University of Illinois Fall 2018

ECE 544NA: Pattern Recognition
Lecture 4: September 6

Lecturer: Alexander Schwing Scribe: Xinhang Song

1 Overview

The goal of this lecture is to understand the basics of optimization. Optimization problems that we have seen so far include linear regression and logistic regression.
Linear Regression:

$$\min_w \frac{1}{2} \sum_{(x^{(i)}, y^{(i)} \epsilon D)} (y^{(i)} - \phi(x^{(i)})^\top w)^2 \tag{1}$$

Logistic Regression:

$$\min_w \sum_{(x^{(i)}, y^{(i)} \epsilon D)} log(1 + exp(-y^{(i)} w^\top \phi(x^{(i)}))) \tag{2}$$

We can find analytically computable optimum for linear regression, but we can't find analytically computable optimum in logistic regression. So we use gradient descent to find optimum in logistic regression. Generally, optimization problem looks like as follow:

$$\begin{aligned} &min_w \quad f_0(w) \\ &s.t. \quad\;\; f_i(w) \leq 0 \quad \forall i \in \{1, ..., C\} \end{aligned} \tag{3}$$

w is the optimization variable of the problem, the function f_0 is the objective function, the functions f_i are the constraint functions. Solution w^* is called optimal, if it has the smallest value $f_0(w^*)$ among all values that satisfy constraints (s.t. is the abbreviation of "subject to", means constraint). There are three different minimal points that are saddle minimums, local minimums, global minimums in Figure 1 (This example is in 1 Dimension, it can also be extended to multi-dimentions). How do we differetiate them?

- Saddle Minimum: In mathematics, a saddle point is a point on the surface of the graph of a function where derivatives in orthogonal directions are both zero, but which is not a local extremum of the function. (See Fig. 1) [1]
- Local Minimum: In local optimization, we seek a point that is only locally optimal, which means that it minimizes the objective function among feasible points that are near it, but is not guaranteed to have a lower objective value than all other feasible points.[1]
- Global Minimum: It has zero gradients in all directions and also has the smallest value. It has a lower objective value than all other feasible points.

General optimization problems are very difficult to solve, often compromise between accuracy and computation time. From Fig. 2, we can think about 3 questions.:

- When can we find the optimum?

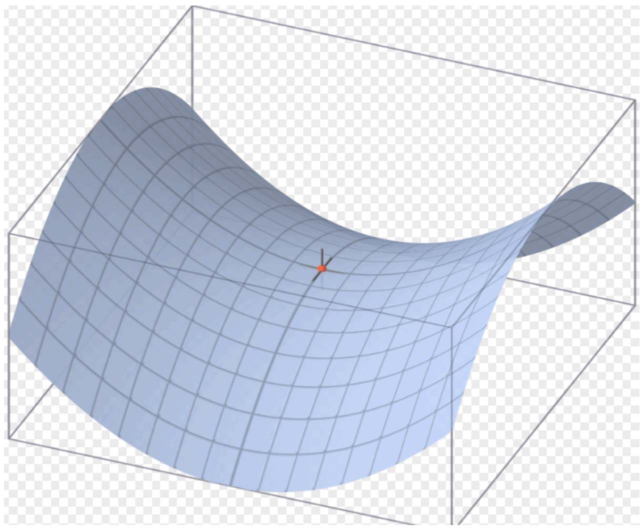

Figure 1: a saddle minimum(point), cited from Wikipedia[1]

- How do we search for the optimum?
- How long does it take to find the optimum?

1.1 When can we find the optimum?

To answer this question, we need to figure out the property of functions that have the optimum. For example, least squares, linear and convex programs can be solved efficiently and reliably. The form of least squares, linear and convex programs are given as follow:

Least squares: A least-squares problem has no constraints and its form is as follows.

$$\min_w \frac{1}{2} \sum_{(x^{(i)},y^{(i)}\in D)} (y^{(i)} - \phi(x^{(i)})^\top w)^2 \tag{4}$$

linear program: the objective and all constraint functions are linear.

$$\min_w c^\top w \qquad s.t. \;\; Aw \leqslant b \tag{5}$$

convex program: the convex program has the form as follow, where all f_i are convex.

$$min_w \; f_0(w)s.t. \quad f_i(w) \leq 0 \quad \forall i \in \{1, ..., C\} \tag{6}$$

Convex program generalizes the above, the least-squares problem and linear programming problem are both special cases of the general convex optimization problem. So, how do we know they are convex? We will talk about what convex means next.

1.1.1 Convex function

Before we talk about convex function, we should know what is convex set first:

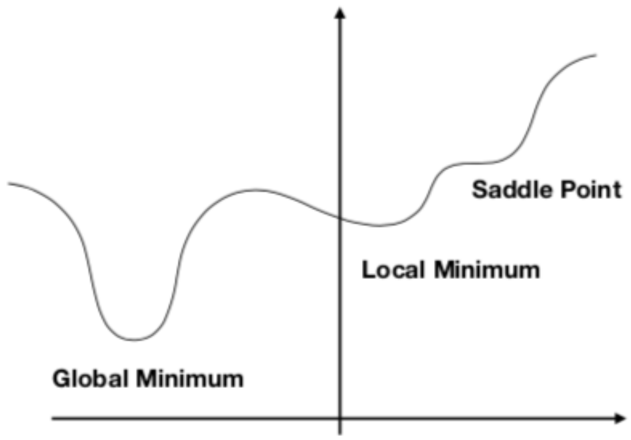

Figure 2: Three different minimal points

Definition 1 *Covex Set: A set is convex if for any two points w_1, w_2 in the set, the line segment $\lambda w_1 + (1-\lambda)w_2$ for $\lambda \in [0,1]$ also lies in the set.*

To better understand this definition, we can see Fig.3, a convex set means that when you take any two points, you draw a line between the two points, all the points in this line will be in the set. So the left of Fig.3 is not convex set while the right is convex. Another well-known example is Polyhedron that is a convex set. The general description of Polyhedron is $\{w \mid Aw \leq b, Cw = d\}$, which is the same as the constraint of linear program. That is, the constraint of linear program is convex.

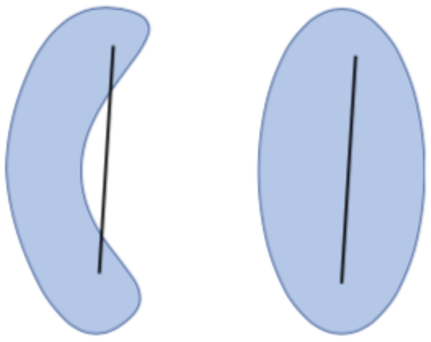

Figure 3: Left: The kidney shaped set is not convex. Right: It is convex set.

Definition 2 *Convex Function: A function f is convex if its domain is a convex set and for any points w_1, w_2 in the domain and any $\lambda \in [0,1]$.*

$$f((1-\lambda)w_1 + \lambda w_2) \leq (1-\lambda)f(w_1) + \lambda f(w_2) \tag{7}$$

To better understand this definition, we can see Fig. 4, for the region betwee w_1 and w_2, the straight line ($(1-\lambda)f(w_1) + \lambda f(w_2)$)is higher(larger) than the polynomial function($f((1-\lambda)w_1 + \lambda w_2)$).

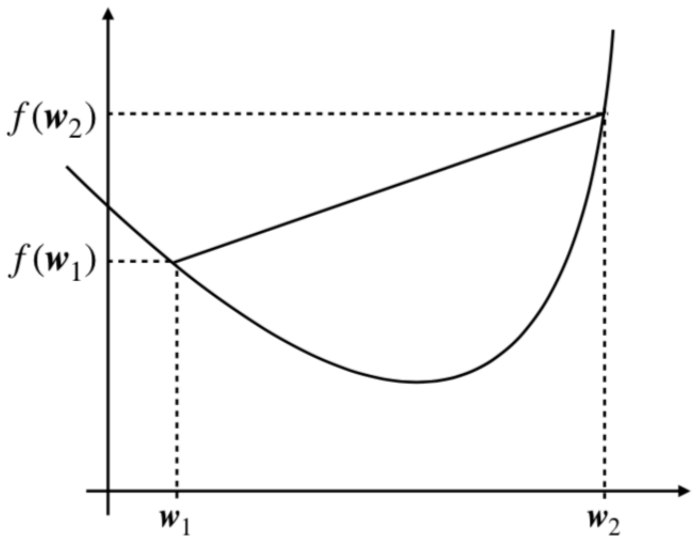

Figure 4: Graph of a convex function. The chord (i.e., line segment) between any two points on the graph lies above the graph.

There are several ways to recognize convex function:

- If f is differentiable, then f is convex if and only if its domain is convex and $f(w_1) \geq f(w_2) + \bigtriangledown f(w_2)^\top (w_1 - w_2)\ \ \forall w_1, w_2$ in the domain. (See Fig. 4)
- If f is differentiable, then f is convex if and only if its domain is convex and $\forall w_1, w_2$ in the domain.(monotone mapping)

  $(\bigtriangledown f(w_1) - \bigtriangledown f(w_2))^\top (w_1 - w_2) \geq 0.$
- If f is twice differentiable, then f is convex if and only if its domain is convex and $\bigtriangledown^2 f(w) \succeq 0\ \ \forall w$in the domain. (It is not working if your function is not twice differentiable.)

Examples of convex functions are given as follow:

- Exponential: $exp(ax)$ is convex on x $\in \mathbb{R}\ \ \forall$ a $\in \mathbb{R}$
- Negative Logarithm: $-log(x)$ is convex on $x \in \mathbb{R}_{++}$
- Negative Entropy: $-H(x) = xlog(x)$ is convex on $x \in \mathbb{R}_{++}$
- Norms: $\|w\|_p$ for $p \leq 1$
- Log-Sum-Exp: $log(exp(w_1) + ... + exp(w_d))$

There are some operations which preserver convexity:

- Non-negative weighted sums: $\alpha_i \geq 0$; if f_i convex $\forall i$, so is

$$g = \alpha_1 f_1 + \alpha_2 f_2 + \cdots \tag{8}$$

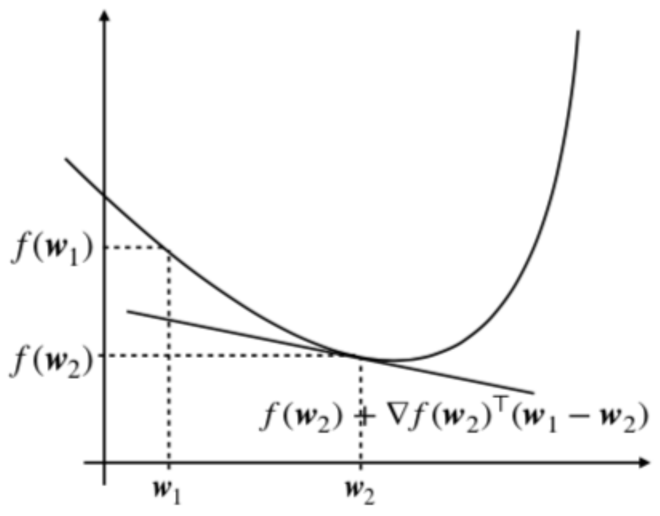
$f(\boldsymbol{w}_1)$
$f(\boldsymbol{w}_2)$
$f(\boldsymbol{w}_2) + \nabla f(\boldsymbol{w}_2)^\top(\boldsymbol{w}_1 - \boldsymbol{w}_2)$
$\boldsymbol{w}_1$
$\boldsymbol{w}_2$

Figure 5: If f is differentiable, then f is convex if and only if its domain is convex and $f(w_1) \geq f(w_2) + \bigtriangledown f(w_2)^\top(w_1 - w_2)\ \ \forall w_1, w_2$ in the domain.

- Composition with an affine mapping: if f is convex, so is

$$g(w) = f(Aw + b) \tag{9}$$

- Pointwise maximum: if f1, f2 are convex, so is

$$g(w) = \max\{f_1(w), f_2(w)\} \tag{10}$$

- $log(1 + exp(x))$ is convex for $x \in \mathbb{R}$. (This can be applied to logistic regression)

Optimality of convex optimization It is guranteed to find the global minimun for convex functions because any local ooptimal has to be global optimal for convex functions (there is no local optimal existing for convex functions), which makes convex optimization special.

- A point w^* is locally optimal if $f(w^*) \leq f(w)\ \ \forall w$ in a neighborhood of w^* ; globally optimal if $f(w^*) \leq f(w)\ \ \forall w$.
- For convex problems global optimality follows directly from local optimality.
- For a local minimum of f, $\bigtriangledown f(w^*) = 0$. If f convex, then $\bigtriangledown f(w^*) = 0$ sufficient for global optimality.

1.2 How do we search for the optimum?

Intuition is the descent method (find a stationary point with $\bigtriangledown f(w) = 0$) for $\min_w f(w)$. Next, we will talk about algorithms to search for the optimum.

Iterative algorithm:

- Start with some guess w

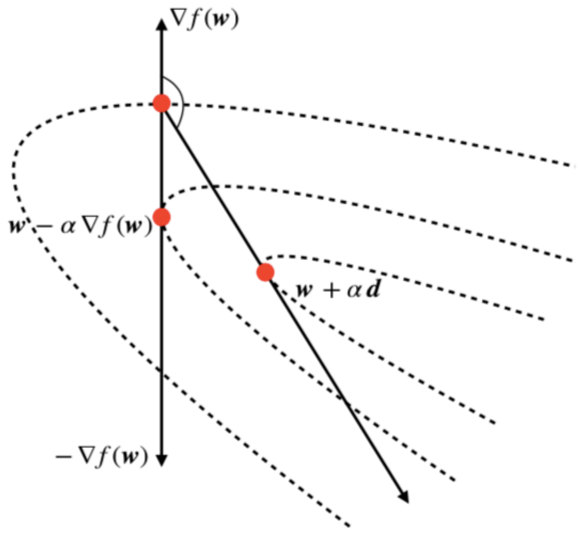
$\nabla f(\boldsymbol{w})$
$\boldsymbol{w} - \alpha \nabla f(\boldsymbol{w})$
$\boldsymbol{w} + \alpha \boldsymbol{d}$
$-\nabla f(\boldsymbol{w})$

Figure 6: When choosing direction d, $w + \alpha d$ must be negtive.

- Iterate k = 1, 2, 3, . . .
  1. Select direction d_k and stepsize α_k
  2. $w \leftarrow w + \alpha_k d_k$
  3. Check whether we should stop (e.g., if $\bigtriangledown f(w) \approx 0$)

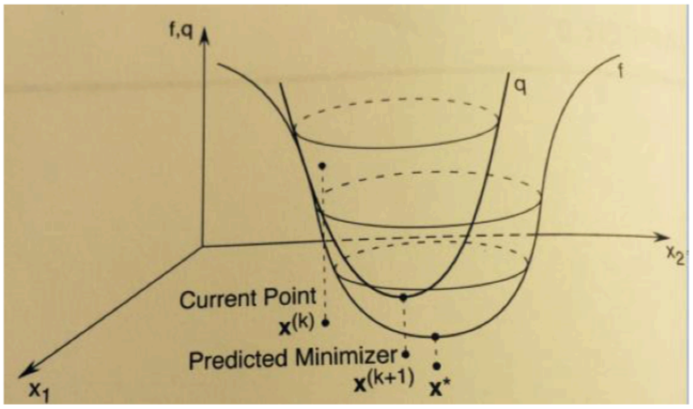

Figure 7: Explanation for Newton's method. We use w in our lecture instead of x in this figure. Cited from [2]

- Descent direction d_k satisfies $\bigtriangledown f(w)^\top d_k < 0$
- d_k: Direction to move along at iteration k.
- α_k: Step size at iteration k.

If we directly go to the $-\bigtriangledown f(w)$ direction, we might meet the local minimum. With the help of d, we can change the direction and find the global minimum. A constraint is that when choosing direction d, $w + \alpha d$ should always be negtive (See Fig. 7).

How to select direction:

- Steepest descent: $d_k = - \bigtriangledown f(w_k)$
- Scaled gradient: $d_k = -D_k \bigtriangledown f(w_k)$ for $D_k \succ 0$
- E.g., Newton's method: $D_k = [\bigtriangledown^2 f(w_k)]^{-1}$

  In next part, I cited the explanation of Newton's method from [2]. In practice, Newton's method can converge with much fewer iterations than gradient methods.[2] For example, for quadratic functions, while we saw that gradient methods can zigzag for a long time (depending on the underlying condition number), Newton's method will always get the optimal solution in a single step.[2] Notation $\bigtriangledown f(w_k)$ and $\bigtriangledown^2 f(w_k)$ respectively denote the gradient and the Hessian at w_k. Iteration only well-defined when the Hessian at w_k is invertible.[2]

  Where does this come from?

  One motivation: minimizing a quadratic approximation of the function sequentially. (See Fig. 8) Around the current iterate w_k, let's Taylor expand our function to second order and minimize the resulting quadratic function.[2]

  $f(w) \approx f(w_k) + \bigtriangledown f(w_k)^\top (w - w_k) + \frac{1}{2}(w - w_k)^\top \bigtriangledown^2 f(w_k)(w - w_k)$

  We let

  $q(w) = f(w_k) + \bigtriangledown f(w_k)^\top (w - w_k) + \frac{1}{2}(w - w_k)^\top \bigtriangledown^2 f(w_k)(w - w_k)$

  What is the minimum of this function?

  First order necessary condition tells us that $\bigtriangledown q(w^*) = 0$.

  $Hence, \bigtriangledown f(w_k) + \bigtriangledown^2 f(w_k) w - \bigtriangledown^2 f(w_k) w_k = 0$

  $w = w_k - (\bigtriangledown^2 f(w_k))^{-1} \bigtriangledown f(w_k)$

  If $\bigtriangledown^2 f(w_k) \succ 0$, then q is convex and hence our stationary point is a global optimum. Newton's method picks this point as the next iterate.[2]

How to select stepsize:

- It is important to select a good α_k because it may not converge if α_k is too large and may be too slow if α_k is too small.
- Exact Line Search: $\alpha_k = arg\ min_{\alpha \geq 0}\ \ f(w_k + \alpha d_k)$
- Constant: $\alpha_k = 1/L$ (for suitable L)
- Diminishing: $\alpha_k \to 0$ but $\sum_k \alpha_k = \infty (e.g., \alpha_k = 1/k)$
- Armijo Rule: Start with α = s and continue with $\alpha = \beta$s, $\alpha = \beta^2 s$, ..., until $\alpha = \beta^m s$ falls within the set of α with $f(w_k + \alpha d_k) - f(w_k) \leq \sigma\alpha \bigtriangledown f(w_k)^\top d_k$, In this case, we can find the α with which we can make enough progress to the optimum.

1.3 How long does it take to find the optimum?

For a convex function, gradient descent will converge to a global optimum.

The goal of this section is to figure out how many iterations k for $f(w_k) - f(w^*) \leq \epsilon$. We choose some $\epsilon, f(w^*)$ is the global minimum. Now we want to know how many k we need (how long we have to run) to close to the global minimum.

In order to answer this question, there are two important properties we need to know first: lipschitz continuous gradient and strong convexity. We will see what they mean and how they are going to help us derive convergence rates if our functions satisfy those two properties.

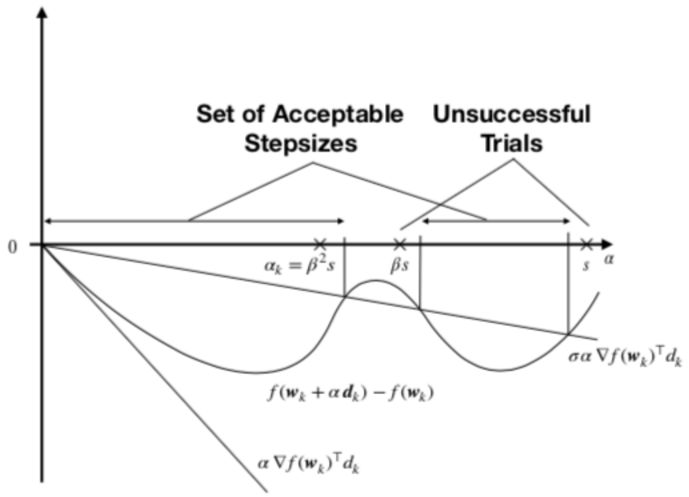
Set of Acceptable Stepsizes
Unsuccessful Trials
0
$\alpha_k = \beta^2 s$
βs
s
α
$\sigma\alpha\nabla f(\mathbf{w}_k)^\top d_k$
$f(\mathbf{w}_k + \alpha \mathbf{d}_k) - f(\mathbf{w}_k)$
$\alpha\nabla f(\mathbf{w}_k)^\top d_k$

Figure 8: Explanation for Armijo Rule, $f(w_k)$ is the horizontal axis here, $f(w_k + \alpha d_k$ is the polinomial function below the horizontal axis. β means how quickly we are moving to the original point. α means how much progress we make to the optimum.

Lipschitz continuous gradient means the gradient doesn't change frastically between two points. We assume f(w) has Lipschitz continuous gradient. The form of the function as follows:

$$\|\triangledown f(w_1) - \triangledown f(w_2)\|_2 \leq L\|w_1 - w_2\|_2 \quad \forall w_1, w_2 \tag{11}$$

So why is this property important and how can we use this property? Intuition: if f(w) has Lipschitz continuous gradient, then $g(w) = \frac{L}{2}\|w\|_2^2 - f(w)$ convex.

We will give the proof that $g(w)$ is convex as follows:

$$(\triangledown f(w_1) - \triangledown f(w_2))^\top (w_1 - w_2) \leq \|\triangledown f(w_1) - \triangledown f(w_2)\|_2 \|w_1 - w_2\|_2 \leq L\|w_1 - w_2\|_2^2 \tag{12}$$

$$\triangledown g(w_1) - \triangledown g(w_2) = L(w_1 - w_2) - (\triangledown f(w_1) - \triangledown f(w_2)) \tag{13}$$

$$(\triangledown g(w_1) - \triangledown g(w_2))^\top (w_1 - w_2) = L\|w_1 - w_2\|_2^2 - (\triangledown f(w_1) - \triangledown f(w_2))^\top (w_1 - w_2) \geq 0 \tag{14}$$

(Eq. 14 utilize the property of monotone mapping we mentioned before.)

If $g(w) = \frac{L}{2}\|w\|_2^2 - f(w)$ convex, then $f(w_2) \leq f(w_1) + \triangledown f(w_1)^\top (w_2 - w_1) + \frac{L}{2}\|w_2 - w_1\|_2^2$, $\forall w_1, w_2$

Proof: plug definition of g into $g(w_2) \geq g(w_1) + \nabla g(w_1)^\top (w_2 - w_1)$

To find how long it takes to find the optimum, we need to figure out how many iterations k for $f(w_k) - f(w^*) \leq \epsilon$ and $w_{k+1} = w_k + \alpha d_k$, so how to pick αd_k? We minimize w.r.t. w_{k+1} right-hand-side of upper bound $f(w_{k+1}) \leq f(w_k) + \triangledown f(w_k)^\top (w_{k+1} - w_k) + \frac{L}{2}\|w_{k+1} - w_k\|_2^2$, Hence

$$\alpha d_k = -\frac{1}{L}\nabla f(w_k) \tag{15}$$

$$f(w_{k+1}) \leq f(w_k) - \frac{1}{2L}\|\nabla f(w_k)\|_2^2 \tag{16}$$

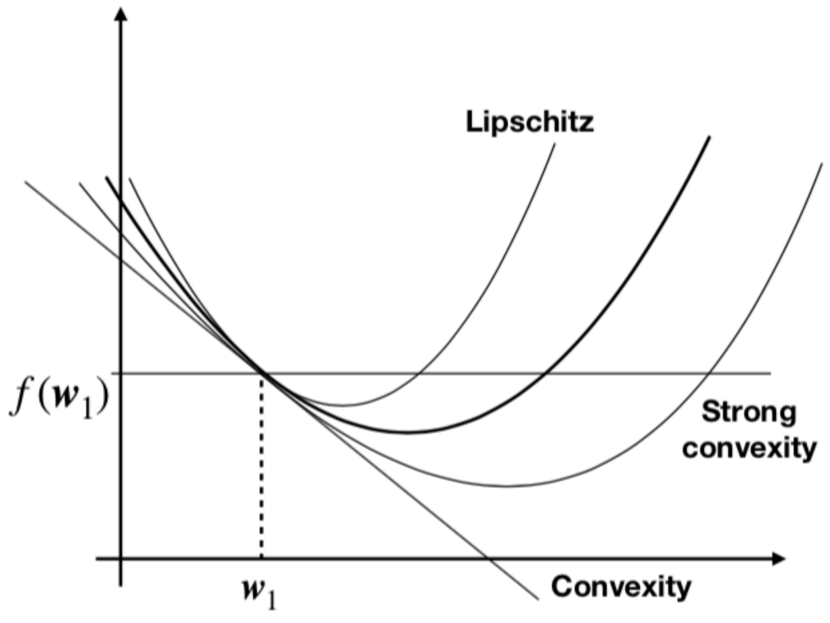

Figure 9: if f twice differentiable, $\sigma I \prec \nabla^2 f(w) \prec LI, \forall w$

The above is bound on guaranteed progress.
Properties for strong convexity is $f(w_2) \geq f(w_1) + \bigtriangledown f(w_1)^\top (w_2 - w_1) + \frac{\sigma}{2} \left\| w_2 - w_1 \right\|_2^2$, $\forall w_1, w_2$
Bound on sub-optimality from strong convexity: $f(w^*) \geq f(w_k) - \frac{1}{2\sigma} \left\| \nabla f(w_k) \right\|_2^2$
So in guaranteed progress:
$f(w_k) - f(w^*) \leq f(w_{k-1}) - f(w^*) - \frac{\sigma}{L}(f(w_{k-1}) - f(w^*)) \leq (1 - \frac{\sigma}{L})^k (f(w_0) - f(w^*)), (\sigma < L)$
Rate: $c(1 - \frac{\sigma}{L})^k \leq \epsilon \Rightarrow k \geq O(log(1/\epsilon))$, sometimes $O(e^k)$
Next, we consider this assumption with no strong convexity:
Lipschitz bound and $w_2 = w_1 \alpha \nabla f(w_1)$ yields $f(w_2) \leq f(w_1) - (1 - \frac{L\alpha}{2}) \left\| \bigtriangledown f(w_1) \right\|_2^2$
Combined with convexity: $f(w_1) + \nabla f(w_1)^\top (w^* - w_1) \leq f(w^*)$, we can get:

$$f(w_2) \leq f(w^*) + \bigtriangledown f(w_1)(w_1 - w^*) - \frac{\sigma}{2} \left\| \nabla f(w_1) \right\|_2^2 \tag{17}$$

using $w_2 - w_1 = -\alpha \nabla f(w_1)$ and rearranging terms gives

$$f(w_2) \leq f(w^*) + \frac{1}{2\alpha}(\left\| w_1 - w^* \right\|_2^2 - \left\| w_2 - w^* \right\|_2^2) \tag{18}$$

Summing over all iterations $\sum_{i=1}^{k} (f(w_i) - f(w^*)) \leq \frac{1}{2\alpha} \left\| w_0 - w^* \right\|_2^2$
$f(w_i)$ is non-increasing, so we can get:

$$f(w_k) - f(w^*) \leq \frac{1}{k} \sum_{i=1}^{k} (f(w_i) - f(w^*)) \leq \frac{1}{2k\alpha} \left\| w_0 - w^* \right\|_2^2 \leq \epsilon \tag{19}$$

Consequently: $k \geq O(1/\epsilon)$. Iteration complexity is linear in the number of samples $|D|$.

A large dataset will make gradient computation slow. So far we didn't consider the time for computing the gradient. Then we talk about Stochastic gradient descent which consider a subset of samples and approximate the gradient based on this batch of data.

- Select a subset of samples $_k$
- Gradient update using approximation $\nabla f(w) \approx \sum_{(x^{(i)},y^{(i)})\in_k} \nabla\nabla(y^{(i)}, F(x^{(i)}, w))$

Convergence rates for stochastic gradient descent:

- Lipschitz continuous gradient and strongly convex: $k \geq O(1/\epsilon)$
- Lipschitz continuous gradient: $k \geq O(1/\epsilon^2)$

To sum up, we compare Stochastic and deterministic (in strongly convex condition) gradient descent as follows:

Batch gradient descent:

- Convergence rate: O(log 1/ϵ)
- Iteration complexity: linear in $|D|$

Stochastic gradient descent:

- Convergence rate: O(1/ϵ)
- Iteration complexity: independent of $|D|$

We can see Convergence rate of Batch gradient descent and Iteration complexity of Stochastic gradient descent have better properties respectively. Can we get the best of both worlds?

There are many other related algorithms:

- SAG (Le Roux, Schmidt, Bach 2012)
- SDCA (Shalev-Shwartz and Zhang 2013)
- SVRG (Johnnson and Zhang 2013)
- MISO (Mairal 2015)
- Finito (Defazio 2014)
- SAGA (Defazio, Bach, Lacoste-Julien 2014)

Take SVRG for example, its steps are as follow:

- Initialize $\hat{w}$
- For epoch 1, 2, 3, ...
- Compute $\nabla f(\hat{w}) = \sum_{i\in D} \nabla_i(\hat{w})$
- Initialize w_0= $\hat{w}$
- For t in length of epochs $w_t = w_{t-1} - \alpha \left[\nabla f(\hat{w}) + \nabla_{i(t)}(w_{t-1}) - \nabla_{i(t)}(\hat{w})\right]$

- Update $\hat{w} = w_t$
- Output $\hat{w}$

Reference

1. Wikipedia, https://zh.wikipedia.org/w/index.php?title=%E9%9E%8D%E9%BB%9E&oldid=49262338, 2018.
2. Amir Ali Ahmadi, Princeton Lecture, http://www.princeton.edu/~amirali/ Public/Teaching/ORF363_COS323/F14/ORF363_COS323_F14_Lec9.pdf, 2014Fall.