M2177.0043 Introduction to Deep Learning

Lecture 4: Optimization[1]

Hyun Oh Song[1]

[1] Dept. of Computer Science and Engineering, Seoul National University

March 26, 2020

[1] Many slides and figures adapted from Stephen Boyd

Last time

- Mathematical optimization
- Unconstrained optimization
- Gradient descent
- Newton method

Outline

Constrained minimization problems

Online method

Preliminary: Why do descent methods work?

$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)} \quad \text{with } f(x^{(k+1)}) < f(x^{(k)})$$

- Δx is the *step*, or *search direction*
- t is the *step size*, or *step length*

General descent method.

given a starting point $x \in \mathrm{dom}\, f$.

repeat

1. Determine a descent direction Δx
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.

Descent direction

Definition 1 (Descent direction)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function over $\mathbb{R}^n$. A vector $0 \neq d \in \mathbb{R}^n$ is called a **descent direction** of f at x if the directional derivative $\nabla f(x)^\intercal d$ is negative, meaning that

$$\nabla f(x)^\intercal d < 0$$

- The most important property of descent direction is that taking small enough steps along these directions lead to a decrease of the objective function.

Descent property of descent directions

Lemma 2 (Descent property of descent directions)

Let f be a continuously differentiable function over $\mathbb{R}^n$, and let $x \in \mathbb{R}^n$. Suppose that d is a descent direction of f at x. Then there exists $\epsilon > 0$ such that

$$f(x + td) < f(x)$$

for any $t \in (0, \epsilon]$.

Descent property of descent directions

Proof

Since $\nabla f(x)^\intercal d < 0$, it follows from the definition of the directional derivative that

$$\lim_{t \to 0^+} \frac{f(x+td) - f(x)}{t} < 0$$

. Therefore, there exists an $\epsilon > 0$ such that

$$\frac{f(x+td) - f(x)}{t} < 0$$

for any $t \in (0, \epsilon]$, which implies the desired result. ■

What about constraints?

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & x \in \mathcal{C} \end{array}$$

- Need to find the minimizer in the constraint set $x \in \mathcal{C}$
- Projected gradient descent (covered today), Frank-Wolfe (conditional gradient) algorithm, *etc.*

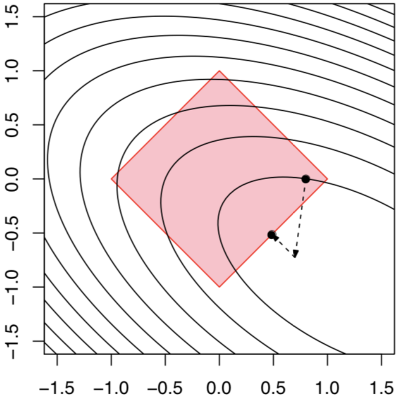

Figure: from R. Tibshirani 10-725

Optimality condition

Theorem 3 (Optimality Condition)

Let f be a continuously differentiable function over a closed convex set $\mathcal{C}$ and let x^* be a local minimum of f. Then, $\nabla f(x^*)^\intercal (x - x^*) \geqslant 0$ for any $x \in \mathcal{C}$.

- Note, the theorem does not assume convexity of f
- We assume convexity of the constrain set $\mathcal{C}$ instead

Proof of the optimality condition theorem

Proof.

Proof by contradiction. Assume in contradiction that x^* is a local minimum and there exists $x \in \mathcal{C}$ such that $\nabla f(x^*)^\intercal(x - x^*) < 0$. Let $d = x - x^*$, then we see that d is a descent direction. By "Descent property of descent direction Lemma", it follows that there exists $\epsilon \in (0, 1)$ such that $f(x^* + td) < f(x^*)$ for all $t \in (0, \epsilon]$.

It remains to show that we are still in the set $\mathcal{C}$ after this descent step. Since $\mathcal{C}$ is convex we have that $x^* + td = (1 - t)x^* + tx \in \mathcal{C}$, leading to conclusion that x^* is *not* a local optimum point, in contradiction to the assumption that x^* is a local minimum point. ■

Example $\mathcal{C} = \mathbb{R}^n$

- If $\mathcal{C} = \mathbb{R}^n$, then a local minimum x^* satisfies $\nabla f(x^*)^\intercal (x - x^*) \geqslant 0$ for all $x \in \mathbb{R}^n$. Plugging in $x = x^* - \nabla f(x^*)$ into the inequality, we get $-\|\nabla f(x^*)\|^2 \geqslant 0$, implying that

$$\nabla f(x^*) = 0$$

- Therefore, it follows that the notion of a stationary point of a function and a local minimum of a minimization problem coincide when the problem is unconstrained.

Example $\mathcal{C} = \mathbb{R}^n_+$

Left as an exercise.

Intermezzo

- We have looked at the characterization of optimal points with the constraints
- This can serve as the stopping criteria in the optimization algorithm
- So what is the algorithm for finding the local optimal points under constraints?

Projected gradient descent

Enforce that the points are feasible by projecting onto $\mathcal{C}$

Projected gradient descent

given a starting point $x \in \mathrm{dom} f$.

repeat

1. *Compute a descent direction* Δx
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.
4. *Projection onto* $\mathcal{C}$. $x := \Pi_{\mathcal{C}}(x)$

until stopping criterion is satisfied.

The projection operator $\Pi_{\mathcal{C}}$ onto $\mathcal{C}$

$$\Pi_{\mathcal{C}}(x) = \min_{z \in \mathcal{C}} \|x - z\|_2$$

Projection onto $\mathcal{C}$, $\quad \Pi_{\mathcal{C}}(x) = \min_{z \in \mathcal{C}} \|x - z\|_2$

- $C = \{x \in \mathbb{R}^n \mid \|x\|_2 \leqslant 1\}$, closed form

$$z^* = \frac{x}{\max\{1, \|x\|\}}$$

[Proof] https://math.stackexchange.com/questions/627034/orthogonal-projection-onto-the-l-2-unit-ball

- $C = \{x \in \mathbb{R}^n \mid \|x\|_\infty \leqslant 1\}$, closed form

$$z_i^* = \begin{cases} -1 & x_i < -1 \\ x_i & -1 \leqslant x_i \leqslant 1 \\ 1 & x_i > 1 \end{cases}$$

[Proof] https://math.stackexchange.com/questions/1825747/orthogonal-projection-onto-the-l-infty-unit-ball

- $C = \{x \in \mathbb{R}^n \mid \|x\|_1 \leqslant 1\},\ O(n \log n)$ algorithm
  [Proof] https://math.stackexchange.com/questions/2327504/orthogonal-projection-onto-the-l-1-unit-ball
- $C = \{x \in \mathbb{R}^n \mid \sum_i x_i = 1, x_i \geqslant 0 \ \ \forall i \in [n]\},\ O(n \log n)$ algorithm
  [Proof] https://math.stackexchange.com/questions/2402504/orthogonal-projection-onto-the-unit-simplex

Convergence properties

- What can we say about the convergence properties of the (projected) gradient methods?
- In general, this is very difficult unless we know somethings about the function
- We get stronger convergence bounds as we know more about the function. *i.e.* Convexity, $L-$Liptshitz, $\alpha-$strong convexity, $\beta-$smoothness, etc.
- Take my graduate machine learning class for formal proofs
- For this class, we will not assume anything about the function for generality and applicability to deep learning

Convergence property of convex f

Theorem 4

Assume that function f is convex, differentiable, and L-Lipschitz over the convex domain Ω. Let R be the upper bound on the distance $\|x_1 - x^*\|_2$ from the initial point x_1 to an optimal point $x^* \in \arg\min_{x\in\Omega} f(x)$. Let $x_1, \ldots, x_t$ be the sequence of iterates computed by t steps of projected gradient descent with constant step size $\eta = \frac{R}{L\sqrt{t}}$. Then,

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f\left(x^*\right) \leqslant \frac{RL}{\sqrt{t}}.$$

- This means that the difference between the functional value of the average point during the optimization process from the optimal value is bounded above by a constant proportional to $\frac{1}{\sqrt{t}}$.

Outline

Stochastic gradient descent

- Consider minimizing an objective function that has the form of a sum of functions:
$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

- Each summand function $f_i(x)$ is typically associated with the i-th observation in a dataset. The standard (or "batch") gradient descent method would perform the following iterations:
$$x := x - \eta \nabla f(x) = x - \eta \sum_{i=1}^{n} \frac{1}{n} \nabla f_i(x)$$

- Computing the gradient can be very expensive if n is large. Stochastic gradient descent method *samples* the summand functions at every step for scalability.
$$x := x - \eta \boxed{\nabla f_i(x)}$$

Minibatch SGD

- A compromise between computing the true gradient and the stochastic gradient of a single example
- Reduces the variance of the gradient estimate
- In practice, vector-process the gradient computation so the minibatch size fits in the memory

Minibatch SGD

Minibatch stochastic gradient descent

given a starting point $x \in \mathrm{dom} f$.

repeat

1. *Shuffle the data*
2. **For every** m **sequence of data** $i = 1, \ldots, \lceil n/m \rceil$
    a. *Compute the minibatch gradient.* $\Delta x = \frac{1}{m} \sum_{j=1}^{m} \nabla f_j(x)$
    b. *Update the stepsize.* $t := Update(t)$.
    c. *Update.* $x := x + t\Delta x$.
3. *Decay the stepsize*

until stopping criterion is satisfied.

Distributed gradient descent for full batch gradient descent

- **Map:** compute gradient on subblock and emit
  Reduce: aggregate parts of the gradients
-

---

given a starting point $x \in \operatorname{dom} f$.

repeat

1. *Compute the full gradient* $\Delta x_{\mathsf{ds}} := \boxed{\sum_{i=1}^{n} \frac{1}{n} \nabla f_i(x).}$
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x_{\mathsf{ds}}$.

until stopping criterion is satisfied.
