ML & Optimization Lecture 12

- Newton's Method
- Stochastic Gradient Descent

Newton's Method

Uses knowledge of the curvature to choose the update direction

$$f(x) \approx \underbrace{f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \frac{1}{2}(x - x_t)^T \nabla^2 f(x_t)(x - x_t)}$$

$$x_{t+1} = \operatorname*{argmin}_x$$

By Fermat's condition we have x_{t+1} satisfies

$$0 = \nabla f(x_t) + \nabla^2 f(x_t)(x_{t+1} - x_t)$$

$$\Rightarrow \quad x_{t+1} = x_t - \left[\nabla^2 f(x_t)\right]^{-1} \nabla f(x_t)$$

"Unguarded Newton's method"

We generally take our search direction

$$g_t = -\left[\nabla^2 f(x_t)\right]^{-1} \nabla f(x_t)$$

and judiciously choose our step-size α_t, then take

$$x_{t+1} = x_t + \alpha_t g_t$$

e.g. backtracking line search

This is called the guarded Newton's method

There are two phases of this alg:

- "damped phase": suboptimality goes down by a constant factor at each iteration: (here $\alpha_t < 1$)

$$f(x_t) - f(x_*) \leq \epsilon \quad \text{after} \quad t = O\left(\log\left(\tfrac{1}{\epsilon}\right)\right)$$

- "purely Newton phase": ($\alpha_t = 1$, BTLS does nothing) at each iteration, the number of digits of accuracy doubles!

$$f(x_t) - f(x_*) \leq \epsilon \quad \text{after} \quad t = O\left(\log\log\left(\tfrac{1}{\epsilon}\right)\right)$$

Ex Consider an ill-conditioned linear system

Solving $x_* = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|Ax-b\|_2^2$

$$f(x) = \frac{1}{2}\|Ax-b\|_2^2 \quad \Rightarrow \quad \nabla^2 f(x) = A^TA$$

and

$$\lambda_{min}(A^TA)\cdot I \preceq A^TA \preceq \lambda_{max}(A^TA)\cdot I$$

so

$$\kappa_f = \frac{\lambda_{max}(A^TA)}{\lambda_{min}(A^TA)}$$

Assume $\kappa_f = e^C$ where C is large.

Opt 1

Use GD with step size $\frac{1}{\beta} = \frac{1}{\lambda_{max}(A^TA)} \ll 1$

$f(x_T) - f(x_*) \leq \epsilon$ when $T = O\left(K \log\left(\frac{1}{\epsilon}\right)\right) \gg 1$

Opt 2

Use Newton's method with unit step size

$$x_1 = x_0 - (A^TA)^{-1}A^T(Ax_0 - b)$$

$$= (A^TA)^{-1}A^Tb$$

$$= x_{OLS}$$

$$= \operatorname{argmin}_x \frac{1}{2}\|Ax - b\|_2^2$$

One iteration of Newton's method gives the exact solution

Takeaway: we converge faster if we account for curvature

Drawbacks to Newton's Method

- need to form $\nabla^2 f(x_t) \in \mathbb{R}^{d \times d}$
  and solve a linear system

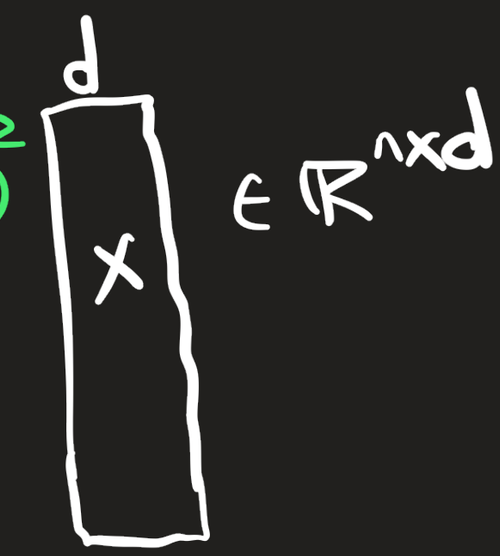
In general, this costs $O(d^2)$ memory and $O(d^3)$ operations

so when d is large, this is not feasible

GD can be expensive

Ex GD for solving a linear system

$$f(w) = \frac{1}{2}\|Xw - y\|_2^2 = \sum_{i=1}^{n} \frac{1}{2}(x_i^T w - y_i)^2$$



$\in \mathbb{R}^{n \times d}$

$$\nabla f(w) = X^T(Xw - y)$$
$$= X^T X w - X^T y$$

Two approaches

1) can store $X^T X \in \mathbb{R}^{d \times d}$ then cost of computing $\nabla f(w)$ is $O(d^2)$

2) can't store $X^T X$ cost of computing $\nabla f(w)$ is $O(nd)$

Usually $n \gg d$, so both costs are $\Omega(d^2)$ so if $d \gg 1$ then GD is expensive

Stochastic Gradient Descent

We want an alg cheaper per iteration than GrD, preferably $O(d)$ per iteration

Idea: notice that $\nabla f(w_t)$ is an approximate search direction

$$f(w) \approx f(w_t) + \langle \nabla f(w_t), w - w_t \rangle + \frac{1}{2\alpha_t} \|w - w_t\|_2^2$$

where

$$f(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(w^T x_i, y_i)$$

$$\Rightarrow \nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} x_i \ell'(w^T x_i, y_i)$$

$\nabla f(w)$ is already a sample average that approximates a population average.

Treat $\nabla f(w)$ itself as a population average, and approximate it with a sample average

- select a random subset $I \subset [n] = \{1, \dots, n\}$ of size k and form an approximation

$$g = \frac{1}{k} \sum_{i \in I} x_i \ell'(w^T x_i, y_i)$$ — stochastic gradient estimate

if we choose I appropriately

$$\mathbb{E} g = \nabla f(w)$$

The set I is called a <u>minibatch</u>

Stochastic Gradient Descent

$$x_* = \operatorname{argmin}_x f(x)$$

Choose a batch size K

for $t = 0, \dots, T-1$

sample $I \subseteq [n]$ of size K

$$g_t \leftarrow \frac{1}{K} \sum_{i \in I} \nabla \ell_i(x_t)$$

$$\ell_i(w) := \ell(w^T x_i, y_i)$$

$$x_{t+1} = x_t - \alpha_t g_t$$

return

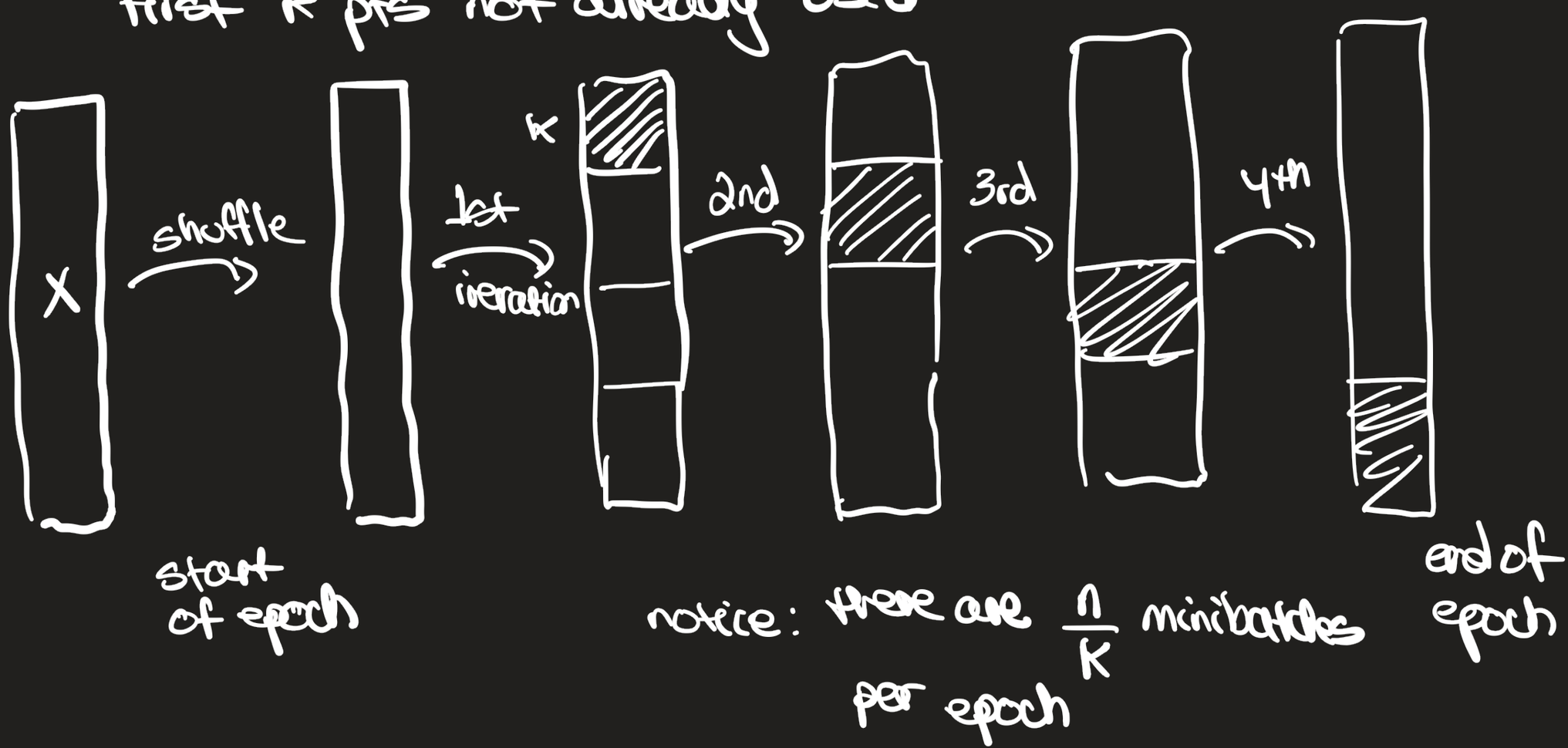
$$\hat{x} = \frac{1}{T} \sum_{i=1}^{T} x_i$$

Polyak average

In practice, to efficiently use all our data, we form our minibatches in the following way:

In each epoch:

- randomize the order of the training data
- everytime we sample a minibatch, use the first k pts not already used



notice: there are $\frac{n}{k}$ minibatches per epoch

Minibatch SGD

Inputs: ∇f oracle, α_t stepsizes, k minibatch size, E epochs

Alg

$z_0 = x_0$

for $e = 1, \dots, E$

$(X, y) \leftarrow$ random shuffling of itself

$x_0 = z_{e-1}$

for $t = 0, \dots, \frac{n}{k} - 1$

$I \Leftarrow$ first k unused samples in this epoch

$$g_t \leftarrow \frac{1}{k} \sum_{i \in I} \nabla \ell_i(x_t)$$

$$x_{t+1} \leftarrow x_t - \alpha g_t$$

end

$z_e = x_{\frac{n}{k}}$

end

return z_E

Convergence rate of SGD (for nonconvex opt prob)

Due to nonconvexity and fact SGD is not a descent method, we measure convergence with

$$\min_{t=0,\dots,T-1} \mathbb{E}\,\|\nabla f(x_t)\|_2^2 \leq \epsilon$$

Assumptions

- f bounded below
- f is β-smooth
- assume $\mathbb{E}\, g_t = \nabla f(x_t)$
- $\mathbb{E}\,\|g_t\|_2^2 \leq \sigma^2$ (could relax to $\mathbb{E}\,\|\nabla f(x_t) - g_t\|_2^2 \leq \sigma^2$)

Idea: bound

$$\sum_{t=0}^{T-1} \alpha_t \mathbb{E} \|\nabla f(x_t)\|_2^2 \leq B$$

which implies

$$\min_{t=0,\cdots,T-1} \mathbb{E} \|\nabla f(x_t)\|_2^2 \leq \frac{B}{\sum_{t=0}^{T-1} \alpha_t}$$

Prf

Recall that b/c f is β-smooth,

$$f(y) \leq f(x) + \langle \nabla f(x), y-x \rangle + \frac{\beta}{2}\|x-y\|_2^2$$

$\Rightarrow$

$$f(x_{t+1}) \leq f(x_t) + \langle \nabla f(x_t), -\alpha_t g_t \rangle + \frac{\beta}{2}\|-\alpha_t g_t\|_2^2$$

$$= f(x_t) - \alpha_t \langle \nabla f(x_t), g_t \rangle + \frac{\alpha_t^2 \beta}{2}\|g_t\|_2^2$$

Take expectations (b/c $x_t > x_{t+1} > g_t$ are random vectors)

$$\mathbb{E} f(x_{t+1}) \leq \mathbb{E} f(x_t) - \alpha_t \mathbb{E} \|\nabla f(x_t)\|_2^2 + \frac{\alpha_t^2 \beta \sigma^2}{2}$$

this implies

$$\alpha_t \mathbb{E} \|\nabla f(x_t)\|_2^2 \leq \mathbb{E}\left[f(x_t) - f(x_{t+1})\right] + \frac{\alpha_t^2 \beta \sigma^2}{2}$$

so

$$\sum_{t=0}^{T-1} \alpha_t \mathbb{E} \|\nabla f(x_t)\|_2^2 \leq \sum_{t=0}^{T-1} \mathbb{E}\left[f(x_t) - f(x_{t+1})\right]$$

telescoping sum

$$+ \frac{\beta \sigma^2}{2} \sum_{t=0}^{T-1} \alpha_t^2$$

$$= f(x_0) - \mathbb{E} f(x_T) + \frac{\beta \sigma^2}{2} \sum_{t=0}^{T-1} \alpha_t^2$$

$$\leq f(x_0) + \frac{\beta \sigma^2}{2} \sum_{t=0}^{T-1} \alpha_t^2$$

So finally

$$\underbrace{\min_{t=0,\dots,T-1} \mathbb{E}\,\|\nabla f(x_t)\|_2^2}_{:= \text{gap}(T)} \le \frac{f(x_0)}{\sum_{t=0}^{T-1}\alpha_t} + \frac{B\sigma^2}{2}\frac{\sum_{t=0}^{T-1}\alpha_t^2}{\sum_{t=0}^{T-1}\alpha_t}$$

Choices of step-sizes:

- $\alpha_t = \alpha$, then $\text{gap}(T) = O\left(\frac{f(x_0)}{T\alpha} + \frac{B\sigma^2}{2}\left(\frac{T\alpha^2}{T\alpha}\right)\right)$

$$= O\left(\frac{f(x_0)}{T} + \sigma^2\alpha\right)$$

- $\alpha_t = \frac{\alpha}{t+1}$, then $\sum_{t=0}^{T-1}\alpha_t = \sum_{t=0}^{T-1}\frac{\alpha}{t+1} = O(\ln T)$

and $\sum_{t=0}^{T-1}\alpha_t^2 = \sum_{t=0}^{T-1}\frac{\alpha^2}{(t+1)^2} = O(1) \Rightarrow \text{gap}(T) = O\left(\frac{1}{\ln T}\right)$

$$-\alpha_t = \frac{\alpha}{\sqrt{t+1}} \quad \text{then} \quad \sum_{t=0}^{T-1} \alpha_t = \sum_{t=0}^{T-1} \frac{\alpha}{\sqrt{t+1}} = O(\sqrt{T})$$

$$\text{and} \quad \sum_{t=0}^{T-1} \alpha_t^2 = O(\ln T), \quad \text{so}$$

$$\text{gap}(T) = O\left(\frac{\ln T}{\sqrt{T}}\right)$$

If f is μ-strongly cvx with const step size α

$$\mathbb{E}\|x_T - x_*\|_2^2 \leq (1-2\alpha\mu)^T \|x_0 - x_*\|_2^2 + \frac{\alpha\sigma^2}{2\mu}$$

after we stall in making progress on our training set, half the step size will allow us to get closer