EXPERIMENTAL EVALUATION OF ITERATIVE METHODS FOR GAMES

by
Mengtong Xu

A thesis submitted to Johns Hopkins University in conformity with the requirements for the degree of Master of Science in Engineering

Baltimore, Maryland
Dec, 2022

Abstract

Min-max optimization problems are a class of problems that are usually seen in game theory, machine learning, deep learning, and adversarial training. Deterministic gradient methods, such as gradient descent ascent (GDA), Extragradient (EG), and Hamiltonian Gradient Descent (HGD) are usually implemented to solve those problems. In large-scale setting, stochastic variants of those gradient methods are prefer because of their cheap per iteration cost. To further increase optimization efficiency, different improvements of deterministic and stochastic gradient methods are proposed, such as acceleration, variance reduction, and random reshuffling.

In this work, we explore advanced iterative methods for solving min-max optimization problems, including deterministic gradient methods combined with accelerated methods and stochastic gradient methods combined with variance reduction and random reshuffling. We use experiments to evaluate the performance of the classical and advanced iterative methods on both bilinear and quadratic games.

With an experimental approach, we show that the most advanced iterative methods in the deterministic and stochastic setting have improvements in iteration complexity.

Thesis Readers

Dr. Nicolas Loizou (Advisor)
Professor
Department of Applied Mathematics and Statistics
Johns Hopkins University

Acknowledgements

I would thank Professor Nicolas Loizou for the support and guidance throughout the planning and writing of this thesis. I would also thank Siqi Zhang, Sayantan Choudhury, Zhichao Jia, Yichuan Wang and Konstantinos Emmanouilidis for informative discussions and feedback.

Contents

List of Figures

Chapter 1

Introduction

Min-max optimization problems play a crucial part in classical game theory and generative adversarial networks,[1] and it is also applied in a wide range of novel Machine Learning problems. Statistics,[2, 3] online learning,[4] deep learning,[5] distributed computing,[6, 7] multi-agent reinforcement learning,[8] and adversarial training[9] are also fields of mathematics and computer science that use using min-max optimization methods. In addition, there is a growing awareness that machine learning systems used for real-life problems that involve scarcity or competition are always consistent with game-theoretic constraints.[10]

Unconstrained min-max optimization problems on games have the form as

$$\min_{x \in \mathbb{R}^{d_1}} \max_{x \in \mathbb{R}^{d_2}} f(x, y) \tag{1.1}$$

where $f : \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \to \mathbb{R}$ is an objective with two vector input x, y. The goal for this problem is to find $(x^*, y^*) \in \mathbb{R}^{d_1+d_2}$ as a min-max solution, also known as a saddle point, or Nash equilibrium of (1.1), such that

$$f(x^*, y) \leq f(x^*, y^*) \leq f(x, y^*) \; for \; every \; x \in \mathbb{R}^{d_1} \; and \; y \in \mathbb{R}^{d_2} \tag{1.2}$$

The problem (1.1) is usually called a deterministic game.

In practice, problems like domain generalization,[11] generative adversarial networks,[1] and some formulations in reinforcement learning[12] all have finite sums game of the

form of (1.1) when doing empirical risk minimization. We name this formulation as a stochastic game, in which the objective f is expanded into a finite sum version (1.3) as below :

$$\min_{x\in\mathbb{R}^{d_1}} \max_{y\in\mathbb{R}^{d_2}} f(x,y) = \frac{1}{n}\sum_{i=1}^{n} f_i(x,y) \tag{1.3}$$

where $f : \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \to \mathbb{R}$. The parameter n represents the number of data a problem has, and the value of n in practice is usually very big.

The min-max solution (x^*, y^*) of (1.1) or (1.3) is first a stationary point of the objective function f such that

$$F(x^*, y^*) := (\nabla_x f(x^*, y^*), -\nabla_y f(x^*, y^*))^T = 0. \tag{1.4}$$

Thus, finding a min-max solution needs to first find a stationary point.

An unconstrained Stochastic Variational Inequality Problem (VIP) is a problem that aims to find a point $z^* \in \mathbb{R}^d$ for function $F_i : \mathbb{R}^d \to \mathbb{R}^d$ such that

$$F(z^*) = \frac{1}{n}\sum_{i=1}^{n} F_i(z^*) = 0. \tag{1.5}$$

If $z := (x, y)$ and $F_i(z) := (\nabla_x f(z), \nabla_y f(z))$, then solving VIP is equivalent to finding a stationary point for (1.3). If (1.3) has a unique stationary point, the stationary point is its global min-max solution. This kind of situation can happen, for example, when f_i is convex-concave functions. Thus, min-max problems can be regarded as a special case of VIP.

Multiple algorithms can be used to solve minimax problems. For first-order methods, which are algorithms that use first-order derivative information in each iteration, some commonly used algorithms are Gradient Descent Ascent (GDA), Optimistic Gradient (OG), and Extragradient (EG). For second-order methods, Hamiltonian Gradient Descent (HGD) and Consensus Optimization (CO), which use the information of the Jacobin matrix in each update, are commonly used. There are also some variants based on those methods, for example, Alternative Gradient Descent Ascent (AGDA),

which alternatingly updates the primal-dual variables. All of those methods are in the deterministic setting that uses all the data information to compute the full gradient in each iteration.

Although significant advances in the deterministic setting have been witnessed in recent years,[13, 14, 15, 16, 17, 18, 19, 20,21] it is shown that deterministic variants require expensive updates, so it becomes infeasible and impractical for model training, especially when the number or dimension of a dataset is large. This is where the stochastic settings of the above algorithms emerge. Since the stochastic setting of gradient methods based on mini-batches reduces computational load and is more relevant and applicable to real-world ML problems, it is important and worth analyzing.

To be specific, deterministic gradient methods are used to solve (1.1), and stochastic gradient methods are used to (1.3). If problems are using the same objective f of (1.1) and (1.3), then the difference between stochastic gradient methods and deterministic ones is only the usage of an unbiased estimator of the full gradient of f instead of the full gradient itself.

Nowadays, people care about both computation and iteration complexity. Accelerated methods such as Heavy-ball momentum and Nesterov's optimal are proposed for deterministic methods, and people even use those accelerated methods in stochastic settings. For stochastic cases, variance reduction and random reshuffling are two commonly used methods that always have good performance in practice.

In this thesis, we will cover some gradient-related methods including GDA, HGD, and their stochastic version, to solve the min-max problem. More details on pseudocodes for those algorithms will be shown in Chapter 2. With an experimental approach, we explored different combinations of gradient methods, including deterministic variants combined with accelerated methods and stochastic variants combined with variance reduction and random reshuffling methods. Details on experiments will be shown in Chapter 4.

Chapter 2

Background

In this chapter, we will provide the necessary background information required for the rest of this thesis, and describe related work. We will present an overview of unconstrained optimization, definitions of some properties, and various gradient methods with their corresponding pseudocode. In addition, two examples of the aforementioned algorithms will be provided as a general outlook.

2.1 Unconstrained Min-max Problem

In this thesis, we are focusing on solving unconstrained optimization problems in the form of (1.1) or (1.3), and aiming on finding a saddle point (x^*, y^*), such that

$$f(x^*, y) \leq f(x^*, y^*) \leq f(x, y^*) \; for \; every \; x \in \mathbb{R}^{d_1} \; and \; y \in \mathbb{R}^{d_2} \tag{2.1}$$

If objective function f is differentiable, then a saddle point must be a stationary point of f, which means that $\nabla f(x^*, y^*) = 0$. Thus, by finding stationary points of objective function f, it is possible to eventually achieve a saddle point. If with the additional assumption that all stationary points are global min-max solutions of f, finding stationary points is equivalent to finding saddle points.

The most common way to find a saddle point is to denote an initial point (x_0, y_0) and update along the opposite side of the gradient in iteratively to find a fixed point. The goal of this method is to find a stationary point (x_k, y_k) that satisfied

$\nabla f(x_k, y_k) = 0$. This point may be the saddle point of the function $f(x, y)$. If all critical points are saddle points, the points the algorithm would find are the final solution. This situation can happen, for example, when objective functions are convex in x and concave in y.

A number of first-order and second-order methods are useful in solving this problem. First order and second order in the name of a method means that a first-order derivative or a second-order derivative information is used respectively during the iterative update. In section 2.3, more details about algorithms are provided.

2.2 Preliminary

Notation

In this thesis, we are using the standard notation for optimization literature, and we are also using $\mathbb{E}_\xi[\cdot]$ to denote the expectation taken w.r.t. the randomness coming from sampling ξ only.

Main Definitions

Definition 1 *Operator $F(x)$ is L-Lipschitz, if there exists $L > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$||F(x) - F(y)|| \leq L||x - y||. \tag{2.2}$$

Definition 2 *We say that a function f is L-smooth, if there exists $L > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$||\nabla f(x) - \nabla f(y)|| \leq L||x - y||. \tag{2.3}$$

Note that it is equivalent to saying the objective function f is L-smooth of a problem if its corresponding operator F is L-Lipschitz.

If $f = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$, then a more refined analysis of stochastic gradient methods has been proposed under new notations of expected smoothness(ES).

Definition 3 *We say that the function $f = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$ satisfies the expected smoothness with parameter $\mathcal{L}$, if there exists $\mathcal{L} > 0$ such that for all $x \in \mathbb{R}^d$,*

$$\mathbb{E}_i[||\nabla f_i(x) - \nabla f_i(x^*)||^2] \leq 2\mathcal{L}(f(x) - f(x^*)). \tag{2.4}$$

Definition 4 *We say that a function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if there exists a constant $\mu > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}||y - x||^2, \tag{2.5}$$

where x^* is the projection of x onto the solution set $\mathcal{X}^*$ minimizing f.

Definition 5 *We say that a function $f : \mathbb{R}^d \to \mathbb{R}$ is convex, if for all $x, y \in \mathbb{R}^d$*

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle. \tag{2.6}$$

Strongly concave or concave have the similar definitions as strongly convex or convex. We say a function is strongly concave if the function with the opposite sign is strongly convex, and we say it is concave if the function with the opposite sign is convex.

Definition 6 *Operator $F(x)$ is μ-quasi-strongly monotone, if for $\mu \geq 0$ and all $x \in \mathbb{R}^d$*

$$\langle F(x), x - x^*\rangle \geq \mu||x - x^*||^2, \tag{2.7}$$

where $x^* \in \mathbb{R}^d$ is the unique solution such that $F(x) = 0$.

Definition 7 *We say that a function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-quasi-strongly convex if there exists a constant $\mu > 0$ such that for all $x \in \mathbb{R}^d$*

$$f^* \geq f(x) + \langle \nabla f(x), x^* - x \rangle + \frac{\mu}{2}||x^* - x||^2, \tag{2.8}$$

where f^* is the minimum value of f and x^* is the projection of x onto the solution set $\mathcal{X}^*$ minimizing f.

Note that quasi-strong monotonicity has its roots in the quasi-strong convexity condition from the optimization literature. Thus, having quasi-strong monotonicity of operator F is equivalent to having quasi-strongly convexity of objective function f.

Definition 8 *We say that an operator F is ℓ-co-coercive, if there exists $\ell > 0$ such that for all $x, y \in \mathbb{R}^d$*

$$||F(x) - F(y)||^2 \leq \ell \langle F(x) - F(y), x - y \rangle. \tag{2.9}$$

We say that the operator is ℓ-co-coercive around w^*, if there exists $w^* \in \mathbb{R}^d$ and $\ell > 0$ such that for all $x \in \mathbb{R}^d$

$$||F(x) - F(w^*)||^2 \leq \ell \langle F(x) - F(w^*), x - w^* \rangle. \tag{2.10}$$

Note that w^* is not necessarily a point where $F(w^*) = 0$.

Definition 9 *We say that an operator F is ℓ_F-co-coercive in expectation with respect to a distribution $\mathcal{D}$, if there exists $\ell_F > 0$ such that for all $x \in \mathbb{R}^d$*

$$\mathbb{E}_{\mathcal{D}}[||F_i(x) - F_i(x^*)||^2] \leq \ell_F \langle F(x), x - x^* \rangle. \tag{2.11}$$

We write $F \in EC(\ell_F)$ to denote that the above inequality is satisfied with the expected co-coercivity constant ℓ_F.

Hamiltonian function

Hamiltonian gradient descent (HGD) has been proposed as an efficient method for solving min-max problems in Balduzzi's paper.[22] The method consists of performing gradient descent on a particular objective function $H(z)$, named the Hamiltonian function. Here $z := (x, y)$ is a vector with information of both x and y. The Hamiltonian function has the following form:

$$H(z) = \frac{1}{2}||F(z)||^2 \tag{2.12}$$

where $F(z)$ is the vector of appropriately assigned partial derivatives as:

$$F(z) = (\frac{\partial f}{\partial x}(z), -\frac{\partial f}{\partial y}(z)). \tag{2.13}$$

In other words, $H(z)$ is the square norm of the gradient $F(z)$.

In min-max problems, we at least need to find stationary points, and that happens when $F(z) = 0$. That is equivalent to $H(z) = 0$. Since $H(z)$ is the square norm of the gradient, it has a value larger or equal to zero. We can find a stationary point by finding a minimizer of $H(z)$. In some special cases or with some additional assumptions, all stationary points are equivalent to saddle points so that we can find the saddle points by minimizing $H(z)$ as :

$$\min_{z \in \mathbb{R}^{d_1+d_2}} H(z) \tag{2.14}$$

In the stochastic setting, we need an unbiased estimator of $\nabla H(z)$ as the update direction in each iteration. When $f(x, y)$ is of finite sum format as (1.3), then $F(z) = \frac{1}{2}\sum_{i=1}^{n} F_i(z)$, where $F_i(z) := (\nabla_x f(z), -\nabla_y f(z))$ for all $i \in [n]$. For Hamiltonian function and its gradient, we have the following form:

$$H(z) = \frac{1}{n^2}\sum_{i,j=1}^{n} \frac{1}{2}\langle F_i(z), F_j(z)\rangle \tag{2.15}$$

$$\nabla H(z) = \frac{1}{n^2}\sum_{i,j=1}^{n} \frac{1}{2}[J_i^T F_j + J_j^T F_i], \ \textit{where } J_i := \begin{bmatrix} \nabla^2_{x,x} f_i & \nabla^2_{x,y} f_i \\ -\nabla^2_{y,x} f_i & -\nabla^2_{y,y} f_i \end{bmatrix} \tag{2.16}$$

So, we can see that

$$\nabla H_{i,j}(z) = \frac{1}{2}[J_i^T F_j + J_j^T F_i] \tag{2.17}$$

is an unbiased estimator of $H(z)$.

2.3 Algorithms

In this part, we will show the pseudocode for some commonly used gradient methods for min-max problems in both deterministic and stochastic settings.

The gradient descent ascent (GDA) algorithm is the most prevalent and straightforward method for solving min-max problems (1.1). It is the first algorithm designed for a min-max problem, in which the descent part of the algorithm optimizes for the minimization part of the problem and vice versa. It uses the gradient of objection function f with respect to x as the update direction for minimization with respect to x. For maximization respect to y, it uses the inverse gradient of objection function f with respect to y as the update direction. The pseudocode of GDA is shown as Algorithm 1.

Algorithm 1 Gradient Descent Ascent (GDA)

Input: Starting step-size $\eta_{1,0} > 0, \eta_{2,0} > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$.

1: **for** k = 0, 1, 2, ..., K **do**
2: Set step-size $\eta_{1,k}, \eta_{2,k}$ following one of the selected choices(constant, decreasing)
3: Set $x_{k+1} = x_k - \eta_{1,k}\nabla_x f(x_k, y_k)$
4: Set $y_{k+1} = y_k + \eta_{2,k}\nabla_y f(x_k, y_k)$
5: **end for**

Output: (x_K, y_K)

The Stochastic Gradient Descent Ascent (SGDA) algorithm is the stochastic version of GDA, and it solves min-max problems of format (1.3). It is very similar as GDA, and the only difference is that SGDA use some unbiased estimators of gradients

instead of the full gradients in each update. The pseudocode of SGDA is shown as Algorithm 2.

Algorithm 2 Stochastic Gradient Descent Ascent (SGDA)

Input: Starting step-size $\eta_{1,0} > 0, \eta_{2,0} > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$. Distribution $\mathcal{D}$ of samples.

1: **for** k = 0, 1, 2, ..., K **do**
2: Generate fresh samples $i \sim \mathcal{D}$, and evaluate $\nabla_x f_i(x_k, y_k)$, $\nabla_y f_i(x_k, y_k)$
3: Set step-size $\eta_{1,k}, \eta_{2,k}$ following one of the selected choices(constant, decreasing)
4: Set $x_{k+1} = x_k - \eta_{1,k} \nabla_x f_i(x_k, y_k)$
5: Set $y_{k+1} = y_k + \eta_{2,k} \nabla_y f_i(x_k, y_k)$
6: **end for**

Output: (x_K, y_K)

The extragradient (EG) method is a standard algorithm to solve min-max problems, and it contains one extrapolation step and one update step in each iteration. The pseudocode of EG is shown as Algorithm 3. Lines 3 and 4 of Algorithm 3 are the extrapolation steps that compute gradient updates from the point of current iterate and get an extrapolated point $(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$. Lines 5 and 6 are the update step that updates the current iterate point (x_k, y_k) according to the direction gradients at the extrapolated point $(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$.

Algorithm 3 Extragradient (EG)

Input: Starting step-size $\eta_{1,0} > 0, \eta_{2,0} > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$.

1: **for** k = 0, 1, 2, ..., K **do**
2: Set step-size $\eta_{1,k}, \eta_{2,k}$ following one of the selected choices(constant, decreasing)
3: Set $x_{k+\frac{1}{2}} = x_k - \eta_{1,k} \nabla_x f(x_k, y_k)$
4: Set $y_{k+\frac{1}{2}} = y_k + \eta_{1,k} \nabla_y f(x_k, y_k)$
5: Set $x_{k+1} = x_k - \eta_{2,k} \nabla_x f(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$
6: Set $y_{k+1} = y_k + \eta_{2,k} \nabla_y f(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$
7: **end for**

Output: (x_K, y_K)

The Same-samples Stochastic Extragradient (S-SEG) algorithm is the stochastic version of EG, and it solves min-max problems of format (1.3). It is very similar to EG, and the only difference is that S-SEG uses some unbiased estimators of gradients

instead of the full gradients in each update. In each iteration, the same sample is used for the extrapolation and update steps. In other words, for the same f_i in one iteration, unbiased estimators for full gradients are calculated respectively. The pseudocode of S-SEG is shown as Algorithm 4.

Algorithm 4 Same-samples Stochastic Extragradient (S-SEG)

Input: Starting step-size $\eta_{1,0} > 0, \eta_{2,0} > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$. Distribution $\mathcal{D}$ of samples.

1: **for** k = 0, 1, 2, ..., K **do**
2: Generate fresh samples $i \sim \mathcal{D}$, and evaluate $\nabla_x f_i(x_k, y_k)$, $\nabla_y f_i(x_k, y_k)$
3: Set $x_{k+\frac{1}{2}} = x_k - \eta_{1,k}\nabla_x f_i(x_k, y_k)$
4: Set $y_{k+\frac{1}{2}} = y_k + \eta_{1,k}\nabla_y f_i(x_k, y_k)$
5: Set $x_{k+1} = x_k - \eta_{2,k}\nabla_x f_i(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$
6: Set $y_{k+1} = y_k + \eta_{2,k}\nabla_y f_i(x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}})$
7: **end for**

Output: (x_K, y_K)

Hamiltonian Gradient Descent (HGD) algorithm consists of performing a gradient on a particular objective function $H(z)$, which is half of the norm of the gradient of the original objective function. It updates the point by doing a gradient descent on the Hamiltonian function $H(z)$. The pseudocode of HGD is shown as Algorithm 5.

Algorithm 5 Hamiltonian Gradient Descent (HGD)

Input: Starting step-size $\eta_0 > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$. Let $z_0 = (x_0, y_0)$.

1: **for** k = 0, 1, 2, ..., K **do**
2: Set step-size η_k following one of the selected choices (constant, decreasing)
3: Set $z_{k+1} = z_k - \eta_k \nabla \mathcal{H}(z_k)$
4: **end for**

Output: $p_K = (x_K, y_K)$

The Stochastic Hamiltonian Gradient Descent (SHGD) algorithm is the stochastic version of HGD, and it solves min-max problems of format (1.3). It is very similar to HGD, and the only difference is that SHGD uses some unbiased estimators of gradients instead of the full gradients in each update. One way to choose an unbiased estimator

of $\nabla H(z)$ is by letting $\nabla H_{i,j}(z) = \frac{1}{2}[J_i^T F_j + J_j^T F_i]$ as (2.17), where $i \sim \mathcal{D}$ and $j \sim \mathcal{D}$ are fresh independent samples. The pseudocode of SHGD is shown as Algorithm 6.

Algorithm 6 Stochastic Hamiltonian Gradient Descent (SHGD)

Input: Starting step-size $\eta_0 > 0, \gamma_0 > 0$. Choose initial points $x_0 \in \mathbb{R}^d$, $y_0 \in \mathbb{R}^d$. Distribution $\mathcal{D}$ of samples. Let $z_0 = (x_0, y_0)$.

1: **for** k = 0, 1, 2, ..., K **do**
2: Generate fresh samples $i \sim \mathcal{D}$ and $j \sim \mathcal{D}$, and evaluate $\nabla \mathcal{H}_{i,j}(p_k)$
3: Set step-size η_k following one of the selected choices (constant, decreasing)
4: Set $z_{k+1} = z_k - \eta_k \nabla \mathcal{H}_{i,j}(z_k)$
5: **end for**

Output: $z_K = (x_K, y_K)$

2.4 Related Work

GDA and SGDA

The gradient descent ascent (GDA) algorithm and its stochastic format (SGDA) are the most prevalent and straightforward methods for solving minimax problems (1.1) and (1.3). They are the first algorithm designed for a min-max problem, in which the descent part of the algorithm optimizes for the minimization part of the problem and vice versa. The pseudocode of GDA and SGDA are shown as Algorithm 1 and 2

The core aim of the determinate version of GDA is to find an ε-approximate stationary point. GDA can find the solution within $O(\kappa^2 log(1/\varepsilon))$ iterations for strongly convex-strongly concave problems, and within $O(\varepsilon^2)$ iterations with decaying step-size for convex-concave games.[23], [24] For nonconvex-strongly concave min-max problems, it is shown that a two-time-scale GDA with $O(\kappa^2\varepsilon^2)$ gradient evaluations is capable of returning an ε-stationary point of the function $\Phi(\cdot) = max_{y \in Y} f(\cdot, y)$, where $\kappa > 0$ is a condition number.[16] In the nonconvex-concave setting, on the other hand, two-time scale GDA requires $O(\varepsilon^6)$ gradient evaluations.[16]

There is a simple deviation within GDA. The only difference is using simultaneous or alternating updates of the primal-dual variables, named GDA and AGDA, respectively. GDA is usually used. But when moving to bilinear games, Zhang et al. show that the iterates of GDA diverge linearly for any positive constant step-size.[25] By contrast, they also show that the iterates of AGDA stay bounded.

For the stochastic version, $F_i(x, y) := (\nabla_x f_i(x, y); \nabla_y f_i(x, y))$ represents the appropriate concatenation of the block-gradients of f_i, and operator $F(x, y) = \frac{1}{n}\sum_{i=1}^{n} F_i(x, y)$. Stochastic Gradient Descent Ascent (SGDA) is similar to GDA, and the only difference is that instead of using a full gradient, SGDA uses an unbiased estimator of the full gradient in each iteration.

When operator F is quasi-strongly monotone and satisfies the expected co-coercivity assumption, it is proven that SGDA with constant step-size converges linearly to a neighborhood, and with the help of decreasing step-size, it would be able to achieve the exact solution in sublinear $O(1/K)$ rate.[26] For nonconvex-strongly-concave min-max problems, two-time scale SGDA requires $O(\kappa^3\varepsilon^4)$ stochastic gradient evaluations a to return an ε-stationary point of the function $\Phi(\cdot) = max_{y\in Y} f(\cdot, y)$, where $\kappa > 0$is a condition number.[16] In the nonconvex-concave setting, $O(\varepsilon^8)$ stochastic gradient evaluations are required for two-time scale SGDA.[16] For all the stochastic cases that have been mentioned, if the problem is extrapolation or over-parameterized in addition, then SGDA with constant can also guarantee linear convergence to the exact solution like the corresponding deterministic case does.

While these algorithms have achieved good results in practice, especially in adversarial training, GDA algorithms with constant step-sizes are known for having the possibility of failing to converge for some general smooth function,[27] including bilinear games.[28] In the cases when they do converge, the stable limit point still can not always be guaranteed to be a local Nash equilibrium.[29, 30]

EG and SEG

The extragradient (EG) method is a standard and well-known algorithm to optimize VIP and min-max problems. The EG algorithm has been originally introduced by Korpelevich[31] and extended by Nesterov[32] and Nemirovski[24]. Stochastic versions of the extragradient have been recently analyzed for stochastic variational inequalities with bounded constraints.[33, 34] A linearly convergent variance-reduced version of the stochastic gradient method has also been proposed for strongly monotone variational inequalities.[35]

The variational inequality problem (VIP) is a slightly more general setting than the min-max setting. In a differentiable problem, its corresponding VIP designates the necessary first-order stationary conditions. Under the assumption that the objective functions of the differentiable game are convex-concave, the solutions of the VIP are also solutions of the original min-max problem.[36]

For the deterministic version, Mokhtari et al. 2020[38] proved a linear convergence rate for both EG in bilinear or strongly convex-strongly concave settings, with an iteration complexity of $O(\kappa log(1/\varepsilon))$, where κ is the condition number. They also established a sublinear convergence rate of $O(1/K)$ in terms of the function value difference of the averaged iterates for EG for convex-concave saddle point problems.[37]

For the stochastic version, the last-iterate SEG algorithm on bilinear games in the same sample and same step-size in the min-max optimization problem has been shown that it cannot converge in general even when the step-sizes are diminishing to zero.[39] Thus, using different step-size becomes important. It was also shown that in the bilinear game setting with some mild assumptions, iteration averaging allows SEG to converge at the sublinear rate of $O(1/\sqrt{K})$, where K is the processed number of samples of the algorithm. The convergence has been further boosted by combining iteration averaging with scheduled restarting.

Gorbunov et al.[36] give convergence analysis for stochastic versions of the extragradient of same-sample and independent-sample for stochastic variational inequalities with Lipschitz continuous and quasi-strongly monotonicity operator F.

HGD and SHGD

Hamiltonian Gradient Descent (HGD) and its stochastic version (SHGD) consists of performing a gradient on a particular objective function $H(z)$, which is half of the norm of the gradient of the original objective function. In practice, those algorithms usually show a fast convergence and perform well.

The last-iterate convergence rates for the deterministic Hamiltonian gradient descent (HGD) are first provided by Abernethy.[39] They included games that satisfy the sufficiently bilinear condition in there study. The authors introduced the stochastic setting and explain how a stochastic variant of HGD with decreasing step-size behaves by using the convergence results of Karimi.[40] However, their theoretical approach was not able to provide an efficient way of selecting the unbiased estimators of the gradient of the Hamiltonian function.

Loizou et al.[41] improve upon the methods that Abernethy[39] provided by introducing the first efficient variants and analysis of SHGD. In their study, they choose a practical unbiased estimator of the full gradient by using the recently proposed assumptions of expected smoothness and expected residual.[42, 43] They also indicate tight convergence guarantees for the deterministic HGD recovering the result of Abernethy. They give out convergence analysis of SHGD for stochastic bilinear games and stochastic sufficient bilinear games and give the convergence rate of deterministic HGD as a special case of SHGD.

Loizou et al. show that by assuming the Hamiltonian function $H(z)$ is quasi-strongly convex and expected smooth, SHGD with constant step-size converges linearly to a

neighborhood of the solution.[26] With the additional assumption that operator F is quasi-strongly monotone and is expected co-coercive, they also indicate that Stochastic Consensus Optimization (SCO) method reaches the exact solution of sublinear $O(1/K)$ rate when using some decreasing step-size.

However, the HGD method does not always helpful, since it may find unstable stationary points of the operator F or other local minima of $H(z) = \frac{1}{2}||F(z)||^2$. That is because even some convex-concave functions may even have a nonconvex Hamiltonian function such that $H(z)$ will have more than one critical point. In that case, people propose the consensus optimization (CO) method to combine the step direction of minimizing the value of $H(z)$ and the step direction of the gradient of the objective function with the appropriate sign.

2.5 Simple Examples

In this part, we will run the algorithms mentioned above on two simple basic bilinear games of the below format V1 and V2:

V1:

$$\min_{x\in\mathbb{R}^{10}} \max_{y\in\mathbb{R}^{10}} \frac{1}{10}\sum_{i=1}^{10} x^T A_i y + b_i^T x + c_i^T y,$$
$$\text{where } b_i, c_i \sim N(0, 1/10), \text{ and } A_i[p,q] = \begin{cases} 1 & \text{if } i = p = q \\ 0 & \text{else.} \end{cases} \tag{2.18}$$

V2:

$$\min_{x\in\mathbb{R}} \max_{y\in\mathbb{R}} xy + \frac{1}{100}\sum_{i=1}^{100} b_i x + c_i y, where\ b_i, c_i \sim N(0, 1/100) \tag{2.19}$$

For the second format example V2, we draw two 3D graphs to give a brief illustration of the properties of its objective function $f(x, y)$ and Hamiltonian function $H(x, y)$, and show as Figures 2-1 and 2-2 respectively. In Figure 2-1, Z-axis denotes the value

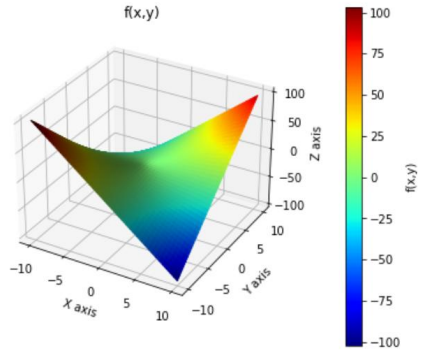

Figure 2-1. f(x,y) of V2

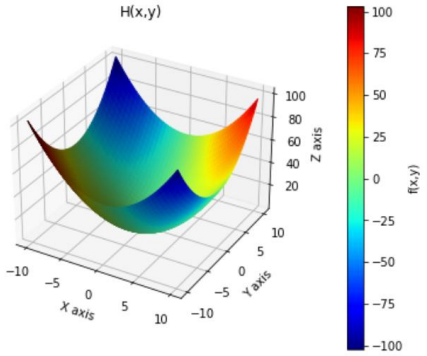

Figure 2-2. H(x,y) of V2

of $f(x, y)$; In Figure 2-2, Z-axis denotes the value of $H(x, y)$. In both figures, surface color represents the value of objective function $f(x, y)$. As shown by the color legend, the bigger the objective function, the warmer the color is.

The update trajectories results using various algorithms will be shown in the following parts.

2.5.1 GDA and SGDA

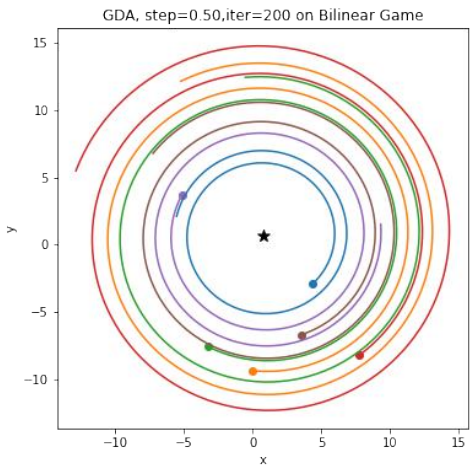

Figure 2-3. GDA on V1

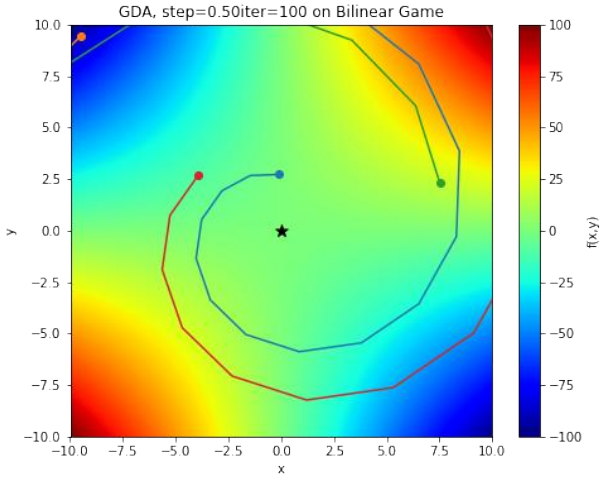

Figure 2-4. GDA on V2

We run GDA with a step-size of 0.5 and 200 iteration times for 6 random starting

points for the first version V1. For the second version V2, we run GDA with a step-size of 0.5 and 100 iteration times for 4 random starting points. Calculate by hand to find the optimum value by hand as $x = -\sum(c_i), y = -\sum(b_i)$ for V1 and $x = -\frac{1}{100}\sum(c_i), y = -\frac{1}{100}\sum(b_i)$ for V2. Use a black star to show the optimum point and together draw the trajectories. The plot is shown in Figures 2-3 and 2-4. From all starting points, we can see that the trajectories cycles. In Figure 2-3, there shows some divergence, and in Figure 2-4, a clear divergence can be witnessed. It shows that GDA diverges for our bilinear games.

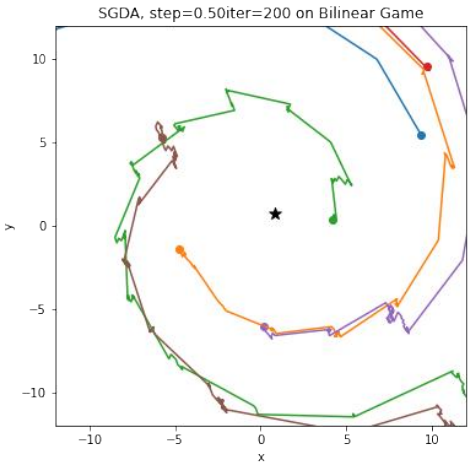

Figure 2-5. SGDA on V1

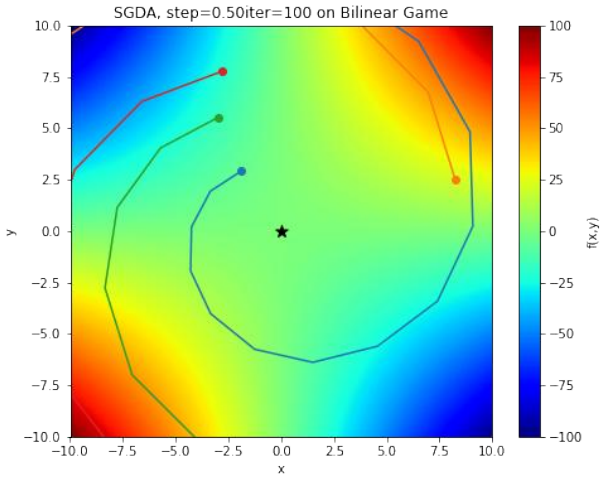

Figure 2-6. SGDA on V2

For the stochastic version, SGDA, we use single-element uniform sampling for each iteration. That is to uniformly choose one sample to calculate the gradient as the estimator of the whole true gradient. Step-sizes are 0.5 for both versions. The numbers of iterations are 200 and 100 for V1 and V2 respectively. Algorithms are run for 4 randomly chosen starting points. Results are shown in Figures 2-5 and 2-6. The trajectories in both figure circle to diverge. And compare with the deterministic case, the trajectories have more vibrating, i.e, not smooth as before.

2.5.2 EG and SEG

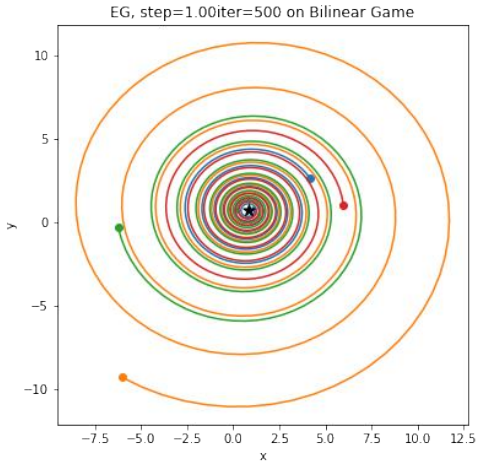

Figure 2-7. EG on V1

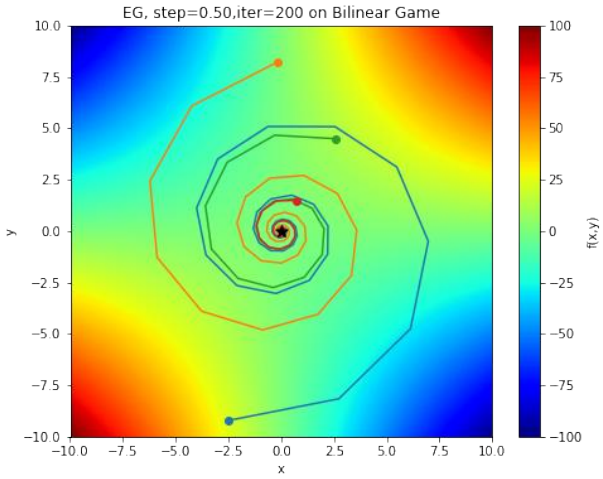

Figure 2-8. EG on V2

For EG, a step-size of 0.5 is again used. We run EG for 4 uniformly random chosen starting points of 500 iterations for V1 and 200 iterations for V2. The star-point is the hand-calculated optimum point. From Figures 2-7 and 2-8, it is shown that this method converges to the exact solution but with circling, i.e., not updates in the best direction.

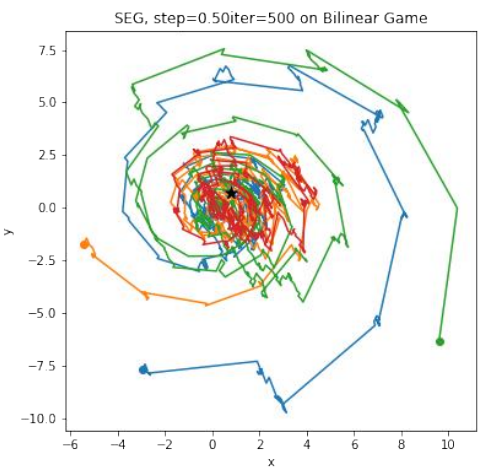

Figure 2-9. SEG on V1

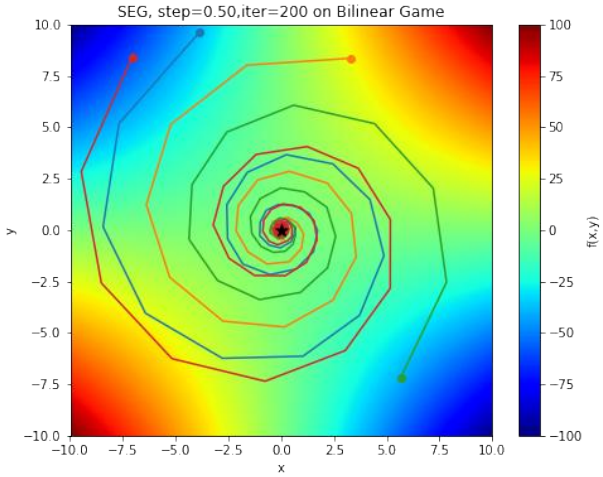

Figure 2-10. SEG on V2

For SEG, the stochastic version, we do the same thing, both with 0.5 step-size, 4 starting points, 500 iterations for V1, and 200 iterations for V2. More specifically about sampling, we use single-element uniform sampling for each iteration. We emphasize that we use the same sample during one iteration. In another word, the same sample is used for calculating the gradient, updating $x_{t+\frac{1}{2}}$, $y_{t+\frac{1}{2}}$ from x_t,y_t and updating x_{t+1},y_{t+1} from $x_{t+\frac{1}{2}}$, $y_{t+\frac{1}{2}}$.

Some more vibrating trajectories are obtained and shown in Figures 2-9 and 2-10. Although with some vibration, it can be seen that in both figure SEG methods circle to converge.

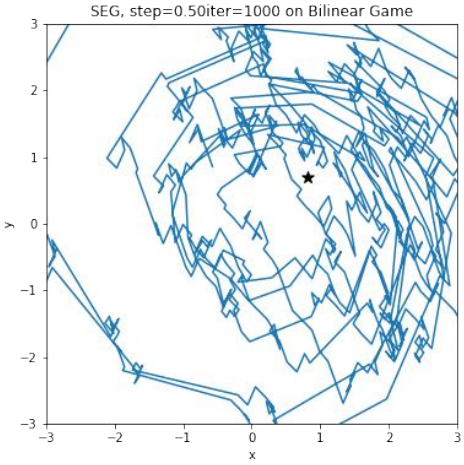

Figure 2-11. SEG on V1 of one starting

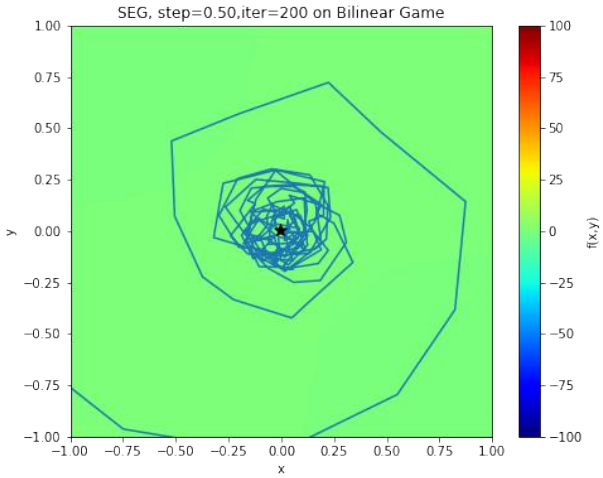

Figure 2-12. SEG on V2 of one starting

To see more clearly what happened around the optimum point, we run the algorithm again for one starting point. All the parameters remain the same except that for SEG on V1 we run 1000 iterations instead of 500. The results are shown in Figure 2-11 and 2-12. It can be witnessed that the trajectory circles at some neighbor of the optimum point with fluctuation.

2.5.3 HGD and SHGD

For HGD, use step-size as 0.5. Run algorithm for 4 uniformly random chosen starting points of 1000 iterations for V1 and of 100 iterations for V2. Star-point is the hand-calculated optimum point. In Figures 2-13 and 2-14, the trajectories converge straightforwardly to the optimum point as straight lines. In another word, they update in the best direction.

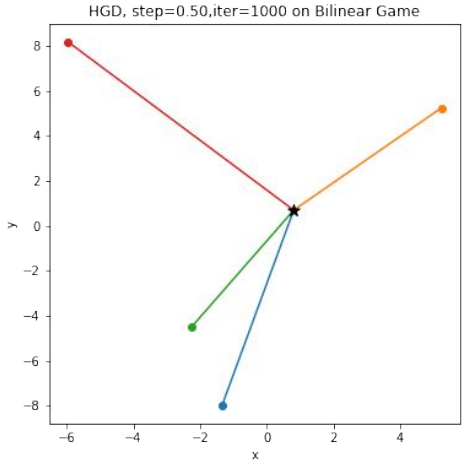

Figure 2-13. HGD on V1

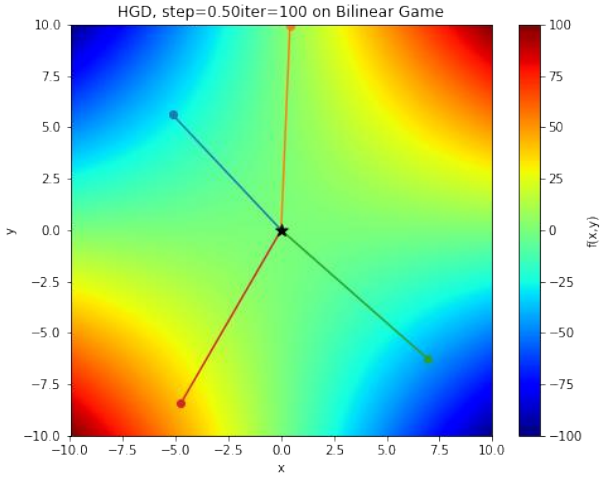

Figure 2-14. HGD on V2

For SHGD, the stochastic version, we do the same thing, both with 0.5 step-size, 4 starting points, 500 iterations for V1, and 200 iterations for V2. More specifically about sampling, we use two single-element uniform samplings for each iteration to calculate the unbiased estimator of true gradients of the Hamiltonian function.

Results are saved in Figures 2-15 and 2-16. Even though some vibrating can be witnessed, the trajectories still converge approximately to the optimum point. In addition, they don't circle the optimum. Their fluctuations are only around their best update directions.

To see more clearly what happened around the optimum point, we run the algorithm again for one starting point, zoom in, and show the result in Figures 2-17 and 2-18.

For V1, SHGD converges with fluctuations around the best updating direction to a neighbor of the optimum point. For V2, it almost converges according to the best direction, but vibrates in a very small neighbour of the the optimum point.

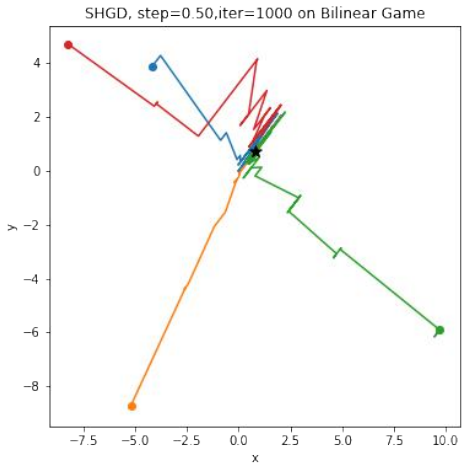

Figure 2-15. SHGD on V1

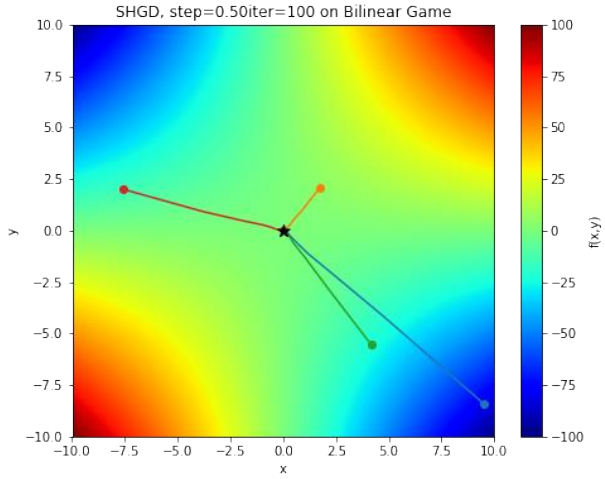

Figure 2-16. SHGD on V2

It can be seen that many natural algorithms, such as simultaneous GDA (diverge) and EG (circle to converge), provably diverge, cycle, or converge with circling even in this simple min-max setting. However, HGD in this problem updates in the direction of the line that points to the optimum point, i.e., updates straightforwardly according to the optimum direction. Thus, somehow the HGD method may increase convergence speed by reducing circling. It is very interesting to analyze and understand the performance of variants of Hamiltonian function-related gradient descent methods on min-max problems.

In this thesis, we will introduce some convergence theorems of GDA, HGD, and their stochastic versions. Then, give an experimental evaluation of iterative methods for games. In more detail, we will compare convergence performance for gradient methods with different parameters, e.g., step size and momentum parameters, on

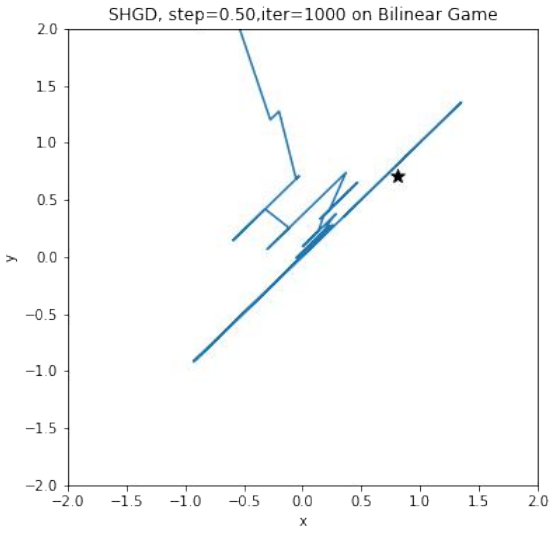

Figure 2-17. SHGD on V1 of one starting

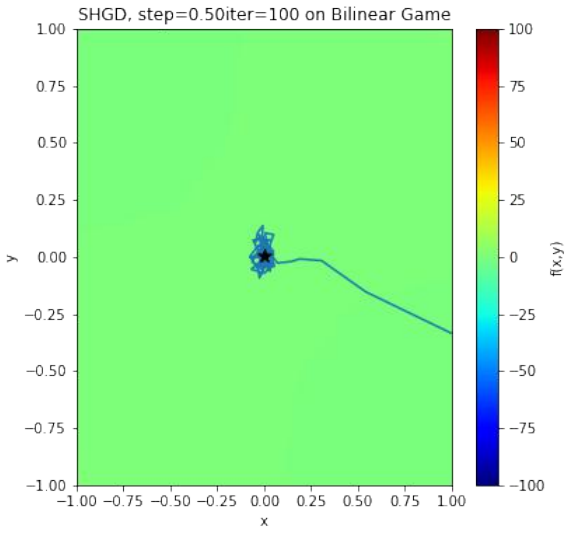

Figure 2-18. SHGD on V2 of one starting

some min-max problems. We will use experiments to evaluate the performance of using accelerated methods on HGD and GDA of different problems. We will also cover experimental performances of using variance reduction or random reshuffling on stochastic gradient methods on different games.

Chapter 3

Convergence Guarantee

In this chapter, we will first present the convergence theorem for GDA and HGD in both deterministic and stochastic settings. Then, we will cover some methods for deterministic or stochastic settings to achieve faster convergences.

Heavy-ball momentum and Nesterov's method are two general momentum methods that can be used to accelerate convergence for deterministic gradient methods. In practice, people usually use them even in the stochastic setting. But there is no convergence guarantee for them. For stochastic gradient algorithms, variance reduction and random reshuffling are two commonly used methods to improve convergence and they always have good performance in practice.

3.1 converge theorem for different algorithms

In this section, we will give the convergence theorem for variant algorithms on different classes of problems. We will cover Alternative Gradient Descent Ascent (AGDA) for bilinear games, and cover GDA and SGDA on min-max problems with quasi-strongly monotone and expected co-coercive operators. Then, we will show the theorems of HGD and SHGD on min-max problems with quasi-strongly convex and smooth Hamiltonian function H. We also cover some theorems of EG and SEG, which we will mention a little bit in the random reshuffling section of experimental

evaluation chapter.

3.1.1 GDA, AGDA, SGDA and SAGDA

Theorem 1 *(SGDA with constant step-size)*
Assume that operator F is μ-quasi strongly monotone and that $F \in EC(\ell_F)$. Choose $\eta_k = \eta \leq \frac{1}{2\ell_F}$ for all k. Then, the iterates of SGDA, given by Algorithm 2, satisfy:

$$\mathbb{E}[||x_k - x^*||^2] \leq (1 - \eta\mu)^k ||x_0 - x^*||^2 + \frac{2\eta\sigma^2}{\mu}, \tag{3.1}$$

where σ^2 is the variance at the optimum.

Proof 3.1.1 *Proved by Loizou, et al., 2021 [26].*

Theorem 1 indicates that SGDA with constant step-size converges linearly to a neighbourhood of the exact solution. For deterministic GDA, we can derive the convergence analysis from the theorem above by using the fact that $\sigma^2 = 0$ for deterministic GDA. Then, we get the following theorem.

Theorem 2 *(GDA with constant step-size)*
Assume that operator F is μ-quasi strongly monotone and is ℓ-co-coercive. Choose $\eta_k = \eta \leq \frac{1}{2\ell}$ for all k. Then, the iterates of GDA, given by Algorithm 1, satisfy: $\mathbb{E}[||x_k - x^*||^2] \leq (1 - \eta\mu)^k ||x_0 - x^*||^2$.

In theorem 1, it shows that SGDA has a linear convergence but it only reaches a neighborhood of the solution. That kind of step should be used when the precision is not important or when computing time is limited. However, there is some situation that precision of the solution is vital and have plenty of time. Using decreasing step-size helps to figure out the problem. It can guarantee convergence to the exact solution x^*.

In the below theorem, a switching step-size rule describes when to switch a constant step-size to a decreasing one. The reason to use a constant step-size in the initial iterations is to maintain the fast linear convergence at the first part. The theorem shows that after switching step-size, a sublinear convergence to the exact solution can be seen.

Theorem 3 *(SGDA with decreasing step-size)*
Assume that operator F is μ-quasi strongly monotone and that $F \in EC(\ell_F)$. Choose decreasing step-size as

$$\eta_k = \begin{cases} \frac{1}{2\ell_F} & for k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu} & for k > 4\lceil\kappa\rceil, \end{cases} \tag{3.2}$$

where $\kappa = \frac{\ell_F}{\mu}$. If $k \geq \lceil\kappa\rceil$, then iterates of SGDA, given by Algorithm 2, satisfy:

$$\mathbb{E}[||x_k - x^*||^2] \leq \frac{\sigma^2}{\mu^2}\frac{8}{k} + \frac{16\lceil K\rceil^2}{e^2k^2}||x_) - x^*||^2 = O(\frac{1}{k}), \tag{3.3}$$

where σ^2 is the variance at the optimum, and e is the Euler number.

Proof 3.1.2 *Proved by Loizou, et al., 2021 [26].*

Moving on to the alternating gradient descent ascent (AGDA), it is very similar to GDA, and the only difference is it updates the value of x_k, y_k alternately instead of simultaneously. It will be used as a base line when checking the performance of AGDA with negative momentum on bilinear games.

For AGDA and its stochastic version, the inaccuracy of each update point(x_k, y_k) is measured through the potential function

$$P_k = a_k + \frac{1}{\lambda}b_k, \tag{3.4}$$

where $a_k = \mathbb{E}[g(x_k) - g^*]$, $b_k = \mathbb{E}[g(x_k) - f(x_k, y_k)]$, and $\lambda > 0$ will be described in different theorems. Here $g(x) = max_y f(x, y)$, and $g^* = min_x g(x)$.

Theorem 4 *(SAGDA with constant step-size)*

Let function $f : \mathbb{R}^d \to \mathbb{R}$ *be differentiable with a global minimum* x^**. Assume that* f *has* ℓ*-Lipschitz gradient respect to* x *and* y *separately. Suppose stochastic gradients* $\nabla_x f_i(x,y)$ *and* $\nabla_y f_i(x,y)$ *are unbiased stochastic estimators of* $\nabla_x f(x,y)$ *and* $\nabla_y f(x,y)$ *and have variance bounded by* $\sigma^2 > 0$*. Let* $f(x,y)$ *satisfy the two-sided PL condition with* μ_1, μ_2*. Define* $P_k := a_k + \frac{1}{\lambda} b_k = a_k + \frac{1}{10} b_k$*, where* $a_k = \mathbb{E}[g(x_k) - g^*]$*,* $b_k = \mathbb{E}[g(x_k) - f(x_k, y_k)]$*,* $g(x) = max_y f(x,y)$*, and* $g^* = min_x g(x)$*. Choose step-size as* $\eta_2 \leq \frac{1}{\ell}$ *and* $\eta_1 \leq \frac{\mu_2^2 \eta_2}{18\ell^2}$ *Then the iterates of stochastic-AGDA satisfy:*

$$P_k \leq (1 - \frac{1}{2}\mu_1\eta_1)^t P_0 + \delta$$

where $\delta = \frac{(1-\mu_2\eta_2)(L+\ell)\eta_1^2 + \ell\eta_2^2 + 10L\eta_\ell^2}{10\mu_1\eta_1}\sigma^2, L = \ell + \ell^2/\mu_2$

The parameter L here means that function g is L-smooth. $\mu = min(\mu_1, \mu_2)$

Proof 3.1.3 *Proved by Yang et al, 2020 [44].*

Theorem 4 indicates that SAGDA with constant step-size converges linearly to a neighbourhood of the exact solution. For deterministic AGDA, we can derive a linear convergence to the exact solution by using the fact that $\sigma^2 = 0$ in the deterministic setting. Theorem for AGDA is shown as below:

Theorem 5 *(AGDA with constant step-size)*

Let function $f : \mathbb{R}^d \to \mathbb{R}$ *be differentiable with a global minimum* x^**. Assume that* f *has* ℓ*-Lipschitz gradient respect to* x *and* y *separately. Let* $f(x,y)$ *satisfy the two-sided PL condition with* μ_1, μ_2*. Define* $P_k := a_k + \frac{1}{\lambda} b_k = a_k + \frac{1}{10} b_k$*, where* $a_k = \mathbb{E}[g(x_k) - g^*]$*,* $b_k = \mathbb{E}[g(x_k) - f(x_k, y_k)]$*,* $g(x) = max_y f(x,y)$*, and* $g^* = min_x g(x)$*. Choose step-size as* $\eta_1 = \frac{\mu_2^2}{18\ell^3}$ *and* $\eta_2 = \frac{1}{\ell}$*. Then the iterates of AGDA satisfy:*

$$P_k \leq (1 - \frac{\mu_1\mu_2^2}{36\ell^3})^t P_0$$

Furthermore, $\{(x_k, y_k)\}_k$ *converges to some saddle point* (x^*, y^*)*, and*

$$||x_k - x^*||^2 + ||y_k - y^*||^2 \leq \alpha(1 - \frac{\mu_1 \mu_2^2}{36\ell^3})^t P_0, \tag{3.5}$$

where α *is a constant depending on* μ_1, μ_2*, and* ℓ

In previous sections, we show that both SGDA, SAGDA, and their deterministic version converge for SC-SC problems. However, when moving to bilinear games, a simple example of a convex-concave problem would cause SGDA and GDA not to converge and even diverge, and causes AGDA to stay bounded. The theorems below describes more details about this situation.

Below theorem indicates that the iterates of GDA on a bilinear game diverge linearly for any positive constant step-size η.

Theorem 6 *(GDA on bilinear)*
For any $\eta > 0$*, the iterates of GDA on a bilinear game of format 3.23 diverges as*

$$\delta_k \in \Omega(\delta_0(1 + \eta^2 \sigma_{max}^2(B))^k, \tag{3.6}$$

where $\delta_k = ||x_k - x^*||^2 + ||y_k - y^*||^2$

Proof 3.1.4 *Proved by Gidel et al., 2019 [28].*

For AGDA on bilinear games, the theorem below indicates that for some step-size η, AGDA stays bounded. However, staying bounded is not as good as converging, even with a very slow rate. The reason we cover this theory is to use this as a base line to compare AGDA with negative momentum, which has a linear convergence to the exact solution.

Theorem 7 *(AGDA on bilinear with constant step-size)*

For any $0 < \eta < \frac{2}{\sigma_{max}(B)}$ *, the iterates of AGDA on a bilinear game of format 3.23 stay bounded as*

$$\delta_k \in O(\delta_0), \tag{3.7}$$

where $\delta_k = ||x_k - x^*||^2 + ||y_k - y^*||^2$

Proof 3.1.5 *Proved by Zhang et al., 2022 [25].*

3.1.2 HGD and SHGD

Hamiltonian gradient descent (HGD) method and its stochastic version (SHGD) two very interesting gradient methods, which can find the best update direction in some kinds of problems. In section 2.5, we have shown the convergence trajectories of HGD and SHGD on some simple bilinear games. In this part, we will give convergence guarantees for HGD and SHGD.

If the Hamiltonian function of a problem is quasi-strongly convex and expected smooth, then using some constant step-size can make SHGD converges linearly to a neighborhood of the exact solution. The neighborhood is related to the gradient variance of the stochastic Hamiltonian function with $\sigma^2 = \mathbb{E}_{i,j}[||\nabla H_{i,j}(x^*)||^2]$

Theorem 8 *(SHGD with constant step-size)*

For a stochastic game with μ_H*-quasi-strongly convex and* $\mathcal{L}_H$*-expected smooth Hamiltonian function, run SHGD with constant step-size* $\eta_k = \eta \in (0, \frac{1}{2\mathcal{L}_H}]$*. Then, the iterations, given by Algorithm 6, satisfy:*

$$\mathbb{E}[||x_k - x^*||^2] \leq (1 - \eta\mu_H)^k ||x_0 - x^*||^2 + \frac{2\eta\sigma^2}{\mu_H}, \tag{3.8}$$

where $\sigma^2 = \mathbb{E}_{i,j}[||\nabla H_{i,j}(x^*)||^2]$.

Proof 3.1.6 *Proved by Loizou, et al., 2021 [26].*

In this theorem, expected smoothness parameters $\mathcal{L}$ is required to use. Gower et al. show that the expected smoothness constants can take several values according to the well-defined distributions $\mathcal{D}$ selected for sampling in stochastic algorithms.[42, 43] Take τ -minibatch sampling as an example, the expected smoothness parameters can be taken of the following values:

$$\mathcal{L}(\tau) = \frac{n^2(\tau-1)}{\tau(n^2-1)}L_H + \frac{n^2-\tau}{\tau(n^2-1)}L_{max}, \tag{3.9}$$

where $L_{max} = max\{L_{H_{i,j}}\}_{i,j=1}^{n}$, and $L_{H_{i,j}}$ is the smoothness parameter for function $H_{i,j}$.

Then, it can be seen that for uniform single-element sampling, we have $\tau = 1$ so that $\mathcal{L} = L_{max}$. The deterministic case can be regarded as using full-batch sampling, which is $\tau = n^2$. Then, for deterministic HGD, the expected smoothness parameter $\mathcal{L} = L_H$, where L_H is the smoothness parameter for the Hamiltonian function.

From theorem 8, we can reach deterministic HGD as a special case by using full-batch sampling $\tau = n^2$. Note that for the deterministic case, $\sigma = 0$ and $\mathcal{L} = L_H$. Then, we can find that HGD with constant step-size converges linearly to the exact solution. The theorem for HGD is shown as below:

Theorem 9 *(HGD with constant step-size)*
For a game with μ_H-quasi-strongly convex and L_H-smooth Hamiltonian function, run HGD with constant step-size $\eta_k = \eta \in (0, \frac{1}{2L_H}]$. Then, the iterations satisfy:

$$\mathbb{E}[||x_k - x^*||^2] \leq (1-\eta\mu_H)^k||x_0 - x^*||^2. \tag{3.10}$$

Similar to SGDA, and actually similar to all stochastic gradient algorithms, once with constant step-size, SHGD has a linear convergence but only reaches a neighborhood of the solution. That kind of step should be used when precision is not

important or computing time is limited. When the requirement of precision surpasses the restriction of time, using decreasing step-size is a beneficial way to figure out the problem. In the below theorem, a switching step-size rule is shown and points out when to switch a constant step-size to a decreasing one. The reason to use the constant step-size in the initial iterations is to maintain the fast linear convergence in the first part. The theorem shows that after switching step-size, a sublinear convergence to the exact solution can be witnessed.

Theorem 10 *(SHGD with decreasing step-size)*
For a stochastic game with μ_H-quasi-strongly convex and $\mathcal{L}_H$-expected smooth Hamiltonian function, run SHGD with decreasing step-size. Let $\kappa = \frac{\mathcal{L}_H}{\mu_H}$. Choose step-size as:

$$\eta_k = \begin{cases} \frac{1}{2\mathcal{L}_H} & fork \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu_H} & fork > 4\lceil\kappa\rceil. \end{cases} \tag{3.11}$$

If $k \geq 4\lceil\kappa\rceil$, then SHGD given in Algorithm 6 satisfy:

$$\mathbb{E}[||x_k - x^*||^2] \leq \frac{\sigma^2}{\mu_H^2}\frac{8}{k} + \frac{16\lceil\kappa\rceil^2}{e^2k^2}||x_) - x^*||^2 = O(\frac{1}{k}), \tag{3.12}$$

Proof 3.1.7 *Proved by Loizou, et al., 2021 [26].*

Later, we illustrate propositions 1 and 2 to show that both bilinear games and strongly-monotone quadratic games have quasi-strongly convex, smooth, and quadratic Hamiltonian functions. In other words, they satisfy the assumptions of theorem 9 to 11. Then those theorems can be directly used in bilinear games or strongly-monotone quadratic games.

3.1.3 SEG

Here, we simply show two theorems for SEG that will be followed to choose step-sizes in an experiment in the later "Random Reshuffling" part. One theorem is

SEG with constant step-size and it can guarantee linear convergence to a neighborhood of the exact solution. The other is SEG with decresing step-size, which makes the algorithm converge sublinearly to the exact solution. It needs to emphasize that using different step-size for extrapolation and update step is very important in the stochastic setting. Usually, the update step has a smaller step-size than the extrapolation step.

Theorem 11 *(SEG with constant step-size)*
We assume that for all sampling ξ there exists $L_\xi > 0$ such that operator $F_\xi(x)$ is L_ξ-Lipschitz, i.e., for all $x \in \mathbb{R}^d$

$$||F_\xi(x) - F_\xi(y)|| \leq L_\xi ||x - y||. \tag{3.13}$$

We assume that for all sampling ξ, operator $F_\xi(x)$ is (μ_ξ, x^*)-strongly monotone, i.e., there exists (possibly negative) $\mu_\xi \in \mathbb{R}$ such that for all $x \in \mathbb{R}^d$

$$< F_\xi(x) - F_\xi(x^*), x - x^* > \geq \mu_\xi ||x - x^*||^2 \tag{3.14}$$

Let $\eta_{2,\xi_k} \leq \alpha \eta_{1,\xi_k}$, $0 < \alpha \leq 1/4$, and $\eta_{1,\xi_k} \leq \frac{1}{4|\mu_{\xi_k}| + \sqrt{2} L_{\xi_k}}$. Let $\rho = \frac{\alpha}{2} \mathbb{E}_{\xi_k}[\eta_{1,\xi_k} \mu_{\xi_k} (\mathbb{1}_{\mu_{\xi_k} \geq 0} + 4 \times \mathbb{1}_{\mu_{\xi_k} < 0}]$ is positive. Then the iterates of S-SEG given by Algorithm 4 satisfy:

$$\mathbb{E}[||x_K - x^*||^2] \leq (1 - \rho)^K ||x_0 - x^*||^2 + \frac{3\alpha(4\alpha + 1)\sigma_{AS^2}}{2\rho}, \tag{3.15}$$

where $\sigma_{AS}^2 = \mathbb{E}_\xi[\eta_{1,\xi}^2 ||F_\xi(x^*)||^2]$.

Proof 3.1.8 *Proved by Gorbunov et al., 2022 [36].*

Theorem 12 *(SEG with decreasing step-size)*
We assume that for all sampling ξ there exists $L_\xi > 0$ such that operator $F_\xi(x)$ is L_ξ-Lipschitz, i.e., for all $x \in \mathbb{R}^d$

$$||F_\xi(x) - F_\xi(y)|| \leq L_\xi ||x - y||. \tag{3.16}$$

We assume that for all sampling ξ operator $F_\xi(x)$ is (μ_ξ, x^*)-strongly monotone, i.e., there exists (possibly negative) $\mu_\xi \in \mathbb{R}$ such that for all $x \in \mathbb{R}^d$

$$< F_\xi(x) - F_\xi(x^*), x - x^* >\geq \mu_\xi ||x - x^*||^2 \tag{3.17}$$

Let $\eta_{2,\xi_k} = \frac{\eta_{1,\xi_k}}{4}$, and $\eta_{1,\xi_k} = \beta_k \eta_{\xi_k}$, where $\eta_{\xi_k} = \frac{1}{4|\mu_{\xi_k}| + \sqrt{2}L_{\xi_k}}$, and $\tilde{\rho} = \frac{1}{8}\mathbb{E}_{\xi_k}[\eta_{\xi_k}\mu_{\xi_k}(\mathbb{1}_{\mu_{\xi_k}\geq 0} + 4 \times \mathbb{1}_{\mu_{\xi_k}<0}$. Assume $\tilde{\rho} > 0$. Then, for the total number of iterations $K \geq 0$ and $\{\beta_k\}_{k\geq 0}$ such that

$$\begin{cases} if\ K \leq \frac{1}{\tilde{\rho}} & \beta_k = 1, \\ if\ K > \frac{1}{\tilde{\rho}}\ and\ k \geq k_0, & \beta_k = 1, \\ if\ K > \frac{1}{\tilde{\rho}}\ and\ k \geq k_0, & \beta_k = \frac{2}{2 + \tilde{\rho}(k - k_0)} \end{cases} \tag{3.18}$$

where $k_0 = \lceil K/2 \rceil$, we have that the iterates of S-SEG satisfy

$$\mathbb{E}[||x_K\text{-}x^*||^2] \leq \frac{32||x_0 - x^*||^2}{\tilde{\rho}} exp(-\frac{\tilde{\rho}K}{2}) + \frac{27\sigma_{AS}^2}{\tilde{\rho}^2 K} \tag{3.19}$$

Proof 3.1.9 *Proved by Gorbunov et al., 2022 [36].*

3.2 Accelerated Methods

3.2.1 Heavy-ball moment

The heavy-ball method was proposed by Polyak in 1964. It has a two-step procedure defined by the following state transitions:

$$p_k = -\nabla f(x_k) + \beta_k p_{k-1} \tag{3.20 a}$$

$$x_{k+1} = x_k + \alpha_k p_k \tag{3.20 b}$$

with some initial points x_0 and p_0, and some positive sequences α_k and β_k. Let $p_0 = 0$, we can rewrite the algorithm as the iteration:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}) \tag{3.20}$$

where $x_{-1} = x_0$. The term $(x_k - x_{k-1})$ is regarded as momentum.

In practice, heavy-ball method requires a lot of tuning, and we only have a guaranteed acceleration theorem for it on quadratic problems. Despite this, momentum, especially heavy-ball momentum is used on top of almost all popular optimization algorithms in machine learning (even in stochastic cases).

Now, we will describe the convergence theorem for heavy-ball method on Quadratic minimization problems as:

$$minf(x) = \frac{1}{2}x^T Qx - b^T + c \tag{3.21}$$

where Q is a positive definite matrix with the assumption that $\mu I \preceq Q \preceq LI$. This problem has a unique solution given by $x^* = Q^{-1}b$. For the heavy-ball method, instead of looking at $||x_{k+1} - x^*||^2$ we examine the value of Lyapunov function $||x_{k+1} - x^*||^2 + ||x_k - x^*||^2$. Theorem 13 shows that heavy-ball method converge at a linear rate.

Theorem 13 *(Heavy-ball Momentum for Quadratic function)*
Let function $f(x) = \frac{1}{2}x^T Qx - b^T + c$ *has positive definite hessian* Q *with* $0 < \mu = \lambda_{min}(Q)$ *and* $L = \lambda_{max}(Q)$. *Choose step-size* $\alpha = \frac{4}{(\sqrt{L}+\sqrt{\mu})^2}$ *and momentum parameter* $\beta = \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}$. *Let* $x_{-1} = x_0$. *Then the iterates of the heavy-ball method given by (3.20) satisfy:*

$$[||x_{k+1} - x^*||^2 + ||x_k - x^*||^2] \leq (\frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}} +)^k C[||x_1 - x^*||^2 + ||x_0 - x^*||^2] \tag{3.22}$$

where is a small positive number and C *is a constant.*

Proof 3.2.1 *Proved by Wright and Recht B, 2022 [45].*

3.2.1.1 H(x) of bilinear games and quadratic games

Now, let's show that Hamiltonian function $H(x)$ of bilinear games and quadratic games are L_H-smooth and μ_H-quasi-strongly convex quadratic functions in proposition 1 and 2. Stochastic bilinear games are of the below format:

$$\min_{x\in\mathbb{R}^{d_1}} \max_{x\in\mathbb{R}^{d_2}} \frac{1}{n}\sum_{i=1}^{n} x^T B_i y + b_i^T x + c_i^T y. \tag{3.23}$$

And stochastic quadratic games are of the format:

$$\min_{x\in\mathbb{R}^{d_1}} \max_{x\in\mathbb{R}^{d_2}} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}x^T A_i x + x^T B_i y - \frac{1}{2}y^T C_i y + a_i^T x + b_i^T y. \tag{3.24}$$

Proposition 1 *For stochastic bilinear games of the form (3.23), the stochastic Hamiltonian function $H(x)$(2.12) is an L_H-smooth and μ_H-quasi-strongly convex quadratic function with constants $L_H = \sigma^2_{max}(B)$ and $\mu_H = \sigma^2_{min}(B)$, where $B = \frac{1}{n}\sum_{i=1}^{n} B_i$ and σ_{max} and σ_{min} are the maximum and minimum non-zero singular values of B.*

Proof 3.2.2 *Proved by Loizou, et al, 2020 [41].*

Proposition 2 *For quadratic games of the form (3.24) with A_i and C_i symmetric with at least on solution x^*, the Hamiltonian function $H(x)$(2.12) is an L_H-smooth and μ_H-quasi-strongly convex quadratic function with constants $L_H = \sigma^2_{max}(J)$ and $\mu_H = \sigma^2_{min}(J)$, where $J = \nabla F$ is that the Jacobian matrix of the game and σ_{max} and σ_{min} are the maximum and minimum non-zero singular values of J. Here, $A = \frac{1}{n}\sum_{i=1}^{n} A_i$ and $C = \frac{1}{n}\sum_{i=1}^{n} C_i$.*

Proof 3.2.3 *Proved by Loizou, et al., 2021 [26].*

Then, we can use the parameter of Heavy-ball momentum to HGD on billinear or strongly-monotone quadratic games.

3.2.1.2 Negative momentum parameter for AGDA.

There is something special when using Heavy-ball momentum of simultaneous/ alternating gradient descent ascent on bilinear games. It is known that both simultaneous and alternating gradient descent ascent doesn't converge for bilinear games. Also, Heavy-ball momentum with any parameters doesn't help simultaneous gradient descent ascent (GDA) to converge. However, it is interesting that some well-chosen negative momentum term can make alternating gradient descent ascent (AGDA) converge to the exact solution with a linear rate. In this part, we focus on bilinear smooth games of the format (3.23).

The theorem below is a choice of step-size and negative momentum parameters that can guarantee a linear convergence in bilinear games.

Theorem 14 *(AGDA with negative momentum)*
For stochastic bilinear games of the form (3.23), let $\sqrt{\eta_1\eta_2} = \eta \leq \frac{1}{\sigma_{max}(B)}$, $\beta_1 = -\frac{1}{2}$ *and* $\beta_2 = 0$. *Then the iterates of AGDA satisfy:*

$$\Delta_{k+1} \in O(max\{\frac{1}{2}, 1 - \frac{\eta\sigma_{min}(A)}{4}\}^k \Delta_0) \tag{3.25}$$

where $\Delta_k = ||x_k - x^*||_2^2 + ||y_k, -y^*||_2^2$, $B = \frac{1}{n}\sum_{i=1}^n B_i$ *and* σ_{max} *and* σ_{min} *are the maximum and minimum non-zero singular values of* B.

Proof 3.2.4 *Proved by Gidel et al., 2019 [28].*

3.2.2 Nesterov's Method

Nesterov's optimal method (also known as Nesterov's accelerated gradient method (AGD)) is defined by the formula:

$$x_{k+1} = x_k - \alpha\nabla f(x_k + \beta(x_k - x_{k-1})) + \beta(x_k - x_{k-1}) \tag{3.26}$$

The only difference from (3.20) is that the gradient ∇f is evaluated at a point $x_k+\beta(x_k-x_{k-1})$ after momentum rather than at the original point x_k. By introducing an intermediate sequence $\{y_k\}$ and allowing α and β to have possibly different values at each iteration, this method can be rewritten as follows:

$$y_k = x_k + \beta_k(x_k - x_{k-1}) \tag{3.26 a}$$

$$x_{k+1} = y_k - \alpha_k \nabla f(y_k) \tag{3.26 b}$$

where $x_{-1} = x_0 = y_0 \in \mathbb{R}^d$.

Nesterov's Optimal Method can accelerate for strongly convex and smooth games, and can also accelerate for convex and smooth games. In this thesis, we only use this method on Hamiltonian functions that are strongly convex and smooth. So, below we only mention the theorem for the first case.

The theorem below shows that Accelerated Gradient Descent (AGD) converges linearly to the exact solution for L-smooth and μ-quasi-strongly convex games.

Theorem 15 *(AGD)*
Let function $f : \mathbb{R}^d \to \mathbb{R}$ be differentiable with a global minimum x^*. Assume that f is L-smooth and μ-quasi-strongly convex. Choose step-size $\alpha = \frac{1}{L}$ and momentum parameter $\beta = \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}$. Let $x_{-1} = x_0$. Let $x_{-1} = x_0 = y_0 \in \mathbb{R}^d$. Then the iterates of AGD given by (3.26) satisfies:

$$f(x_k) - f(x^*) \leq (1 - \sqrt{\frac{\mu}{L}})^k [f(x_0) - f(x^*) + \frac{\mu}{2}||x_0 - x^*||^2]. \tag{3.27}$$

Proof 3.2.5 *Proved by Wright and Recht B, 2022 [45].*

Nesterov's Method can be used directly on HGD in bilinear games and quadratic games. As shown in proposition 1 and 2, the Hamiltonian function $H(x)$ is an L_H-smooth and μ_H-quasi-strongly convex quadratic function for bilinear games and quadratic games. Then, AGD method can be used on Hamiltonian function with step-size $\alpha = \frac{1}{L_H}$ and momentum parameter $\beta = \frac{\sqrt{L_H}-\sqrt{\mu_H}}{\sqrt{L_H}+\sqrt{\mu_H}}$.

3.3 Variance reduction

Variance reduction is a method used in stochastic gradient methods that reduce the neighborhood when using constant step-size. It is one of the most remarkable algorithmic breakthroughs in recent years for solving minimization problems of finite sum format. This method is significantly faster than SGD in practice and theory.

In this section, we will talk about the Loopless Stochastic Variance Reduced Gradient Descent method (L-SVRG) and the Loopless Stochastic Variance Reduced Gradient Descent Ascent method (L-SVRGDA). L-SVRG is used for minimization problems. Thus, it can be used on the Hamiltonian function to solve the min-max problems of bilinear games and quadratic games, and this method is named L-SVRHG. The pseudocode for L-SVRH is similar to Algorithm 7, the only difference is to regard the Hamiltonian function $H(z)$ as the objective function. The pseudocode of L-SVRGDA is shown as Algorithm 8.

Variance reduction has revolutionized stochastic methods in optimization. This technique applies to finite sum minimization problem of the form $min_z = \frac{1}{n}\sum_{i=1}^{n} f_i(z)$. Instead of using a random sample $g_k = \nabla f_i(z_k)$ as SGD does, variance reduction methods use

$$g_{z_k} = \nabla f_i(z_k) - \nabla f_i(w_k) + \nabla f(w_k). \tag{3.28}$$

A good choice of w_k decreases the "variance" $\mathbb{E}[||g_k - \nabla f(z_k)||^2]$ compared to $\mathbb{E}[||\nabla f_i(z_k) - \nabla f(z_k)||^2]$ that SGD has.

In that way, stochastic variance reduction gradient descent has an inner loop to update z_k and an outer loop to update w_k after some fixed number of updates of z_k. In order to simplify the inner and outer loops to one loop, the method of Loopless Stochastic Variance Reduced Gradient Descent (L-SVRG) has been proposed. Its pseudocode is shown as Algorithm 7.

Since SHGD is doing minimization on the Hamiltonian function, L-SVRG method

Algorithm 7 Loopless Stochastic Variance Reduced Gradient Descent (L-SVRG)

Input: Step-size $\eta > 0$, probability $p \in (0, 1]$. Choose initial points $w_0 = x_0 \in \mathbb{R}^d$. Distribution $\mathcal{D}$ of samples.

1: **for** k = 0, 1, 2, ..., K **do**

2: Generate fresh samples $i \sim \mathcal{D}$, and evaluate $g_k = \nabla f_i(w_k) - \nabla f_i(x_k) + \nabla f(x_k)$

3: Set $x_{k+1} = x_k - \eta g_k$

4: Set $w_{k+1} = \begin{cases} x_k & \text{with probability } p \\ w_k & \text{with probability } 1-p \end{cases}$

5: **end for**

Output: x_K

can be directly used on to Hamiltonian function to get the algorithm Loopless Stochastic Variance Reduced Hamiltonian Gradient Descent (L-SVRHG). The pseudocode for L-SVRH is similar to Algorithm 7, the only difference is to regard the Hamiltonian function $H(z)$ as the objective function.

Moving from minimization to min-max problems, a similar variance reduction method can be used on stochastic gradient methods. Take stochastic variance reduction gradient descent ascent as an example. Instead of using a random sample $g_k = (g_{x_k}, g_{y_k}) = (\nabla_x f_i(x_k), \nabla_y f_i(y_k))$ as SGDA does, variance reduction methods use

$$\begin{aligned} g_{x_k} &= \nabla_x f_i(x_k, y_k) - \nabla_x f_i(w_{x_k}, w_{y_k}) + \nabla_x f(w_{x_k}, w_{y_k}), \\ g_{y_k} &= \nabla_y f_i(x_k, y_k) - \nabla_y f_i(w_{x_k}, w_{y_k}) + \nabla_y f(w_{x_k}, w_{y_k}). \end{aligned} \tag{3.29}$$

Again, those stochastic variance reduction methods all have an inner loop and an outer loop. In order to simplify the inner and outer loops to one loop, the method of Loopless Stochastic Variance Reduced Gradient Descent Ascent (L-SVRGDA) has been proposed. Its pseudocode is shown as Algorithm 8.

The below theorem shows that L-SVRG with constant step-size converges linearly to the exact solution for smooth and strongly convex problems.

Theorem 16 *(L-SVRG)*

Assume f(x) is a L-smooth, μ-strongly convex function. Let step-size $\eta \leq \frac{1}{6L}$ and

Algorithm 8 Loopless Stochastic Variance Reduced Gradient Descent Ascent (L-SVRGDA)

Input: Step-size $\eta > 0$, probability $p \in (0, 1]$. Choose initial points Choose initial points $w_{x_0} = x_0 \in \mathbb{R}^d$, $w_{y_0} = y_0 \in \mathbb{R}^d$. Distribution $\mathcal{D}$ of samples.

1: **for** k = 0, 1, 2, ..., K **do**

2: Generate fresh samples $i \sim \mathcal{D}$, and let $g_{x_k} = \nabla_x f_i(x_k, y_k) - \nabla_x f_i(w_{x_k}, w_{y_k}) + \nabla_x f(w_{x_k}, w_{y_k})$, and $g_{y_k} = \nabla_y f_i(x_k, y_k) - \nabla_y f_i(w_{x_k}, w_{y_k}) + \nabla_y f(w_{x_k}, w_{y_k})$

3: Set $x_{k+1} = x_k - \eta g_{x_k}$, and $y_{k+1} = y_k - \eta g_{y_k}$

4: Set $(w_{x_{k+1}}, w_{y_{k+1}}) = \begin{cases} (x_k, y_k) & \text{with probability } p \\ (w_{x_k}, w_{y_k}) & \text{with probability } 1 - p \end{cases}$

5: **end for**

Output: (x_K, y_K)

probability $p = \frac{1}{n}$. *Then iterates of L-SVRG, given in Algorithm 7, satisfy:*

$$\mathbb{E}[\Phi_k] \leq max\{1 - \frac{\mu}{6L}, 1 - \frac{1}{2n}\}^k \Phi_0, \tag{3.30}$$

where $\Phi_k := ||x_k - x^*||^2 + D_k$, *and* $D_k := \frac{4\eta^2}{pn} \sum_{i=1}^n ||_i(w_k) - \nabla f_i(x^*)||^2$.

Proof 3.3.1 *Proved by Kovalev et al., 2020 [46].*

In proposition 1 and 2, we have shown that the Hamiltonian function of bilinear games and strongly-monotone quadratic games are smooth, quasi-strongly convex, and quadratic functions. So, we can use L-SVRG on the Hamiltonian function with constant step-size to get a linear convergence to the exact solution. The algorithm is named L-SVRHG.

Now, we will illustrate a theorem of Loopless Stochastic Variance Reduced Gradient Descent Ascent method (L-SVRGDA) on quasi-strongly monotone functions for min-max problem. The theorem denotes that L-SVRGDA can converge linearly to the exact solution with constant step-sizes.

Theorem 17 *(L-SVRGDA in the quasi-strongly monotone case)*
Let operator F *be* μ*-quasi-strongly monotone. Assume that operators* F_i *are averaged star-cocoercivity with parameter* $\hat{\ell}$*, and assume that there is a unique solution for the*

problem. Let step-size satisfy $0 < \eta \leq \frac{1}{6\hat{\ell}}$. *Then for all* $k \geq 0$ *the iterates produced by L-SVRGDA, given by Algorithm 8, satisfy:*

$$\mathbb{E}[||x_k - x^*||^2] \leq (1 - min\{\eta\mu, \frac{p}{2}\})^k V_0, \tag{3.31}$$

where $V_0 = ||x_k - x^*||^2 + \frac{4\eta^2\sigma_0^2}{p}$, *and* $\sigma_0^2 := \frac{1}{n}\sum_{i=1}^n ||F_i(x_k) - F_i(x^*)||^2$.

Proof 3.3.2 *Proved by Beznosikov et al., 2022 [47].*

3.4 Random Reshuffling

For stochastic gradient methods, we can get the unbiased estimator of the gradient by uniformly sampling for index i or by following some distribution. In practice, there is another way named random reshuffling (RR) which is normally used and usually has good results. RR is an algorithm for finite-sum functions that utilizes iterative gradient descent steps in conjunction with data reshuffling. It uniformly samples a random permutation of $[n]$ at the start of every epoch, and processes the data points within that epoch according to the order specified by the permutation. The pseudocode for using RR on any stochastic gradient methods is shown as Algorithm 9.

Usually, RR has better performance and quicker convergence than using random sampling. However, the empirical benefits of RR are hard to get theoretical proof. Here, we will just demonstrate two recent related theorems for RR with constant step-size. While in a later chapter, we will use RR in my experiments with some decreasing step-sizes. Details for decreasing step-sizes and experiments can be found in the next chapter.

THe first theorem is using random reshuffling on gradient descent (GD-RR) for minimizing an objective function $f = \sum_{i=1}^n f_i$ of finite sum format. By using constant

Algorithm 9 Stochastic Gradient Method with Random Reshuffling (RR)

Input: Number of epochs K, Step-size $\eta > 0$, and initial points z_0. Distribution $\mathcal{D}$ of samples. Select a stochastic gradient method.

Initialize $z_0^1 = z_0$
for k = 0, 1, 2, ..., K **do**
  RR: Sample permutation $\tau_k \sim$Uniform($\mathbb{S}_n$)
  for i = 1, 2, ..., n **do**
    Use sample $\tau_k(i)$, and evaluate $g_{\tau_k(i)}(z_{i-1}^k)$ as the update direction for the selected gradient method.
    Set $z_i^k = z_{i-1}^k - \eta g_{\tau_k(i)}(z_{i-1}^k)$
  end for
  Set $z_0^{k+1} = z_n^k$
end for

Output: z_n^K

step-size, the iterates converge linearly to a neighborhood of the exact solution.

Theorem 18 *(GD-RR with constant step-size for minimization)*

Suppose that the functions $f_1, ..., f_n$ are μ-strongly convex and both objective function $f = \sum_{i=1}^n f_i$ and functions $f_1, ..., f_n$ are all L-smooth. Assume that f is lower bounded by some $f_* \in R$. Then for GD-RR with a constant step-size $\eta \leq \frac{1}{L}$, the iterates satisfy:

$$\mathbb{E}[||x_k - x^*||^2] \leq (1 - \eta\mu)^k ||x_0 - x^*||^2 + \frac{2\eta\sigma_{Shuffle}^2}{\mu}, \tag{3.32}$$

where $\sigma_{Shuffle}^2 := max_{i=i,...n-1}[\frac{1}{\eta}E[D_{f_{\pi_i}}(x_*^i, x_*)]$ for a random permutation π of $\{1, 2, ..., n\}$, and $x_*^i := x_* - \eta - \sum_{j=0}^{i-1} \nabla f_{\pi_j}(x_*)$ for $i = 1, ..., n-1$. Here for any i, the quantity $D_{f_i}(x, y) := f_i(x) - f_i(y) - < \nabla f_i(x), x - y >$ is the Bregman divergence between x and y associated with f_i.

Proof 3.4.1 *Proved by Mishchenko et al., 2020 [48].*

Then, when the Hamiltonian function satisfies the assumption of the above theorem, we can use GD-RR on the Hamiltonian function, named HGD-RR, and obtain a linear convergence to a neighborhood.

The second theorem is for using random reshuffling on gradient descent ascend (GDA-RR) on min-max problems. Below is a convergence theorem of GDA-RR for each epoch.

Theorem 19 *(GDA-RR with constant step-size)*
Assume objective function f has μ-strongly monotone operator $F(z) = \frac{1}{n}\sum_{i=1}^{n} F_i(z)$, where each F_i is ι-Lipschitz, but not necessarily monotone. Let z^* denotes the unique root of F. Select the step-size as $\alpha \leq min\{\mu/5n\iota^2, 2log(||F(z_0|n^{1/2}K/\mu)/\}$. Here, $K > 0$ is the number of epochs that need to run. Then, after the last epoch by running GDA-RR, the result satisfy:

$$\mathbb{E}[||Z_n^K - Z^8||^2] \leq 2e^{-K/5\kappa^2}||z_0 - z^*||^2 + \frac{2\mu^2 + 8\kappa^2\sigma^2 log^3(||F(z_0)||n^{1/2}K/\mu)}{\mu^2 nK^2}, \quad (3.33)$$

where z_n^K is the points in n^{th} iteration of K^{th} epoch, $\kappa = \iota/mu$ is the condition number and $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}||F_i(z^*)||^2$ is the gradient variance at the optimum.

Proof 3.4.2 *Proved by Das et al., 2022 [49].*

Chapter 4

Experimental Evaluation

4.1 Evaluation of Accelerate Methods

In this chapter, we will show the results of HGD and GDA or AGDA with or without accelerate methods on bilinear games and strongly-monotone quadratic games (which is an example of SC-SC). Chosen method and parameters are described separately for each experiment.

In before chapter, we have shown that GDA or GDA with momentum doesn't converge for bilinear games, while AGDA with negative momentum converges for bilinear games. So, in this part, we will give results and some comparison of numerical experiments using HGD, HGD with momentum, AGDA, and AGDA with negative momentum for numerical examples of bilinear games of the below format:

$$\min_{x \in R_1^d} \max_{x \in R_2^d} \frac{1}{n} \sum_{i=1}^{n} x^T B_i y + b_i^T x + c_i^T y, \tag{4.1}$$

where $[b_i]_k, [c_i]_k \sim N(0, 1/n)$, $\mu_B^2 I \preceq B_i^T B_i \preceq L_B^2 I$, and I is the identity matrix. The sampling method of B_i is the same as (4.2), which will be described below.

We will also cover HGD, HGD with momentum, GDA, GDA with momentum, AGDA, and AGDA with momentum on numerical experiments of strongly-monotone quadratic games of the format:

$$\min_{x\in R_1^d}\max_{x\in R_2^d}\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}x^TA_ix+x^TB_iy-\frac{1}{2}y^TC_iy+a_i^Tx+b_i^Ty, \tag{4.2}$$

where $[a_i]_k, [b_i]_k \sim N(0, 1/n)$. To make the game be strongly-monotone and co-coercive, matrices are sampled such that $\mu_A \preceq A_i \preceq L_A I$, $\mu_B^2 I \preceq B_i^T B_i \preceq L_B^2 I$, and $\mu_C I \preceq C_i \preceq L_C I$, where I is the identity matrix.

The sampling method of matrices A, B and C comes from Loizou et al.[26] The sampling of the matrices A_i (resp. C_i) is based on the generation of a random orthogonal matrix Q_i (resp. Q_i'), and subsequently sampling a random diagonal matrix D_i (resp. D_i') where the elements on the diagonal are sampled uniformly in $[\mu_A, L_A]$ (resp. $[\mu_C, L_C]$), such that at least one of the matrices has a minimum eigenvalue equal to μ_A (resp. μ_C) and one matrix has a maximum eigenvalue equal to L_A (resp. L_C). Finally, the matrices are constructed by computing $A_i = Q_i D_i Q_i^\top$ (resp. $C_i = Q_i' D_i' Q_i'^\top$). This ensures that the matrices A_i and C_i for all $i \in [n]$, are symmetric and positively definite. The sampling of matrices B_i follows a similar fashion with the diagonal matrix D_i to lie between $[\mu_B, L_B]$. Note that matrices B_i are not necessarily symmetric.

4.1.1 HGD

In proposition 1, we have shown that the Hamiltonian function of bilinear games is a smooth and quasi-strongly convex quadratic function. Then, theorem 9 of HGD with constant step size can be used in this experiment. Use constant step-size $\eta_k = \frac{1}{2L_H}$ for HGD, where L_H is the smoothness parameter of the Hamiltonian function and can be calculated according to proposition 1 as $L_H = \sigma_{max}^2(B)$.

Also, theorem 13 and 15 of Heavy-ball momentum and Nesterov's optimal method can be directly used on HGD. We run HGD, HGD with heavy-ball momentum, and HGD with Nesterov's optimal method of 100 iterations on bilinear games of format

(4.1) with $\mu_B = 1, L_B = 10$, and compare their performance in Figure 4-1.

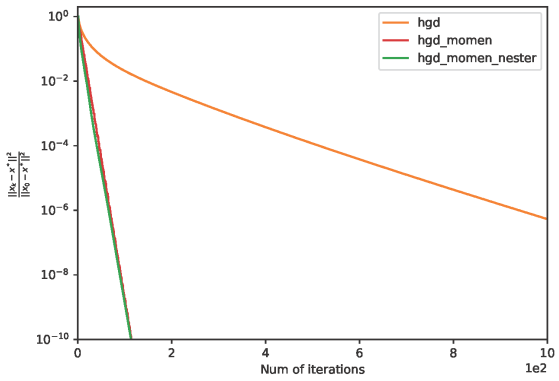

Figure 4-1. HGD on Bilinear with $1 \preceq B_i^T B_i \preceq 10^2$. HGD uses constant steps as theorem 9;[26] HGD with heavy-ball momentum, and HGD with Nesterov's optimal method choose parameters as theorem 13 and 15.[45]

From the figure, it can be seen that all those algorithms converge linearly to the exact solution by using constant step-sizes. When combining HGD with Heavy-ball momentum or Nesterov's optimal method, the convergence becomes faster than purely using HGD. That matches the theorem and indicates that both Heavy-ball momentum and Nesterov's optimal method can accelerate the convergence of HGD on smooth bilinear games.

Now, let's move on to strongly-monotone quadratic games. In proposition 2, we have shown that the Hamiltonian function of a quadratic game is a smooth and quasi-strongly convex quadratic function. Then, theorem 9 of HGD with constant step-size can be used in this experiment. Use constant step-size $\eta_k = \frac{1}{2L_H}$ for HGD, where L_H is the smoothness parameter of the Hamiltonian function. The smoothness parameter L_H can be calculated according to proposition 2 as $L_H = \lambda_{max}(J^T J)$, where J is the Jacobian matrix of the game.

Also, theorem 13 and 15 of Heavy-ball momentum and Nesterov's optimal method can be directly used on HGD. We run HGD, HGD with Heavy-ball momentum, and HGD with Nesterov's optimal method of 100 iterations on strongly-monotone

quadratic games of format (4.2) with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 10$, and compare their performance in Figure 4-2.

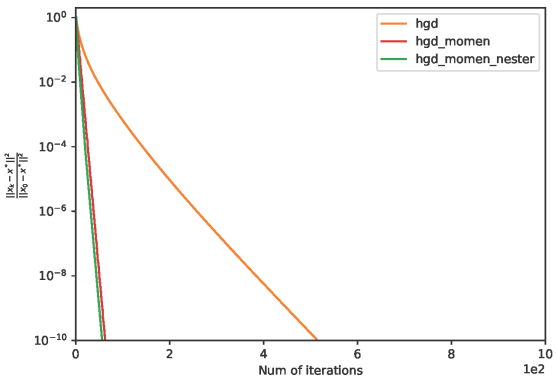

Figure 4-2. HGD on Quadratic with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 1$. HGD uses constant steps as theorem 9;[26] HGD with heavy-ball momentum, and HGD with Nesterov's optimal method choose parameters as theorem 13 and 15.[45]

From the figure, it can be seen that all those algorithms converge linearly to the exact solution by using constant step-sizes. When combining HGD with Heavy-ball momentum or Nesterov's optimal method, the convergence becomes faster than purely using HGD. That matches the theorem and indicates that both Heavy-ball momentum and Nesterov's optimal method can accelerate the convergence of HGD on strongly-monotone quadratic games.

4.1.2 GDA or AGDA

For the bilinear game $\mu_B = 1, L_B = 2$, we run AGDA, and AGDA with negative momentum. The step-sizes and momentum parameters are chosen according to the theorems that we mentioned in before part. For AGDA, use step-size $\eta = \frac{2}{\sigma_{max}(B)}$ according to theorem 7, where $\sigma_{max}(B)$ is the maximum singular value of $B = \frac{1}{n}\sum_{i=1}^{n} B_i$. For AGDA with negative momentum, choose step-size as $\eta_1 = \eta_2 = \frac{1}{\sigma_{max}(B)}$, and choose momentum parameters as $\beta_1 = -\frac{1}{2}$, $\beta_2 = 0$ as Theorem 14 described. Both

methods run for 100 iterations. The results are shown in Figure 4-3.

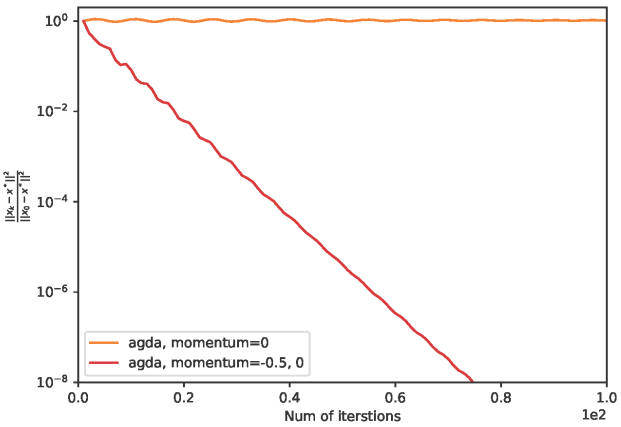

Figure 4-3. AGDA for Bilinear with $1 \preceq B_i^T B_i \preceq 2^2$. AGDA with a constant step as theorem 7;[25] AGDA with negative momentum as theorem 14.[28]

It can be seen that AGDA stays bounded, while AGDA with negative momentum converges linearly to the exact solution. It is very interesting and matches the theorem that adding a well-chosen negative momentum terms on AGDA for bilinear games changes the situation from not converging (only bounded) to converging.

For strongly-monotone quadratic games, there is no theorem guarantee that Heavy-ball momentum will help to improve the convergence rate. So, in this part, we merely try some values of heavy-ball momentum parameters on GDA and AGDA separately to see what will happen. The experiments we use are of format (4.2) with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 10$.

Following theorem 2, choose step-size $\eta = \frac{1}{2\ell}$ for GDA, where ℓ means that operator F is ℓ-co-coercive. To be specific, for Lipschitz operator F, strongly monotone can imply co-coercive operator. Also, a strongly monotone operator is equivalent to SC-SC objective function. Our strongly monotone quadratic game is SC-SC with Lipschitz operator. The values of the co-coercive parameters ℓ is computed using $\frac{1}{\ell} = min_{\lambda \in Sp(J)} R(\frac{1}{\lambda})$.[26],[50] Here $Sp(J)$ denotes the spectrum of the Jacobian matrix J for objective function $f(x, y)$ and $R(\cdot)$ denotes the real part of a complex number.

Thus, our experiment has a co-coercive operator and uses $\eta = \frac{1}{2\ell}$ for GDA.

Then, with this fixed step, we try different values of heavy-ball momentum parameters and run 100 iterations. We find that most of the values, both negative and positive, may slow the convergence. Only very small positive momentum values don't damage the convergence, shown in Figure 4-4. Unfortunately, we don't obtain acceleration when using Heavy-ball momentum on GDA on strongly-monotone quadratic games.

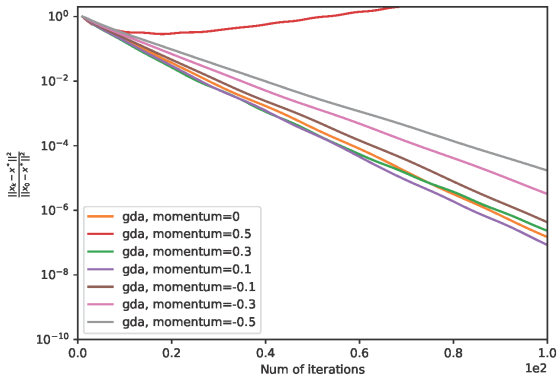

Figure 4-4. GDA on Quadratic with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 10$. All situations (with or without momentum) use step-size as theorem 2.[26]

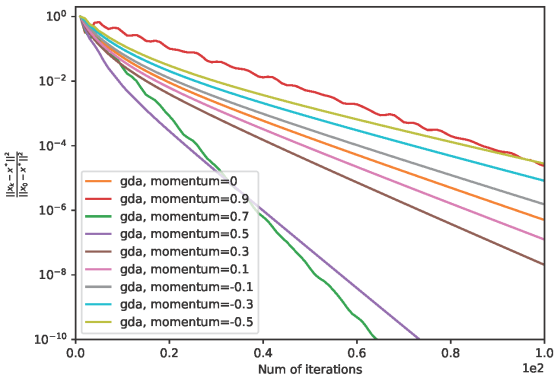

Figure 4-5. GDA on Quadratic with $\mu_A = \mu_C = 1$, $L_A = L_C = 10$, and $B_i = 0$. All situations (with or without momentum) use step-size as theorem 2.[26]

We try one more experiment of format (4.2) with $\mu_A = \mu_C = 1$,$L_A = L_C = 10$, and all $B_i = 0$. By letting all $B_i = 0$, the quadratic game gets rid of the interaction term $x^T By$. Again, we follow the theorem to choose a constant step-size as $\eta = \frac{1}{2\ell}$ for GDA, where ℓ means that operator F is ℓ-co-coercive. Then, with this fixed step, we try different values of heavy-ball momentum parameters and run 100 iterations. Results are shown in Figure 4-5.

We find that Heavy-ball momentum with positive parameters helps to improve the convergence for GDA on quadratic games without interaction term $x^T By$. And among all parameters we try, Heavy-ball momentum with the value of 0.7 has the fastest convergence.

We think the difference in performance of GDA with Heavy-ball momentum on those two quadratic games is caused by the interaction term. When without interaction term, the game can be separated into two independent minimization problems on two independent quadratic functions respected to x or y (inverse the function sign respected to y to change maximizing to minimizing). And we know that Heavy-ball momentum can help to improve the convergence of Gradient Descent on quadratics functions. So, for those kinds of problems, Heavy-ball momentum can improve the convergence. But when there is an interaction term, we cannot separate the min-max problem into two independent optimization problems. So, it becomes a different case, and Heavy-ball momentum may not accelerate them.

We also want to have a look at the performance of AGDA combing with Heavy-ball momentum on strongly-monotone quadratic games. Since our example is SC-SC with parameter μ_1, μ_2, then it implies that the experiment follows a two-sided PL condition with μ_1 and μ_2. Then, we can choose the step-size following theorem 5 as $\eta_1 = \frac{\mu_2^2}{18\ell^3}$, and $\eta_2 = \frac{1}{\ell}$, here ℓ means Lipschitz gradient with parameter $\ell > 0$. We try different values of heavy-ball momentum parameters on strongly-monotone quadratic games with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 10$. All situations run for 100 iterations. Results are shown in Figure 4-6. It can be seen that with some well-chosen momentum parameters, it indeed accelerates the convergence of AGDA. Among all the parameters we choose, the momentum parameter of 0.6 seems to accelerate AGDA method the most.

4.2 Evaluations for Variance Reduction

In this part, we move on to checking the performance of variance reduction on stochastic gradient methods. Like the before section, we contain experiments of bilinear

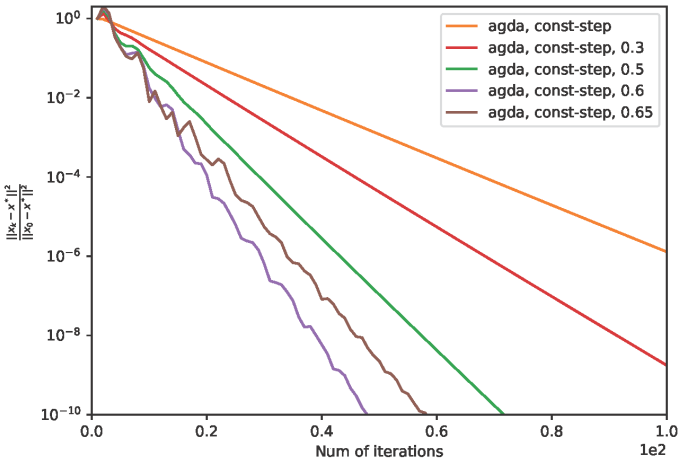

Figure 4-6. AGDA on Quadratic with $\mu_A = \mu_B = \mu_C = 1$, and $L_A = L_B = L_C = 10$. All situations (with or without momentum) use step-size as theorem 5.[44]

games and strongly-monotone quadratic games. On bilinear games, we compare the performance between SHGD and L-SVRHG. On strongly-monotone quadratic games, we compare the performance between SHGD and L-SVRHG, and between SGDA and L-SVRGDA. Parameters selection will be described respectively.

4.2.1 SHG and L-SVRHG

First, we move on to the stochastic version of bilinear games and run an experiment in format (4.1) with $\mu_B = 0, L_B = 2$ to check the performance of SHGD and L-SVRHG.

In proposition 1, we have shown that the Hamiltonian function of a bilinear game is L_H-smooth, μ_H-quasi-strongly convex, and quadratic. Then for SHGD, we use decreasing step-sizes according to theorem 10 as

$$\eta_k = \begin{cases} \frac{1}{2\mathcal{L}} & \text{for } k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu} & \text{for } k > 4\lceil\kappa\rceil. \end{cases} \tag{4.3}$$

Here, $\kappa = \frac{\mathcal{L}}{\mu_H}$ and $\mathcal{L}$ is the expected smoothness parameter. For single element sampling, $\mathcal{L} = L_{max} = max_{\{1,\dots,n^2\}}\{L_{H_{i,j}}\}$ is the maximum smoothness constant of the functions $H_{i,j} = \frac{1}{2} < F_i(z), F_j(z) >$.

According to theorem 16, we choose constant step-size $\eta = \frac{1}{6L_H}$ with $p = \frac{1}{n}$ for

L-SVRHG. we run both SHGD and L-SVRHG for 5 different runs of 5000 iterations. The results are shown in Figure 4-7. Form now on, the horizontal axis denotes the number of samples passed by the algorithm. The line is the average performance of 5 runs for each algorithm. The lower and upper boundary of each shade around the line represents the best and worst cases of convergence in our experiments.

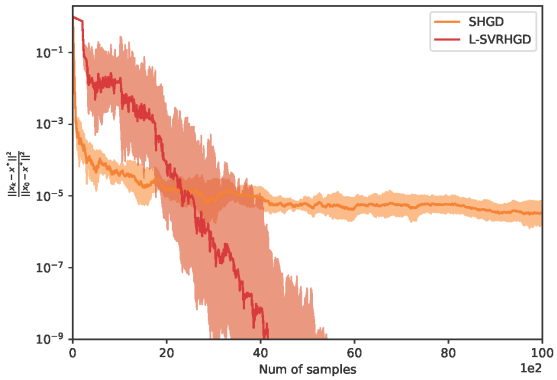

Figure 4-7. SHGD and L-SVRHG for Bilinear with $0 \preceq B_i^T B_i \preceq 2^2$. Use decreasing step-size for SHGD as theorem 10;[26] Use constant step-size for L-SVRHG as theorem 16.[46]

From Figure 4-7 we can see that SHGD with decreasing step-size first has a linear convergence which is because in the initial iterations constant and big step-size is used. Then it has a sublinear convergence to the exact solution after switching to using decreasing step-size. For L-SVRHG with constant step-size, although it shows a slower convergence than SHGD in the initial steps, it has a linear convergence to the exact solution in total. Thus, the variance reduction method not only helps to improve the convergence in total but also vanish the variance, which is the bound of convergence.

Now, we move on to the stochastic version of strongly-monotone quadratic games and run an experiment in format (4.2) with $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$ to check the performance of SHGD and L-SVRHG.

In proposition 2, we have shown that the Hamiltonian function of a strongly-

monotone quadratic game is L_H-smooth, μ_H-quasi-strongly convex, and quadratic. Then for SHGD, we use decreasing step-sizes according to theorem 10 as

$$\eta_k = \begin{cases} \frac{1}{2\mathcal{L}} & \text{for } k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu} & \text{for } k > 4\lceil\kappa\rceil, \end{cases} \tag{4.4}$$

where $\kappa = \frac{\mathcal{L}}{\mu_H}$ and $\mathcal{L}$ is the expected smoothness parameter. For single element sampling, $\mathcal{L} = L_{max} = max_{\{1,\dots,n^2\}}\{L_{H_{i,j}}\}$ is the maximum smoothness constant of the functions $H_{i,j} = \frac{1}{2} < F_i(z), F_j(z) >$.

Also, it can use the parameters in theorem 16 of L-SVRG on its Hamiltonian function to get better performance. That is choosing constant step-size $\eta = \frac{1}{6L_H}$ with $p = \frac{1}{n}$ for L-SVRHG. We run both SHGD and L-SVRHG for 5 different runs of 5000 iterations. Results are shown in Figure 4-8.

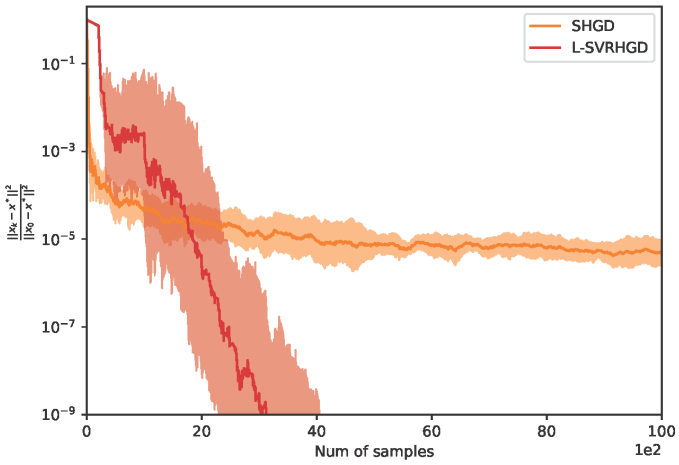

Figure 4-8. SHGD and L-SVRHG on Quadratic with $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$. Use decreasing step-size for SHGD as theorem 10;[26] Use constant step-size for L-SVRHG as theorem 16.[46]

Again, we can see that SHGD with decreasing step-size first has a linear convergence which is because in the initial iterations constant and big step-size are used. Then it has a sublinear convergence to the exact solution after switching to using decreasing step-size. For L-SVRHG with constant step-size, although it shows a slower convergence than SHGD in the initial steps, it has a linear convergence to the exact solution in

total. Thus, the variance reduction method indeed helps to improve the convergence.

4.2.2 SGDA and L-SVRGDA

We know that SGDA or GDA doesn't converge for bilinear games, so in this part, we only run SGDA and L-SVRGDA on strongly-monotone quadratic games.

For SGDA, we select decreasing step-size according to theorem 3 as

$$\eta_k = \begin{cases} \frac{1}{2\ell_F} & \text{for } k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu} & \text{for } k > 4\lceil\kappa\rceil, \end{cases} \tag{4.5}$$

where $\kappa = \frac{\ell_F}{\mu}$ and ℓ_F means that $F \in EC(\ell_F)$. For single-element sampling, $\ell_F = \ell_{max} = max\{\ell_i\}_{i=1}^n$, and the values of the co-coercive parameters ℓ_i for all $i \in [n]$ are computed using $\frac{1}{\ell_i} = min_{\lambda \in Sp(J_i)} R(\frac{1}{\lambda})$.[26],[50] Here $Sp(J_i)$ denotes the spectrum of the Jacobian matrix J_i for all $i \in [n]$ and $R(\cdot)$ denotes the real part of a complex number.

For L-SVRGDA, theorem 17 indicates that we can use step-size as $\eta = \frac{1}{6\hat{\ell}}$. Here $\hat{\ell}$ means that operator F_i is averaged star-co-coercivity with parameter $\hat{\ell}$. We know that if F_i is ℓ_i-co-coercive for all $i \in [n]$, then we can derive averaged star-co-coercivity with parameter $\hat{\ell} \leq \ell_F = \ell_{max} = max\{\ell_i\}_{i=1}^n$. So, in the experiment, we use step-size $\eta = \frac{1}{6\ell_{max}}$, and probability $p = \frac{1}{n}$.

We use a strongly-monotone quadratic game of format (4.2) with $\mu_A = \mu_B = \mu_C = 0.5$, and $L_A = L_B = L_C = 1$ as an experiment. We run both SGDA and L-SVRGDA for 5 runs of 2000 iterations. The results are shown in Figure 4-9.

From the plot, we can see that SGDA with decreasing step-size first has a linearly convergence which is because in the initial iterations constant and big step-size are used. Then it has a sublinear convergence to the exact solution after switching to using decreasing step-size. For L-SVRGDA with constant step-size, although it shows a slower convergence than SGDA in the initial steps, it has a linear convergence to the exact solution in total. Thus, the variance reduction method not only helps to

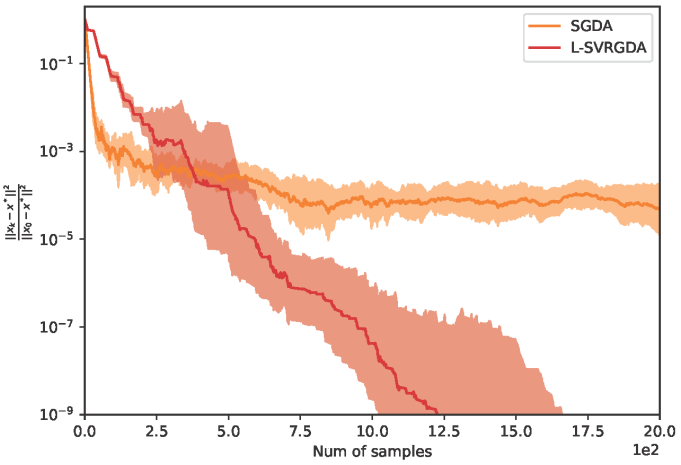

Figure 4-9. SGDA and L-SVRGDA on Quadratic with $\mu_A = \mu_B = \mu_C = 0.5$, and $L_A = L_B = L_C = 1$. Use decreasing step-size for SGDA as theorem 3;[26] Use constant step-size for L-SVRGDA as theorem 17.[47]

improve the convergence in total but also vanishes the neighborhood when using a constant step-size.

4.3 Random Reshuffling

In this part, we compare the results between single uniform sampling and random reshuffling on gradient methods. For bilinear games, we run SHGD and HGD-RR. For strongly-monotone quadratic games, we compare SHGD and Hamiltonian Gradient Descent with Random Reshuffling (HGD-RR), SGDA and Gradient Descent Ascent with Random Reshuffling (GDA-RR), and between S-SEG and Same-sample Exagradient with Random Reshuffling (S-EG-RR).

4.3.1 SHGD and HGD-RR

We take two bilinear game examples of format (4.1) with parameters $\mu_B = 0.2, L_B = 1$, and $\mu_B = 0.5, L_B = 1$ respectively.

For both SHGD and HGD-RR, we both use decreasing step-size as (3.11) as

$$\eta_k = \begin{cases} \frac{1}{2\mathcal{L}_H} & \text{for } k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu_H} & \text{for } k > 4\lceil\kappa\rceil. \end{cases} \tag{4.6}$$

Here, $\kappa = \frac{\mathcal{L}}{\mu_H}$ and $\mathcal{L}$ is the expected smoothness parameter. For single element sampling, $\mathcal{L} = L_{max} = max_{\{1,\dots,n^2\}}\{L_{H_{i,j}}\}$ is the maximum smoothness constant of the functions $H_{i,j}$.

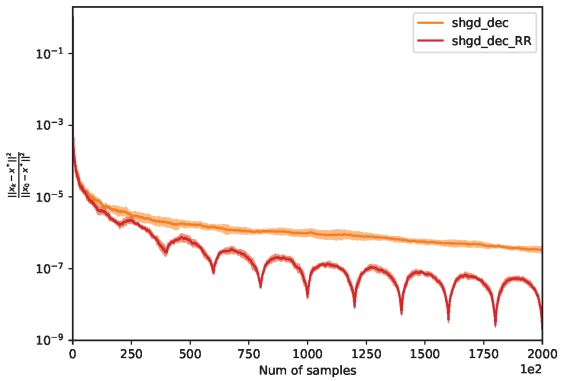

Figure 4-10. SHGD and HGD-RR for Bilinear with $0.2^2 \preceq B_i^T B_i \preceq 1$. Use decreasing step-size according to theorem 10 for SHGD;[26] For better comparison, HGD-RR use the same decreasing step as SHGD does.

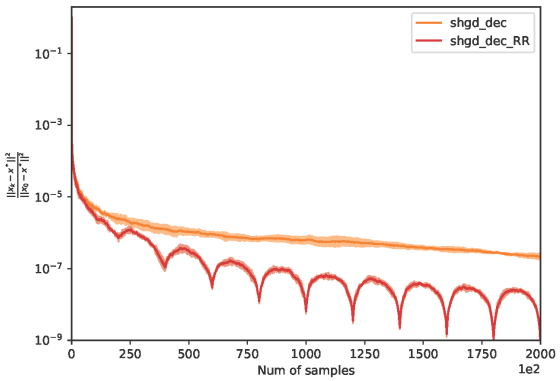

Figure 4-11. SHGD and HGD-RR for Bilinear with $0.5^2 \preceq B_i^T B_i \preceq 1$. Use decreasing step-size according to theorem 10 for SHGD;[26] For better comparison, HGD-RR use the same decreasing step as SHGD does.

We run 5 times of 100000 iterations for SHGD and HGD-RR of different problems. The results of those two bilinear games can be seen in Figures 4-10 and 4-11. It can be witnessed that in both plots, SHGD convergence sublinearly to the exact solution, and HGD-RR converges faster with vibration. Thus, we say that random reshuffling help to improve the convergence for using SHGD with decreasing step-size on bilinear games.

Then, we take two strongly-monotone quadratic games of format (4.2) with parameter $\mu_A = \mu_B = \mu_C = 0.2$, $L_A = L_B = L_C = 1$, and $\mu_A = \mu_B = \mu_C = 0.5$, $L_A = L_B = L_C = 1$, respectively.

Again, for both SHGD and HGD-RR, we use decreasing step-size as (3.11) as

$$\eta_k = \begin{cases} \frac{1}{2\mathcal{L}_H} & \text{for } k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu_H} & \text{for } k > 4\lceil\kappa\rceil. \end{cases} \tag{4.7}$$

Here, $\kappa = \frac{\mathcal{L}}{\mu_H}$ and $\mathcal{L}$ is the expected smoothness parameter. For single element sampling, $\mathcal{L} = L_{max} = max_{\{1,\dots,n^2\}}\{L_{H_{i,j}}\}$ is the maximum smoothness constant of the functions $H_{i,j}$.

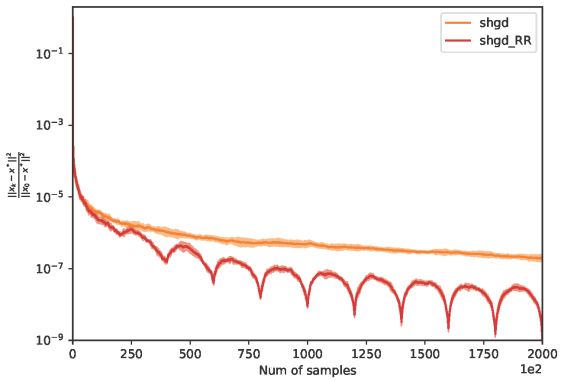

Figure 4-12. SHGD and HGD-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$. Use decreasing step-size according to theorem 10 for SHGD;[4] For better comparison, HGD-RR use the same decreasing step as SHGD does.

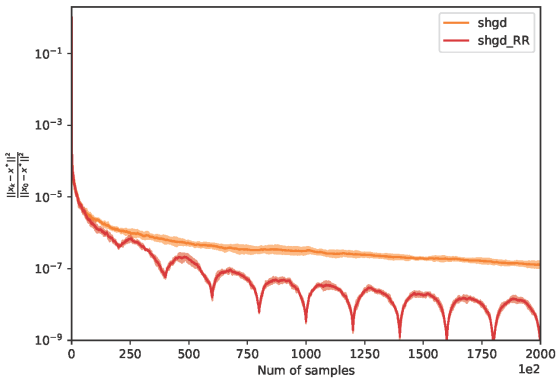

Figure 4-13. SHGD and HGD-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0.5$, and $L_A = L_B = L_C = 1$. Use decreasing step-size according to theorem 10 for SHGD;[4] For better comparison, HGD-RR use the same decreasing step as SHGD does.

We run 5 times of 100000 iterations for SHGD and HGD-RR of different problems. The results of those two strongly-monotone quadratic games can be seen in Figure 4-12 and 4-13. It can be witnessed that in both plots, SHGD convergence sublinearly to the exact solution, and HGD-RR converges faster with vibration. Random reshuffling help to improve the convergence for using SHGD with decreasing step-size on strongly-monotone quadratic games. Since our experiment is an example of SC-SC problems, we guess that for SC-SC problems, random reshuffling may also be a method that can help SHGD get a faster convergence rate.

4.3.2 SGDA and GDA-RR

We know that SGDA or GDA doesn't converge for bilinear games, so in this part, we only run SGDA and GDA-RR on strongly-monotone quadratic games.

We take two strongly-monotone quadratic games of format (4.2) with parameter $\mu_A = \mu_B = \mu_C = 0.2$, $L_A = L_B = L_C = 1$, and $\mu_A = \mu_B = \mu_C = 0.5$, $L_A = L_B = L_C = 1$, respectively.

For SGDA, we select decreasing step-size according to theorem 3 as

$$\eta_k = \begin{cases} \frac{1}{2\ell_F} & \text{for} k \leq 4\lceil\kappa\rceil \\ \frac{2k+1}{(k+1)^2\mu} & \text{for } k > 4\lceil\kappa\rceil, \end{cases} \tag{4.8}$$

where $\kappa = \frac{\ell_F}{\mu}$ and ℓ_F means that operator $F \in EC(l_\xi)$. For single-element sampling, $\ell_F = \ell_{max} = max\{\ell_i\}_{i=1}^n$, and the values of the co-coercive parameters ℓ_i for all $i \in [n]$ are computed using $\frac{1}{\ell_i} = min_{\lambda \in Sp(J_i)} R(\frac{1}{\lambda})$.[26],[50] Here $Sp(J_i)$ denotes the spectrum of the Jacobian matrix J_i for all $i \in [n]$ and $R(\cdot)$ denotes the real part of a complex number. We also use the same decreasing step-size for GDA-RR.

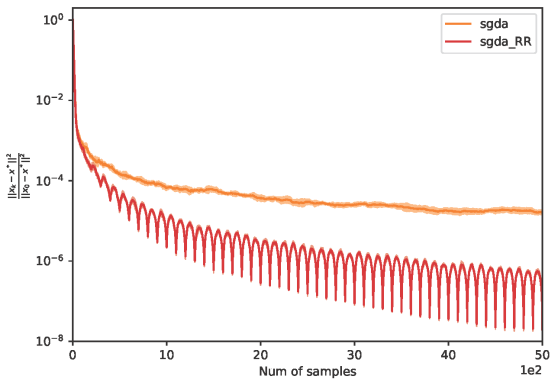

Figure 4-14. SGDA and GDA-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$. Use decreasing step-size according to theorem 3 for SGDA;[26] For better comparison, GDA-RR use the same decreasing step as SGDA does.

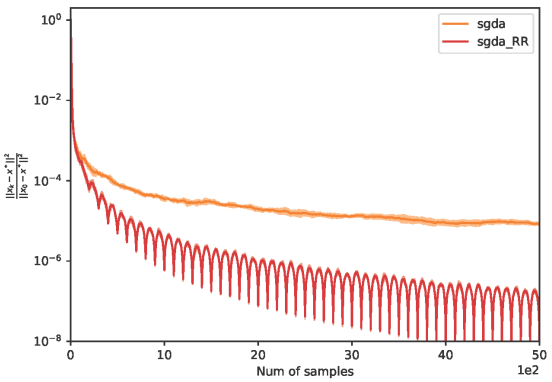

Figure 4-15. SGDA and GDA-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0.5$, and $L_A = L_B = L_C = 1$. Use decreasing step-size according to theorem 3 for SGDA;[26] For better comparison, GDA-RR use the same decreasing step as SGDA does.

We run 5 times of 5000 iterations for SGDA and GDA-RR of different problems.

Results of those two strongly-monotone quadratic games can be seen in Figures 4-14 and 4-15. It can be witnessed that in both plots, SGDA convergence sublinearly to the exact solution, and GDA-RR converges faster with vibration. Thus, we say that random reshuffling help to improve the convergence for using SGDA with decreasing step-size on strongly-monotone quadratic games. Since our experiment is an example of SC-SC problems, we guess that for SC-SC problems, random reshuffling may also be a method that can help SGDA get a faster convergence rate.

4.3.3 S-SEG and S-EG-RR

We take a strongly-monotone quadratic game of format (4.2) with parameter $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$. Parameter selections are shown below.

We follow Theorem 11 to choose a constant step-size for S-SEG. That is, using $\eta_1 = \frac{1}{6L_\xi}, \eta_2 = \frac{\eta_1}{4}$ for constant step-size. We also use the same constant step-size for S-EG-RR to compare.

For S-SEG with decreasing step-size, by following theorem 12, let $\eta_{2,\xi_k} = \frac{\eta_{1,\xi_k}}{4}$, and $\eta_{1,\xi_k} = \beta_k \eta_{\xi_k}$, where $\eta_{\xi_k} = \frac{1}{4|\mu_{\xi_k}| + \sqrt{2}L_{\xi_k}}$, and $\tilde{\rho} = \frac{1}{8} E_{\xi_k}[\eta_{\xi_k}\mu_{\xi_k}(1_{\mu_{\xi_k}\geq 0} + 4 \times 1_{\mu_{\xi_k}<0})]$. Assume $\tilde{\rho} > 0$. Then, for all total number of iterations $K \geq 0$ and $\{\beta_k\}_{k\geq 0}$ such that

$$\begin{cases} if\ K \leq \frac{1}{\tilde{\rho}} & \beta_k = 1, \\ if\ K > \frac{1}{\tilde{\rho}}\ and\ k \geq k_0, & \beta_k = 1, \\ if\ K > \frac{1}{\tilde{\rho}}\ and\ k \geq k_0, & \beta_k = \frac{2}{2+\tilde{\rho}(k-k_0)} \end{cases} \tag{4.9}$$

where $k_0 = \lceil K/2 \rceil$. The total iteration number we chose is $K = 1000$. We also use the same decreasing step-size for S-EG-RR to compare. All methods on different games are run 5 times of $K = 1000$ iterations. Results are shown in Figure 4-16 and 4-17.

Figure 4-16 shows the results for running S-SEG and S-EG-RR with constant or decreasing step-sizes on a strongly-monotone quadratic game with $\mu_A = \mu_B = \mu_C = 0$ and $L_A = L_B = L_C = 2$. Figure 4-17 shows the results for running S-SEG and S-EG-

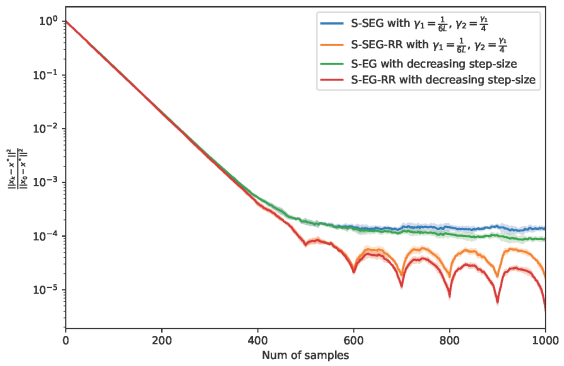

Figure 4-16. SEG and S-EG-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0$, and $L_A = L_B = L_C = 2$. Use constant or decreasing step-size for S-SEG according to theorem 11 and 12 respectively.[36] For better comparison, S-EG-RR use the same decreasing step as S-SEG does.

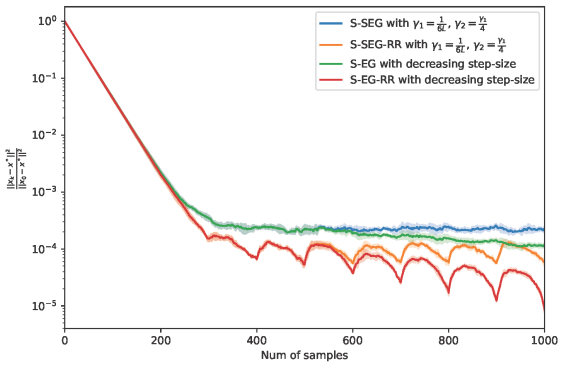

Figure 4-17. SEG and S-EG-RR on Quadratic with $\mu_A = \mu_B = \mu_C = 0.2$, and $L_A = L_B = L_C = 1$. Use constant or decreasing step-size for S-SEG according to theorem 11 and 12 respectively.[36] For better comparison, S-EG-RR use the same decreasing step as S-SEG does.

RR with constant or decreasing step-sizes on a strongly-monotone quadratic game with $\mu_A = \mu_B = \mu_C = 0.2$ and $L_A = L_B = L_C = 1$. It can be seen that with constant step-size, both S-SEG and S-EG-RR converge linearly to a neighborhood of the exact solution. The difference is that S-EG-RR has a smaller neighbor compared with S-SEG, and S-EG-RR is with some vibration when reaching the neighborhood. For decreasing step-size, both S-SEG and S-EG-RR first have a linear convergence because of the constant step-size in the initial iterations. Then they experience a sublinear convergence to the exact solution. Again, S-EG-RR shows a faster convergence with vibration, compared with S-SEG.

In our experiment, random reshuffling improves the convergence for S-SEG on the strongly-monotone quadratic game with either constant step-size or decreasing step-size. Since our experiment is an example of SC-SC problems, we guess that for SC-SC problems, random reshuffling may be helpful for S-SEG to get a faster convergence rate.

Conclusions

For the deterministic gradient methods of min-max problems, we can see from our experiment part that using Heavy-ball momentum or Nesterov's optimal method on HGD on bilinear games or quadratic games helps to get faster convergence. For bilinear games, using some well-selected negative momentum makes AGDA converge linearly to the exact solution. When moving to quadratic games with interaction terms, we don't find any Heavy-ball momentum parameters that can accelerate GDA. But for AGDA on quadratic games with or without interaction terms, there exist some heavy-ball momentum parameters that can fasten the convergence. So, it is worth to further studying and analyzing AGDA with accelerated method on quadratic games, and, in general, on some strongly convex-strongly concave games.

For the stochastic gradient methods of min-max problems, loopless variance reduction methods are very helpful. Using L-SVRHG on bilinear games or quadratic games both have linear convergence to the exact solution. Using L-SVRGDA on quadratic games also shows a linear convergence to the exact solution. the loopless variance reduction methods help to vanish the neighborhood when using the constant step sizes for stochastic gradient methods. So, variance reduction seems to be a very sufficient method for stochastic gradient methods.

Random reshuffling is a method that is also helpful to improve the convergence of stochastic gradient methods on min-max problems. From our experiments, the stochastic gradient method with RR converges faster than the same stochastic gradient method with single uniform sampling with the same decreasing step-sizes. Improve-

ments can be seen in HGD-RR on bilinear games and quadratic games, GDA-RR on quadratic games, and S-EG-RR on quadratic games. So, it is worth to further studying and analyzing the method of random reshuffling on various stochastic gradient methods on different problems.

In summary, the accelerated method, variance reduction method, and random reshuffling are all useful and sufficient methods that can improve convergence for some deterministic or stochastic problems. Further study and analysis of their performance are meaningful and crucial.

Bibliography

[1] Goodfellow, Ian J., et al. "Generative adversarial nets (Advances in neural information processing systems)(pp. 2672–2680)." Red Hook, NY Curran (2014).

[2] Xu, Huan, Constantine Caramanis, and Shie Mannor. "Robustness and Regularization of Support Vector Machines." Journal of machine learning research 10.7 (2009).

[3] Shafieezadeh Abadeh, Soroosh, Peyman M. Mohajerin Esfahani, and Daniel Kuhn. "Distributionally robust logistic regression." Advances in Neural Information Processing Systems 28 (2015).

[4] Cesa-Bianchi, Nicolo, and Gábor Lugosi. Prediction, learning, and games. Cambridge university press, 2006.

[5] Sinha, Aman, Hongseok Namkoong, and John C. Duchi. "Certifiable Distributional Robustness with Principled Adversarial Training. CoRR, abs/1710.10571." arXiv preprint arXiv:1710.10571 (2017).

[6] Shamma, Jeff, ed. Cooperative control of distributed multi-agent systems. John Wiley Sons, 2008.

[7] Mateos, Gonzalo, Juan Andrés Bazerque, and Georgios B. Giannakis. "Distributed sparse linear regression." IEEE Transactions on Signal Processing 58.10 (2010): 5262-5276.

[8] Zhang, Kaiqing, Zhuoran Yang, and Tamer Başar. "Multi-agent reinforcement learning: A selective overview of theories and algorithms." Handbook of Reinforcement Learning and Control (2021): 321-384.

[9] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).

[10] Jordan, Michael I. "Artificial intelligence—the revolution hasn't happened yet." (2019).

[11] Albuquerque, Isabela, et al. "Generalizing to unseen domains via distribution matching." arXiv preprint arXiv:1911.00804 (2019).

[12] Pfau, David, and Oriol Vinyals. "Connecting generative adversarial networks and actor-critic methods." arXiv preprint arXiv:1610.01945 (2016).

[13] Chen, Ziyi, Shaocong Ma, and Yi Zhou. "Accelerated proximal alternating gradient-descent-ascent for nonconvex minimax machine learning." 2022 IEEE International Symposium on Information Theory (ISIT). IEEE, 2022.

[14] Huang, Feihu, Xidong Wu, and Heng Huang. "Efficient mirror descent ascent methods for nonsmooth minimax problems." Advances in Neural Information Processing Systems 34 (2021): 10431-10443.

[15] Jin, Chi, Praneeth Netrapalli, and Michael Jordan. "What is local optimality in nonconvex-nonconcave minimax optimization?." International conference on machine learning. PMLR, 2020.

[16] Lin, Tianyi, Chi Jin, and Michael Jordan. "On gradient descent ascent for nonconvex-concave minimax problems." International Conference on Machine Learning. PMLR, 2020.

[17] Lin, Tianyi, Chi Jin, and Michael I. Jordan. "Near-optimal algorithms for minimax optimization." Conference on Learning Theory. PMLR, 2020.

[18] Lu, Songtao, et al. "Hybrid block successive approximation for one-sided non-convex min-max problems: algorithms and applications." IEEE Transactions on Signal Processing 68 (2020): 3676-3691.

[19] Ostrovskii, Dmitrii M., Andrew Lowy, and Meisam Razaviyayn. "Efficient search of first-order nash equilibria in nonconvex-concave smooth min-max problems." SIAM Journal on Optimization 31.4 (2021): 2508-2538.

[20] Thekumparampil, Kiran K., et al. "Efficient algorithms for smooth minimax optimization." Advances in Neural Information Processing Systems 32 (2019).

[21] Xu, Zi, et al. "A unified single-loop alternating gradient projection algorithm for nonconvex-concave and convex-nonconcave minimax problems." arXiv preprint arXiv:2006.02032 (2020).

[22] Balduzzi, David, et al. "The mechanics of n-player differentiable games." International Conference on Machine Learning. PMLR, 2018.

[23] Nedić, Angelia, and Asuman Ozdaglar. "Subgradient methods for saddle-point problems." Journal of optimization theory and applications 142.1 (2009): 205-228.

[24] Nemirovski, Arkadi. "Prox-method with rate of convergence O (1/t) for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems." SIAM Journal on Optimization 15.1 (2004): 229-251.

[25] Zhang, Guodong, et al. "Near-optimal local convergence of alternating gradient descent-ascent for minimax optimization." International Conference on Artificial Intelligence and Statistics. PMLR, 2022.

[26] Loizou, Nicolas, et al. "Stochastic gradient descent-ascent and consensus optimization for smooth games: Convergence analysis under expected co-coercivity." Advances in Neural Information Processing Systems 34 (2021): 19095-19108.

[27] Mescheder, Lars, Andreas Geiger, and Sebastian Nowozin. "Which training methods for GANs do actually converge?." International conference on machine learning. PMLR, 2018.

[28] Gidel, Gauthier, et al. "Negative momentum for improved game dynamics." The 22nd International Conference on Artificial Intelligence and Statistics. PMLR, 2019.

[29] Daskalakis, Constantinos, et al. "Training gans with optimism." arXiv preprint arXiv:1711.00141 (2017).

[30] Chasnov, Benjamin, et al. "Convergence analysis of gradient-based learning in continuous games." Uncertainty in Artificial Intelligence. PMLR, 2020.

[31] G.M. Korpelevich. The extragradient method for finding saddle points and other problems. Ekonomika i Matematicheskie Metody, 12:747–756, 1976.

[32] Nesterov, Yurii. "Dual extrapolation and its applications to solving variational inequalities and related problems." Mathematical Programming 109.2 (2007): 319-344.

[33] Juditsky, Anatoli, Arkadi Nemirovski, and Claire Tauvel. "Solving variational inequalities with stochastic mirror-prox algorithm." Stochastic Systems 1.1 (2011): 17-58.

[34] Iusem, Alfredo N., et al. "Extragradient method with variance reduction for stochastic variational inequalities." SIAM Journal on Optimization 27.2 (2017): 686-724.

[35] Palaniappan, Balamurugan, and Francis Bach. "Stochastic variance reduction methods for saddle-point problems." Advances in Neural Information Processing Systems 29 (2016).

[36] Gorbunov, Eduard, et al. "Stochastic extragradient: General analysis and improved rates." International Conference on Artificial Intelligence and Statistics. PMLR, 2022.

[37] Mokhtari, Aryan, Asuman Ozdaglar, and Sarath Pattathil. "A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach." International Conference on Artificial Intelligence and Statistics. PMLR, 2020.

[38] Li, Chris Junchi, et al. "On the Convergence of Stochastic Extragradient for Bilinear Games using Restarted Iteration Averaging." International Conference on Artificial Intelligence and Statistics. PMLR, 2022.

[39] Abernethy, Jacob, Kevin A. Lai, and Andre Wibisono. "Last-iterate convergence rates for min-max optimization." arXiv preprint arXiv:1906.02027 (2019).

[40] Karimi, Hamed, Julie Nutini, and Mark Schmidt. "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition." Joint European conference on machine learning and knowledge discovery in databases. Springer, Cham, 2016.

[41] Loizou, Nicolas, et al. "Stochastic hamiltonian gradient methods for smooth games." International Conference on Machine Learning. PMLR, 2020.

[42] Gower, Robert Mansel, et al. "SGD: General analysis and improved rates." International Conference on Machine Learning. PMLR, 2019.

[43] Gower, Robert, Othmane Sebbouh, and Nicolas Loizou. "Sgd for structured non-convex functions: Learning rates, minibatching and interpolation." International Conference on Artificial Intelligence and Statistics. PMLR, 2021.

[44] Yang, Junchi, Negar Kiyavash, and Niao He. "Global convergence and variance-reduced optimization for a class of nonconvex-nonconcave minimax problems." arXiv preprint arXiv:2002.09621 (2020).

[45] Wright, Stephen J., and Benjamin Recht. Optimization for data analysis. Cambridge University Press, 2022.

[46] Kovalev, Dmitry, Samuel Horváth, and Peter Richtárik. "Don't jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop." Algorithmic Learning Theory. PMLR, 2020.

[47] Beznosikov, Aleksandr, et al. "Stochastic gradient descent-ascent: Unified theory and new efficient methods." arXiv preprint arXiv:2202.07262 (2022).

[48] Mishchenko, Konstantin, Ahmed Khaled, and Peter Richtárik. "Random reshuffling: Simple analysis with vast improvements." Advances in Neural Information Processing Systems 33 (2020): 17309-17320.

[49] Das, Aniket, Bernhard Schölkopf, and Michael Muehlebach. "Sampling without Replacement Leads to Faster Rates in Finite-Sum Minimax Optimization." arXiv preprint arXiv:2206.02953 (2022).

[50] Azizian, Waïss, et al. "A tight and unified analysis of gradient-based methods for a whole spectrum of differentiable games." International Conference on Artificial Intelligence and Statistics. PMLR, 2020.