OPTIMIZATION FOR MACHINE LEARNING

November 4, 2024

This week: Stochastic gradient!

Today : Basics (+ early stopping?)

Course project: Coming up!

STOCHASTIC GRADIENT METHODS

AKA SGD

Motivation: • Most problems in ML involve data, and have the typical form

Loss function: measures agreement between model & data

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(a_i; x), y_i)$$

Model parameterized by x

Data (a_i to be matched with y_i)

↳ If all functions in the sum are C^1, then f is C^1

and $\nabla f(x) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x)$ with $f_i(x) = \ell(h(a_i; x), y_i)$

→ Computing $\nabla f(x)$ (for, e.g., performing an iteration of GD) requires to compute all gradients $\nabla f_i(x)$ (sample gradients)

⇒ Can be expensive if $n \gg 1$

⇒ If $\{(a_i, y_i)\}_{i=1..n}$ are sampled from some data distribution, then a good approximation for $\nabla f(x)$ may be found using less than n sample gradients

→ More generally, this observation applies to stochastic optimization problems of the form

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x) = \mathbb{E}_{(a,y)}\left[\ell(h(a;x), y)\right]$$

1 Basics of stochastic gradient

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \; f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

$f_i \in C^1 \; \forall i = 1..n$

$n \geq 1$

Implicitly, f_i depends on the ith data point in a dataset of size n.

Gradient Descent

Start with $x_0 \in \mathbb{R}^d$

Iteration k: $x_{k+1} = x_k - \underbrace{\alpha_k}_{\alpha_k > 0} \nabla f(x_k) = x_k - \frac{\alpha_k}{n} \sum_{i=1}^{n} \nabla f_i(x_k)$

choosing all ∇f_i's deterministically

Stochastic Gradient (basic form) 1952
aka Vanilla SG

Start with $x_0 \in \mathbb{R}^d$

Iteration k: $x_{k+1} = x_k - \underbrace{\alpha_k}_{\alpha_k > 0} \nabla f_{i_k}(x_k) = x_k - \frac{\alpha_k}{1} \nabla f_{i_k}(x_k)$

index drawn randomly within $\{1, \dots, n\}$

Example: The perceptron algorithm

$a_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$

Problem $$\underset{x \in \mathbb{R}^d}{\text{minimize}} \; \frac{1}{n} \sum_{i=1}^{n} \max(1 - y_i a_i^T x, 0)$$

Perceptron algorithm (Rosenblatt) . Start with x_0, $\alpha > 0$

. Iteration k: → Pick a data point (a_{i_k}, y_{i_k}) at random

→ If $\max(1 - y_{i_k} a_{i_k}^T x_k, 0) = 0$, set $x_{k+1} = x_k$.

$\rightarrow$ If $\max(1 - y_{i_k} a_{i_k}^T x_k, 0) = 1 - y_{i_k} a_{i_k}^T x_k > 0$

then set $x_{k+1} = x_k + \alpha\, y_{i_k} a_{i_k}$

$\Rightarrow$ This is a stochastic (sub) gradient method!

NB: Stochastic gradient techniques can be generalized to nonsmooth functions by using subgradients instead of gradients

The stochastic gradient family

- Choosing the random index at every iteration
  - $\rightarrow$ Choose them in cyclic order $\{1, 2, \ldots, n, 1, \ldots, n\}$
    - $\Rightarrow$ Deterministic
    - "epoch" [$\Rightarrow$ Guarantee that every sequence of n iterations corresponds to looking at the entire dataset
  - Random Re-shuffling { $\rightarrow$ Draw a random permutation of $\{1, \ldots, n\}$ $\{\sigma(1), \ldots, \sigma(n)\}$, then use
    $i_0 = \sigma(1)$
    $i_1 = \sigma(2)$
    $\vdots$
    $i_{n-1} = \sigma(n)$
    Then repeat the process every n iterations.
    $\Rightarrow$ Every block of n iterations corresponds to a pass over the entire dataset ("epoch")
  - $\rightarrow$ Draw $\{i_k\}_k$ iid uniformly at random

$\forall k, \quad \mathbb{P}(i_k = 1) = \dots = \mathbb{P}(i_k = m) = \frac{1}{m}$

$\Rightarrow$ Sampling with replacement: no guarantee that the entire dataset has been seen after m iterations

$\Rightarrow$ On average, however, likely that all data points will be seen in comparable proportions

$\Rightarrow$ In that setting, an epoch = m iterations

$\longrightarrow$ Why not sample more than 1 index?

$\Rightarrow$ Batch stochastic gradient methods

Start with $x_0 \in \mathbb{R}^d$

Iteration k

- Sample a set S_k of indices ("batch") within $\{1, \dots, m\}$ with or without replacement.
- $x_{k+1} = x_k - \frac{\alpha_k}{|S_k|} \sum_{i \in S_k} \nabla f_i(x_k)$

$\alpha_k > 0$

- $S_k = \{1, \dots, m\} \ \forall k$ (m samples without replacement)

  $\Rightarrow$ GD / "batch (S)G"

  $x_{k+1} = x_k - \frac{\alpha_k}{m} \sum_{i=1}^{m} \nabla f_i(x_k)$

- $S_k = \{i_k\}$ (1 index per iteration)

  $x_{k+1} = x_k - \alpha_k \nabla f_i(x_k)$

  $\Rightarrow$ Vanilla SG!

$\longrightarrow$ Batch methods include SG and GD as special cases

$\longrightarrow$ Other cases of interest:

$|S_k|$: batch size

- $|S_k| \gg 1$ but not exactly m ("large batch regime")
- $1 < |S_k| \ll m$ ("mini batch regime")

$\longrightarrow$ Often used when evaluations of ∇f_i can be parallelized

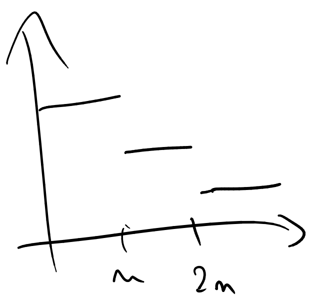
Choose the stepsize (aka learning rate)

→ A constant stepsize can be surprisingly effective in practice!

→ Decreasing stepsizes typically come with the best theoretical guarantees (proposed for the first stochastic gradient method: pick $\{\alpha_k\}_k$ such that $\sum_{k=0}^{\infty} \alpha_k = \infty$, $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$)

(ex): $\frac{1}{k+1}$

→ Hybrid strategies:



. Fix α_k to a constant value for n iterations ("an epoch")

. Decrease that value before starting the next epoch

Variant: Use α for T epochs
then $\alpha/2$ for $2T$ epochs
then $\alpha/4$ for $4T$ epochs
⋮

→ Line search?

Recall: For GD, line search consists in testing values $\{\alpha, \theta\alpha, \theta^2\alpha, \dots\}$ for some $\theta \in (0,1)$ until a value $\theta^j\alpha$ is found such that

$$f(x - \theta^j\alpha \nabla f(x)) < f(x) - c\,\theta^j\alpha \|\nabla f(x)\|^2$$

→ For a stochastic gradient method, line search can be defined in two ways

- Looking for $\alpha > 0$ such that $f(x_k - \alpha \nabla f_{i_k}(x_k)) < f(x_k)$

  $\rightarrow$ Requires to evaluate $f = \frac{1}{m} \sum_{i=1}^{m} f_i$, hence to look at all data points, which defeats the purpose of SG (might have to evaluate ∇f as well)

  $\rightarrow$ Typically not considered in SG methods

- Looking for $\alpha > 0$ such that $f_{i_k}(x_k - \alpha \nabla f_{i_k}(x_k)) < f_{i_k}(x_k)$

  $\rightarrow$ No guarantee that $f(x_k - \alpha \nabla f_{i_k}(x_k)) < f(x_k)$

  $\rightarrow$ Some success in overparameterized models (when the solution satisfies $\nabla f_i(x) = 0 \quad \forall i = 1..m$)

  $\rightarrow$ Used in the PyTorch implementation of L-BFGS

  - For every sample / batch, there is an inner loop of iterations using the same batch and line search

- Advanced aspects (see thursday's lecture)

  $\rightarrow$ Momentum (like in heavy ball)

  $\rightarrow$ Variance reduction (Combine gradient and stochastic gradient steps)

  $\rightarrow$ Output statistics of the iterates

  $\hookrightarrow$ Instead of x_K, output $\frac{1}{K} \sum_{k=0}^{K-1} x_k$ (as in subgradient methods!)

$\hookrightarrow$ Instead of x_K, return x_k $(k \leq K)$ with probability $\dfrac{\alpha_k}{\sum_{h=0}^{K-1} \alpha_h}$.

---

(2) Theory of stochastic gradient

Setup: $\displaystyle \min_{x \in \mathbb{R}^d} f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x)$, f_i C^1

f $C^{1,1}_L$ (∇f L-lipschitz continuous)

$\{x^*\} := \operatorname{argmin}_{x \in \mathbb{R}^d} f(x)$ f μ-strongly convex

$f^* := f(x^*) = \min_{x \in \mathbb{R}^d} f(x)$ $(\mu \leq L)$

f $C^{1,1}_L \Rightarrow \forall (x, w) \in (\mathbb{R}^d)^2$,

$$f(w) \leq f(x) + \nabla f(x)^T (w - x) + \frac{L}{2} \|w - x\|^2$$

Instrumental in proving convergence for GD

$x = x_k$, $w = x_k - \alpha_k \nabla f(x_k) = x_{k+1}$

$$f(x_{k+1}) \leq f(x_k) - \alpha_k \|\nabla f(x_k)\|^2 + \frac{L \alpha_k^2}{2} \|\nabla f(x_k)\|^2$$

Q) what happens when $x_{k+1} = x_k - \alpha_k \underbrace{\nabla f_{i_k}(x_k)}_{\text{"stochastic gradient"}}$?

$$f(x_{k+1}) \leq f(x_k) - \alpha_k \nabla f(x_k)^T \nabla f_{i_k}(x_k) + \frac{L\alpha_k^2}{2} \|\nabla f_{i_k}(x_k)\|^2$$

Random Variable

No guarantee that this is > 0

No guarantee that this is bounded compared to $\nabla f(x_k)^T \nabla f_{i_k}(x_k)$

Taking expected values with respect to i_k on both sides, we get

$$(*) \quad \mathbb{E}_{i_k}[f(x_{k+1})] \leq \mathbb{E}_{i_k}[f(x_k)] - \mathbb{E}_{i_k}[\alpha_k \nabla f(x_k)^T \nabla f_{i_k}(x_k)] + \frac{L}{2}\mathbb{E}_{i_k}[\alpha_k^2 \|\nabla f_{i_k}(x_k)\|^2]$$

Assumptions:

i) For every k, i_k is drawn independently of x_k and $i_0, i_1, \ldots, i_{k-1}$

ii) For every k, $\mathbb{E}_{i_k}[\nabla f_{i_k}(x_k)] = \nabla f(x_k)$

iii) For every k, $\mathbb{E}_{i_k}[\|\nabla f_{i_k}(x_k)\|^2] \leq \|\nabla f(x_k)\|^2 + \sigma^2$ for some $\sigma^2 \geq 0$

Under these assumptions, $(*)$ becomes

$$\mathbb{E}_{i_k}[f(x_{k+1})] \overset{i)}{\leq} f(x_k) - \alpha_k \nabla f(x_k)^T \mathbb{E}_{i_k}[\nabla f_{i_k}(x_k)] + \frac{L}{2}\alpha_k^2 \mathbb{E}_{i_k}[\|\nabla f_{i_k}(x_k)\|^2]$$

$$\overset{ii)}{=} f(x_k) - \alpha_k \|\nabla f(x_k)\|^2 + \frac{L}{2}\alpha_k^2 \mathbb{E}_{i_k}[\|\nabla f_{i_k}(x_k)\|^2]$$

$$\overset{iii)}{\leq} \underline{f(x_k) - \alpha_k \|\nabla f(x_k)\|^2 + \frac{L\alpha_k^2}{2}\|\nabla f(x_k)\|^2} + \boxed{\frac{L\alpha_k^2 \sigma^2}{2}}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{Right hand side of the GD inequality}} \qquad \underset{\text{"Noise" component}}{\downarrow}$$

Remarks

- Assumptions i) and ii) are satisfied by most sampling strategies (ex: $\{i_k\}$ iid uniformly distributed in $\{1, \dots, m\}$)
- Assumption iii) holds when $\|\nabla f_{i_k}(x_k)\|$ are bounded $\forall k$ and it is implied by $\mathbb{E}_{i_k}[\|\nabla f_{i_k}(x_k)\|^2] \le M^2 \quad M \ge 0$

- What we require is:

  i). Stochastic gradients are unbiased estimates of the true gradient ("on average, a SG step is a gradient step")

  $\mathbb{E}_{i_k}[x_k - \alpha_k \nabla f_{i_k}(x_k)] \overset{(i)}{=} x_k - \alpha_k \mathbb{E}_{i_k}[\nabla f_{i_k}(x_k)] \overset{(ii)}{=} x_k - \alpha_k \nabla f(x_k)$

  (i) + (ii)

  iii). Stochastic gradients should not deviate too much from gradients in norm

$$\mathbb{E}_{i_k}\left[\|\nabla f_{i_k}(x_k)\|^2\right] - \|\nabla f(x_k)\|^2 \le \sigma^2 \quad (iii)$$

$$\underbrace{\mathbb{E}_{i_k}\left[\|\nabla f_{i_k}(x_k)\|^2\right] - \|\mathbb{E}_{i_k}[\nabla f_{i_k}(x_k)]\|^2}_{\simeq \text{ variance for } \|\nabla f_{i_k}(x_k)\|} \le \sigma^2 \quad (iii)+(ii)$$

$\uparrow$ "Variance/ Noise parameter"

For a broader class of assumptions, see e.g.

L. Bottou, F. E. Curtis and J. Nocedal, Optimization Methods for Large-Scale Machine Learning, SIAM Review (2018)

Theorem: SG with fixed step size

Suppose we run SG with $\{i_k\}$ satisfying Assumptions (i) – (iii) and $\forall k, \alpha_k = \alpha \in \left(0, \frac{1}{L}\right]$ $\left(f \in C_L^{1,1}, \mu\text{-strongly convex} \right)$

Then, for any $K \geq 1$,

$$\mathbb{E}_{i_0, \dots, i_{K-1}}\left[f(x_K) - f^* \right] \leq \underbrace{(1-\alpha\mu)^{\boxed{K}} \left[f(x_0) - f^* - \frac{\alpha L \sigma^2}{2\mu} \right]}_{\substack{\to 0 \\ K \to \infty}} + \underbrace{\frac{\alpha L \sigma^2}{2\mu}}_{\text{Constant (does not depend on } K)}$$

$(1-\alpha\mu) \geq \underbrace{1 - \frac{\mu}{L}}_{\in(0,1)}$

"Expected convergence result": $\mathbb{E}_{i_0, \dots, i_{K-1}}[\cdot]$

"Last-iterate convergence result": x_K

Recall: For GD on $C_L^{1,1}$, μ-strongly convex f, we have

$$f(x_K) - f^* \leq \underbrace{(1-\alpha\mu)^K (f(x_0) - f^*)}_{\substack{\to 0 \\ K \to \infty}} \quad \forall \alpha \in \left(0, \frac{1}{L}\right]$$

↳ For GD, we can show that $f(x_K) - f^* \xrightarrow[K \to \infty]{} 0$ at a rate $(1-\alpha\mu)^K$

↳ For SG, we can show that $\mathbb{E}[f(x_K) - f^*] \xrightarrow[K \to \infty]{} \left[0, \frac{\alpha L \sigma^2}{2\mu}\right]$ at a rate $(1-\alpha\mu)^K$

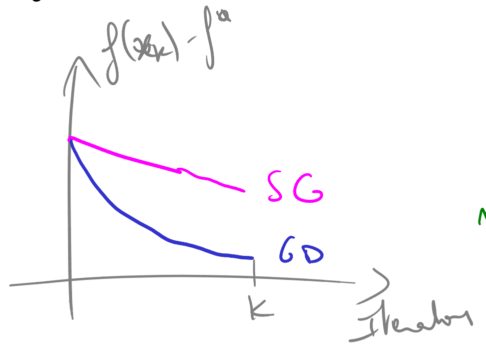
- Convergence in expected value (instead of deterministic)
- Convergence to a neighborhood of the optimal value (instead of convergence to the optimal value)

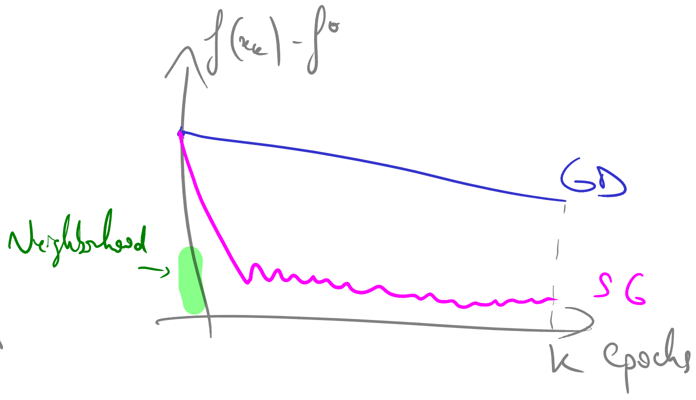
- But same rate of CV!

$\Rightarrow$ From that result, SG seems worse than GD ...

$\Rightarrow$ ... however the comparison is conducted with a fixed iteration budget, and iterations of GD and SG have different costs.

Typical plots





$\rightarrow$ 1 iteration of GD $\equiv$ 1 calculation of $\nabla f(\cdot)$

$\equiv$ n calculations of $\nabla f_i(\cdot)$

$\rightarrow$ 1 iteration of SG $\equiv$ 1 calculation of $\nabla f_i(\cdot)$

$\Rightarrow$ With that metric, 1 iteration of SG is n times cheaper than 1 iteration of GD.

$\Rightarrow$ Given that an epoch corresponds to n accesses to data points, 1 epoch $\equiv$ 1 iteration of GD

1 epoch $\equiv$ n iterations of SG

Corollary Suppose we run SG and GD for a fixed number of epochs $N \geq 1$, i.e. nN iterations of SG and N iterations of GD.

Then, for the final iterates x_{nN}^{SG} and x_N^{GD}, we have

$$\mathbb{E}\left[f(x_{nN}^{SG})-f^*\right] \leq (1-\alpha\mu)^{nN}\left(f(x_0)-f^*-\frac{\alpha L \sigma^2}{2\mu}\right)+\frac{\alpha L \sigma^2}{2\mu}$$

$$f(x_N^{GD})-f^* \leq (1-\alpha\mu)^N\left(f(x_0)-f^*\right)$$

$\longrightarrow$ SG still converges in expectation to a neighborhood of the solution, however when $n \gg 1$ it does so at a faster CV rate than GD.

Further results

- From expected value to high probability results: $\hat{x}_K \in \{x_0, x_1, \ldots, x_K\}$

  $\Rightarrow$ By drawing an iterate at random and using it as output, you can get guarantees of the form

$$\mathbb{P}\left(f(\hat{x}_K) - f^* \leq \ldots \right) \geq p$$

- To guarantee CV to a solution rather than a neighborhood,
  - Can use decreasing stepsizes

    Ex) $\alpha_k = O\left(\frac{1}{k}\right) \Longrightarrow \mathbb{E}\left[f(x_K) - f^*\right] \leq O\left(\frac{1}{K}\right)$

    (NB: still worse than GD in terms of iterations, better in terms of epochs)

  - Can use averaged iterates

$$\bar{x}_K = \frac{1}{K} \sum_{k=0}^{K-1} x_k$$

    (as in subgradient methods)

  - Can also shrink the neighborhood using batch methods

$$\left[0, \frac{\alpha L \sigma^2}{2\mu}\right] \xrightarrow[\text{Batch size } n_b]{} \left[0, \frac{\alpha L \sigma^2}{2\mu\, n_b}\right]$$

- From strongly convex to nonconvex and others
  - There are results for the convex setting and the nonconvex setting
    $\rightarrow$ For nonconvex, look at $\mathbb{E}\left[\frac{1}{\sum_{k=0}^{K-1}\alpha_k}\sum_{k=0}^{K-1}\alpha_k\|\nabla f_{i_k}(x_k)\|^2\right]$
  - Can extend this to nonsmooth functions, typically considering averaged iterates again

---

TAKEAWAYS:
- SG is designed to perform less calculations than GD at every iteration
- Performance depends on the step size and the sampling strategy
- SG has worse iteration CV rates than GD but has better rates in terms of epochs

Exercise : Importance sampling

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x) \qquad f_i \in C^{1,1}_{L_i}$$

$$c_i = \frac{n L_i}{\sum_{j=1}^{n} L_j}$$

Stochastic gradient variant

$\to$ Pick i_k such that $\mathbb{P}(i_k = i) = \dfrac{c_i}{\sum_{j=1}^{n} c_j}$

$\to$ Set $x_{k+1} = x_k - \dfrac{\alpha_k}{c_{i_k}} \nabla f_{i_k}(x_k)$, $\quad \alpha_k > 0$

a) Show that $\mathbb{E}_{i_k}\left[\dfrac{1}{c_{i_k}} \nabla f_{i_k}(x_k) \right] = \nabla f(x_k)$

b) f is $C^{1,1}_L$ with $L = \frac{1}{n} \sum_{i=1}^{n} L_i$

i) If $\alpha_k = \frac{1}{L}$, what is $\dfrac{\alpha_k}{c_{i_k}}$ equal to ?

ii) what can then be the advantage of importance sampling over uniform sampling ?