Non-Convex Optimization

Rishabh Iyer

Lecture 17

Optimization in Machine Learning, UT Dallas

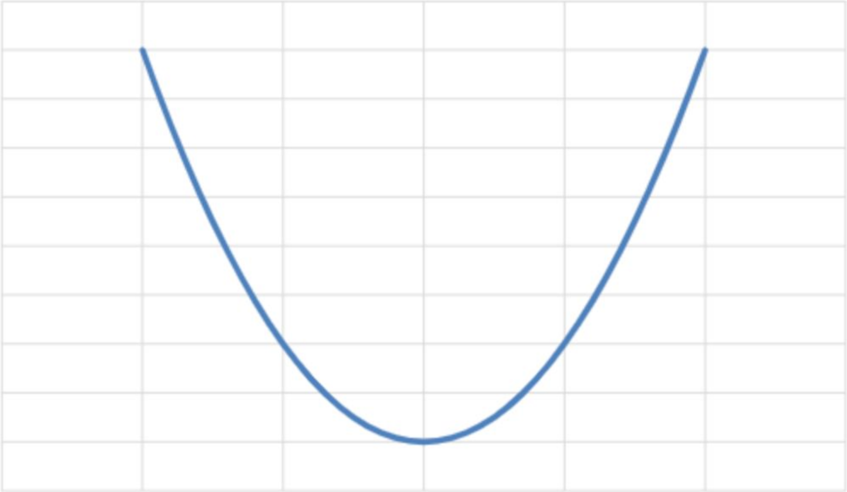
Outline

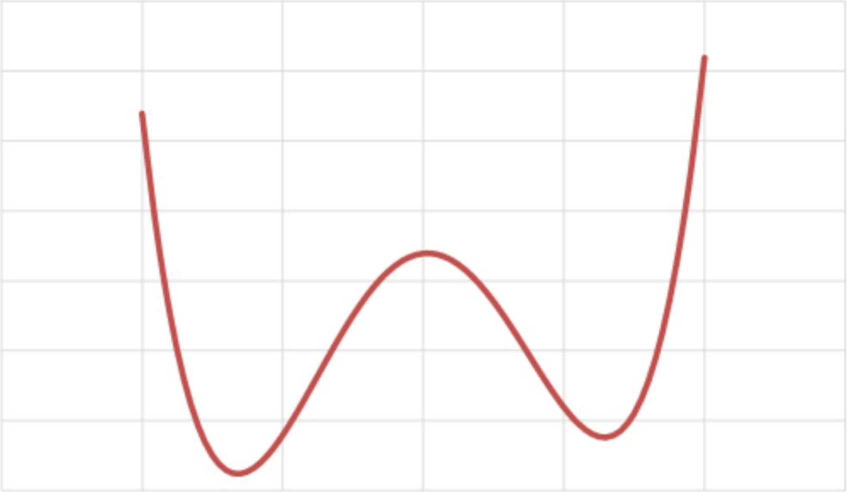
1. Non Convex Optimization: Introduction
2. Why is Non-Convex Optimization Hard
3. Some Convergence Results for Non Convex Optimization
4. Using SGD for Deep Learning

Thanks to Prof. Christopher De Sa from Cornell University for the slides

Non-Convex Problems

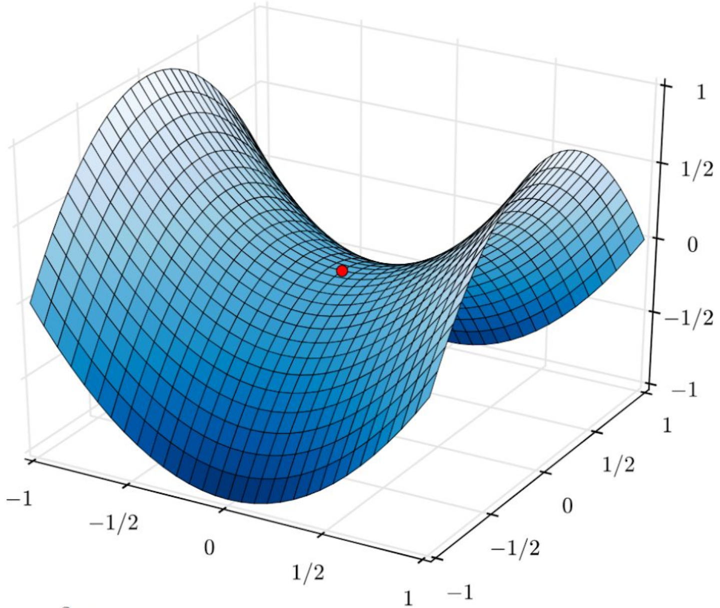
- Anything that's **not convex**





What makes non-convex optimization hard?

- Potentially many local minima
- Saddle points
- Very flat regions
- Widely varying curvature



Source:
https://commons.wikimedia.org/wiki/File:Saddle_point.svg

But is it actually that hard?

- Yes, non-convex optimization is at least **NP-hard**
  - Can encode most problems as non-convex optimization problems

- Example: subset sum problem
  - **Given a set of integers, is there a non-empty subset whose sum is zero?**
  - Known to be NP-complete.

- How do we encode this as an optimization problem?

Subset sum as non-convex optimization

- Let $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ be the input integers

- Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ be 1 if $\mathbf{a}_i$ is in the subset, and 0 otherwise

- Objective:

$$\text{minimize} \;\; \left(a^T x\right)^2 + \sum_{i=1}^{n} x_i^2 (1 - x_i)^2$$

- **What is the optimum if subset sum returns true? What if it's false?**

So non-convex optimization is pretty hard

- There can't be a general algorithm to solve it efficiently in all cases

- Downsides: **theoretical guarantees are weak** or nonexistent
  - Depending on the application
  - There's usually no theoretical recipe for setting hyperparameters

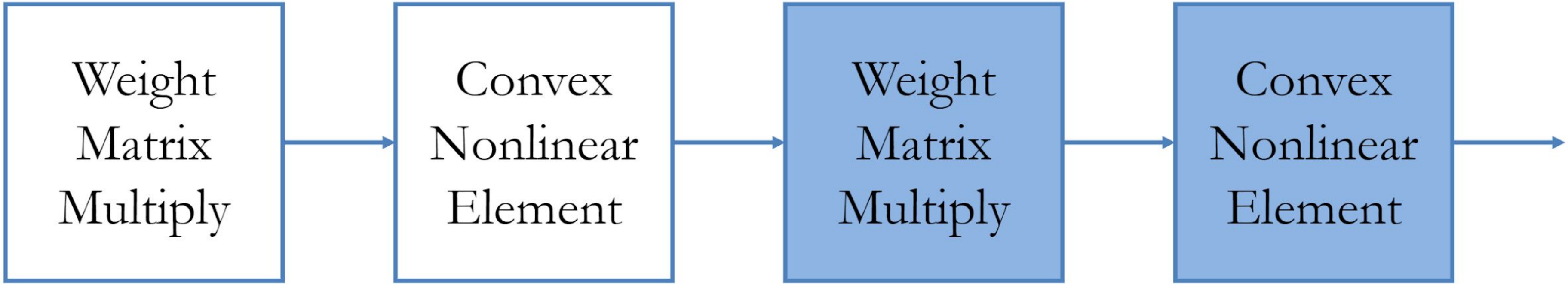
- Upside: an endless array of **problems to try to solve better**
  - And gain theoretical insight about
  - And improve the performance of implementations

Examples of non-convex problems

- Matrix completion, principle component analysis
- Low-rank models and tensor decomposition
- Maximum likelihood estimation with hidden variables
  - Usually non-convex
- The big one: **deep neural networks**

Why are neural networks non-convex?

- They're often made of convex parts!
  - **This by itself would be convex**.



- **Composition of convex functions is not convex**
  - So deep neural networks also aren't convex

Why do neural nets need to be non-convex?

- Neural networks are **universal function approximators**
  - With enough neurons, they can learn to approximate any function arbitrarily well
- To do this, they need to be able to approximate non-convex functions
  - **Convex functions can't approximate non-convex** ones well.
- Neural nets also have many symmetric configurations
  - For example, exchanging intermediate neurons
  - This symmetry means they can't be convex. **Why?**

How to solve non-convex problems?

- Can use many of the **same techniques as before**
  - Stochastic gradient descent
  - Mini-batching
  - SVRG
  - Momentum

- There are also specialized methods for solving non-convex problems
  - Alternating minimization methods
  - Branch-and-bound methods
  - These generally **aren't very popular for machine learning** problems

Varieties of theoretical convergence results

- Convergence **to a stationary point**
- Convergence **to a local minimum**
- **Local convergence** to the global minimum
- **Global convergence** to the global minimum

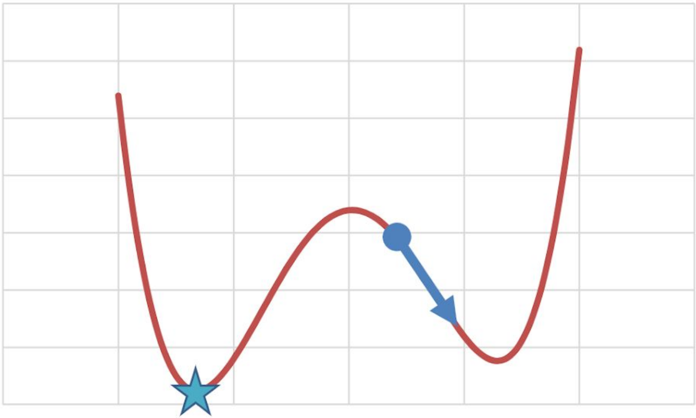
Stochastic Gradient Descent

- The update rule is the same for non-convex functions

$$w_{t+1} = w_t - \alpha_t \nabla \tilde{f}_t(w_t)$$

- Same intuition of **moving in a direction that lowers objective**

- **Doesn't necessarily go towards optimum**
  - Even in expectation



Non-convex SGD: A Systems Perspective

- It's **exactly the same as the convex case**!
- The **hardware doesn't care** whether our gradients are from a convex function or not
- This means that all our intuition about computational efficiency from the convex case directly applies to the non-convex case
- But **does our intuition about statistical efficiency also apply?**

When can we say SGD converges?

- First, we need to decide what type of convergence we want to show
  - Here I'll just show **convergence to a stationary point**, the weakest type
- Assumptions:
  - Second-differentiable objective

$$-LI \preceq \nabla^2 f(x) \preceq LI$$

  - Lipschitz-continuous gradients
  - Noise has bounded variance

$$\mathbf{E}\left[\left\|\nabla \tilde{f}_t(x) - f(x)\right\|^2\right] \le \sigma^2$$

  - But **no convexity assumption**!

Convergence of Non-Convex SGD

- Start with the update rule:

$$w_{t+1} = w_t - \alpha_t \nabla \tilde{f}_t(w_t)$$

- At the next time step, by Taylor's theorem, the objective will be

$$\begin{aligned} f(w_{t+1}) &= f(w_t - \alpha_t \nabla \tilde{f}_t(w_t)) \\ &= f(w_t) - \alpha_t \nabla \tilde{f}_t(w_t)^T \nabla f(w_t) + \frac{\alpha_t^2}{2} \nabla \tilde{f}_t(w_t)^T \nabla^2 f(y_t) \nabla \tilde{f}_t(w_t) \end{aligned}$$

After a number of slides of Math…

Convergence (continued)

- This means that for some fixed constant **C**

$$\mathbf{E}\left[\|\nabla f(z_T)\|^2\right] \le \frac{C}{\log T}$$

- And so in the limit

$$\lim_{T\to\infty} \mathbf{E}\left[\|\nabla f(z_T)\|^2\right] = 0.$$

Convergence Takeaways

- So even **non-convex SGD converges!**
  - In the sense of getting to points where the **gradient is arbitrarily small**
- But this **doesn't mean it goes to a local minimum**!
  - Doesn't rule out that it goes to a saddle point, or a local maximum.
  - Doesn't rule out that it goes to a region of very flat but nonzero gradients.
- Certainly **doesn't mean that it finds the global optimum**
- And the theoretical **rate here was really slow**

Strengthening these theoretical results

Convergence to a local minimum

- Under stronger conditions, can prove that SGD **converges to a local minimum**
  - For example using the **strict saddle property** (Ge et al 2015)

- Using even stronger properties, can prove that **SGD converges to a local minimum with an explicit convergence rate** of **1/T**

- **But, it's unclear whether common classes of non-convex problems, such as neural nets, actually satisfy these stronger conditions.**

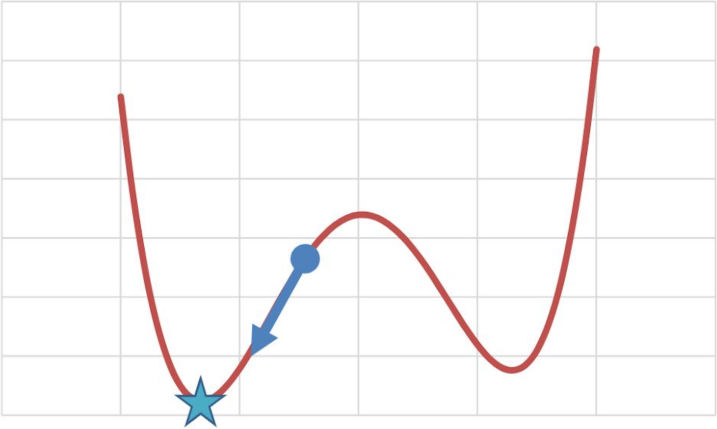
Strengthening these theoretical results

Local convergence to the global minimum

- Another type of result you'll see are **local convergence results**

- Main idea: if we **start close enough to the global optimum**, we will converge there with high probability

- Results often give **explicit initialization scheme** that is guaranteed to be close
  - But it's often **expensive to run**
  - And **limited to specific problems**



Strengthening these theoretical results

Global convergence to a global minimum

- The strongest result is **convergence no matter where we initialize**
  - Like in the convex case
- To prove this, we need a **global understanding of the objective**
  - So it can only apply to a **limited class of problems**
- For many problems, we **know empirically that this doesn't happen**
  - Deep neural networks are an example of this

Other types of results

- Bounds on generalization error
  - Roughly: we can't say it'll converge, but we can say that it won't overfit
- Ruling out "spurious local minima"
  - Minima that **exist in the training loss, but not in the true/test loss**.
- Results that use the Hessian to escape from saddle points
  - By using it to find a descent direction, but rarely enough that it doesn't damage the computational efficiency

One Case Where We Can Show Global Convergence: PCA

Recall: Principal Component Analysis

- Setting: find the dominant eigenvalue-eigenvector pair of a positive semidefinite symmetric matrix **A**.

$$u_1 = \arg\max_x \frac{x^T A x}{x^T x}$$

- Many ways to write this problem, e.g.

$\|B\|_F$ is *Frobenius norm*

$$\|B\|_F^2 = \sum_i \sum_j B_{i,j}^2$$

$$\sqrt{\lambda_1} u_1 = \arg\min_x \|x x^T - A\|_F^2$$

Recall: PCA is Non-Convex

- PCA is **not convex** in any of its formulations

- **Why?** Think about the solutions to the problem: **u** and **–u**
  - Two distinct solutions → can't be convex

- But it turns out that we can still show that with appropriately chosen step sizes, **gradient descent converges globally**!
  - This is one of the easiest non-convex problems, and a good place to start to understand how a method works on non-convex problems.

PCA: Every Local Minima is a Global Minima!

Not true for other problems but a good starting point to understand!

Can we generalize these results?

- **Difficult to generalize!**
  - Especially to problems like neural nets that are hard to analyze algebraically
- This PCA objective is one of **the simplest non-convex problems**
  - It's just a degree-4 polynomial
- But these results can **give us intuition** about how our methods apply to the non-convex setting
  - To understand a method, PCA is a good place to start

Lots of Interesting Problems are Non-Convex

- Including deep neural networks

- Because of this, we almost always **can't prove convergence** or anything like that when we run backpropagation (SGD) on a deep net

- But can we use intuition from PCA and convex optimization to understand **what could go wrong when we run non-convex optimization** on these complicated problems?

What could go wrong?
We could converge to a bad local minimum

- Problem: we converge to a local minimum which is bad for our task
  - Often in a **very steep potential well**

- One way to debug: re-run the system with **different initialization**
  - Hopefully it will converge to some other local minimum which might be better

- Another way to debug: **add extra noise to gradient updates**
  - Sometimes called "stochastic gradient Langevin dynamics"
  - Intuition: extra noise pushes us out of the steep potential well

What could go wrong?
We could converge to a saddle point

- Problem: we converge to a saddle point, which is not locally optimal

- Upside: **usually doesn't happen with plain SGD**
  - Because noisy gradients push us away from the saddle point
  - But can happen with more sophisticated SGD-like algorithms

- One way to debug: find the **Hessian** and compute a descent direction

Takeaway

- Non-convex optimization is **hard to write theory about**

- But it's **just as easy to compute SGD on**
  - This is why we're seeing a renaissance of empirical computing

- We can use the techniques we have discussed to get speedup here too

- We can **apply intuition from the convex case and from simple problems like PCA** to learn how these techniques work

What could go wrong?

We get stuck in a region of low gradient magnitude

- Problem: we converge to a region where the gradient's magnitude is small, and then stay there for a very long time
  - Might not affect asymptotic convergence, but very bad for real systems

- One way to debug: use specialized techniques like batchnorm
  - There are many **methods for preventing this problem for neural nets**

- Another way to debug: design your network so that it doesn't happen
  - Networks using a **RELU activation** tend to avoid this problem

What could go wrong?

Due to high curvature, we do huge steps and diverge

- Problem: we go to a region where the gradient's magnitude is very large, and then we make a series of very large steps and diverge
  - Especially bad for real systems using floating point arithmetic

- One way to debug: use adaptive step size
  - Like we did for PCA
  - **Adam**

- A simple way to debug: just limit the size of the gradient step
  - But this can lead to the low-gradient-magnitude issue

Conclusions

1. Long way to go for the Theory of Non Convex Optimization to Match Practice
2. However, a lot of the insights for Convex Optimization are used to design optimization algorithms for non-convex optimization
3. Intuition on why some of these algorithms work