay_{t_i}+b} \right| \text{ and } \alpha < \left| \frac{2x_{t_{i+1}}}{ay_{t_{i+1}}+b} \right|$$

The condition is then satisfied for the upcoming iteration.

Remark: In addition, in the two-dimensional case where $a = 1$, $b = 0$ and $c = 0$; thus, $f(x, y) = xy$. The update metrics determined by the momentum are $m_x = y_t x_{t-1}$ and $m_y = -x_t y_{t-1}$. Updating x and y depends on the signs of these metrics. Thus, MBGDA splits the 2-dimensional plane in to the four quadrants defined by $x = 0$ and $y = 0$. Then, at every iteration, either one of the coordinates x or y is updated until it converges to the optimal solution at the origin.

CHAPTER IV EXPERIMENTS

We conducted several experiments to study the characteristics of the new proposed algorithm MBGDA. The purpose of these experiments range between understanding the convergence dynamics in bilinear problems and understanding the factors that affect the convergence rate from the choice of starting points to the size of data and their correlation. We also studied the behavior of MBGDA in non-convex non-concave problems. We compared the convergence dynamics to existing algorithms used for solving general saddle point problems: Gradient Descent Ascent GDA, Optimistic Gradient Descent Ascent OGDA, Extra Gradient EG and Proximal Point PP methods.

In the first experiment, we show that MBGDA updates the iterates in the desired directions. Our results show that MBGDA reduces irrelevant oscillations and achieves an overall faster convergence rate. In the second experiment, a set of tests were done to relate the convergence rate of MBGDA to the scalability of \mathbf{A} , the initialization of the problem, and the intercorrelation of the matrix. Our results show that MBGDA, under similar conditions, achieves better performance compared to existing algorithms. In the third experiment, we solved two non-convex non-concave problems and monitored the fraction of times we get convergence of MBGDA compared to GDA and OGDA algorithms. MBGDA achieved higher frequency of convergence to stable solutions than other algorithms.

A. Experiment I Convergence Dynamics

The purpose of the experiment is to compare the convergence of our proposed algorithm with Optimistic Gradient Descent Ascent OGDA, Extra-gradient EG method,

and the Proximal Point PP method (Mokhtari, Ozdaglar, & Pattathil, 2020). This experiment tends to solve the min-max bilinear problem:

$$\min_{\mathbf{x} \in \mathbb{R}^d} \max_{\mathbf{y} \in \mathbb{R}^d} \mathbf{x}^T \mathbf{A} \mathbf{y},$$

where $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a full rank matrix, $d = 10$ is the dimension of the problem, \mathbf{x}_0 and \mathbf{y}_0 are random initializations, and the step size being 0.005 in the algorithms. The distance to the optimal solution at every iteration is recorded as shown in figure 6. The accuracy is fixed at 10^{-2} .

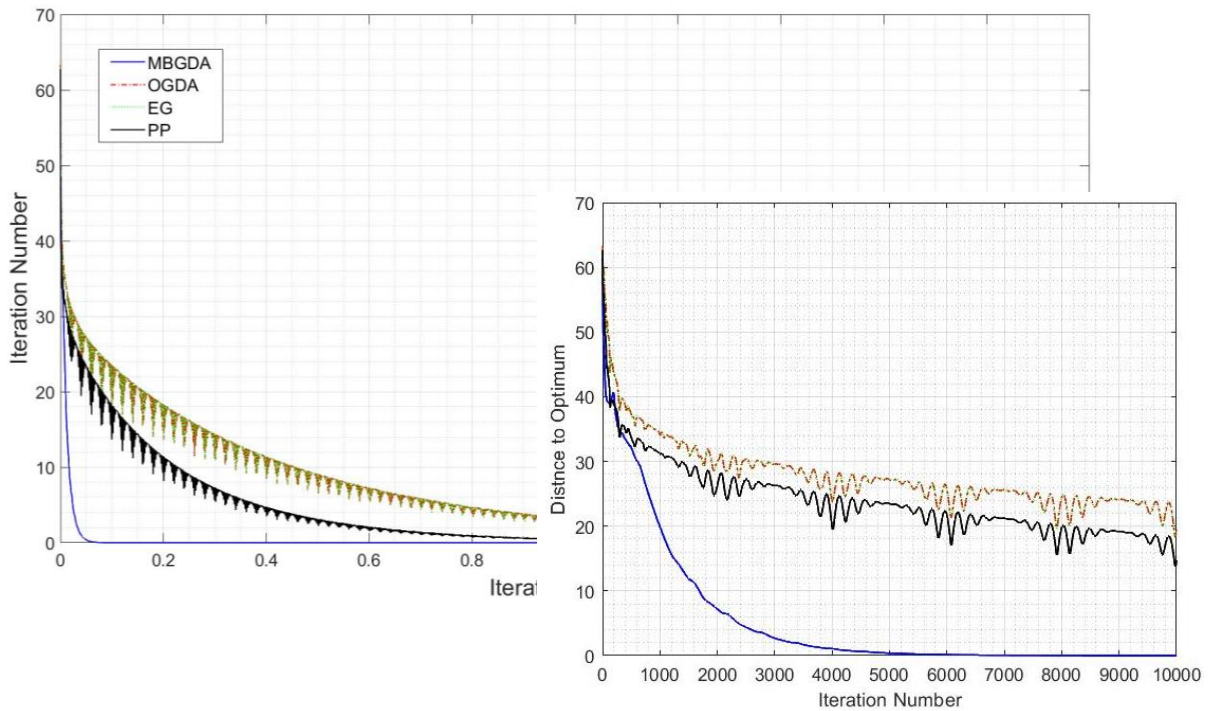


Figure 6 The distance to the optimal solution of MBGDA, GDA, OGDA, EG and PP at every iteration

Figure 6 illustrates the distance to the optimal solution of this bilinear problem versus the number of iterations for each of the considered algorithms: OGDA, EG and MBGDA. OGDA

and EG show similar behavior while MBGDA shows a significantly better performance with a faster convergence to the optimal solution. Unlike other algorithms, MBGDA shows a continuous decrease in the distance between its output vectors and the optimal solution without any oscillations. This proves that the points were only updating in the right direction determined by the momentum, thus creating an efficient process which aligns with the main motivation of this algorithm.

B. Experiment II Convergence Rate

We conducted the second set of experiments to study the convergence dynamics of MBGDA algorithm in the bilinear problem. We aim at showing the convergence of the algorithm as a function of the size of matrix \mathbf{A} , the position of the initial points, and the kappa statistic and compare its performance with Optimistic Gradient Descent Ascent, Extra Gradient, and Proximal Point methods.

We first start by exploring the convergence behavior of MBGDA as the **size of the matrix scales** (Lei, Nagarajan, Panageas, & Wang, 2018). We conduct the experiment with an input square matrix \mathbf{A} of size $n \times n$ for $n= 5, 10, 15, \dots, 100$. Matrix \mathbf{A} is a full rank matrix generated with random variables sampled from $[-1, +1]$. The starting vectors $\mathbf{x}_0 = \mathbf{y}_0 = [0.1, \dots, 0.1]$ and the accuracy is fixed at 0.01. The step size is sufficiently small and fixed at 0.01. The step size is sufficiently small and fixed at 0.01. For more accurate results, the experiment is repeated 10 times per matrix size and the average of number of iterations is documented. Figure 7 shows the number of iterations needed by MBGDA method to reach the optimal solution as a function of the number of rows of the matrix \mathbf{A} . Figure 8 illustrates the number of iterations of each of MBGDA,

OGDA, EG and PP as a function of size of matrix \mathbf{A} . The plot shows that MBGDA outperforms other algorithms with smaller convergence rate regardless of the size of \mathbf{A} . However, similar to EG and OGDA, MBGDA witnessed almost three times increase in the number of iterations as shown in figure 8. This shows that the convergence rate of MBGDA is affected by the size of the matrix \mathbf{A} but remains more efficient than the other algorithms.

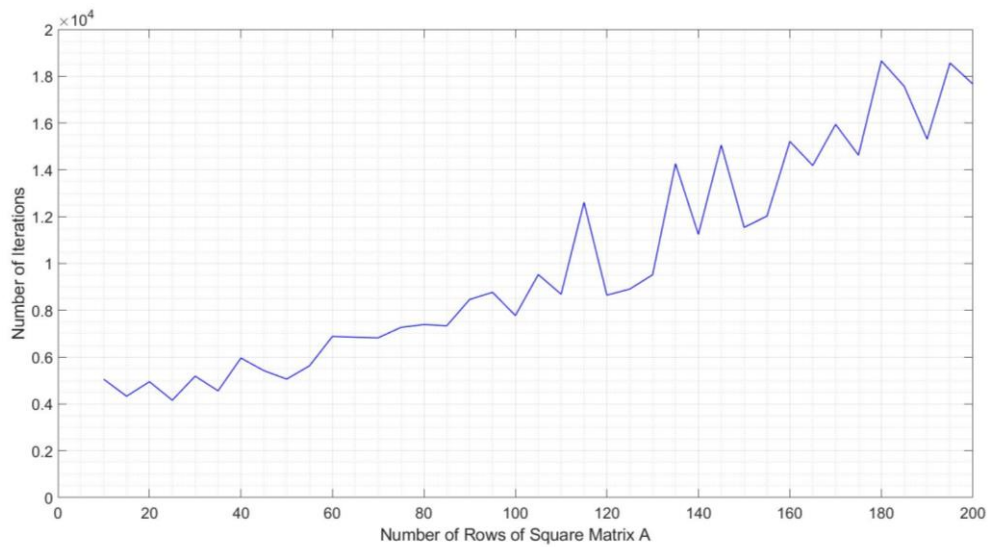


Figure 7 The number of iterations as function of matrix size using MBGDA algorithm.

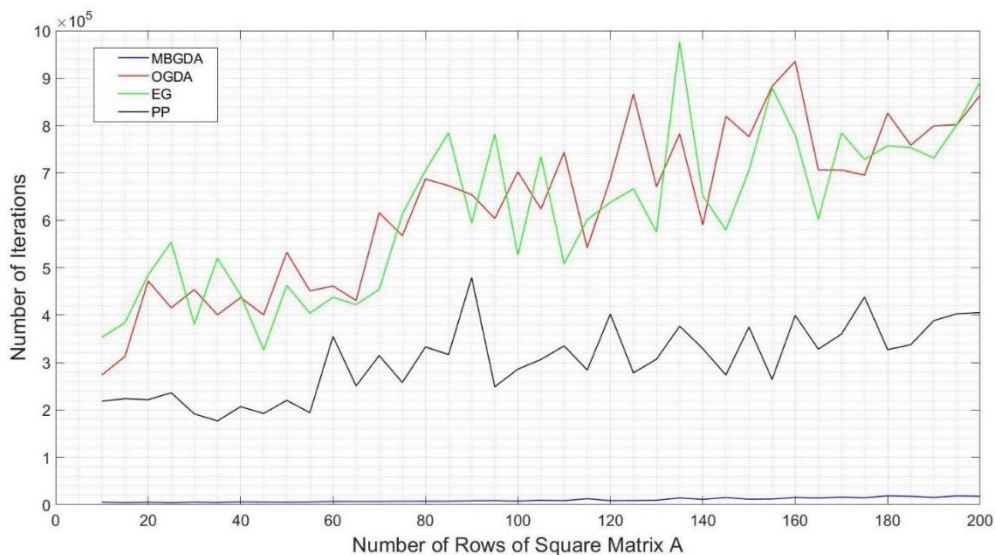


Figure 8 The number of iterations to converge using each of MBGDA, OGDA, EG and PP methods as function of matrix size.

We then conduct the second experiment to study the convergence rate of MBGDA as the **initialization varies**. We repeat the experiment to monitor the variation of number of iterations as a function of radius and distance from the initial point to the optimum. The initial point $\mathbf{x}_0 = \mathbf{y}_0 = (1/d) \times [0.1, \dots, 0.1]$ where $d = \{1, 1.1, 1.2, \dots, 20\}$. The step size is fixed at 0.01. The input matrix \mathbf{A} of size $n \times n$ for $n = 100$ is generated with random variables sampled from $[-1; +1]$.

Figure 9 illustrates the number of iterations of each of MBGDA, OGDA, EG and PP as a function of vector initialization. MBGDA shows constantly faster convergence rate regardless of the distance between the initial point and the optimal solution. OGDA and EG showed similar performance. However, when closely inspecting the performance of MBGDA, we notice that the convergence rate is affected by the initial distance to the optimum.

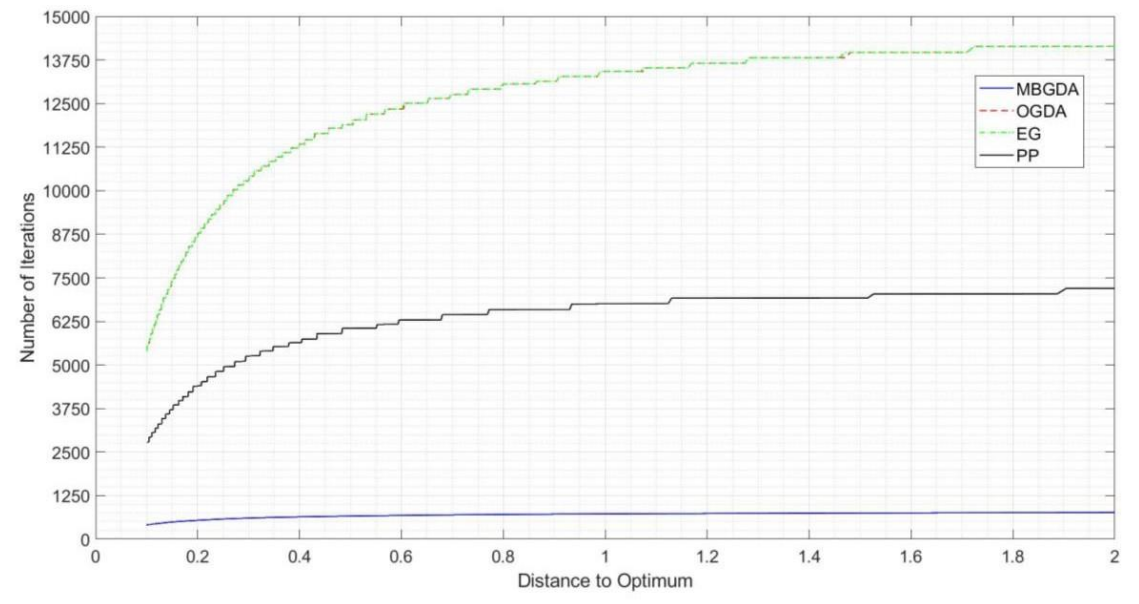


Figure 9 The number of iterations to converge using each of MBGDA, OGDA, EG and PP methods as a function of vector initializations.

We also notice that the convergence rate of several algorithms in traditional optimization depend on the range of eigenvalues of the Hessian matrix, we denote as the

kappa statistic κ . It is a parameter used to measure the interrater reliability and agreement of the data under study (Cohen, 1960). This is achieved by measuring the intercorrelation between the data of the matrix (McHugh, 2012). The convergence rates of the existing algorithms for solving bilinear optimization problems, as shown in Table 1, are dependent on the kappa statistic. In this experiment, we measure how MBGDA performs for different kappa values.

We conduct this experiment with a square matrix \mathbf{A} . The elements of the starting vectors \mathbf{x} and \mathbf{y} are generated randomly between $[-1; +1]$ and are constant throughout the experiment. The step size is sufficiently small at 0.001. To ensure a variable κ , we generate a matrix \mathbf{M} with size $n = 15$ and elements as normal random variables then subtract a diagonal matrix with equal elements m . The experiment is repeated for $m = 1, 1.25, 1.5, \dots, 10$. As we increased the value of m , the ratio of the largest to the lowest value in the matrix decreased, thus decreasing the value of the kappa statistic κ . The results show that the number of iterations was not directly affected by the value of m .

In conclusion, these sets of experiments show that the convergence rate of MBGDA is affected by the matrix size and initialization. They also prove that MBGDA achieve a faster convergence when compared to that of Optimistic Gradient Descent Ascent, Proximal Point and Extra Gradient methods for solving bilinear problems under similar conditions.

C. Experiment III Non-Convex Non-Concave Cases

We apply the experiment constructed by (Daskalakis & Panageas, 2018) to test the efficiency of MBGDA in the non-convex non-concave settings and compare it to that

of GDA and OGDA. The problem has a local min-max and {GDA, OGDA}- critical points that are linearly stable with respect with GDA/OGDA dynamics respectively for a fixed step size α (Daskalakis & Panageas, 2018).

Consider the following functions $f_1(x, y)$ and $f_2(x, y)$

$$\begin{cases} f_1(x, y) = -\frac{1}{8}x^2 - \frac{1}{2}y^2 + \frac{6}{10}xy \\ f_2(x, y) = \frac{1}{2}x^2 + \frac{1}{2}y^2 + 4xy \end{cases}.$$

The function f_1 has a property that the point $(0,0)$ is GDA critical point while f_2 has a property that the point $(0,0)$ is a GDA critical point but not a local min-max. The function f constructed by (Daskalakis & Panageas, 2018) behaves like f_1 around $(0, 0)$ and like f_2 around $(1, 1)$. The constructed polynomial function is

$$f(x, y) = f_1(x, y)(x - 1)^2(y - 1)^2 + f_2(x, y)x^2y^2$$

We will start by defining the stationary points as a couple (x^*, y^*) with zero gradient; thus, there is no feasible descent direction (Gidel, Berard, Vincent, & Lacoste-Julien, 2018) and

$$\|\nabla_x f(x^*, y^*)\| = \|\nabla_y f(x^*, y^*)\| = 0$$

We use random initializations generated in the range $[-1, 1]$. The accuracy is set to be 0.001. After repeating the experiment 10,000 times, we record the points that each of the algorithms finds. This allows us to get the probability of convergence of MBGDA, GDA and OGDA to each of the critical points.

Table 2 shows the results for each of these points. The frequency of convergence of MBGDA was greater than that of GDA and OGDA for each of the critical points. It

had better performance with larger step size. We also observe that the points (1, 0) and (1, 1) are unstable critical points.

Table 2 Convergence properties of MBGDA, GDA and OGDA on the critical points of f

Step size	0.01			0.1			0.2		
Critical Point	MBGDA	GDA	OGDA	MBGDA	GDA	OGDA	MBGDA	GDA	OGDA
(0,0)	58.97%	54.20%	54.63%	55.93%	50.24%	48.47%	50.45%	43.93%	35.62%
(0,1)	24.32%	22.00%	23.31%	20.00%	19.33%	14.86%	17.25%	15.83%	9.51%
(1,0)	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
(1,1)	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Convergence Rate	85.28%	76.62%	78.25%	76.14%	69.83%	63.56%	67.90%	59.94%	45.22%
Average nb. of iterations	5472	2445	2434	472	245	274	238	125	146

We then monitor the performance of MBGDA, GDA and OGDA in the nonconvex nonconcave problem $g(x, y) = axy + by^2 - cx^2$ where a , b and c are positive integers. The problem is also constructed with 10,000 random initializations randomly generated in the range $[-1, 1]$. The accuracy is set to be 0.001. The convergence properties of the three algorithms as the step size increases are shown in table 3. Table 3 shows that MBGDA and GDA achieve similar convergence probabilities with increasing step size value. This result was not achieved by the OGDA method.

Table 3 Convergence properties of MBGDA, GDA and OGDA on the critical points of f

Step size	0.01			0.1			0.2		
Convergence Rate	MBGDA	GDA	OGDA	MBGDA	GDA	OGDA	MBGDA	GDA	OGDA
	24.73%	25.82%	20.46%	23.88%	24.54%	13.87%	25.05%	25.51%	12.69%

After analyzing the MBGDA dynamics around critical points, we compared it to the first order methods GDA and OGDA. The results indicate that all three methods failed to

completely converge to the local min-max solutions. Yet, MBGDA was able to achieve a better convergence rate.

CHAPTER V

CONCLUSION

The interest to solve saddle point problems has increased with the increase of these problems in machine learning problems, game theory and Generative Adversarial Networks (GANs). Several first order algorithms have been developed to solve these problems. The most commonly used algorithm is the Gradient Descent Ascent (GDA) that even fails to converge in bilinear settings. Other algorithms include but are not limited to Optimistic Gradient Descent Ascent (OGDA), Extra Gradient (EG), and Proximal Point (PP).

We developed an algorithm that solves the bilinear problem by dynamically updating the coordinates of \mathbf{x} and \mathbf{y} in the favorable direction. The Momentum Block Gradient Descent Ascent algorithm (MBGDA) is a GDA based algorithm that performs a gradient descent on \mathbf{x} and a gradient ascent on \mathbf{y} . It also utilizes the negative momentum used as an OGDA update metrics to dynamically determine the coordinates to be updated at every iteration. Its main characteristic is this dynamic selection that allows it to reach Nash equilibrium efficiently and with minimum oscillations.

We have conducted several experiments to compare the performance of MBGDA with GDA, OGDA, PP and EG. MBGDA experimentally proved better performance compared to existing first order algorithms. MBGDA can solve bilinear problems efficiently with a better rate regardless of the initialization, size of data and kappa statistic. MBGDA also achieves better convergence rates and similar convergence probabilities in non-convex non-concave settings.

REFERENCES

- Adler, I. (2013, February). The equivalence of Linear Programs and Zero-sum Games. *International Journal of Game Theory*, 42, 165-177. doi:10.1007/s00182-012-0328-8
- Adolphs, L., Daneshmand, H., Lucchi, A., & Hofmann, T. (2019). Local Saddle Point Optimization: A Curvature Exploitation Approach. *22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*. 89. Naha, Okinawa, Japan: Department of Computer Science, ETH Zurich. doi:1805.05751
- Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., & Marchand, M. (2015). Domain-Adversarial Neural Networks. *I*. doi:1412.4446
- B. Dai, A. S. (2018). Convergent reinforcement learning with nonlinear function approximation. *International Conference on Machine Learning*, (pp. 1133–1142).
- Baygun, B., & Hero, A. O. (1996, June). An iterative solution to the min-max simultaneous detection and estimation problem. *Proceedings of the 8th IEEE Signal Processing Workshop on Statistical Signal and Array Processing (SSAP '96)*. doi:10.5555/829533.831433
- Beck, A., & Tsetuashvili, L. (2013). On the Convergence of Block Coordinate Descent Type Methods. *SIAM Journal on Optimization*, 23(4), 2037–2060. doi:10.1137/120887679
- Berger, U. (2007, 02 01). Brown's Original Fictitious Play. *Journal of Economic Theory*, 135, 572-578. doi:10.1016/j.jet.2005.12.010

- Borji, A. (2021, March 17). Pros and Cons of GAN Evaluation Measures: New Developments. doi:arXiv preprint arXiv:2103.09396.
- Brownlee, J. (2019, August 28). *How to Implement the Inception Score (IS) for Evaluating GANs*. Retrieved from Machine Learning Mastery.
- Bubeck, S. (2015). Convex Optimization: Algorithms and Complexity. *In Foundations and Trends in Machine Learning*, 8, 231-357. doi:10.1561/22000000050
- Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, XX(1).
- Daskalakis, C., & Panageas, I. (2018). *Last-Iterate Convergence: Zero-Sum Games and Constrained Min-Max Optimization*.
- Daskalakis, C., & Panageas, I. (2018). *The Limit Points of (Optimistic) Gradient Desceny in Min-Max Optimization*.
- Daskalakis, C., & Panageas, I. (2018). Last-Iterate Convergence: Zero-Sum Games and Constrained Min-Max Optimization.
- Daskalakis, C., Ilyas, A., Syrgkanis, V., & Zeng, H. (2017, August 13). Training GANs with Optimism. *In International Conference on Learning Representations*. doi:arXiv preprint arXiv:1711.00141
- Gidel, G., Askari Hemmat, R., Pezeshki, M., Le Prio, R., Huang, G., Lacoste-Julien, S., & Mitliagkas, I. (2020). Negative Momentum for Improved Game Dynamics. doi:arXiv:1807.04740v5
- Gidel, G., Berard, H., Vincent, P., & Lacoste-Julien, S. (2018). A variational inequality perspective on generative adversarial nets. doi:arXiv preprint arXiv:1802.10551

- Gill, E. P., Murray, W., & Wright, H. M. (1981). *Practical optimization*. London: Academic Press Inc.
- Glowinski, R. (1984). *Numerical Methods for Nonlinear Variational Problems* (1 ed.). New York: Springer-Verlag Berlin Heidelberg. doi:10.1007/978-3-662-12613-4
- Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative Adversarial Nets. (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger, Eds.) *Advances in Neural Information Processing Systems*, 2672-2680. Retrieved from <http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf>
- Jason, B. (2019, August 30). *Machine Learning Mastery*. Retrieved from How to Implement the Frechet Inception Distance (FID) for Evaluating GANs.
- Jin, C., Netrapalli, P., & Jordan, M. I. (2020). What is Local Optimality in Nonconvex-Nonconcave Minimax Optimization? *37th International Conference on Machine Learning*. Proceedings of Machine Learning Research. doi:119:4880-4889
- Joseph, M., Kearns, M., Morgenstern, J. H., & Roth, A. (2016). Fairness in Learning: Classiv and contextual bandits. *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, (pp. 325-333).
- Konno, H. (1975). Bilinear Programming.
- Korpelevich, G. (1976). The extragradient method for finding saddle points and other problems. *Matecon*, 12, 747-756.

- Lei, Q., Nagarajan, G. S., Panageas, I., & Wang, X. (2018, Feb 13). Last iterate convergence in no-regret learning: constrained min-max optimization for convex-concave landscapes. 2.
- Liang, T., & Stokes, J. (2019). Interaction matters: A note on non-asymptotic local convergence of generative adversarial networks. *22nd International Conference on Artificial Intelligence and Statistics* (pp. 907-915). Naha, Okinawa, Japan: AISTATS.
- Liu, C.-S. (2013). An Optimally Generalized Steepest-Descent Algorithm for Solving Ill-Posed Linear Systems. (H.-S. Shen, Ed.) *Journal of Applied Mathematics*, 2013. doi:10.1155/2013/154358
- M. Sanjabi, J. B. (2018). On the convergence and robustness of. *Advances in Neural Information Processing*, 7091–7101.
- Martinet, B. (1970). Brève communication. Régularisation d'équations variationnelles par approximations successives. *Revue française d'informatique et de recherche opérationnelle.*, 4, 154-158.
- McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochem Med (Zagreb)*, 22(3), 276-282.
- Modersitzki, J., & Haber, E. (2007, March). Image Registration with Guaranteed Displacement Regularity. *International Journal of Computer Vision* 71, 361–372. doi:10.1007/s11263-006-8984-4
- Mokhtari, A., Ozdaglar, A., & Pattathil, S. (2020). A Unified Analysis of Extra-gradient and Optimistic Gradient Methods for Saddle Point Problems: Proximal Point

Approach. *23rd International Conference on Artificial Intelligence and Statistics (AISTATS), 108*. Palermo, Italy.

Nahapetyan, A. G. (2007). *Bilinear Programming*.

Nocedal, J., & Wright, S. (1999). *Numerical Optimization* (Vol. Springer Series in Operation Research and Financial Engineering). New York: Springer, New York, NY. doi:10.1007/b98874

Nouiehed, M., Sanjabi, M., Huang, T., Lee, J. D., & Razaviyayn, M. (2019). Solving a Class of Non-Convex Min-Max Games Using Iterative First Order Methods. *33rd Conference on Neural Information Processing Systems*. Vancouver, Canada: NeurIPS.

Rafique, H., Liu, M., Lin, Q., & Yang, T. (2019). *Non-convex min-max optimization: Provable Algorithms and Applications in Machine Learning*. doi:arXiv preprint arXiv:1810.02060

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2018, June 1). Do CIFAR-10 Classifiers Generalize to CIFAR-10? *CoRR*. doi:1806.00451

Rockafellar, R. T. (1976). Augmented lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of operations research*, 97–116.

Tseng, P. (1995). On linear convergence of iterative methods for the variational inequality problem. *Journal of Computational and Applied Mathematics*., 60(1-2), 237-252.

Yan, Y., Xu, Y., Lin, Q., Liu, W., & Yang, T. (2020, June 17). Optimal Epoch Stochastic Gradient Descent Ascent Methods for Min-Max Optimization.

Zhang, G., & Wang, Y. (2021). On the Suboptimality of Negative Momentum of
Minimax Optimization. doi:arXiv:2008.07459v4