ON THE CONVERGENCE OF ADAM-TYPE ALGORITHMS FOR SOLVING STRUCTURED SINGLE NODE AND DECENTRALIZED MIN-MAX SADDLE POINT GAMES

Babak Barazandeh[1,⋆], *Kristal Curtis*[1], *Chandrima Sarkar*[1], *Ram Sriharsha*[2], *George Michailidis*[3]

[1]Splunk, [2]Pinecone, [3]University of Florida

ABSTRACT

Many modern machine learning problems require solving min-max saddle point games, whose computational complexity is NP-hard in general. To overcome this issue, most available algorithms aim for finding a first-order Nash equilibrium solution that always exists under mild assumptions. However, the proposed algorithms for obtaining such solutions are non-adaptive and also exhibit slow convergence rates in real settings. Further, most algorithms are centralized in nature and cannot be adapted to a decentralized architecture in a straightforward manner. This study aims to address these issues by introducing general two-step adaptive algorithms for obtaining first-order Nash equilibrium solutions of min-max games in both single-node and decentralized architectures. We also obtain the non-asymptotic convergence rates of the algorithms assuming the objective functions satisfy the weak Minty variational inequality condition which is standard in recent literature. Finally, we illustrate the performance of the proposed algorithms by using them to train neural networks which are more robust against adversarial attacks compared to neural networks trained using existing algorithms.

***Index Terms*—** Min-max games, Generative adversarial networks, Adaptive algorithms, Decentralized optimization

1. INTRODUCTION

Min-max saddle point games have recently drawn a lot of attention due to their relationship with various machine learning tasks, including training Generative Adversarial Networks (GANs) [1] and neural networks that are robust against adversarial attacks [2], fairness in machine learning [3, 4], and many others [5].

However, solving a non-convex-non-concave min-max problem is not only NP-hard [6], but the Nash equilibrium does not exist in general [7]. As a result, all recent efforts are focused on structured min-max games whose goal is to compute a first-order Nash equilibria (see Section 2 for additional details). [8] considers deterministic min-max games that are non-convex-concave or non-convex-PL, in which the max player satisfies the Polyak-Łojasiewicz (PL) condition [9]. [10] studies similar games but assumes that the objective function is non-smooth. [11] studies a stochastic version of the min-max game in which both players satisfy the PŁ condition. [12] studies the saddle point game under the Minty variational inequality (MVI) condition. [13] recently introduced the weak MVI assumption and proposed a generalization of the extra-gradient method that obtains stationary points of min-max problems under this condition. We describe this assumption in more detail in Section 4. There have also been efforts to solve these structured min-max games using zeroth-order methods [14, 15].

One of the interesting lines of research in optimization is studying the performance of decentralized methods compared to centralized ones. In centralized optimization problems, there exists a central computing node which holds information on the model and updates the model's parameters by appropriately aggregating information received from all other nodes [16]. A centralized setting is not desirable when there are network bandwidth limitations or privacy considerations [17, 18]. On the other hand, in a decentralized setting [17, 19], each node stores a local copy of the model, and each node communicates with its selected subset of all nodes. This setup addresses both network bandwidth issues and data privacy concerns, which might arise due to nodes' unwillingness to share any information with a central node [20, 21].

There are two important aspects to note regarding min-max problems. First, most of the above-mentioned studies for solving min-max games are non-adaptive and are thus either very slow or computationally expensive [22, 23]. Second, all these methods are for the single-node case and cannot be easily generalized for decentralized min-max games. As a matter of fact, unlike classical minimization problems, there are very few works that aim to solve min-max problems for decentralized structures. As an example, [18] proposes a decentralized non-adaptive method; however, due to its slow convergence rate in real world problems, the authors do not use their own proposed method in the numerical experiments, despite the theoretical guarantees they establish. As another example, the method in [24] requires a closed-form solution for the proximal point step, which makes it infeasible for solving practical large-scale min-max problems. [25] has recently proposed an adaptive method that is applicable only in a narrow problem domain. This illustrates a major gap and a need for developing adaptive methods that are broadly

⋆ Corresponding author, email: bbarazandeh@splunk.com

applicable and easily generalizable for both single-node and decentralized scenarios. In this paper, we address these issues by proposing generalized two-step adaptive optimizers for solving both single-node and decentralized min-max problems using a wide variety of adaptive update rules. In addition, we obtain their non-asymptotic convergence rates under more relaxed assumptions.

The remainder of the paper is organized as follows: in Section 2, we define the problem of interest formally. In Sections 3 and 4, we discuss the details of our proposed algorithms and analyze their theoretical behavior, respectively. Finally, Section 5 describes our experiments.

Notation: We use $\mathcal{S} \triangleq \mathbb{R}^{d_s}$ and $\mathcal{T} \triangleq \mathbb{R}^{d_t}$ to represent the domain for minimization and maximization parameters. We also define $\boldsymbol{g}(\boldsymbol{x}) = [\nabla_{\boldsymbol{s}} F(\boldsymbol{s},\boldsymbol{t}), \nabla_{\boldsymbol{t}} F(\boldsymbol{s},\boldsymbol{t})]$ and $\hat{\boldsymbol{g}}(\boldsymbol{x};\boldsymbol{\xi}) = [\nabla_{\boldsymbol{s}} f(\boldsymbol{s},\boldsymbol{t};\boldsymbol{\xi}), \nabla_{\boldsymbol{t}} f(\boldsymbol{s},\boldsymbol{t};\boldsymbol{\xi})]$ to be the values of the exact gradient and its estimate at the point $\boldsymbol{x} = (\boldsymbol{s},\boldsymbol{t}) \in \mathcal{S} \times \mathcal{T}$. In the decentralized setup with M machines, we use $\boldsymbol{g}_i(\boldsymbol{x})$ and $\boldsymbol{g}_i(\boldsymbol{x};\boldsymbol{\xi})$ for $i \in \{1,\dots,M\}$ to represent the exact and estimated gradient for machine i. In Section 3, we use $\boldsymbol{g}_{i,k}(\boldsymbol{x})$ and $\boldsymbol{g}_{i,k}(\boldsymbol{x};\boldsymbol{\xi})$ for the gradients obtained for machine i at iteration k. The weighting matrix and its second largest eigenvalue are represented by $\boldsymbol{W} = [w_{i,j}]$ and τ, respectively. We also define the matrix $\boldsymbol{W}^r = [w_{i,j}^r]$.

2. PROBLEM FORMULATION

We are interested in solving the following min-max optimization problem:

$$\min_{\boldsymbol{s}\in\mathcal{S}} \max_{\boldsymbol{t}\in\mathcal{T}} F(\boldsymbol{s},\boldsymbol{t}) \tag{1}$$

for a general function $F(\cdot,\cdot)$, i.e., the function $F(\boldsymbol{s},\cdot)$ is non-convex and $F(\cdot,\boldsymbol{t})$ is non-concave. The ultimate goal for this problem is obtaining a Nash equilibrium, which might not exist in general. So, we settle at finding a first-order Nash equilibrium. For more details, refer to [8, 22].

Definition 1 (ϵ-Stochastic First Order Nash Equilibrium (ϵ-SFNE)). *A point $\boldsymbol{x}^* \in \mathcal{S} \times \mathcal{T}$ is called an ϵ-Stochastic First Order Nash Equilibrium (ϵ-SFNE) of the game defined in* (1) *if* $\mathbb{E}\left[\|\nabla F(\boldsymbol{x}^*)\|^2\right] \le \epsilon^2$.

In this paper, the goal is to obtain an ϵ-SFNE of the game defined in (1) in both the single-node and decentralized cases in which

$$F(\boldsymbol{s},\boldsymbol{t}) = \mathbb{E}_{\boldsymbol{\xi}\sim\mathcal{D}}[f(\boldsymbol{s},\boldsymbol{t};\boldsymbol{\xi})] \tag{2}$$

or

$$F(\boldsymbol{s},\boldsymbol{t}) = \frac{1}{M}\sum_{j=1}^{M}[F_j(\boldsymbol{s},\boldsymbol{t}) = \mathbb{E}_{\boldsymbol{\xi}\sim\mathcal{D}_j}[f(\boldsymbol{s},\boldsymbol{t};\boldsymbol{\xi})], \tag{3}$$

respectively. Here, $\mathcal{D}$ and $\mathcal{D}_j$ represent the data for central node and machine j. In the next section, we propose general two-step algorithms that are capable of obtaining such points under standard assumptions.

Number	Update Rule
1	$\boldsymbol{v}_k = \beta_2 \boldsymbol{v}_{k-1} + (1-\beta_2)\boldsymbol{g}_k^2$ $\hat{\boldsymbol{v}}_k = \max(\hat{\boldsymbol{v}}_{k-1}, \boldsymbol{v}_k)$
2	$\hat{\boldsymbol{v}}_k = \beta_2 \hat{\boldsymbol{v}}_{k-1} + (1-\beta_2)\boldsymbol{g}_k^2$
3	$\boldsymbol{v}_k = \beta_2 \boldsymbol{v}_{k-1} + (1-\beta_2)\boldsymbol{g}_k^2$ $\hat{\boldsymbol{v}}_k = \beta_3 \hat{\boldsymbol{v}}_k + (1-\beta_3)\max(\hat{\boldsymbol{v}}_{k-1}, \boldsymbol{v}_k)$

Table 1. Various choices for updating the second moment, wherein $0 < \beta_2, \beta_3 < 1$. Regarding the first moment, we have $\beta_{1,k} = 0$ for the second rule and $\beta_{1,k} = \beta_{1,1}\kappa^{k-1}$ for the other ones with $0 < \kappa, \beta_{1,1} < 1$, and k corresponds to the iteration number.

3. PROPOSED ALGORITHMS

We start by presenting our single-node adaptive algorithm, and then develop its decentralized variant. To gain insights into the proposed two-step algorithms, consider two variables $\boldsymbol{x}_k, \boldsymbol{y}_k \in \mathcal{S} \times \mathcal{T}$. At iteration k, in step 1 we update the variable $\boldsymbol{y}_k$, and then after getting the estimated gradient at point $\boldsymbol{y}_k$, we update the variable $\boldsymbol{x}_k$ in the second step using an *adaptive* gradient. In the decentralized case, each node $i \in \{1,\dots,M\}$ has its own copy of these variables, namely $\boldsymbol{x}_{i,k}, \boldsymbol{y}_{i,k}$, and after updating them locally, each node aggregates its values with those of its neighbours using the weighting matrix $\boldsymbol{W}^r$.

Table 1 shows some of the popular choices for updating the second moment estimate. It is important to note that despite similarity in notation, these update rules are not the same as AMSGrad [26], RMSProp [27], or FEMA [28] since the estimated gradient contains the gradients with respect to both minimization and maximization parameters. However, if we assume that the objective function *does not* depend on the parameters of the max player, the rules in our table reduce to those update rules. In Section 4, to obtain the convergence behavior, we assume that the second moment is updated by any of the three cases in Table 1.

Single Node: Algorithm 1 details how to obtain an ϵ-SFNE in the single-node setting. We first update $\boldsymbol{y}_k$. Then, we obtain a mini-batch of data $\boldsymbol{\xi}_k = (\boldsymbol{\xi}_k^1, \cdots, \boldsymbol{\xi}_k^m)$ to estimate the stochastic gradient at $\boldsymbol{y}_k$. The box in Algorithm 1 shows the adaptive update rule. First, we obtain a biased first moment estimate by combining the previous iteration's estimate for the first moment with the newly-estimated stochastic gradient. Then, we update the second moment estimate, using one of the rules from Table 1. We then scale the first moment using our updated second moment estimate, which we then use to update $\boldsymbol{x}_k$.

Decentralized: In Algorithm 2, we show how to obtain an ϵ-SFNE in the decentralized setting. The first step updates the local estimate of $\boldsymbol{y}_{i,k}$, which is achieved by initially obtaining a local estimate and then aggregating it with the other nodes' values. Then, each node obtains a mini-batch of data

Algorithm 1: Generalized two-step single node adaptive min-max optimizer (AMMO)

Input : Update rule from Table 1, η, $m \in \mathbb{N}$, $\beta_{1,1}$, κ: if applicable

initialize $\boldsymbol{y}_0 = \boldsymbol{x}_0 = \boldsymbol{\alpha}_0 = \boldsymbol{m}_0 = \boldsymbol{v}_0 = \mathbf{0}_{d,1}$.

for $k = 1 : N$ **do**

$\boldsymbol{y}_k = \boldsymbol{x}_{k-1} - \eta\ \boldsymbol{\alpha}_{k-1}$ //step 1: update local estimate of $\boldsymbol{y}_k$

$\hat{\boldsymbol{g}}_k = \frac{1}{m}\sum_{j=1}^{m} \hat{\boldsymbol{g}}_k(\boldsymbol{y}_k; \boldsymbol{\xi}_k^j)$ //estimate the stochastic gradient at $\boldsymbol{y}_k$

Adaptive update rule

$\boldsymbol{m}_k = \beta_{1,k}\boldsymbol{m}_{k-1} + (1-\beta_{1,k})\hat{\boldsymbol{g}}_k$ //update biased 1st moment estimate

$\hat{\boldsymbol{v}}_k = h_k(\hat{\boldsymbol{g}}_1, \hat{\boldsymbol{g}}_2, \ldots, \hat{\boldsymbol{g}}_k)$ //update 2nd moment estimate

$\boldsymbol{\alpha}_k = \hat{\boldsymbol{v}}_k^{-\frac{1}{2}} \odot \boldsymbol{m}_k$ //scale the adaptive gradient

$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} - \eta\ \boldsymbol{\alpha}_k$ //step 2:update local estimate of $\boldsymbol{x}_k$

end

Algorithm 2: Generalized two-step decentralized adaptive min-max optimizer (DAMMO)

Input : Update rule from Table 1, η, $m \in \mathbb{N}$, $\boldsymbol{W}$, $\beta_{1,1}$, κ: if applicable

for all $i \in \mathcal{V}$, initialize

$\boldsymbol{y}_{i,0} = \boldsymbol{x}_{i,0} = \boldsymbol{\alpha}_{i,0} = \boldsymbol{m}_{i,0} = \boldsymbol{v}_{i,0} = \mathbf{0}_{d,1}$.

for $k = 1 : N$ **do**

for $i \in \mathcal{V}$ **do**

$\boldsymbol{y}_{i,k} = \sum_{j=1}^{M} w_{ij}^r(\boldsymbol{x}_{k-1} - \eta\ \boldsymbol{\alpha}_{i,k-1})$ //step 1: update local estimate of $\boldsymbol{y}_{i,k}$

$\hat{\boldsymbol{g}}_{i,k} = \frac{1}{m}\sum_{j=1}^{m} \hat{\boldsymbol{g}}_{i,k}(\boldsymbol{y}_{i,k}; \boldsymbol{\xi}_{i,k}^j)$ //estimate the stochastic gradient at $\boldsymbol{y}_{i,k}$

Adaptive update rule

$\boldsymbol{m}_{i,k} = \beta_{1,k}\boldsymbol{m}_{i,k-1} + (1-\beta_{1,k})\hat{\boldsymbol{g}}_{i,k}$ //update biased 1st moment estimate

$\hat{\boldsymbol{v}}_{i,k} = h_k(\hat{\boldsymbol{g}}_{i,1}, \hat{\boldsymbol{g}}_{i,2}, \ldots, \hat{\boldsymbol{g}}_{i,k})$ //update 2nd moment estimate

$\boldsymbol{\alpha}_{i,k} = \hat{\boldsymbol{v}}_k^{-\frac{1}{2}} \odot \boldsymbol{m}_{i,k}$ //scale the adaptive gradient

$\boldsymbol{x}_{i,k} = \sum_{j=1}^{M} w_{ij}^r(\boldsymbol{x}_{j,k-1} - \eta\ \boldsymbol{\alpha}_{i,k})$ //step 2:update local estimate of $\boldsymbol{x}_{i,k}$

end

end

$\boldsymbol{\xi}_{i,k} = (\boldsymbol{\xi}_{i,k}^1, \cdots, \boldsymbol{\xi}_{i,k}^m)$ to estimate the stochastic gradient at $\boldsymbol{y}_{i,k}$. The box shows the adaptive update rule. We update the biased first moment estimate, and then use one of the update rules in Table 1 to update the second moment estimate. We then use the second moment estimate to scale the first moment. The second step involves updating the local estimate $\boldsymbol{x}_{i,k}$, again by first locally updating the value and then aggregating the local value with the other nodes' values using the weighting matrix.

4. CONVERGENCE ANALYSIS

[1] The following assumptions commonly used in the literature of classical minimization and also min-max problems are required for the ensuing analysis.

Assumption 1. *We assume that the function $F(\cdot,\cdot)$ is Lipschitz continuous gradient in its domain and we also have access to its unbiased gradient estimator with bounded variance.*

The above assumption is commonly used in the literature and implies that there exists an oracle that can obtain an unbiased estimate of the gradient of the objective function with bounded variance.

Assumption 2 (Weak MVI)**.** *There exists a point $\boldsymbol{x}^* \in \mathcal{S}\times\mathcal{T}$ such that for any point $\boldsymbol{x} \in \mathcal{S}\times\mathcal{T}$, we have*

$$\langle \boldsymbol{g}(\boldsymbol{x}), \boldsymbol{x} - \boldsymbol{x}^* \rangle \geq -\frac{\rho}{2}\|\boldsymbol{g}(\boldsymbol{x})\|^2, \tag{4}$$

for some parameters $\rho \in [0, \frac{1}{4L})$.

This assumption has been recently proposed and studied in [13]. This can be considered as a generalization to the commonly-used MVI assumption. There are practical cases (refer to the game described in Proposition 2 in [29]) that MVI is not satisfied, while weak MVI holds true.

Assumption 3. *The network graph $\mathcal{G} = (V, E)$ is undirected and strongly connected. Additionally, its associated matrix $\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{M\times M}$ is doubly stochastic. Also, $\tau = \max\{|\lambda_2(\boldsymbol{W})|, |\lambda_M(\boldsymbol{W})|\} < 1$.*

This is a standard assumption in the literature of decentralized optimization problems and is required to obtain convergence results for both classical minimization [30, 28] and also min-max problems [18, 25].

Theorem 1 (Informal Statement)**.** *Consider the min-max game defined in* (2) *. In Algorithm 1 under update rules of Table 1 and Assumptions 1-2, by selecting $\eta = \mathcal{O}(1)$, there exists an iterate $k \in \{1, \ldots, N\}$ such that $\boldsymbol{x}_k$ is an ϵ-SFNE.*

[1] To conserve space, the clarification of assumptions with our notation, the details of the proofs and experiments, including design of neural network in Exp. 5.2 and results of experiments in 5.1-5.2 for the decentralized case, will be in arXiv version of the paper. The code will also be released on GitHub.

Theorem 2 (Informal Statement)**.** *Consider the min-max game defined in* (3) *that is optimized by Algorithm 2 using* $M = \mathcal{O}(\epsilon^{-2})$*. Under update rules of Table 1 and Assumptions 1-3, by selecting* $\eta = \mathcal{O}(1)$ *and* $r = \Omega(\log_{1/\rho}(1 + \epsilon^{-4}))$*, there exists an iterate* $k \in \{1, \dots, N\}$ *such that* $\bar{\boldsymbol{x}}_k = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{x}_{i,k}$ *is an* ϵ*-SFNE.*

In Theorem 2, we could also obtain an ϵ-SFNE using a constant number of machines but variable-sized mini-batch, i.e., $M = \mathcal{O}(1), \boldsymbol{m}_i = \mathcal{O}(\epsilon^{-2})\ \forall i \in \{1, \cdots, M\}$.[2]

5. NUMERICAL EXPERIMENTS

In this section, we evaluate the performance of our proposed generic algorithms using both synthetic and practical experiments.

5.1. Synthetic Experiment

Simultaneous methods, gradient-based or adaptive, are commonly used for solving complex min-max optimization problems in which the parameters $\boldsymbol{s}$ and $\boldsymbol{t}$ in problem (1) are updated simultaneously using a classical minimization approach. However, our proposed generic methods in Algorithms 1-2 have an ancillary variable $\boldsymbol{y}_k$ or $\boldsymbol{y}_{i,k}$ and two updating steps. Removing the update rule for these variables reduces the algorithms to the simultaneous type.

In this case study, we illustrate that this step is critical for convergence and how its removal causes the algorithms to fail to obtain an FNE for even simple min-max problems. Consider the following bi-linear stochastic min-max problem:

$$f(\boldsymbol{s}, \boldsymbol{t}; \xi) = \xi\ [\boldsymbol{s}^\intercal \mathbf{A} \boldsymbol{t} + \boldsymbol{b}^\intercal \boldsymbol{s} + \boldsymbol{c}^\intercal \boldsymbol{t} + \boldsymbol{d}]$$

in which $\xi \sim \mathcal{N}(\mu = 1.0, \sigma = 0.1)$, $\mathbf{A} = \mathbf{I} \succ 0$ and each element of $\boldsymbol{b}, \boldsymbol{c}$ and $\boldsymbol{d}$ is sampled from $\mathcal{N}(\mu = 0.0, \sigma = 1.0)$. The function $F(\boldsymbol{s}, \boldsymbol{t})$ is

$$F(\boldsymbol{s}, \boldsymbol{t}) = \boldsymbol{s}^\intercal \mathbf{A} \boldsymbol{t} + \boldsymbol{b}^\intercal \boldsymbol{s} + \boldsymbol{c}^\intercal \boldsymbol{t} + \boldsymbol{d}$$

which has a single FNE at $(\boldsymbol{s}^*, \boldsymbol{t}^*) = (-\mathbf{A}^{-1}\boldsymbol{c}, -\mathbf{A}^{-1}\boldsymbol{b})$.

To measure the performance of different algorithms, in each iteration we calculate the ℓ_2 distance of the current estimate from the optimal point, i.e., at iteration k we have $e_k = \sqrt{\|\boldsymbol{s} - \boldsymbol{s}^*\|^2 + \|\boldsymbol{t} - \boldsymbol{t}^*\|^2}$. For ease of exposition, we only use update rules 1 and 2 in Table 1 with parameters $\beta_1 = 0.1, \beta_2 = 0.99, \eta = 0.1$, but the convergence results are similar for all rules. Figure 1a) shows the performance of Algorithm 1 under updating rules 1 and 2 and compares it with their simultaneous variants. As seen in the figure, simultaneous-type methods fail to converge to the optimal solution. This shows that the intuitive generalization of the adaptive methods into their simultaneous type may fail to solve even a simple min-max problem, so development of new algorithms is required.

[2]We will discuss these cases in more detail in the arXiv version.

5.2. Real Data Experiment

Recent studies have shown that trained neural networks are not robust against adversarial attacks and that a small perturbation to the data can deteriorate the performance of neural networks dramatically [31, 32]. One of the approaches to overcome this issue and train robust neural networks is reformulating the training process to be the following min-max optimization problem [33, 8]:

$$\min_{\boldsymbol{s}} \max_{\boldsymbol{t}}\ F(\boldsymbol{s}, \boldsymbol{t}) = \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y}) \sim \mathcal{D}}[\ell(g_{\boldsymbol{s}}(\boldsymbol{x} + \boldsymbol{t}), \boldsymbol{y})] \qquad (5)$$

in which $(\boldsymbol{x}, \boldsymbol{y})$ is the data's feature vector and its label with distribution $\mathcal{D}$, $g_{\boldsymbol{s}}(\cdot)$ is the output of the neural network with parameters $\boldsymbol{s}$, $\ell(\cdot, \cdot)$ is a loss function measuring the difference between the true and predicted label, and $\boldsymbol{t}$ is the perturbation added to the data.

In this experiment, we use $\|\boldsymbol{t}\|_0 \leq \epsilon$ with $\epsilon = 0.3$. The benchmark method with theoretical guarantees for solving problem 5 relies on optimizing the inner maximization problem (up to some accuracy) for each data batch and then using the augmented data to update the parameters of the neural network [34, 8]. In order to make our proposed algorithm applicable to this problem, we run Algorithm 1 for each batch of data and project the data back in order to have $\|\boldsymbol{t}\|_0 \leq \epsilon$. For this experiment, we consider the MNIST dataset. The classifier is a neural network with two convolutional layers and two fully connected layers that maps the output to 10 different classes. We set $\eta = 0.01$, $N = 25$, $\epsilon = 0.3$, and number of epochs to 10. We set the same parameters for the benchmark method with the inner problem solved for $N = 25$ times for each batch of data before feeding it to the model. Figure 1b) compares the performance of the neural network against two commonly used attacks, projected gradient descent (PGD) [2] and fast gradient sign method (FSGM) [35], after training them using formulation 5 that is optimized using the proposed Algorithm 1 and the benchmark method. As seen from the figure, the proposed algorithm outperforms the benchmark method in training a neural network that is robust against adversarial attacks.

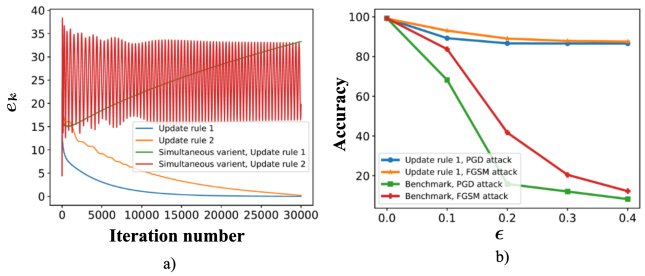

Fig. 1. a) Performance of two-step algorithms versus their simultaneous variants, b) Accuracy of the trained neural network using formulation (5) against adversarial attacks with different attack powers and fixed ϵ in the training process

6. REFERENCES

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

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

[3] Depeng Xu, Shuhan Yuan, Lu Zhang, and Xintao Wu, "Fairgan: Fairness-aware generative adversarial networks," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 570–575.

[4] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel, "Learning adversarially fair and transferable representations," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3384–3393.

[5] Meisam Razaviyayn, Tianjian Huang, Songtao Lu, Maher Nouiehed, Maziar Sanjabi, and Mingyi Hong, "Nonconvex min-max optimization: Applications, challenges, and recent theoretical advances," *IEEE Signal Processing Magazine*, vol. 37, no. 5, pp. 55–66, 2020.

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

[7] Farzan Farnia and Asuman Ozdaglar, "GANs May Have No Nash Equilibria," *arXiv preprint arXiv:2002.09124*, 2020.

[8] Maher Nouiehed, Maziar Sanjabi, Tianjian Huang, Jason D Lee, and Meisam Razaviyayn, "Solving a class of non-convex min-max games using iterative first order methods," *arXiv preprint arXiv:1902.08297*, 2019.

[9] Hamed Karimi, Julie Nutini, and Mark Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 795–811.

[10] Babak Barazandeh and Meisam Razaviyayn, "Solving non-convex non-differentiable min-max games using proximal gradient method," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3162–3166.

[11] Junchi Yang, Negar Kiyavash, and Niao He, "Global convergence and variance reduction for a class of nonconvex-nonconcave minimax problems," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[12] Mingrui Liu, Hassan Rafique, Qihang Lin, and Tianbao Yang, "First-order convergence theory for weakly-convex-weakly-concave min-max problems," *arXiv preprint arXiv:1810.10207*, 2018.

[13] Jelena Diakonikolas, Constantinos Daskalakis, and Michael Jordan, "Efficient methods for structured nonconvex-nonconcave min-max optimization," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2746–2754.

[14] Sijia Liu, Songtao Lu, Xiangyi Chen, Yao Feng, Kaidi Xu, Abdullah Al-Dujaili, Mingyi Hong, and Una-May O'Reilly, "Min-max optimization without gradients: Convergence and applications to black-box evasion and poisoning attacks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6282–6293.

[15] Zhongruo Wang, Krishnakumar Balasubramanian, Shiqian Ma, and Meisam Razaviyayn, "Zeroth-order algorithms for nonconvex minimax problems with improved complexities," *arXiv preprint arXiv:2001.07819*, 2020.

[16] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *arXiv preprint arXiv:1705.09056*, 2017.

[17] X. Lian, C. Zhang, H. Zhang, C. J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms?," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[18] Mingrui Liu Liu, Youssef Mroueh, Wei Zhang, Xiaodong Cui, Jerret Ross, and Payel Das, "Decentralized parallel algorithm for training generative adversarial nets," in *Conference on Neural Information Processing Systems*, 2020.

[19] W. Shi, Q. Ling, G. Wu, and W. Yin, "An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[20] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *Prooceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with the 28th Chinese Control Conference*. IEEE, 2009, pp. 3581–3586.

[21] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, 2012.

[22] Babak Barazandeh, Davoud Ataee Tarzanagh, and George Michailidis, "Solving a class of non-convex min-max games using adaptive momentum methods," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3625–3629.

[23] Mingrui Liu, Youssef Mroueh, Jerret Ross, Wei Zhang, Xiaodong Cui, Payel Das, and Tianbao Yang, "Towards better understanding of adaptive gradient algorithms in generative adversarial nets," *arXiv preprint arXiv:1912.11940*, 2019.

[24] Weijie Liu, Aryan Mokhtari, Asuman Ozdaglar, Sarath Pattathil, Zebang Shen, and Nenggan Zheng, "A decentralized proximal point-type method for saddle point problems," *arXiv preprint arXiv:1910.14380*, 2019.

[25] Babak Barazandeh, Tianjian Huang, and George Michailidis, "A decentralized adaptive momentum method for solving a class of min-max optimization problems," *arXiv preprint arXiv:2106.06075*, 2021.

[26] J Reddi Sashank, Kale Satyen, and Kumar Sanjiv, "On the convergence of adam and beyond," in *International Conference on Learning Representations*, 2018, vol. 5, p. 7.

[27] T Tieleman and G Hinton, "Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning," *Technical Report*, 2017.

[28] Parvin Nazari, Davoud Ataee Tarzanagh, and George Michailidis, "Adaptive first-and zeroth-order methods for weakly convex stochastic optimization problems," *arXiv preprint arXiv:2005.09261*, 2020.

[29] Constantinos Daskalakis, Dylan J Foster, and Noah Golowich, "Independent policy gradient methods for competitive reinforcement learning," *arXiv preprint arXiv:2101.04233*, 2021.

[30] Kun Yuan, Qing Ling, and Wotao Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, 2016.

[31] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al., "Adversarial examples in the physical world," 2016.

[32] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel, "Adversarial attacks on neural network policies," *arXiv preprint arXiv:1702.02284*, 2017.

[33] Aleksandr Beznosikov, Gesualdo Scutari, Alexander Rogozin, and Alexander Gasnikov, "Distributed saddle-point problems under similarity," *arXiv preprint arXiv:2107.10706*, 2021.

[34] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

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