Minimax Optimization with Smooth Algorithmic Adversaries

Praneeth Netrapalli
Google Research India

Joint work with

Tanner Fiez, Lillian Ratliff and Chi Jin
Univ. of Washington and Princeton
Minimax optimization/ Two player zero sum games

\[ \min_\theta f(\theta, \mu) \quad \text{vs} \quad \max_\mu f(\theta, \mu) \]

- In contrast to standard optimization: \( \min_\theta f(\theta) \)
- Variants: Simultaneous vs sequential (Stackelberg), one-shot vs repeated
- Our work: Sequential, one-shot, minimax optimization
Application: Robust ML

Minimize maximum perturbation error

\[ \min_{\theta} \max_{\mu} f(\theta, \mu) \]

Classifier \( \theta \)

Adversarial perturbation \( \mu \)

\[ f(x, y) = \mathbb{E}[\text{Error of classifier } \theta \text{ after perturbations } \mu] \]

Several more: Fair ML, generative adversarial networks (GANs),...
Quick summary of our work

• **Theory**: Most work restricted to convex-concave setting, while modern applications involve nonconvex and/or nonconcave functions

• **Our goal**: Minimax optimization beyond convex-concave settings

**Main result**

Novel & computationally efficient, framework & algorithm for solving general nonconvex-nonconcave minimax optimization problems.
Classical work: Minimax theorem and equilibria

[vonNeumann 1928, Sion 1958]

If \( f(\theta, \mu) \) is convex in \( \theta \) and concave in \( \mu \), then

\[
\min_{\theta} \max_{\mu} f(\theta, \mu) = \max_{\mu} \min_{\theta} f(\theta, \mu)
\]

- Optimal strategy \((\theta^*, \mu^*)\) is called \textbf{Nash equilibrium}

\[
\theta^* \in \arg\min_{\theta} f(\theta, \mu^*) \quad \text{and} \quad \mu^* \in \arg\max_{\mu} f(\theta^*, \mu)
\]

- Minimax theorem plays a fundamental role in both existence and efficient computation of Nash equilibria
  - extensive work in convex-concave setup
Going beyond convex-concave games

• **Reason**: Most modern applications involve nonconvex and/or nonconcave objectives

• **Technical challenges**: minimax theorem does not hold, so Nash equilibrium need not exist!

• Most popular algorithm in practice: gradient descent ascent (GDA)
  • Known to suffer from limit cycles and/or divergence
  • Other variants: *extra gradient (EG)*, optimistic gradient descent ascent (OGDA) etc. have been proposed, but theory mostly restricted to convex-concave settings

Figure credit: Chaos, Extremism and Optimism: Volume Analysis of Learning in Games, Cheung and Piliouras
First key idea

\[
\min_{\theta} \max_{\mu} f(\theta, \mu)
\]

\(f(\theta, \mu)\) nonconcave in \(\mu\) ⇒ Exact maximization intractable.

Model the algorithm(s) used by the \(\mu\)-player!

\[
\min_{\theta} \max_i f(\theta, A_i(\theta))
\]

• For example, \(A_i(\theta)\) could be the output of gradient ascent (GA) on \(f(\theta, \cdot)\) starting at some \(\mu = \mu_0\).

• Captures multiple algorithms/hyperparametersinitializations etc.

• For stochastic algorithms \(A_i(\theta)\), consider

\[
\min_{\theta} \mathbb{E} \left[ \max_{i=1,\ldots,k} f(\theta, A_i(\theta)) \right]
\]
Second key idea

\[ \min_\theta \max_\mu f(\theta, \mu) \quad \text{versus} \quad \min_\theta \max_{i=1,\cdots,k} f(\theta, A_i(\theta)) \]

- Nonconvex-nonconcave
- Nonconvex-concave

- Denote \( g_i(\theta) \overset{\text{def}}{=} f(\theta, A_i(\theta)) \) and \( g(\theta) \overset{\text{def}}{=} \max_{i=1,\cdots,k} g_i(\theta) \).

- Well established theory for nonconvex-concave minimax optimization but requires smoothness of \( g_i(\theta) \).

- **Question**: Is \( g_i(\theta) \) smooth i.e., has Lipschitz gradients?
Computing the gradient

• **Yes**, if $\mathcal{A}_i(\cdot): \theta \to \mu$ are smooth i.e., Jacobian $D\mathcal{A}_i(\theta)$ is Lipschitz.

• Reason: $\nabla g_i(\theta) = \nabla \theta f(\theta, \mathcal{A}_i(\theta)) + D\mathcal{A}_i(\theta) \cdot \nabla \mu f(\theta, \mathcal{A}_i(\theta))$

• **Technical result**: If $f(\cdot, \cdot)$ is sufficiently smooth, then $D\mathcal{A}_i(\theta)$ is Lipschitz for $T$—step stochastic GA (SGA), stochastic Nesterov’s accelerated gradient (SNAG) etc.
  
  • Proof idea generalizes and holds for most of the algorithms used in practice.
  • Smoothness parameter depends exponentially on $T$ in general.

• $\nabla g(\theta) = \nabla g_{i^*}(\theta)$ where $i^* \in \text{argmax}_i g_i(\theta)$
Gradient descent algorithm

**Algorithm 1:** Subgradient descent (SGD)

**Input:** initial point $x_0$, step size $\eta$

1. for $s = 0, 1, \ldots, S$ do
2.   Compute $\nabla g(\theta_s) = \nabla g_{i^*}(\theta)$
3.   where $i^* \in \arg\max_{i\in[k]} g_i(\theta)$.
4.   $\theta_{s+1} \leftarrow \theta_s - \eta \hat{\nabla} g(\theta_s)$.
5. return $\bar{\theta} \leftarrow \theta_s$, where $s$ is uniformly sampled from $\{0, \cdots, S\}$.

Key differences from existing approaches in GANs and adversarial training.

1. Restarting max player’s algorithm in every step
2. Differentiating through max player’s algorithm
Main result

If \( f(\cdot,\cdot) \) is sufficiently smooth, then for adversaries’ algorithm choices of \( T \) —step SGA and/or SNAG, our algorithm finds an \( \epsilon \) —stationary point i.e., \( \|\nabla g(\theta)\| \leq \epsilon \) in \( O(2^O(T)\epsilon^{-\frac{4}{T}}) \) iterations.

- Exponential dependence on \( T \) is inherited from smoothness of SGA/SNAG.
- However, empirical results suggest a milder dependence on \( T \).
- Similar results can also be obtained for more sophisticated, recent algorithms for nonconvex-concave settings that give faster rates in \( \epsilon \) [Thekumparampil et al. 2019, Lin et al. 2020,...].
Sanity-check empirical results

• **Promising results**: GAN on MNIST dataset produces reasonable looking images.

• For each generator step, discriminator was initialized from scratch and $T = 10$ SGA steps were run.

• **Dependence on $T$**: Empirically not exponential.

More thorough empirical evaluation required.
Conclusion and future directions

• Our results:
  • Propose a simple but novel and interesting framework and algorithm for general nonconvex-nonconcave minimax optimization.
  • Prove convergence results for our algorithms.
  • Results naturally extend to non zero sum games.
  • Empirical results on simple examples sanity check the practical performance of proposed algorithm.

• Future directions: Thorough empirical evaluation required.
  • Restarting adversary’s algorithm in each iteration increases computational requirements. Good initialization e.g., with pretrained networks might help.
  • Extension to multi-player games, repeated games.