AGE: Enhancing the Convergence on GANs using Alternating extra-gradient with Gradient Extrapolation

Huan He
Department of Computer Science
Emory University
Atlanta, GA 30033
huan.he@emory.edu

Shifan Zhao
Department of Computer Science
Emory University
Atlanta, GA 30033
shifan.zhao@emory.edu

Yuanzhe Xi
Department of Computer Science
Emory University
Atlanta, GA 30033
yuanzhe.xi@emory.edu

Joyce C Ho
Department of Computer Science
Emory University
Atlanta, GA 30033
joyce.c.ho@emory.edu

Abstract

Generative adversarial networks (GANs) are notably difficult to train since the parameters can get stuck in a local optimum. As a result, methods often suffer not only from degeneration of the convergence speed but also from limitations in the representational power of the trained network. Existing optimization methods to stabilize convergence require multiple gradient computations per iteration. We propose AGE, an alternating extra-gradient method with nonlinear gradient extrapolation, that overcomes these computational inefficiencies and exhibits better convergence properties. It estimates the lookahead step using a nonlinear mixing of past gradient sequences. Empirical results on CIFAR10, CelebA and several synthetic datasets demonstrate that the introduced approach significantly improves convergence and yields better generative models.

1 Introduction

Generative adversarial networks (GANs) have become the standard approach to generate plausible new samples [14]. Typically, GANs are trained by a generator and a discriminator in an adversarial way. Generators are trained to mimic real samples, whereas discriminators are trained to classify between real samples and fake samples drawn from the generators. While very powerful, GANs are notoriously hard to train and often fail to converge. One reason for non-convergence is cycling around the optimum or even slow outward spiraling [24, 30]. There have been attempts to improve stability and tackle this training issue by proposing new formulations of the GAN objective [3, 29, 22]. [33] uses historical averaging of both generator and discriminator parameters as a regularization term in the objective function to improve stability. Wasserstein GANs (WGANs) [3] remedy the mode collapse that appears in the standard GANs by using the Wasserstein distance.

Orthogonal to the above strategies to make the GAN-game well-defined and stable, a few studies have focused on developing specific optimization methods for GAN training. In particular, the extra-gradient method has drawn significant attention [11, 25, 10, 13]. These algorithms treat GANs as a variational inequality problem and aim to find the Nash equilibrium of a minimax problem from a game theory perspective. One key idea behind the extra-gradient method is to update the parameter using a lookahead gradient at each iteration. Although these attempts have provided better results, there is still scope to improve the convergence of GANs. Of particular interest is to accelerate the convergence of extra-gradient as it provides better guarantees. In addition, the extra-gradient method requires multiple gradient computations which are computationally expensive.

We propose *AGE (Alternating extra-gradient with Gradient Extrapolation)*, a novel alternating update scheme that can accelerate the convergence of several minimax optimization problems. Tangential to the previous approaches, it uses an adaptive gradient extrapolation procedure to replace the extrapolation step in extra-gradient by an adaptive nonlinear mixing of previous gradient sequences. This avoids multiple gradient computations while also mitigating the noise arising from gradient computations. In addition, *AGE* uses an alternating update scheme to obtain better generative models. We summarize our main contributions:

- A method to extrapolate the current gradient based on an efficient and adaptive nonlinear mixing of previous gradient sequences.
- Convergence and stability analysis of *AGE*. An asymptotic quadratic convergence rate is provided as an illustrative example and asymptotically optimal convergence rates for optimizing stochastic problems are proved in Sec 4.
- Empirical evaluation of the advantages of *AGE* on both toy settings and two real datasets. For all tasks, using *AGE* leads to improved convergence and higher-quality generative models.

Outline: We present the background on GANs and extra-gradient, and show a nonlinear extrapolation technique in Sec 2. Then we propose our method AGE in Sec 3 and give its convergence properties in Sec 4. We review the related work in Sec A.4 and provide experimental results in Sec 5.

2 Background

In this section, we first review the extra-gradient method. We then briefly describe the standard nonlinear extrapolation algorithm, which will be used to extrapolate gradient sequences.

2.1 Extra-gradient method

GANs consist of two models: a generator G_θ, that aims to generate samples from a noise distribution q_θ that best matches the true distribution p of the data, and a discriminator D_ϕ whose purpose is to classify genuine samples against generated ones, thereby training the generator. It can be seen as a saddle point problem and solved by finding the Nash equilibrium (stationary point) [25]. However, saddle point problems are known to be hard to optimize. Both simultaneous and alternating gradient methods fail to converge but instead exhibit a cycling behavior. Solving GAN can be considered as a variational inequality problem [2, 13]. First introduced in [20], extra-gradient method (EG) performs a "prediction" step to obtain an extrapolated point $(\theta_{t+\frac{1}{2}}, \phi_{t+\frac{1}{2}})$, and then applies the gradients at the extrapolated point to the current iterate (θ_t, ϕ_t) as follows:

$$\text{Extrapolation:} \begin{cases} \theta_{t+\frac{1}{2}} = \theta_t - \eta \nabla_\theta \mathcal{L}_\theta \left(\theta_t, \phi_t\right) \\ \phi_{t+\frac{1}{2}} = \phi_t + \eta \nabla_\phi \mathcal{L}_\phi \left(\theta_t, \phi_t\right) \end{cases} \tag{1}$$

$$\text{Update:} \begin{cases} \theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}_\theta \left(\theta_{t+\frac{1}{2}}, \phi_{t+\frac{1}{2}}\right) \\ \phi_{t+1} = \phi_t + \eta \nabla_\phi \mathcal{L}_\phi \left(\theta_{t+\frac{1}{2}}, \phi_{t+\frac{1}{2}}\right) \end{cases} \tag{2}$$

where η denotes the step size.

In the concept of a two-player game, extrapolation corresponds to an opponent shaping step: each player anticipates its opponent's next moves to update its strategy. For bilinear games, a slightly more generalized version was proposed in [23] with linear convergence proved. It guarantees convergence for any convex-concave function $\mathcal{L}$ and any closed convex sets [16]. It has been shown that EG converges to saddle points sublinearly for smooth convex-concave games [27].

2.2 Nonlinear Extrapolation Acceleration

Extrapolation techniques are designed to accelerating an arbitrary sequence of objects in a vector space, such as vectors, matrices or tensors. These methods perform a linear combination of previous k iterates of a given sequence to obtain one which can converge faster to the solution. One such technique for matrices is the Vector Epsilon Algorithm (VEA) [38]. The algorithm uses the generalized matrix inverse to extend the scalar ϵ-algorithm to sequences of matrices. Assuming that $\{x_t\}$ is a sequence of matrices, VEA produces an extrapolated matrix from every k consecutive iterates x_t and can be implemented by an elegant recursive formula without performing the unstable determiannt computations.

Let ϵ_{2k}^{n} denote the output of VEA algorithm at iteration n. It can be shown that ϵ_{2k}^{n} essentially represents the k-term Shanks transforms of the original sequence $\{x_t\}$:$\epsilon_{2k}^{(n)} = e_k\left(x_n\right)$, where $e_k\left(x_n\right) = \sum_{t=0}^{k} \gamma_t x_{n+t}$ with γ_t being the solution to the linear system

$$\sum_{t=0}^{k} \gamma_t \Delta x_{m+t} = 0, \quad m = n, \ldots, n+k-1, \quad \sum_{t=0}^{k} \gamma_t = 1$$

and $\Delta x_t = x_{t+1} - x_t$. When the sequence satisfies the k-term Shanks kernel exactly, VEA will yield a constant sequence equal to the limit value. In general, the extrapolated sequence has been shown to converge faster in various nonlinear problems [7, 32]. Under the assumption that the size of the epsilon table k is equal to the degree of minimal polynomial with respect to initial point during the beginning of each iteration, it can be shown that extrapolated sequence can achieve quardratic convergence rate asymptotically [36].

3 Proposed Method

3.1 Gradient extrapolation via VEA

Inspired by recent advances in parameter averaging and variance reduction schemes [1, 17], we propose an adaptive gradient extrapolation procedure on top of the VEA framework. Different from most parameter averaging schemes that use an inner-outer pullback fashion, we apply VEA on gradient sequences directly during the training procedure. This framework also mitigates the noise arising from gradient computation over different batches.

Assuming that $\{x_t\}$ is a sequence of gradient matrices, VEA produces an extrapolated gradient from every k consecutive iterates x_t in the following manner:

$$\begin{aligned} &\epsilon_{-1}^{(t)} = 0, \epsilon_{0}^{(t)} = x_t \\ &\epsilon_{k+1}^{(t)} = \epsilon_{k-1}^{(t+1)} + \frac{(\epsilon_{k}^{(t+1)} - \epsilon_{k}^{(t)})}{\left\|\epsilon_{k}^{(t+1)} - \epsilon_{k}^{(t)}\right\|_F^2} \quad \text{for } k > 0. \end{aligned} \tag{3}$$

The final output of those sequences defined in ϵ-algorithm equivalently represents the k-term Shanks transforms of the original sequence $\{x_t\}$ as shown in (2.2) [6]. As soon as a new gradient $\mathbf{x}_t$ becomes available, we can immediately compute $\epsilon_1^{(t-1)}, \epsilon_2^{(t-2)}, \ldots$ based on (3). The final output will be used as the extra-gradient in the next iteration. Since the procedure is highly parallel and is dominated by dense matrix/vector operations, this extrapolation can be effectively implemented on GPUs. [34] showed that the complexity of VEA is $O(Nk^2)$, in which N is the parameter size and k is the sequence length (set as 3 in all experiments). The relationship between ϵ-algorithm and Shanks transformation and implementation details can be found in the Appendix.

The relation between the gradient extrapolation via VEA and the one-step linear extrapolation proposed in [13] is shown in the following theorem.

Theorem 3.1. *Assume x_i's are the gradient sequence generated by the extra-gradient method. Then we have $e_0(x_{i-1}) = x_i$.*

This theorem shows that when k is set to be 0, the gradient extrapolation via VEA is equivalent to alternating gradient descent with one-step linear extrapolation. For general k, we can show that the gradient extrapolation via VEA leads to faster convergence than the linear extrapolation extra-gradient method in Sec 4.

Algorithm 1 AGE

```
1:  Given $\theta, \phi$, maximum iteration number $T$and learning rate $\eta$
2:  Set $\gamma = 0.5$. Build a empty list for $\theta$ and $\phi$, $A_\theta, A_\phi$
3:  Let $VEA(x_t)$ denote $e_k(x_{n-k-1})$.
4:  **while** $t < T$ **do**
5:      Use VEA to extrapolate gradients of $\theta$ and $\phi$ at $t$ with the length size $k$:
6:      $\nabla_\theta \mathcal{L}_\theta^* (\theta_t, \phi_t) = VEA(\nabla_\theta \mathcal{L}_\theta \left(\theta_{t-\frac{1}{2}}, \phi_{t-\frac{1}{2}}\right))$
7:      $\nabla_\phi \mathcal{L}_\phi^* (\theta_t, \phi_t) = VEA(\nabla_\phi \mathcal{L}_\phi \left(\theta_t, \phi_{t-\frac{1}{2}}\right))$
8:      $\eta_\theta = \min\{\eta, \frac{\gamma\eta}{\|\nabla_\theta \mathcal{L}_\theta^*(\theta_t, \phi_t)\|}\}$
9:      $\eta_\phi = \min\{\eta, \frac{\gamma\eta}{\|\nabla_\phi \mathcal{L}_\phi^*(\theta_t, \phi_t)\|}\}$
10:     $\theta_{t+\frac{1}{2}} = \theta_t - \eta_\theta \nabla_\theta \mathcal{L}_\theta^* (\theta_t, \phi_t)$
11:     $\phi_{t+\frac{1}{2}} = \phi_t + \eta_\phi \nabla_\phi \mathcal{L}_\phi^* (\theta_t, \phi_t)$
12:     Calculate gradient of $\theta_{t+\frac{1}{2}}$ using $\theta_{t+\frac{1}{2}}$ and $\phi_{t+\frac{1}{2}}$
13:     $\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}_\theta \left(\theta_{t+\frac{1}{2}}, \phi_{t+\frac{1}{2}}\right)$
14:     Store $\nabla_\theta \mathcal{L}_\theta \left(\theta_{t+\frac{1}{2}}, \phi_{t+\frac{1}{2}}\right)$
15:     Calculate gradient of $\phi_{t+\frac{1}{2}}$ using $\theta_{t+1}$ and $\phi_{t+\frac{1}{2}}$
16:     $\phi_{t+1} = \phi_t + \eta \nabla_\phi \mathcal{L}_\phi \left(\theta_{t+1}, \phi_{t+\frac{1}{2}}\right)$
17:     Store $\nabla_\phi \mathcal{L}_\phi \left(\theta_{t+1}, \phi_{t+\frac{1}{2}}\right)$
18: **end while**
19: **return** $\theta$ and $\phi$
```

3.2 AGE

AGE combines the gradient extrapolation technique proposed in the previous section with the extra-gradient method for solving the minimax problem. First, we replace the extrapolation step in extra-gradient with gradient extrapolation. After we obtain the extrapolated gradient using (2.2), we use it to update model parameters with the same learning rate as the baseline optimizer. Different from extra-gradient method where the parameters are updated simultaneously as shown in (2), we propose an alternating extra-gradient updating scheme. That is, after the gradient extrapolation, we update the parameters in an alternating way since well-performing GAN generators are closer to a saddle-point instead of a local minimum, which suggests that the local Nash, the typical solution concept for simultaneous games, may not be the most appropriate one for GANs [5].

The *AGE* algorithm is summarized in Algorithm 1. Lines 4-6 perform gradient extrapolation where gradients computed up to iteration $t-1$ (Lines 12 and 15) are used by VEA to estimate the gradients at iteration t. Note that when two consecutive gradient sequences are too close, VEA may encounter numerical stability issues. To resolves these issues, clipping (Lines 7-8) is used as a safeguard [41]. An extrapolation step is performed on the estimated (clipped) gradients in Lines 9-10. We then calculate the gradient of $\theta_{t+\frac{1}{2}}$ in Line 11 and the gradient of $\phi_{t+\frac{1}{2}}$ using the latest θ_t in Line 14. We repeat this procedure until convergence or the max iteration number is reached. Note that the gradient extrapolation step in Lines 5-6 has a constant computational overhead by fixing the length of the VEA sequence for each parameter. The latest gradient information is discarded once VEA extrapolation is completed.

4 Theoretical analysis

In this section, we demonstrate the convergence properties of the proposed method (*AGE*) for solving several minmax optimization problems with detailed comparisons to the extra-gradient method [13].

4.1 Uni-dimensional game

We first consider the following simple model problem:

$$\min_{\theta \in \mathbb{R}} \max_{\phi \in \mathbb{R}} \theta \cdot \phi \tag{4}$$

This toy example has been extensively studied for illustrating the properties of the methods of interest [13, 8, 26]. We compare several optimization methods on solving this problem and plot the results in Fig 1. The plots show that extra-gradient methods are more preferable for solving this kind of problem. In particular, *AGE* shows a significant improvement of convergence *over all methods*.

Proposition 4.1. *The squared norm N_t^2 for one dimensional min-max problem solved by alternating extra-gradient has the convergence rate $N_{t+1}^2 = O(1-\eta^2)N_t^2$.*

The above proposition shows the convergence rate of alternating extra-gradient without VEA. Next, we analyze the convergence rate of AGE for solving (4). Applying Algorithm 1 to solve (4) leads to the following updating rule:

$$\text{Extrapolation:} \begin{cases} \theta_{t+\frac{1}{2}} = \theta_t - \eta_\theta VEA(\phi_t) \\ \phi_{t+\frac{1}{2}} = \phi_t + \eta_\phi VEA(\theta_t) \end{cases} \quad \text{Update:} \begin{cases} \theta_{t+1} = \theta_t - \eta\phi_{t+\frac{1}{2}} \\ \phi_{t+1} = \phi_t + \eta\theta_{t+1} \end{cases} \tag{5}$$

Note that applying VEA on a separable minimax problem like (4) is equivalent to applying VEA only on one variable θ. Notice that the Nash equilibrium of this problem is $(\theta^*, \phi^*) = (0, 0)$. Denote $N_t^2 \overset{\text{def}}{=} \theta_t^2 + \phi_t^2$ as the squared two norm of the gradient sequence. The convergence rate can be monitored by the magnitude of N_t^2 and is analyzed in the next theorem.

Theorem 4.1. *When t is large enough, N_t^2 computed based on the updating rule* (5) *decreases for any $\eta < 1$ as*

$$N_{t+1}^2 = O(1-\eta^2)N_t^2 \tag{6}$$

In the next theorem, we show that the extrapolated sequence $e_k(\theta_t)$ has the asymptotic quadratic convergence rate.

Theorem 4.2. *When (θ_t, ϕ_t) is near the optimum (θ^*, ϕ^*), $e_k(\theta_t)$ has the convergence rate*

$$|e_k(\theta_t)| = O(|e_k(\theta_{t-1})|^2).$$

Finally we can show that AGE can achieve quadratic convergence in certain cases.

Proposition 4.2. *Suppose θ_t, ϕ_t and $e_k(\theta_t)$ have the same order. Then quadratic convergence for gradient norm $N_t = C_t N_{t-1}^2$ can be achieved for some bounded constant C_t.*

[13] analyzes the convergence rates of two extra-gradient methods for solving (4). These results are are summarized in the next theorem to facilitate the theoretical comparison.

Theorem 4.3 (EG [13]). *N_t^2 computed based on the following two updating rules decreases geometrically for any $0 < \eta < 1$ as*

$$\begin{aligned} &\textit{Implicit}: N_{t+1}^2 = \left(1 - \eta^2 + \eta^4 + \mathcal{O}\left(\eta^6\right)\right) N_t^2, \\ &\textit{Extrapolation}: N_{t+1}^2 = \left(1 - \eta^2 + \eta^4\right) N_t^2, \quad \forall t \geq 0. \end{aligned}$$

A comparison among the convergence rates of these methods further confirms that the proposed method shown in Algorithm 1 can be faster than standard extra-gradient methods for solving (4) when t is large.

4.2 Bilinear problem and Stochastic optimization

Next, we study the convergence property of the proposed method on a more general unconstrained bilinear problem:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^l} \max_{\boldsymbol{\varphi} \in \mathbb{R}^m} \boldsymbol{\theta}^\top \boldsymbol{A} \varphi - \boldsymbol{b}^\top \boldsymbol{\theta} - \boldsymbol{c}^\top \boldsymbol{\varphi} \tag{7}$$

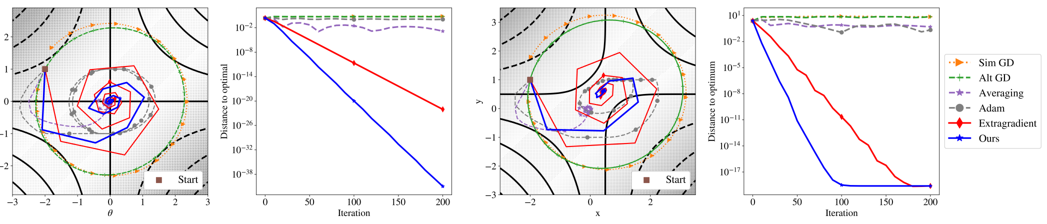


Figure 1: **Left:** A simple example of a minimax problem as described in Sec 4.1: $\min_\theta \max_\phi \theta \cdot \phi$. **Right:** Another nontrival two-dimensional quadratic problem $\min_x \max_y = (x - \frac{1}{2})(y - \frac{1}{2}) + \frac{e^{(-(x-\frac{1}{4})^2-(y-\frac{3}{4})^2)}}{3}$, which is arguably the simplest nontrivial problem. For each problem, the left subplot depicts the trajectory of our method and other algorithms in low dimensional toy problems. The right subplot shows the distance to the Nash equilibrium $(0, 0)$ and $(0.4028, 0.5972)$ for each method using the best learning rate respectively. We can observe that our method significantly improves the convergence rate of extra-gradient methods on two illustrative examples.

where $\boldsymbol{A} \in \mathbb{R}^{l\times m}$, $\boldsymbol{b} \in \mathbb{R}^l$ and $\boldsymbol{c} \in \mathbb{R}^m$. Assuming the existence of the optimal solution allows us to rewrite the problem as

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^l} \max_{\boldsymbol{\varphi}\in\mathbb{R}^m} \left(\boldsymbol{\theta} - \boldsymbol{\theta}^*\right)^\top \boldsymbol{A}\left(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*\right) + \boldsymbol{c}$$

where $\boldsymbol{c} := -\boldsymbol{\theta}^{*\top}\boldsymbol{A}\boldsymbol{\varphi}^*$ is a constant that does not depend on $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$. If we define $\hat{N}_t^2 := \operatorname{dist}\left(\boldsymbol{\theta}_t, \Theta^*\right)^2 + \operatorname{dist}\left(\boldsymbol{\varphi}_t, \Phi^*\right)^2$, where (Θ^*, Φ^*) is the solution, we can obtain the following estimate on the bound of $\hat{N}_t^2$ for the above bilinear unconstrained problem.

Theorem 4.4. *When t is large enough, for any $0 < \eta < \frac{1}{\sigma_{\max}(\boldsymbol{A})}$ we have*

$$\hat{N}_{t+1}^2 < O\left(1 - \left(\sigma_{\min}(\boldsymbol{A})\eta\right)^2\right)\hat{N}_t^2, \quad \forall t \geq 0$$

Particularly, for $\eta = \frac{1}{2\sigma_{\max}(\boldsymbol{A})}$, $\hat{N}_{t+1}^2 < O\left(1 - \frac{1}{4\kappa}\right)^t \hat{N}_0^2, \quad \forall t \geq 0$, where $\sigma_{\max}$ and $\sigma_{\min}$ are the maximal and minimal singular values of A, respectively, and $\kappa := \frac{\sigma_{\max}(\boldsymbol{A})^2}{\sigma_{\min}(\boldsymbol{A})^2}$ is the condition number of $\boldsymbol{A}^\top\boldsymbol{A}$.

A similar convergence rate has been proved in Corollary 1 of [13] for extra-gradient methods as

$$\hat{N}_{t+1}^2 \leq \left(1 - \left(\sigma_{\min}(\boldsymbol{A})\eta\right)^2 + \left(\sigma_{\min}(\boldsymbol{A})\eta\right)^4\right)\hat{N}_t^2, \quad \forall t \geq 0.$$

Thus, AGE can converge faster on this unconstrained bilinear problem than extra-gradient methods when t is large.

5 Experiments

In this section, we investigate whether the theoretical guarantees of *AGE* carry over from simple to practical problems. Particularly, our experiments have three main aims: (1) to test if *AGE* converges to local minimax and improves the convergence rate compared to extra-gradient methods, (2) to test the capability of *AGE* to address the notorious mode collapse problem in GAN training, (3) to test the effectiveness of generating images in GANs via *AGE*.

5.1 Motivating examples

First, we verify our claim on improved local convergence rates as suggested by Proposition 4.2. We run *AGE* on the simple low-dimensional problems in (4). It is a simple two-dimensional problems, yet traditional gradient methods (e.g., Sim GD and Alt GD) fail to converge to the Nash equilibrium. The results are shown in Figure 1, which empirically confirms the improved convergence rate of *AGE*.

Table 1: Averaged best Wasserstein-1 (W-1) distance under different learning rates (LR) over 5 runs for GAN trained on mixture of Gaussians.

LR	SGD	ExtraSGD	*AGE-SGD*
1×10^{-3}	0.75 ± 0.018	0.73 ± 0.020	0.32 ± 0.035
5×10^{-3}	0.21 ± 0.042	0.18 ± 0.041	$\mathbf{0.08} \pm \mathbf{0.015}$

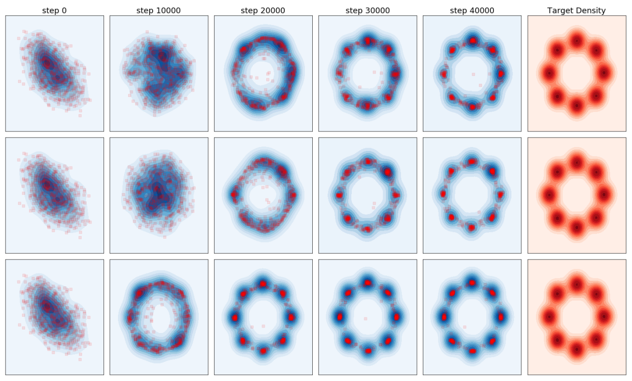
5.2 Generative adversarial networks

One particularly promising application of minimax optimization algorithms is training GANs. The goal in this experimental section is to show that our proposed method achieves faster convergence of GANs and its capability of yielding better generative models across different tasks. It is important to note that our method will not provide new state-of-the-art results with architectural improvements or a new GAN formulation. We demonstrate our method on mixtures of Gaussians, the CIFAR10 dataset, and the CelebA dataset. All experiments were run on a NVIDIA V100 GPU.

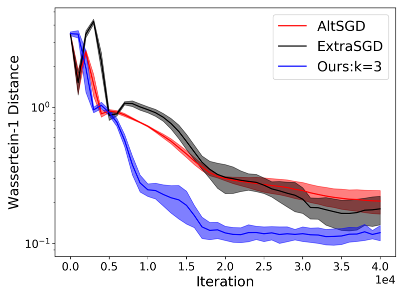
5.2.1 Mixture of Gaussians

To test our proposed method, we evaluate our method on a simple 2-D example from a mixture of 8 Gaussians with standard deviations equal to 1×10^{-2} and modes uniformly distributed around the unit circle. For both generator and discriminator, we use fully connected neural networks with 3 hidden layers and 64 hidden units in each layer. Except for the output layer of discriminator that uses a sigmoid activation, we use tanh-activation for all other layers. We use an 8-dimensional Gaussian prior and the original GAN objective function as in [14].

The details of the experiment are as follows. We run alternating SGD, Extra-gradient (denoted *ExtraSGD*) [13], and *AGE* with SGD for 40000 steps since simultaneous SGD suffers from mode collapse [4]. The learning rate is set as 5×10^{-3} after an extensive grid search which is close to the maximal possible stepsize under which the methods rarely diverge. Fig 2a shows the output after $\{0, 10K, 20K, 30K, 40K\}$ iterations. It can be seen that our method converges faster to the target distribution without oscillation (see step 10K) and offers a significant improvement over baselines.



(a) Evolution plot



(b) W-1 distance

Figure 2: Left:From top to bottom: SGD, ExtraSGD, *AGE-SGD*. The images depict from left to right the resulting densities (in blue) and generated samples (in red) of the algorithm after $0, 10000, 20000, 30000$ and 40000 iterations as well as the target density (in red). The former two methods performed extremely similarly, although when zooming it should be clear that Extra-SGD converges better. On the other hand, our method converges fastest and leads to the best result. The generated samples using our method gather around the circle and are less connected with other circles. Right: Mean and standard deviation of the Wasserstein-1 distance (the lower the better) computed over 5 runs for each method on GAN trained on mixture of Gaussians.

Table 2: Best inception scores achieved on CIFAR10 (averaged over 5 runs) for different learning rates.

Learning rate		Method	
η_{dis}	η_{gen}	ExtraAdam	*AGE-Adam*
1×10^{-4}	1×10^{-5}	$7.65 \pm .09$	$\mathbf{7.85 \pm .08}$
3×10^{-4}	3×10^{-5}	$7.82 \pm .08$	$\mathbf{8.21 \pm .06}$
5×10^{-4}	5×10^{-5}	$7.90 \pm .11$	$\mathbf{7.98 \pm .09}$

Similar to [40], we measure the Wasserstein-1 distance[1], a metric of the difference between two distributions [12], with 2048 samples from the original 8-dimensional Gaussian mixture and the estimated mixture distributions at every 1000 iterations. The averaged Wasserstein-1 distance curves of each method are shown in Fig 2b. We can observe that our method is closer to the true Gaussian mixture model (i.e., better results) and also converges faster than the baselines.

Since different learning rates can impact the convergence, we also run the 3 methods with a less aggressive learning rate 1×10^{-3}. Table 1 reports the averaged best Wasserstein-1 distance achieved for each method over 5 runs.

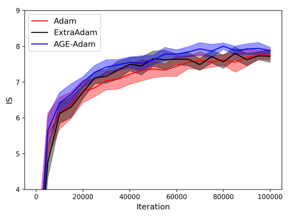
The results demonstrate that our method outperforms SGD and ExtraSGD for both learning rates.

5.2.2 CIFAR-10

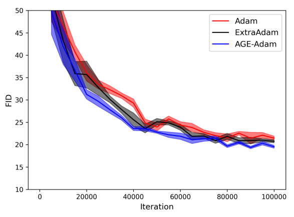
In our second experiment, we apply our method to the CIFAR10 dataset [21] and use the ResNet architecture with the WGAN-GP objective as in [15, 13]. We compare with Adam [19] and ExtraAdam [13] which offers significant improvements over OMD [25, 10].

Since the loss serves as an ambiguous metric for method comparison, we use the Inception score [33] and FID [18] as the performance measure for generative models. The Inception score (IS) and FID are computed on 50,000 samples every 5,000 iterations. Experiments were run with 5 random seeds for 100,000 updates of the generator. For all methods, we did an extensive search over the hyperparameters of Adam. It turns out the best performing learning rate is $\eta_{dis} = 2 \times 10^{-4}, \eta_{gen} = 2 \times 10^{-4}$. We set $\beta_1 = 0.0, \beta_2 = 0.9$ for Adam and $\lambda = 10$ for gradient penalty.

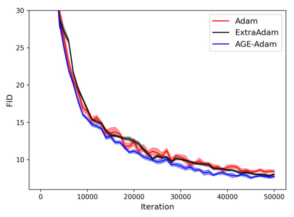
Figure 3a and 3b depict the IS and FID obtained by each method. We note that our method converges to a better FID (from *21.45* to *19.32*) and IS (from *7.76* to *8.02*). Furthermore, we observe *AGE-Adam* exhibits smaller variance during training. We postulate the gradient obtained by gradient extrapolation is less noisy.



(a) Inception Score for CIFAR10



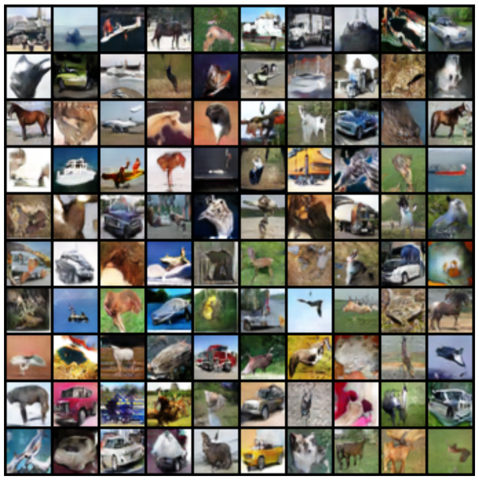
(b) FID for CIFAR10



(c) FID for CelebA

To understand the impact of learning rates, we ran *AGE-Adam* and AGE-Adam with three different learning rates as reported in Table 2. We also conduct experiments using a more common training technique, TTUR [18]. It uses different learning rates for discriminator and generator. Experiments were run with 5 random seeds for 500,000 updates of the generator. The results indicate that *AGE-Adam* generally yields better IS than ExtraAdam regardless of the learning rate. A comparison of wall-clock time of the two methods also support the claim that *AGE* improves the training as it achieves a lower wall-clock time (16 hours to 22 hours).

[1]https://pythonot.github.io/


(a) CIFAR10

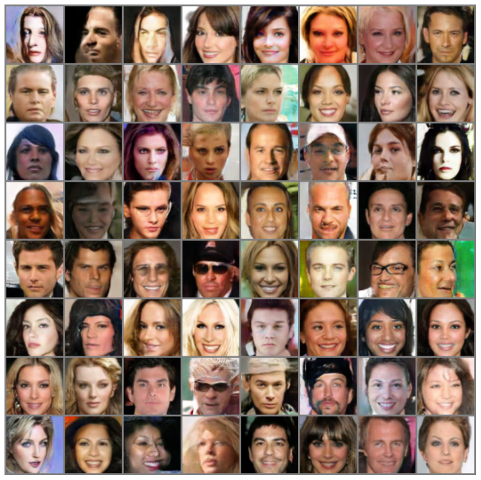

(b) CelebA

Figure 4: Random samples from a generator trained with the WGAN-GP objective using *AGE-Adam*

5.2.3 CelebA

We also test our method's effectiveness on the cropped CelebA data (64×64) which is a higher-resolution image dataset compared to CIFAR10. We use the ResNet architecture with the WGAN-GP objective as described previously.

Since the Inception score serves as an less informative metric for CelebA, we use FID [18] as the performance measure for generative models. The FID is computed on 50,000 samples every 1,000 iterations. The pre-calculated statistics is computed on the cropped images for the purpose of FID evaluation. Experiments were run with 5 random seeds for 50,000 updates of the generator. For all methods, we did an extensive search over the hyperparameters and use the same setting used for CIFAR-10 as it still provides best performance. Figure 3c depicts the FID obtained by each method and Figure 4b shows the generated images using *AGE-Adam*. It can be observed that *AGE-Adam* still outperforms the baselines as our method achieves a lower FID (from 8.2 to 7.88).

6 Conclusion

We have proposed a new way to extrapolate gradient using *AGE* that enhances the convergence rate of adversarial training. Because *AGE* is based on mixing past gradient information nonlinearly, it can avoid noise in the gradient estimate. We have also proven the fast convergence rate of *AGE* on a motivating example and bilinear problems. We empirically show better performance using *AGE* on both toy datasets and a real-world deep learning setting.

References

[1] Zeyuan Allen-Zhu. "Katyusha: the first direct acceleration of stochastic gradient methods". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing* (2017).

[2] S. Antman. "The influence of elasticity on analysis: Modern developments". In: *Bulletin of the American Mathematical Society* 9 (1983), pp. 267–291.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein Generative Adversarial Networks". In: *ICML*. 2017, pp. 214–223.

[4] D. Balduzzi et al. "The Mechanics of n-Player Differentiable Games". In: *ArXiv* (2018).

[5] Hugo Berard et al. "A Closer Look at the Optimization Landscapes of Generative Adversarial Networks". In: *ArXiv* abs/1906.04848 (2020).

[6] C. Brezinski, M. Redivo-Zaglia, and Y. Saad. "Shanks Sequence Transformations and Anderson Acceleration". In: *SIAM Rev.* 60 (2018), pp. 646–669.

[7] P. Brown and Y. Saad. "Hybrid Krylov Methods for Nonlinear Systems of Equations". In: *SIAM J. Sci. Comput.* 11 (1990), pp. 450–481.

[8] Tatjana Chavdarova et al. "Reducing Noise in GAN Training with Variance Reduced Extragradient". In: *NeurIPS*. 2019.

[9] Tatjana Chavdarova et al. "Taming GANs with Lookahead". In: *ArXiv* (2020).

[10] Constantinos Daskalakis et al. "Training GANs with Optimism". In: *ICLR*. 2018.

[11] Carles Domingo Enrich et al. "Extragradient with player sampling for faster Nash equilibrium finding". In: *ICML*. 2020, pp. 4736–4745.

[12] Aude Genevay, G. Peyré, and M. Cuturi. "Learning Generative Models with Sinkhorn Divergences". In: *AISTATS*. 2018.

[13] Gauthier Gidel et al. "A Variational Inequality Perspective on Generative Adversarial Networks". In: *ICLR*. 2019.

[14] Ian J. Goodfellow et al. "Generative Adversarial Nets". In: *NIPS*. 2014, pp. 2672–2680.

[15] Ishaan Gulrajani et al. "Improved Training of Wasserstein GANs". In: *NIPS*. 2017.

[16] P. Harker and J. Pang. "Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications". In: *Mathematical Programming* 48 (1990), pp. 161–220.

[17] Huan He, Yuanzhe Xi, and Joyce C Ho. "Fast and Accurate Tensor Decomposition without a High Performance Computing Machine". In: *2020 IEEE International Conference on Big Data*. 2020.

[18] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. 2017.

[19] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* (2015).

[20] G. M. Korpelevich. "The extragradient method for finding saddle points and other problems". In: 1976.

[21] A. Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: 2009.

[22] Chun-Liang Li et al. "MMD GAN: Towards Deeper Understanding of Moment Matching Network". In: *NIPS*. 2017, pp. 2200–2210.

[23] Tengyuan Liang and J. Stokes. "Interaction Matters: A Note on Non-asymptotic Local Convergence of Generative Adversarial Networks". In: *AISTATS*. 2019.

[24] Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. "Cycles in Adversarial Regularized Learning". In: *SODA*. 2018, pp. 2703–2717.

[25] Panayotis Mertikopoulos et al. "Optimistic Mirror Descent in Saddle-Point Problems: Going the Extra (Gradient) Mile". In: *ICLR*. 2019.

[26] Lars M. Mescheder, Sebastian Nowozin, and Andreas Geiger. "The Numerics of GANs". In: *NIPS*. 2017.

[27] Aryan Mokhtari, A. Ozdaglar, and S. Pattathil. "Proximal Point Approximations Achieving a Convergence Rate of $O(1/k)$ for Smooth Convex-Concave Saddle Point Problems: Optimistic Gradient and Extra-gradient Methods". In: *ArXiv* abs/1906.01115 (2019).

[28] A. Nedic and A. Ozdaglar. "Subgradient Methods for Saddle-Point Problems". In: *Journal of Optimization Theory and Applications* 142 (2009), pp. 205–228.

[29] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. "f-GAN: Training Generative Neural Samplers Using Variational Divergence Minimization". In: *NIPS*. 2016, pp. 271–279.

[30] Christos Papadimitriou and Georgios Piliouras. "From Nash Equilibria to Chain Recurrent Sets: Solution Concepts and Topology". In: *ITCS*. 2016.

[31] L. Popov. "A modification of the Arrow-Hurwicz method for search of saddle points". In: *Mathematical notes of the Academy of Sciences of the USSR* 28 (1980), pp. 845–848.

[32] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[33] Tim Salimans et al. "Improved Techniques for Training GANs". In: *NIPS*. 2016.

[34] A. Sidi, W. Ford, and D. Smith. "Acceleration of convergence of vector sequences". In: *SIAM Journal on Numerical Analysis* 23 (1986), pp. 178–196.

[35] Avram Sidi. *Vector extrapolation methods with applications*. SIAM, 2017.

[36] David A Smith, William F Ford, and Avram Sidi. “Extrapolation methods for vector sequences”. In: *SIAM review* 29.2 (1987), pp. 199–233.

[37] Yuanhao Wang, Guodong Zhang, and Jimmy Ba. “On Solving Minimax Optimization Locally: A Follow-the-Ridge Approach”. In: *ArXiv* abs/1910.07512 (2020).

[38] Peter Wynn. “Acceleration techniques for iterated vector and matrix problems”. In: *Mathematics of Computation* 16.79 (1962), pp. 301–322.

[39] Peter Wynn. “On a device for computing the e m (S n) transformation”. In: *Mathematical Tables and Other Aids to Computation* (1956), pp. 91–96.

[40] Yasin Yazici et al. “The Unusual Effectiveness of Averaging in GAN Training”. In: *ArXiv* (2019).

[41] J. Zhang et al. “Why Gradient Clipping Accelerates Training: A Theoretical Justification for Adaptivity”. In: (2020).

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes] See 1
   (b) Did you describe the limitations of your work? [Yes] See 3.2
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] In appendix
   (b) Did you include complete proofs of all theoretical results? [Yes] In appendix
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Algorithm 2 Extrapolation of gradient sequence

```
1: Set a window size $k$, 3 in this work. For the first time: Initialize a table $\mathbf{T}$ with $2k+1$ columns
2: **Input**: The latest gradient matrix $\Delta\mathbf{A}_i$, current iteration number $i$, table $\mathbf{T}$
3: **Output:** Extrapolated gradient $\mathbf{E}$, table $\mathbf{T}$
4: $j=1, \mathbf{z}=0$ and an empty array $\mathbf{Y}$
5: Set $\mathbf{Y}(:,1)=\Delta\mathbf{A}_i$
6: **while** $j<2k+1$ and $j<i$ **do**
7:     Compute $\triangle\varepsilon=\mathbf{Y}(:,j)-\mathbf{T}(:,j)$
8:     Calculate $\mathbf{z}=\mathbf{z}+\triangle\varepsilon/\left\|\triangle\varepsilon\right\|_F^2$
9:     $\mathbf{Y}(:,j+1)=\mathbf{z}$
10:    Reset $\mathbf{z}=\mathbf{T}(:,j)$
11:    $j=j+1$
12: **end while**
13: Set $\mathbf{T}=\mathbf{Y}$ and $\mathbf{E}=\mathbf{T}(:,2k+1)$
14: **if** $i\leq 2k$ and $j$ mod 2 == 1 **then** {when $i$ is too small}
15:     $\mathbf{E}=\mathbf{E}/\left\|\mathbf{E}\right\|_F^2$
16: **end if**
17: **return** $\mathbf{E}$, $\mathbf{T}$
```

A Appendix

A.1 Detailed algorithm for Vector Epsilon Algorithm (VEA)

In Sec 3, we proposed a gradient extrapolation technique that adopts the vector epsilon algorithm to extrapolate gradient and achieves faster convergence in the context of extra-gradient method. A general form of vector epsilon algorithm is shown in Eq 2.2. As it can be quite effective if we implement it correctly, we present a brief summarization of efficient implementation in Algorithm 1.

A.2 Relationship between Shanks Transformation and VEA

The relationship between VEA and Shanks transformation is illustrated in the following theorem:

Theorem A.1 ([39]). *Let $\{x_m\}$ be an arbitrary sequence. Then, for any two integers n and k, the Shanks transformation produces an approximation to the limit or antilimit of this sequence that is given by*

$$e_k\left(x_n\right)=\sum_{i=0}^{k}\gamma_i x_{n+i}$$

with the γ_i being the solution to the linear system

$$\sum_{i=0}^{k}\gamma_i\Delta x_{m+i}=0,\quad m=n,n+1,\ldots,n+k-1,\quad \Delta x_{m+i}=x_{m+i+1}-x_{m+i},\quad \sum_{i=0}^{k}\gamma_i=1$$

and $e_k\left(x_n\right)$ has the determinant representation

$$e_k\left(x_n\right)=\frac{\begin{vmatrix} x_n & x_{n+1} & \ldots & x_{n+k} \\ \Delta x_n & \Delta x_{n+1} & \ldots & \Delta x_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta x_{n+k-1} & \Delta x_{n+k} & \ldots & \Delta x_{n+2k-1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \ldots & 1 \\ \Delta x_n & \Delta x_{n+1} & \ldots & \Delta x_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta x_{n+k-1} & \Delta x_{n+k} & \ldots & \Delta x_{n+2k-1} \end{vmatrix}}$$

A.3 Theoretical analysis

In this section, we demonstrate the convergence properties of the proposed method (*AGE*) for solving several minmax optimization problems with detailed comparisons to the extra-gradient method [13].

A.3.1 Uni-dimensional game

We first consider the following simple model problem:

$$\min_{\theta\in\mathbb{R}}\max_{\phi\in\mathbb{R}} \theta \cdot \phi \tag{8}$$

Proposition A.1. *The squared norm N_t^2 for one dimensional min-max problem solved by alternating extra-gradient has the convergence rate $N_{t+1}^2 = O(1-\eta^2)N_t^2$.*

The above proposition shows the convergence rate of alternating extra-gradient without VEA. Next, we analyze the convergence rate of AGE for solving (8). Applying Algorithm 1 to solve (8) leads to the following updating rule:

$$\text{Extrapolation:} \begin{cases} \theta_{t+\frac{1}{2}} = \theta_t - \eta_\theta VEA(\phi_t) \\ \phi_{t+\frac{1}{2}} = \phi_t + \eta_\phi VEA(\theta_t) \end{cases} \tag{9}$$

$$\text{Update:} \begin{cases} \theta_{t+1} = \theta_t - \eta\phi_{t+\frac{1}{2}} \\ \phi_{t+1} = \phi_t + \eta\theta_{t+1} \end{cases} \tag{10}$$

Note that applying VEA on a separable minimax problem like (8) is equivalent to applying VEA only on one variable θ. Notice that the Nash equilibrium of this problem is $(\theta^*, \phi^*) = (0, 0)$. Denote $N_t^2 \overset{\text{def}}{=} \theta_t^2 + \phi_t^2$ as the squared two norm of the gradient sequence. The convergence rate can be monitored by the magnitude of N_t^2 and is analyzed in the next theorem.

Theorem A.2. *When t is large enough, N_t^2 computed based on the updating rule* (9)-(10) *decreases for any $\eta < 1$ as*

$$N_{t+1}^2 = O(1-\eta^2)N_t^2 \tag{11}$$

Proof Sketch. When t is large enough, the iterations satisfy the following equation

$$\begin{bmatrix}\theta_{t+1}\\ \phi_{t+1}\end{bmatrix} = \begin{bmatrix}1-\eta^2 & -\eta \\ \eta-\eta^2 & 1-\eta^2\end{bmatrix}\begin{bmatrix}\theta_t\\ \phi_t\end{bmatrix} + \begin{bmatrix}\eta^2\\ \eta^3\end{bmatrix}(\theta_t - e_k(\theta_{t-k-1})). \tag{12}$$

We can show that, $\theta_t \sim O(1-\eta^2)^t$ and $\theta_t - e_k(\theta_{t-k-1}) \sim o\left(\eta^{2(k+1)(t-k-1)}\right) \sim o(\theta_t)$. Thus for t large enough, the convergence rate is at least $N_{t+1}^2 = O(1-\eta^2)N_t^2$. □

In the next theorem, we show that the extrapolated sequence $e_k(\theta_t)$ has the asymptotic quadratic convergence rate.

Theorem A.3. *When (θ_t, ϕ_t) is near the optimum (θ^*, ϕ^*), $e_k(\theta_t)$ has the convergence rate*

$$|e_k(\theta_t)| = O(|e_k(\theta_{t-1})|^2).$$

Proof Sketch. This proof is mainly based on Sec 8 of [36]. Firstly, we can show the error in the intial data will propagate linearly using epsilon algorithm. We decompose the computed value into

$$e_k^*(x_n) = e_k(x_n) + \epsilon_k^n, \tag{13}$$

where $e_k(x_n)$ is the value extrapolated using $\{x_i\}_{i=n-2k-1}^n$. ϵ_k^n is the error in the computation of last entry in epsilon table defined in (2.2). Thus the errors in the initial data is given by $\|\epsilon_0^n\| = \|x_n^* - x_n\|$. Define $\|\epsilon\| = \max_{0\le n\le 2k}\|\epsilon_0^n\|$,where k is the degree of the minimal polynomial of iteration law with respect to $x_j - x$, it implies the following recursion formula $\sum_{i=0}^k c_i x_i = (\sum_{i=0}^k c_i)x$, where x is the limit of the sequence. Then by induction we can show that

$$\|\epsilon_k^n\| = O(\|\epsilon\|) \tag{14}$$

for all k, n if the epsilon table exits. In fact, this is true for ϵ_{2k}^n which is the error for the extrapolated value we will use from the epsilon table. According to the VEA algorithm, the computed entries of epsilon table satisfy

$$e_k^*(x_n) = e_{k-1}^*(x_{n+1}) + \frac{\Delta e_k^*(x_n)}{\|\Delta e_k^*(x_n)\|}. \tag{15}$$

Extracting the exact values, we have

$$\epsilon_{k+1}^{n} = \epsilon_{k-1}^{n+1} + [(\alpha - 1)\Delta\bar{e}_k(x_n) + \alpha\Delta\bar{\epsilon}_k^{n}] \tag{16}$$

where

$$\alpha = \frac{\|\Delta e_k(x_n)\|^2}{\|\Delta e_k^*(x_n)\|^2} \tag{17}$$

and

$$\alpha - 1 = \frac{(\|\Delta e_k(x_n)\|^2 - \|\Delta e_k^*(x_n)\|^2)}{\|\Delta e_k^*(x_n)\|^2}. \tag{18}$$

According to the induction hypothesis Eq.(14), we know $\alpha - 1 = O(\|\epsilon_k^n\|) = O(\|\epsilon\|)$. Thus we finally have $\|\epsilon_{k+1}^n\| = O(\|\epsilon\|)$, which means the errors during the process of epsilon algorithm is controlled linearly by errors in the initial data. Given the sequence generated through the nonlinear iteration defined by $x_{j+1} = F(x_j)$, where F is a vector-valued function defined on an open and connected domain $\mathcal{D}$ with a Lipschitz continuous derivative. If s is a fixed point in $\mathcal{D}$ of F, then Taylor expansion yields $F(x) - s = F^{'}(s)(x-s) + O(\|x-s\|^2)$ for all $x \in \mathcal{D}$. Then we can consider the following iteration $x_{j+1} = Ax_j + b + r_j$, where $A = F^{'}(s)$ and r_j is the error in a linearly generated sequence $x_{j+1} = Ax_j$. Now we consider the extrapolation in Eq.(9), and for each cycle, we know $\|e_k(\theta_t)\| \sim C_k\|e_k(\theta_{t-1})\|^2$, since $(\theta^*, \phi^*) = (0, 0)$. □

Finally we can show that AGE can achieve quadratic convergence in certain cases.

Proposition A.2. *Suppose θ_t, ϕ_t and $e_k(\theta_t)$ have the same order. Then quadratic convergence for gradient norm $N_t = C_tN_{t-1}^2$ can be achieved for some bounded constant C_t.*

Proof Sketch. θ_t, ϕ_t and $e_k(\theta_t)$ have the same order, $\theta_{t+1} \sim O(e_k(\theta_t))$. According to the quadratic convergence of $e_k(\theta_t)$, θ_{t+1} inherits the quadratic convergence directly. Since safeguard constants are used to clip extrapolated gradient and gradients, C_t is bounded. Thus we can derive the desired result. □

[13] analyzes the convergence rates of two extra-gradient methods for solving (8). These results are are summarized in the next theorem to facilitate the theoretical comparison.

Theorem A.4 (EG [13]). *N_t^2 computed based on the following two updating rules decreases geometrically for any $0 < \eta < 1$ as*

$$\begin{aligned} &\textit{Implicit}: N_{t+1}^2 = \left(1 - \eta^2 + \eta^4 + \mathcal{O}\left(\eta^6\right)\right) N_t^2, \\ &\textit{Extrapolation}: N_{t+1}^2 = \left(1 - \eta^2 + \eta^4\right) N_t^2, \quad \forall t \geq 0. \end{aligned}$$

A comparison among the convergence rates of these methods further confirms that the proposed method shown in Algorithm 1 can be faster than standard extra-gradient methods for solving (4) when t is large.

A.3.2 Bilinear problem and Stochastic optimization

Next, we study the convergence property of the proposed method on a more general unconstrained bilinear problem:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^l} \max_{\boldsymbol{\varphi} \in \mathbb{R}^m} \boldsymbol{\theta}^\top \boldsymbol{A}\varphi - \boldsymbol{b}^\top\boldsymbol{\theta} - \boldsymbol{c}^\top\boldsymbol{\varphi} \tag{19}$$

where $\boldsymbol{A} \in \mathbb{R}^{l \times m}, \boldsymbol{b} \in \mathbb{R}^l$ and $\boldsymbol{c} \in \mathbb{R}^m$. We assume the optimal solution $(\boldsymbol{\theta}^*, \boldsymbol{\varphi}^*)$ exists which is equivalent to

$$\begin{cases} \boldsymbol{A}\boldsymbol{\varphi}^* = \boldsymbol{b} \\ \boldsymbol{A}^\top\boldsymbol{\theta}^* = \boldsymbol{c} \end{cases}$$

Assuming the existence of the optimal solution allows us to rewrite the problem as

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^l} \max_{\boldsymbol{\varphi} \in \mathbb{R}^m} (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \boldsymbol{A}(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) + \boldsymbol{c}$$

where $\boldsymbol{c} := -\boldsymbol{\theta}^{*\top}\boldsymbol{A}\boldsymbol{\varphi}^*$ is a constant that does not depend on $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$. If we define

$$\hat{N}_t^2 := \operatorname{dist}(\boldsymbol{\theta}_t, \Theta^*)^2 + \operatorname{dist}(\boldsymbol{\varphi}_t, \Phi^*)^2$$

where (Θ^*, Φ^*) is the solution, we can obtain the following estimate on the bound of $\hat{N}_t^2$ for the above bilinear unconstrained problem.

Theorem A.5. *When t is large enough, for any* $0 < \eta < \frac{1}{\sigma_{\max}(\boldsymbol{A})}$ *we have*

$$\hat{N}_{t+1}^2 < O\left(1 - (\sigma_{\min}(\boldsymbol{A})\eta)^2\right)\hat{N}_t^2, \quad \forall t \geq 0$$

Particularly, for $\eta = \frac{1}{2\sigma_{\max}(\boldsymbol{A})}$*,*

$$\hat{N}_{t+1}^2 < O\left(1 - \frac{1}{4\kappa}\right)^t \hat{N}_0^2, \quad \forall t \geq 0$$

where $\sigma_{\max}$ *and* $\sigma_{\min}$ *are the maximal and minimal singular values of A, respectively, and* $\kappa := \frac{\sigma_{\max}(\boldsymbol{A})^2}{\sigma_{\min}(\boldsymbol{A})^2}$ *is the condition number of* $\boldsymbol{A}^\top\boldsymbol{A}$*.*

Proof Sketch. For our updating rule, the iterates can be written in the following form:

$$\theta_{t+1} = \theta_0 + \sum_{s=0}^{t+1} \gamma_{st} A\left(\varphi_s - \varphi^*\right)$$
$$\varphi_{t+1} = \varphi_0 + \sum_{s=0}^{t+1} \rho_{st} A^\top\left(\theta_s - \theta^*\right)$$

where $\gamma_{st}, \rho_{st} \in \mathbb{R}$. Using the SVD of A $A = U^\top \Sigma V$, we can get the following update: $\begin{cases} \tilde{\theta}_{t+1} = \tilde{\theta}_0 + \sum_{s=0}^{t+1} \lambda_{st}\Sigma\tilde{\varphi}_s \\ \tilde{\varphi}_{t+1} = \tilde{\varphi}_0 + \sum_{s=0}^{t+1} \mu_{st}\Sigma^\top\tilde{\theta}_s \end{cases}$ Since Σ is a diagonal matrix of singular values of A, this can be transformed into the following equivalent problem:

$$\begin{cases} \left[\tilde{\theta}_{t+1}\right]_i = \left[\tilde{\theta}_t\right]_i - \sigma_i\eta\left[\tilde{\phi}_t\right]_i - (\sigma_i\eta)^2 e_k\left[\tilde{\theta}_{t-k-1}\right]_i \\ \left[\tilde{\varphi}_{t+1}\right]_i = \left[\tilde{\varphi}_t\right]_i + \sigma_i\eta\left[\tilde{\theta}_{t+1}\right]_i \end{cases}$$

For the coordinates that are vanishing, they can be set to $\left[\tilde{\theta}^*\right]_i$ and $\left[\tilde{\phi}^*\right]_i$, which yields $N_{it}^2 < (1 - (\sigma_i\eta)^2)N_{i(t-1)}^2$, where $N_{it}^2 = (\left[\tilde{\theta}_t\right]_i - \left[\tilde{\theta}^*\right]_i)^2 + ([\tilde{\varphi}_t]_i - [\tilde{\varphi}^*]_i)^2$. Since $\hat{N}_t^2 := \operatorname{dist}(\theta_t, \Theta^*)^2 + \operatorname{dist}(\varphi_t, \Phi^*)^2 = \sum_{i=1}^r N_{it}^2$ and $N_{it}^2 < (1 - (\sigma_i\eta)^2)N_{i(t-1)}^2$, we have $\hat{N}_{t+1}^2 < \left(1 - (\sigma_{\min}(A)\eta)^2\right)\hat{N}_t^2, \quad \forall t \geq 0$.

□

A similar convergence rate has been proved in Corollary 1 of [13] for extra-gradient methods as

$$\hat{N}_{t+1}^2 \leq \left(1 - (\sigma_{\min}(\boldsymbol{A})\eta)^2 + (\sigma_{\min}(\boldsymbol{A})\eta)^4\right)\hat{N}_t^2, \quad \forall t \geq 0.$$

Thus, AGE can converge faster on this unconstrained bilinear problem than extra-gradient methods when t is large.

We now consider a stochastic version of Algorithm 1. Define $\omega \overset{\text{def}}{=} (\boldsymbol{\theta}, \boldsymbol{\varphi}), \Omega \overset{\text{def}}{=} \Theta \times \Phi$, an exact gradient operator $F(\boldsymbol{\omega})$ and a stochastic estimate of it, $F(\boldsymbol{\omega}, \xi)$, where ξ is a mini-batch of points coming from $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. Similar to the variational inequality perspective in [13], we consider the following merit function:

$$\operatorname{Err}(\boldsymbol{\omega}) \overset{\text{def}}{=} \begin{cases} \max_{(\boldsymbol{\theta}', \boldsymbol{\varphi}') \in \Omega} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi}') - \mathcal{L}(\boldsymbol{\theta}', \boldsymbol{\varphi}) \\ \text{if } F(\boldsymbol{\theta}, \boldsymbol{\varphi}) = [\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi}) - \nabla_{\boldsymbol{\varphi}}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\varphi})]^\top \\ \max_{\boldsymbol{\omega}' \in \Omega} F(\boldsymbol{\omega}')^\top(\boldsymbol{\omega} - \boldsymbol{\omega}') \\ \text{otherwise} \end{cases}, \tag{20}$$

If the following four assumptions hold, we can prove an upper bound for the merit function (20) in the following theorem. Note that assumptions 1 and 2 are standard for the convergence of SGD and assumptions 3 and 4 make a zero-sum game well-defined.

Assumption 1. Bounded variance by σ^2 : $\mathbb{E}_\xi\left[\|F(\boldsymbol{\omega})-F(\boldsymbol{\omega},\xi)\|^2\right] \leq \sigma^2, \quad \forall \boldsymbol{\omega} \in \Omega$.

Assumption 2. Bounded expected squared norm by M^2 : $\mathbb{E}_\xi\left[\|F(\omega,\xi)\|^2\right] \leq M^2, \forall \omega \in \Omega$.

Assumption 3. F **is monotone almost surely** : $\langle F(x)-F(y), x-y\rangle \geq 0$ for all $x, y \in \mathbb{R}^d$.

Assumption 4. Ω **is a compact convex set with** $\max\limits_{\boldsymbol{\omega},\boldsymbol{\omega}' \in \Omega} \|\boldsymbol{\omega}-\boldsymbol{\omega}'\|^2 \leq R^2$.

Theorem A.6. *Under Assump. 1, 2, 3 and 4, the SGD version of Algorithm 1 with averaging and a constant step-size gives*

$$\mathbb{E}\left[\operatorname{Err}\left(\overline{\boldsymbol{\omega}}_T\right)\right] \leq \frac{R^2}{2\eta T}+\eta \frac{M^2+\sigma^2}{2}$$

where $\overline{\boldsymbol{\omega}}_T \overset{def}{=} \frac{1}{T}\sum_{t=0}^{T-1} \boldsymbol{\omega}_t, \quad \forall T \geq 1.$

Proof Sketch. We adopt the variational inequality perspective by taking the merit function introduced in [13] to measure the error of the iterates. Using a result from [35] about VEA and assuming the gradient function is L-Lipschitz, we can obtain the following inequality:

$$\begin{aligned} \|\omega_{t+1}-\omega\|_2^2 \leq & \|\omega_t-\omega\|_2^2-2\eta_t F\left(\omega_t', \xi_t\right)^{\top}\left(\omega_t'-\omega\right) \\ & -\|\omega_t-\omega_t'\|_2^2 \\ & +3\eta_t^2(\|F\left(\omega_t\right)-e_k\left(F(\omega_{t-k-1}, \xi_{t-k-1})\right)\|_2^2 \\ & +\|F\left(\omega_t'\right)-F\left(\omega_t', \xi_t\right)\|_2^2+L^2\|\omega_t-\omega_t'\|_2^2). \end{aligned}$$

If we define $N_t=\|\omega_t-\omega\|_2^2, \;\; M_1\left(\omega_t, \xi_t\right)=3\|F\left(\omega_t\right)-e_k(F\left(\omega_{t-k-1}, \xi_{t-k-1}\right)\|_2^2$, we obtain the following inequality where L is the Lipschitz constant of F and C_Ω is the diameter of Ω:

$$\|F\left(\omega_t\right)-e_k(F\left(\omega_{t-k-1}, \xi_{t-k-1}\right))\|_2^2 \leq 3\sigma^2+LC_\Omega.$$

By Assump. 1, $M_1=M_2=3\sigma^2$ and $\mathbb{E}\left[F\left(\boldsymbol{\omega}_t'\right)-F\left(\boldsymbol{\omega}_t', \xi_t\right) \mid \boldsymbol{\omega}_t', \Delta_0, \ldots, \Delta_{t-1}\right] = \mathbb{E}\left[\mathbb{E}\left[F\left(\boldsymbol{\omega}_t'\right)-F\left(\boldsymbol{\omega}_t', \xi_t\right) \mid \boldsymbol{\omega}_t'\right] \mid \Delta_0, \ldots, \Delta_{t-1}\right]=0$,

$$\mathbb{E}\left[\operatorname{Err}_R\left(\bar{\omega}_T\right)\right] \leq \frac{R^2}{S_T}+\frac{12\sigma^2+3LC_\Omega}{2S_T} \sum_{t=0}^{T-1} \eta_t^2.$$

A suitable choice of η then yields

$$\mathbb{E}\left[\operatorname{Err}\left(\overline{\boldsymbol{\omega}}_T\right)\right] \leq \frac{R^2}{2\eta T}+\eta \frac{M^2+\sigma^2}{2}$$

□

Although the stochastic version of AGE has a similar theoretical convergence bound with the extra-gradient methods [13], we can show that AGE still outperforms extra-gradient methods empirically on both real and synthetic datasets in Sec 5.

A.4 Related Work

There is a rich literature on different strategies to improve GAN training, such as using different objectives [3, 29, 22], using inner-outer update fashion [9], adding an asymmetric preconditioner on top of gradient descent ascent (GDA) [37], or keeping an average over iterates outside of the training loop [28, 40].Particularly, [13] proposed to apply extragradient to overcome the cycling behaviour of GDA. One issue with the existing extra-gradient method is that it requires computing the gradient at two different positions for every single update which is likely to be inefficient for large scale problems. Optimistic gradient descent (OGD), originally proposed in [31] and rediscovered in [10], is more efficient by storing and re-using the extrapolated gradient for the extrapolation step. Without projection, OGD is equivalent to extrapolation from past. Nevertheless, its performance in a stochastic setting is worse than extra-gradient. This is because the method only uses the most recent gradient information for extrapolation and may be subject to noisy updates as well.