Efficient Regret Minimization in Non-Convex Games
Elad Hazan, Karan Singh, Cyril Zhang
Department of Computer Science, Princeton University

Motivation: Non-Convex Games
Repeated games with non-convex utility functions. Example: GAN (generative adversarial network) training. Two players G and D play a zero-sum game, with continuous strategies θG and θD, and non-convex objective

\[ \ell(θG, θD) := E_{x∼D,z∼D'} [\log D(x) + \log (1 − D(G(z)))]. \]

Also: natural model for irrational players (risk/loss aversion).

Question: What kinds of equilibria are achievable?

Framework: Online Non-Convex Optimization
Each player makes iterative predictions. For round \( t = 1, 2, \ldots \)
1. Player commits to a decision \( x_t \in K \), where \( K \subseteq \mathbb{R}^n \) is a convex decision set.
2. Nature presents a smooth non-convex loss function \( f_t : K \rightarrow \mathbb{R} \).
3. Player suffers loss \( f_t(x_t) \in \mathbb{R} \) for the chosen decision \( x_t \).

Usual goal from online convex optimization [5]: achieve low regret

\[
\text{Regret}(T) = \sum_{t=1}^T f_t(x_t) - \min_{x \in K} \sum_{t=1}^T f_t(x) \quad \text{global min. for fixed decision.}
\]

Global non-convex optimization is computationally intractable. We introduce a new framework of local regret, with efficient algorithms and tight lower bounds, generalizing known results in offline and stochastic non-convex optimization.

A Time-Smoothed Regret Measure
Define local regret (with window parameter \( w \)):

\[
\mathcal{R}_w(T) = \mathbb{E}_{x \sim D} \left[ \sum_{t=1}^T \nabla f_t(x_t) \right] \quad \text{time-smoothed gradient}
\]

\[
\mathcal{R}_w(T) = \mathbb{E}_{x \sim D} \left[ \sum_{t=1}^T \frac{1}{w} \sum_{i=0}^{w-1} f_t(x_t) \right] \quad \text{windowed average}
\]

Compare to typical non-convex convergence guarantees [2]:

\[
\sum_{t=1}^T \left| \nabla f(x_t) \right|^2 \geq o(T).
\]

Randomized construction gives lower bound \( \mathcal{R}_w(T) \geq \Omega(T/w^2) \).

Time-Smoothed Algorithms
To minimize local regret, take gradient (or second-order) steps on time-smoothed losses \( F_{t,w}(x) = \frac{1}{w} \sum_{i=0}^{w-1} f_{t-i}(x) \).

Algorithm 1: Time-Smoothed Online Gradient Descent
1: for \( t = 1, \ldots, T \) do
2: Predict \( x_t \), observe \( f_t \).
3: Update \( x_{t+1} \leftarrow x_t − \eta \nabla F_{t,w}(x_{t+1}) \).
4: end for

Algorithm 2: Time-Smoothed Follow-The-Leader
1: for \( t = 1, \ldots, T \) do
2: Predict \( x_t \), observe \( f_t \). Initialize \( x_{t+1} := x_t \).
3: while \( \left| \nabla F_{t,w}(x_{t+1}) \right| > \delta \) do
4: Update \( x_{t+1} \leftarrow x_{t+1} − \eta \nabla F_{t,w}(x_{t+1}) \).
5: end while
6: end for

Algorithm 3: Time-Smoothed Second-Order Method
1: for \( t = 1, \ldots, T \) do
2: Predict \( x_t \), observe \( f_t \). Initialize \( x_{t+1} := x_t \).
3: while \( \left| \nabla F_{t,w}(x_{t+1}) \right| > \delta \) or \( \left| \nabla^2 F_{t,w}(x_{t+1}) \right| < -\delta f_t \) do
4: Compute \( (λ, v) \leftarrow \text{MinEig} (\nabla^2 F_{t,w}(x_{t+1})) \).
5: Let \( g_{t+1} := x_{t+1} − \eta \nabla F_{t,w}(x_{t+1}) \). (gradient step)
6: Let \( h_{t+1} := x_{t+1} + \eta g_{t+1} \). (Hessian steps)
7: Take whichever step makes the most progress on \( F_{t,w} \).
8: end while
9: end for

Local Regret Bounds and Offline Reductions
Upper bounds for local regret (with optimal parameters \( \delta, \eta \)):

- **Algorithm 1** achieves \( \mathcal{R}_w(T) \leq O(Tw) \), with \( O(Tw) \) gradient oracle calls.
- **Algorithm 2** achieves (optimal) \( \mathcal{R}_w(T) \leq O(Tw^2) \). With acceleration by SVRG [1], requires \( O(Tw^5/\delta^2) \) gradient oracle calls.
- **Algorithm 3** achieves \( \mathcal{R}_w(T) \leq O(Tw^2) \), while escaping second-order saddle points.

Online framework generalizes offline convergence:

- Guarantees from **Algorithm 2** recovers standard \( O(1/\varepsilon) \) convergence rate of GD.
- Guarantees from **Algorithm 1** implies \( O(\sigma^4/\varepsilon^2) \) convergence rate of SGD. (Optimal dependence on \( \varepsilon \) but not \( \sigma^4 \).)
- Reductions are black-box (like in online convex optimization).

Solution Concept for Non-Convex Games
Answer: “Smoothed local equilibrium”, if players minimize local regret.

\[
\forall i \in [k], \| \frac{1}{w} \sum_{t=0}^{w-1} \text{Utility}(x_i(t), x_{i-1}(t)) - \mathbb{E} \| \leq \varepsilon.
\]

Solution concept in repeated non-convex games:
- No player gains too much by deviating locally from their recommended play...
- Provided that everyone else plays strategies sampled uniformly from the past \( w \) iterations.

Black-box guarantee: online algorithms minimizing local regret achieve an approximate smoothed local equilibrium, with

\[
\varepsilon \leq \sqrt{\frac{k \cdot \mathcal{R}_w(T)}{T-w}} \leq O(\sqrt{\frac{\mathbb{E}}{w}}).
\]

In GAN training, our time-smoothed algorithms are better known as experience replay, a known technique for promoting stability. [4]

Note: Projected Gradients for Constraint Sets
Constrained non-convex optimization can be information-theoretically hard. To handle constraints, our algorithms use projected gradient descent, and achieve bounds on the projected gradient, the vector \( \nabla \mathcal{L}_x(x) \) such that \( x_{t+1} \leftarrow x_t − \eta \nabla \mathcal{L}_x(x_t) \) is a projected gradient descent step.

Finding a small \( \nabla f \) is intractable... but PGD finds a small \( \nabla \mathcal{L}_x \).

References