How to Escape Saddle Points Efficiently?

Praneeth Netrapalli
Microsoft Research India

Chi Jin
UC Berkeley

Michael I. Jordan
UC Berkeley

Rong Ge
Duke Univ.

Sham M. Kakade
U Washington
Nonconvex optimization

**Problem:** \( \min_x f(x) \quad f(\cdot): \text{nonconvex function} \)

**Applications:** Deep learning, compressed sensing, matrix completion, dictionary learning, nonnegative matrix factorization, ...
Gradient descent (GD) [Cauchy 1847]

$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

**Question**
How does it perform?
Gradient descent (GD) [Cauchy 1847]

\[ x_{t+1} = x_t - \eta \nabla f(x_t) \]

**Question**
How does it perform?

**Answer**
Converges to first order stationary points
Gradient descent (GD) [Cauchy 1847]

\[ x_{t+1} = x_t - \eta \nabla f(x_t) \]

**Question**
How does it perform?

**Answer**
Converges to first order stationary points

**Definition**
\( \epsilon \)-First order stationary point (\( \epsilon \)-FOSP):
\[ ||\nabla f(x)|| \leq \epsilon \]
Gradient descent (GD) [Cauchy 1847]

\[ x_{t+1} = x_t - \eta \nabla f(x_t) \]

**Question**
How does it perform?

**Answer**
Converges to first order stationary points

**Definition**
\( \epsilon \)-First order stationary point (\( \epsilon \)-FOSP): \( \| \nabla f(x) \| \leq \epsilon \)

**Concretely**
\( \epsilon \)-FOSP in \( O \left( \frac{1}{\epsilon^2} \right) \) iterations
[Folklore]
How do FOSPs look like?
How do FOSPs look like?

Hessian PSD
\[ \nabla^2 f(x) \succeq 0 \]
Second order stationary points (SOSP)
How do FOSPs look like?

Hessian PSD
\[ \nabla^2 f(x) \geq 0 \]
Second order stationary points (SOSP)

Hessian NSD
\[ \nabla^2 f(x) \leq 0 \]
How do FOSPs look like?

Hessian PSD
\[ \nabla^2 f(x) \geq 0 \]
Second order stationary points (SOSP)

Hessian NSD
\[ \nabla^2 f(x) \leq 0 \]

Hessian indefinite
\[ \lambda_{\text{min}}(\nabla^2 f(x)) \leq 0 \]
\[ \lambda_{\text{max}}(\nabla^2 f(x)) \geq 0 \]
FOSPs in popular problems

• Very well studied
  • Neural networks [Dauphin et al. 2014]
  • Matrix sensing [Bhojanapalli et al. 2016]
  • Matrix completion [Ge et al. 2016]
  • Robust PCA [Ge et al. 2017]
  • Tensor factorization [Ge et al. 2015, Ge & Ma 2017]
  • Smooth semidefinite programs [Boumal et al. 2016]
  • Synchronization & community detection [Bandeira et al. 2016, Mei et al. 2017]
Two major observations

• FOSPs: proliferation (exponential #) of saddle points
  • Recall FOSP $\triangleq \nabla f(x) = 0$
  • Gradient descent can get stuck near them

• SOSPs: not just local minima; as good as global minima
  • Recall SOSP $\triangleq \nabla f(x) = 0 \& \nabla^2 f(x) \succeq 0$

Upshot
1. FOSP not good enough
2. Finding SOSP sufficient
Can gradient descent find SOSPs?

- Yes, perturbed GD finds an $\epsilon$-SOSP in $O\left(poly\left(\frac{d}{\epsilon}\right)\right)$ iterations [Ge et al. 2015]

- GD is a first order method while SOSP captures second order information
Can gradient descent find SOSPs?

• Yes, perturbed GD finds an $\epsilon$-SOSP in $O\left(\text{poly}\left(\frac{d}{\epsilon}\right)\right)$ iterations [Ge et al. 2015]

• GD is a first order method while SOSP captures second order information

Question 1
Does perturbed GD converge to SOSP **efficiently**?
In particular, **independent of $d$**?
Can gradient descent find SOSPs?

• Yes, perturbed GD finds an $\epsilon$-SOSP in $O\left(poly\left(\frac{d}{\epsilon}\right)\right)$ iterations [Ge et al. 2015]

• GD is a first order method while SOSP captures second order information

---

**Question 1**

Does perturbed GD converge to SOSP **efficiently**?

In particular, **independent of $d$**?

**Our result**

Almost yes, in $\tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^2}\right)$ iterations!
Accelerated gradient descent (AGD) [Nesterov 1983]

• Optimal algorithm in the convex setting

• **Practice:** Sutskever et al. 2013 observed AGD to be much faster than GD

• Widely used in training neural networks since then

• **Theory:** Finds an $\epsilon$-FOSP in $O\left(\frac{1}{\epsilon^2}\right)$ iterations [Ghadimi & Lan 2013]

• No improvement over GD
Question 2: Does essentially pure AGD find SOSPs faster than GD?

- **Our result:** Yes, in \( \tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^{1.75}}\right) \) steps compared to \( \tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^2}\right) \) for GD.

- Perturbation + negative curvature exploitation (NCE) on top of AGD
  - NCE inspired by Carmon et al. 2017

- Carmon et al. 2016 and Agarwal et al. 2017 show this improved rate for a more complicated algorithm
  - Solve sequence of regularized problems using AGD.
Summary

- Convergence to SOSPs very important in practice
- Pure GD and AGD can get stuck near FOSPs (saddle points)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Paper</th>
<th># Iterations</th>
<th>Simplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perturbed gradient descent</td>
<td>Ge et al. 2015, Levy 2016</td>
<td>$O\left(\text{poly}\left(\frac{d}{\epsilon}\right)\right)$</td>
<td>Single loop</td>
</tr>
<tr>
<td></td>
<td>Jin, Ge, N., Kakade, Jordan 2017</td>
<td>$\tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^2}\right)$</td>
<td>Single loop</td>
</tr>
<tr>
<td>Sequence of regularized subproblems with AGD</td>
<td>Carmon et al. 2016, Agarwal et al. 2017</td>
<td>$\tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^{1.75}}\right)$</td>
<td>Nested loop</td>
</tr>
<tr>
<td>Perturbed AGD + NCE</td>
<td>Jin, N., Jordan 2017</td>
<td>$\tilde{O}\left(\frac{\text{polylog}(d)}{\epsilon^{1.75}}\right)$</td>
<td>Single loop</td>
</tr>
</tbody>
</table>
Part I
Main Ideas of the Proof of Gradient Descent
Setting

- **Gradient Lipschitz:** $\| \nabla f(x) - \nabla f(y) \| \leq \| x - y \|$

- **Hessian Lipschitz:** $\| \nabla^2 f(x) - \nabla^2 f(y) \| \leq \| x - y \|$

- **Lower bounded:** $\min_x f(x) > -\infty$
How does GD behave?

Recall
FOSP: $\nabla f(x)$ small
SOSP: $\nabla f(x)$ small & $\lambda_{\text{min}}(\nabla^2 f(x)) \succeq 0$

GD step

$x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$
How does GD behave?

Recall
FOSP: $\nabla f(x)$ small
SOSP: $\nabla f(x)$ small & $\lambda_{\text{min}}(\nabla^2 f(x)) \geq 0$

GD step
$$x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$$

$\|\nabla f(x_t)\|$ small
- SOSP
- Saddle point

$\|\nabla f(x_t)\|$ large

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2} \|\nabla f(x_t)\|^2$$

$f(x_t)$
$-\eta \nabla f(x_t)$
$f(x_{t+1})$
How does GD behave?

GD step:
\[ x_{t+1} \leftarrow x_t - \eta \nabla f(x_t) \]

- **FOSP**:
  \[ \nabla f(x) \text{ small} \]
- **SOSP**: \[ \nabla f(x) \text{ small} \& \lambda_{\min}(\nabla^2 f(x)) \gtrless 0 \]

SOSP:
- Saddle point
- \[ f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2} \|\nabla f(x_t)\|^2 \]
How to escape saddle points?
Perturbed gradient descent

1. For $t = 0, 1, \ldots$ do
2. if perturbation_condition_holds then
3. $x_t \leftarrow x_t + \xi_t$ where $\xi_t \sim Unif(B_\epsilon)$
4. $x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$
Perturbed gradient descent

1. For $t = 0, 1, \ldots$ do
2. if perturbation_condition_holds then
3. $x_t \leftarrow x_t + \xi_t$ where $\xi_t \sim Unif(B_0(\epsilon))$
4. $x_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$

Between two perturbations, just run GD!
Perturbed gradient descent

1. For \( t = 0, 1, \ldots \) do
2. if perturbation_condition_holds then
3. \( x_t \leftarrow x_t + \xi_t \) where \( \xi_t \sim \text{Unif}(B_0(\epsilon)) \)
4. \( x_{t+1} \leftarrow x_t - \eta \nabla f(x_t) \)

Between two perturbations, just run GD!

1. \( \nabla f(x_t) \) is small
2. No perturbation in last several iterations
How can perturbation help?
Key question

- $S \overset{\text{def}}{=} \text{set of points around saddle point from where gradient descent does not escape quickly}$

- Escape $\overset{\text{def}}{=} \text{function value decreases significantly}$

- How much is $\text{Vol}(S)$?

- $\text{Vol}(S) \text{ small } \Rightarrow \text{perturbed GD escapes saddle points efficiently}$
Two dimensional quadratic case

- \( f(x) = \frac{1}{2} x^\top \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} x \)

- \( \lambda_{\text{min}}(H) = -1 < 0 \)

- \( (0,0) \) is a saddle point

- GD: \( x_{t+1} = \begin{bmatrix} 1 - \eta & 0 \\ 0 & 1 + \eta \end{bmatrix} x_t \)

- \( S \) is a thin strip, \( \text{Vol}(S) \) is small
Three dimensional quadratic case

• $f(x) = \frac{1}{2} x^\top \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} x$

• $(0,0,0)$ is a saddle point

• GD: $x_{t+1} = \begin{bmatrix} 1 - \eta & 0 & 0 \\ 0 & 1 - \eta & 0 \\ 0 & 0 & 1 + \eta \end{bmatrix} x_t$

• $S$ is a thin disc, Vol($S$) is small
General case

Key technical results

$S \sim$ thin deformed disc

$\text{Vol}(S)$ is small
Two key ingredients of the proof

**Improve or localize**

\[
f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2} \| \nabla f(x_t) \|^2 \\
= f(x_t) - \frac{\eta}{2} \left\| \frac{x_t - x_{t+1}}{\eta} \right\|^2
\]

\[
\| x_t - x_{t+1} \|^2 \leq 2\eta (f(x_t) - f(x_{t+1}))
\]

\[
\| x_0 - x_t \|^2 \leq t \sum_{i=0}^{t-1} \| x_i - x_{i+1} \|^2 \leq 2\eta t (f(x_0) - f(x_t))
\]
Two key ingredients of the proof

Improve or localize

Upshot

Either function value decreases significantly or iterates do not move much

\[
\|x_0 - x_t\|^2 \leq t \sum_{i=0}^{t-1} \|x_i - x_{i+1}\|^2 \leq 2\eta t(f(x_0) - f(x_t))
\]
Proof idea

• If GD from either $u$ or $w$ goes outside a small ball, it escapes (function value $\downarrow$)

• If GD from both $u$ and $w$ lie in a small ball, use local quadratic approximation of $f(\cdot)$

• Show the claim for exact quadratic, and bound approximation error using Hessian Lipschitz property
Putting everything together

GD step:
\[ x_{t+1} \leftarrow x_t - \eta \nabla f(x_t) \]

\[ \|\nabla f(x_t)\| \text{ large} \]

\[ \|\nabla f(x_t)\| \text{ small} \]

Saddle point

SOSP

Perturbation + GD

Stays at SOSP

Moves away from SOSP

f(·) decreases

\[ f(\cdot) \text{ decreases} \]
Part II
Main Ideas of the Proof of Accelerated Gradient Descent
Nesterov’s AGD

Iterate $x_t$ & Velocity $v_t$

1. $x_{t+1} = (x_t + (1 - \theta)v_t) - \eta \nabla f(x_t + (1 - \theta)v_t)$
2. $v_{t+1} = x_{t+1} - x_t$

Gradient descent at $x_t + (1 - \theta)v_t$

Challenge

Known potential functions depend on optimum $x^*$
Differential equation view of AGD

• AGD is a discretization of the following ODE [Su et al. 2015]

\[ \ddot{x} + \tilde{\theta} \dot{x} + \nabla f(x) = 0 \]

• Multiplying by \( \dot{x} \) and integrating from \( t_1 \) to \( t_2 \) gives us

\[ f(x_{t_2}) + \frac{1}{2} \left\| \dot{x}_{t_2} \right\|^2 = f(x_{t_1}) + \frac{1}{2} \left\| \dot{x}_{t_1} \right\|^2 - \tilde{\theta} \int_{t_1}^{t_2} \left\| \dot{x}_t \right\|^2 dt \]

• Hamiltonian \( f(x_t) + \frac{1}{2} \left\| \dot{x}_t \right\|^2 \) decreases monotonically
After discretization

Iterate: $x_t$ and velocity: $v_t := x_t - x_{t-1}$

- Hamiltonian $f(x_t) + \frac{1}{2\eta} \|v_t\|^2$ decreases monotonically if $f(\cdot)$ “not too nonconvex” between $x_t$ and $x_t + v_t$
  - too nonconvex = negative curvature
  - Can increase if $f(\cdot)$ is “too nonconvex”

- If the function is “too nonconvex”, reset velocity or move in nonconvex direction – negative curvature exploitation
Hamiltonian decrease

\[ f(\cdot) \text{ between } x_t \text{ and } x_t + v_t \]

**Not too nonconvex**

- AGD step

**Too nonconvex**

- \( \|v_t\| \) large
  - \( v_{t+1} = 0 \)
  - \( f(x_t) + \frac{1}{2\eta} \|v_t\|^2 \) decreases

- \( \|v_t\| \) small
  - Move in \( \pm v_t \) direction
Negative curvature exploitation – $\|v_t\|$ small

One of $\pm v_t$ directions decreases $f(x_t)$
Hamiltonian decrease

\[ f(\cdot) \text{ between } x_t \text{ and } x_t + v_t \]

Not too nonconvex

Too nonconvex

(Negative curvature exploitation)

AGD step

\[ ||v_t|| \text{ large} \]

\[ v_{t+1} = 0 \]

\[ f(x_t) + \frac{1}{2\eta} ||v_t||^2 \text{ decreases} \]

Move in \pm v_t \text{ direction}

\[ ||v_t|| \text{ small} \]

Need to do amortized analysis

Enough decrease in a single step
Improve or localize

\[ f(x_{t+1}) + \frac{1}{2\eta} \|v_{t+1}\|^2 \leq f(x_t) + \frac{1}{2\eta} \|v_t\|^2 - \frac{\theta}{2\eta} \|v_t\|^2 \]

\[ \sum_{t=0}^{T-1} \|x_{t+1} - x_t\|^2 \leq \frac{2\eta}{\theta} \cdot (f(x_0) - f(x_T)) \]

- Approximate locally by a quadratic and perform computations
  - Precise computations are technically challenging
Summary

• Simple variations to GD/AGD ensure efficient escape from saddle points

• Fine understanding of geometric structure around saddle points

• Novel techniques of independent interest

• Some extensions to stochastic setting
Open questions

➢ Is NCE really necessary?

➢ Lower bounds – recent work by Carmon et al. 2017, but gaps between upper and lower bounds

➢ Extensions to stochastic setting

➢ Nonconvex optimization for faster algorithms
Thank you!

Questions?