Machine Learning: Optimization 2

Hao Tang

February 2, 2026

Convexity on more points

- If a function f is convex,

$$f(\alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3) \leq \alpha_1 f(x_1) + \alpha_2 f(x_2) + \alpha_3 f(x_3) \tag{1}$$

  for $\alpha_1, \alpha_2, \alpha_3 \geq 0$ and $\alpha_1 + \alpha_2 + \alpha_3 = 1$.

Convexity on more points

- If a function f is convex,

$$f(\alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3) \leq \alpha_1 f(x_1) + \alpha_2 f(x_2) + \alpha_3 f(x_3) \tag{1}$$

for $\alpha_1, \alpha_2, \alpha_3 \geq 0$ and $\alpha_1 + \alpha_2 + \alpha_3 = 1$.

- If a function f is convex,

$$f\left(\sum_{i=1}^{n} \alpha_i x_i\right) \leq \sum_{i=1}^{n} \alpha_i f(x_i) \tag{2}$$

for $\alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i = 1$.

Jensen's inequality

- If a function f is convex,

$$f(\mathbb{E}_{x\sim p(x)}[x]) \leq \mathbb{E}_{x\sim p(x)}[f(x)]. \tag{3}$$

- Jensen's inequality will get used when we talk about expectation maximization (EM).

One optimization strategy

- To find $\min_x f(x)$, the first step is to check whether f is convex.

- The second step is to solve $\nabla_x f(x) = 0$.

A quick reminder on notation again

- We will use x and $f(x)$ for generic optimization, but will use w and $L(w)$ in the context of machine learning (say, optimizing log loss).

- The gradient $\nabla_x f(x)$ should be parenthesized as $(\nabla_x f)(x)$.

- In particular,

$$\nabla_x f(x) \quad (\nabla_x f)(x) \quad D_x f(x) \quad (D_x) f(x)$$

$$\frac{\partial}{\partial x} f(x) \quad \left(\frac{\partial}{\partial x} f\right)(x) \quad \frac{\partial f}{\partial x}(x) \quad \left(\frac{\partial f}{\partial x}\right)(x) \quad \frac{\partial f(x)}{\partial x}$$

  all mean the same thing.

The case for log loss

- The log loss in the binary case is

$$L(w) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \exp(-y_i w^\top x_i)\right). \tag{4}$$

- We have shown that L is convex in w.

The case for log loss

$$\nabla_w L(w) = \frac{1}{n}\sum_{i=1}^{n} \frac{\exp(-y_i w^\top x_i)}{1+\exp(-y_i w^\top x_i)}(-y_i x_i) \tag{5}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \left(1 - \frac{1}{1+\exp(-y_i w^\top x_i)}\right)(-y_i x_i) \tag{6}$$

$$= \frac{1}{n}\sum_{i=1}^{n} (1 - p(y_i|x_i))\,(-y_i x_i) \tag{7}$$

We need a new optimization strategy

- What happens when we cannot solve $\nabla_x f(x) = 0$?
- Do we need to get to the optimal solution?
- Can we get an approximate solution? What does it mean to approximate?

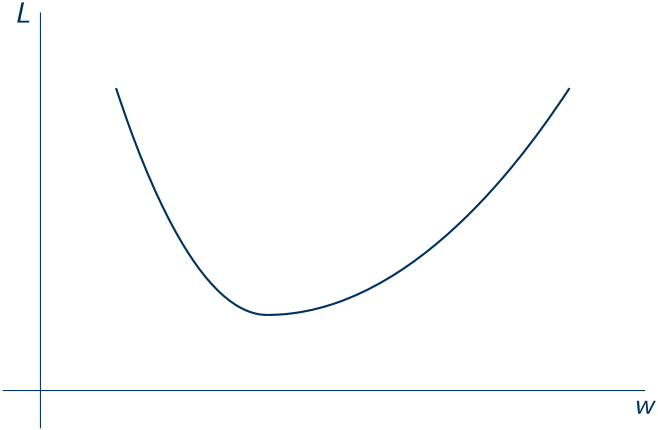
Gradient descent

- Gradient descent is an iterative algorithm that tries to lower the objective value by following the gradient.

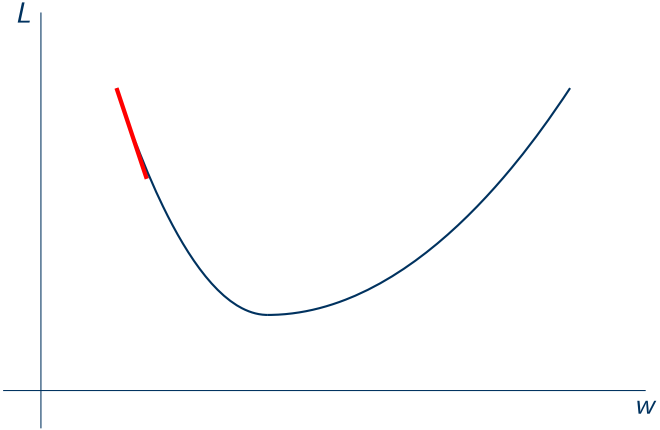
  for $t = 1, 2, \ldots$ **do**
  $x_{t+1} = x_t - \eta_t \nabla_x f(x_t)$
  end for

- The variable $\eta_t > 0$ is called the **step size** (or learning rate), and can depend on t.

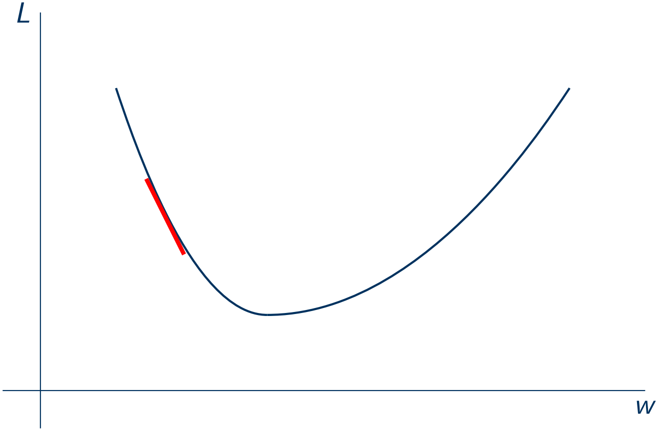
Gradient descent



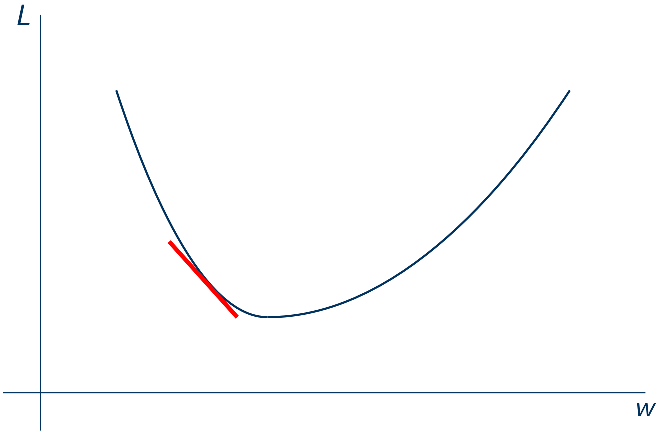
Gradient descent



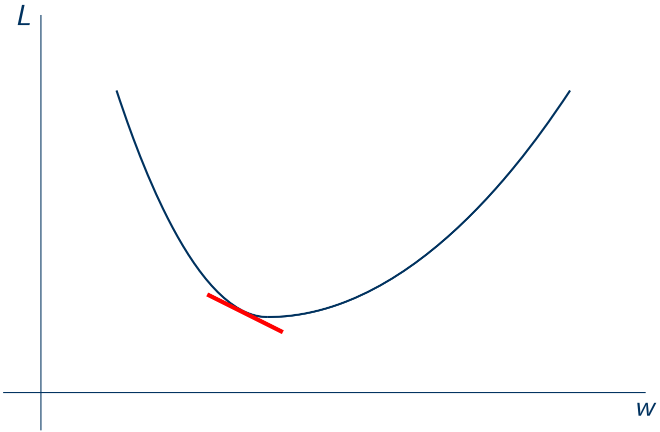
Gradient descent



Gradient descent



Gradient descent



Gradient descent on log loss

- The update rule for gradient descent on log loss is

$$w_{t+1} = w_t - \eta_t \nabla L(w_t) \tag{8}$$

$$w_{t+1} = w_t - \eta_t \frac{1}{n} \sum_{i=1}^{n} \left(1 - \frac{1}{1 + \exp(-y_i w_t^\top x_i)}\right) (-y_i x_i). \tag{9}$$

- Note that ∇L is a function of w_t.
- Note that $1/n$ can technically be subsumed into η_t.

Problems with gradient descent

- Each update needs to go over the entire data set once.
- A single update takes $O(nd)$, where d is the dimension of w and n is the number of samples in the data set.
- This scales poorly, especially when the data set is large.

A potential solution

- We can rewrite the gradient as

$$L(w) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \exp(-y_i w^\top x_i)\right) = \frac{1}{n}\sum_{i=1}^{n} \ell(w; x_i, y_i), \tag{10}$$

where $\ell(w; x, y) = \log(1 + \exp(-yw^\top x))$.

- We can now treat the loss as an expectation because

$$L(w) = \sum_{i=1}^{n} \frac{1}{n}\ell(w; x_i, y_i) = \mathbb{E}_{(x,y)\sim U(S)}[\ell(w; x, y)], \tag{11}$$

where $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ is our data set and the expectation $\mathbb{E}_{(x,y)\sim U(S)}$ is taken uniformly over the samples in S.

A potential solution

- We also treat the gradient as an expectation because

$$\nabla L(w) = \nabla \mathbb{E}_{(x,y)\sim S}[\ell(w; x, y)] = \mathbb{E}_{(x,y)\sim S}[\nabla \ell(w; x, y)]. \tag{12}$$

- The gradient is an estimate with the entire data set.

- What happens if we estimate the gradient with a smaller set?

Estimating the gradient

- Choose a subset of B indices $I = \{i_1, i_2, \ldots, i_B\}$ uniformly from $\{1, 2, \ldots, n\}$.
- In expectation, we have an unbiased estimation of the gradient.

$$\mathbb{E}_I \left[\frac{1}{|I|} \sum_{i \in I} \nabla \ell(w; x_i, y_i) \right] = \mathbb{E}_{(x,y) \sim U(S)} [\nabla \ell(w; x, y)] \tag{13}$$

- This holds even when the size of the subset $B = 1$.

Stochastic gradient descent

- Instead of computing the gradient on the entire data set, we choose a subset of samples S_B uniformly at random (with replacement) from the the data set S.

for t = 1, 2, . . . **do**
  Choose a subset of samples $S_B \subseteq S$ uniformly at random
  $w_{t+1} = w_t - \eta_t \frac{1}{B} \sum_{(x,y) \in S_B} \nabla \ell(w_t; x, y)$
end for

- The subset of samples is sometimes called a **mini-batch**.
- When $S_B = S$, this falls back to gradient descent, and is sometimes also called full-batch gradient descent.

Stochastic gradient descent

- In practice, we don't actually sample a subset S_B at random.
- Instead, we group samples in the data set into non-overlapping mini-batches of size B.
- We then go over all mini-batches at a random order.
- All samples are guaranteed seen when we do a pass over the data set.
- A pass over the data set is called an **epoch**.
- We usually need multiple epochs to get a satisfactory result.

Evaluating (stochastic) gradient descent

- When do we know the result is satisfactory?
- Is stochastic gradient descent always better?
- How do we compare gradient descent and stochastic gradient descent?

Approximate solutions in optimization

- We say that $\hat{x}$ is an approximate solution of the minimizer x^* if, for a given $\epsilon > 0$,

$$f(\hat{x}) - f(x^*) < \epsilon. \tag{14}$$

- Note that it is close in function value, not close in the input.

Approximate solutions for iterative algorithms

- An iterative algorithm creates a sequence $x_1, \dots, x_t$.
- How many updates do we need to achieve an approximate solution?
- Given $\epsilon > 0$, how large does t needs to be to achieve

$$f(x_t) - f(x^*) < \epsilon? \tag{15}$$

- We want to express ϵ as a function of t.

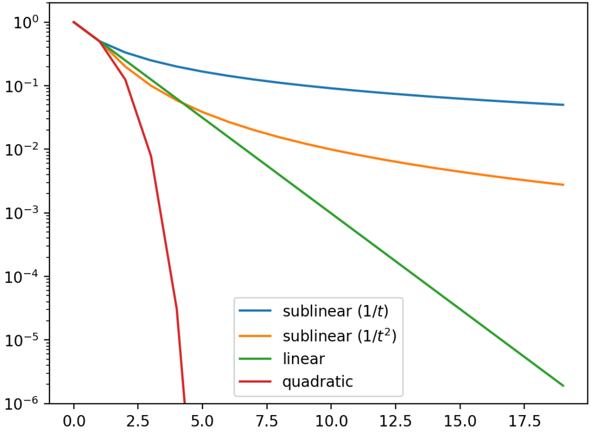
Potential results

- Sublinear
  - $f(x_t) - f(x^*) \leq \frac{c}{t^2}$
- Linear
  - $f(x_t) - f(x^*) \leq cr^t$ for $0 < r < 1$
- Quadratic
  - $f(x_t) - f(x^*) \leq cr^{2^t}$ for $0 < r < 1$

Potential results

- Sublinear
  - $f(x_t) - f(x^*) \leq \frac{c}{t^2}$
  - $\epsilon = O\left(\frac{1}{t^2}\right)$ or $t = O(\frac{1}{\sqrt{\epsilon}})$
- Linear
  - $f(x_t) - f(x^*) \leq cr^t$ for $0 < r < 1$
  - $\epsilon = O\left(2^{-t}\right)$ or $t = O(\log \frac{1}{\epsilon})$
- Quadratic
  - $f(x_t) - f(x^*) \leq cr^{2^t}$ for $0 < r < 1$
  - $\epsilon = O\left(2^{-2^t}\right)$ or $t = O(\log \log \frac{1}{\epsilon})$

Convergence rates



Convergence results

- Convergence results come with assumptions.
- Usually, stronger assumptions lead to faster convergence.
- Many assumptions are beyond the scope of this course.
- When reading the convergence results, focus on the convergence rates.

Guarantees for gradient descent

- If we do gradient descent on a M-smooth, μ-strongly convex function with $\eta = 1/2M$, then

$$f(x_t) - f(x^*) \leq \left(1 - \frac{\mu}{2M}\right)^t (f(x_0) - f(x^*)). \tag{16}$$

Guarantees for gradient descent

- If we do gradient descent on a M-smooth, μ-strongly convex function with $\eta = 1/2M$, then

$$f(x_t) - f(x^*) \leq \left(1 - \frac{\mu}{2M}\right)^t (f(x_0) - f(x^*)). \tag{16}$$

- The convergence rate in this case $O(2^{-t})$ is linear.

Guarantees for gradient descent

- If we do gradient descent on a M-smooth convex function with $\eta \leq 1/M$, then

$$f(x_t) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\eta t} \tag{17}$$

Guarantees for gradient descent

- If we do gradient descent on a M-smooth convex function with $\eta \leq 1/M$, then

$$f(x_t) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\eta t} \tag{17}$$

- The convergence rate in this case $O(1/t)$ is sublinear.

Guarantees for stochastic gradient descent

- If we do SGD on a convex function with $\eta = \frac{\|w_0 - w^*\|_2}{R\sqrt{t}}$, then

$$\mathbb{E}_{x,y \sim U(S)}[L(\bar{w}_t)] - L(w^*) \leq \frac{\|w_0 - w^*\| R}{\sqrt{t}} \tag{18}$$

  where $\|\nabla \ell(w_t; x, y)\|_2 \leq R$ for any t, x, and y, and $\bar{w}_t = \frac{w_1 + \cdots + w_t}{t}$.

- The convergence rate is $O(1/\sqrt{t})$, independent of the data set size n!

Guarantees for stochastic gradient descent

- If we do SGD on a M-smooth convex function, then

$$\mathbb{E}_{x,y\sim U(S)}[L(\bar{w}_t)] - L(w^*) \leq 2\frac{DR}{\sqrt{t}} + \frac{BMD^2}{t} \tag{19}$$

  where $\|\nabla \ell(w_t; x, y)\|_2 \leq R$ for any t, x, and y, $\bar{w}_t = \frac{w_1 + \cdots + w_t}{t}$, and $\|w - w'\| \leq D$ for all reachable w and w'.

- Note where the size of the mini-batch B is.
- When $B = O(\sqrt{t})$, the convergence is $O(1/\sqrt{t})$.
- A mini-batch size B bigger than $O(\sqrt{t})$ might give a slower convergence.