

ECE 532 - lecture 21 - SGD

We saw gradient descent: $x_{k+1} = x_k - \gamma \nabla f(x_k)$

and subgradient method: $x_{k+1} = x_k - \gamma v_k$ where $v_k \in \partial f(x_k)$.

★ what if even evaluating ∇f is too much work? Can we do something even simpler? Yes! Stochastic gradient descent!

★ basic idea: decompose ∇f into pieces: $\nabla f = \sum_{i=1}^N \nabla f_i$

where each gradient ∇f_i is much easier to evaluate than ∇f .

then, perform gradient descent at each step on a different piece:

$$x_{k+1} = x_k - \gamma \nabla f_k(x_k)$$

different function every time.

If there are only finitely many ∇f_i 's, we can loop over all of them in a deterministic fashion, or pick a random ∇f_i at each iteration.

many ways to split the function into pieces, Two most popular one: (2)

1) coordinate descent: only compute one component of gradient vector. Often this can be done efficiently without computing entire gradient.

2) sampling: in LS or classification problems, we have possibly many samples. Rather than collecting all of them, take one at a time for each step.

★ Each idea above can also be organized into "batches". eg descent along 10 coordinates at a time, or take 10 samples at a time. Different decisions can yield better or worse performance depending on e.g. hardware, even if theoretical flop count is the same.

Example: least squares:

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2, \quad A \in \mathbb{R}^{m \times n}$$

standard GD:
$$x_{k+1} = x_k - \gamma \nabla f(x_k)$$
$$= x_k - \gamma A^T (Ax_k - b)$$

computing $A^T(Ax_k - b)$ costs $O(mn)$ for $Ax_k - b$ and $O(mn)$ for multiplication by A^T . so a total of $O(mn)$.

3

coordinate descent : i^{th} component of $2A^T(Ax_k - b)$ is

$$\underbrace{(2e_i^T A^T A)}_{g_i^T} x_k - \underbrace{(2e_i^T A^T b)}_{c_i}$$

The g_i and c_i can be pre-computed. Instead of using

$$\nabla f(x_i), \text{ use } e_i (\underbrace{g_i^T x_i}_{\text{compute } O(n)} - c_i)$$

$\rightarrow O(n)$ to compute.

so only one component of x_i is updated at each iteration.

Sampling : $\underbrace{\|Ax - b\|^2}_f = \sum_{i=1}^m \underbrace{(\tilde{a}_i^T x - b_i)^2}_{f_i}$ where $A = \begin{bmatrix} \tilde{a}_1^T \\ \tilde{a}_2^T \\ \vdots \\ \tilde{a}_m^T \end{bmatrix}$

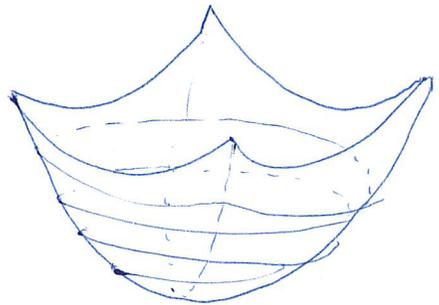
$$\nabla f_i(x) = 2\tilde{a}_i (\underbrace{\tilde{a}_i^T x - b_i}_{\text{compute } O(n)}}$$

$\rightarrow O(n)$ to compute.

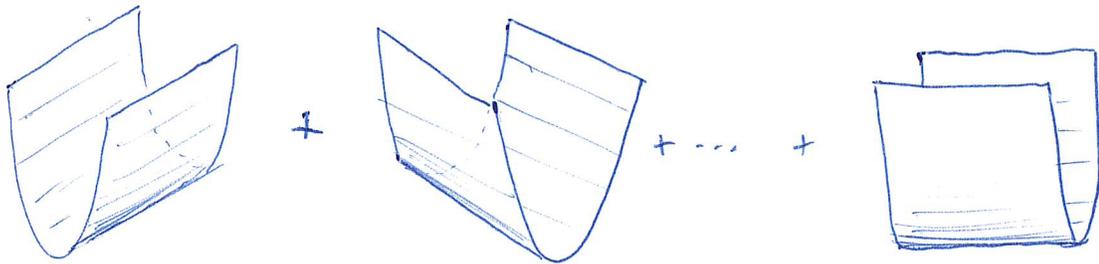
Again, descent at i^{th} step uses only i^{th} data point.

Generally speaking, SGD works. Basically the only method used in practice for large-scale problems. (with several tweaks, of course).

geometrical interpretation



$$f(x) = \|Ax - b\|^2$$



$$f_1(x) = (\tilde{a}_1^T x - b_1)^2 \quad f_2(x) = (\tilde{a}_2^T x - b_2)^2 \quad \dots \quad f_m(x) = (\tilde{a}_m^T x - b_m)^2$$

each f_i has one curved direction and $n-1$ flat directions.

each $\nabla f_i(x)$ costs $O(n)$ to evaluate. (there are m different ∇f_i 's)

whereas $\nabla f(x)$ costs $O(mn)$ to evaluate. (no total savings)

★ we can expect to do many more iterations of SGD compared to regular GD, but each iteration of SGD is cheaper.

We can prove similar convergence results as well.

Suppose each f_t is convex and $\|\nabla f_t(x)\| \leq G$ for all t, x .

define the optimal value: $x_* = \operatorname{argmin}_x \sum_{t=1}^T f_t(x)$.

further suppose f_t is differentiable (proof can be modified to use subgradients)

and we use the algorithm:

$$x_{t+1} = x_t - \gamma \nabla f_t(x_t)$$

then, we have:

$$\begin{aligned} \|x_{t+1} - x_*\|^2 &= \|x_t - \gamma \nabla f_t(x_t) - x_*\|^2 \\ &= \|x_t - x_*\|^2 - 2\gamma \nabla f_t(x_t)^T (x_t - x_*) + \gamma^2 \|\nabla f_t(x_t)\|^2 \end{aligned}$$

using convexity, $f_t(x_*) - f_t(x_t) \geq \nabla f_t(x_t)^T (x_* - x_t)$. Incorporating this into the equation above along with $\|\nabla f_t(x_t)\| \leq G$, we obtain:

$$f_t(x_t) - f_t(x_*) \leq \frac{1}{2\gamma} (\|x_t - x_*\|^2 - \|x_{t+1} - x_*\|^2) + \frac{\gamma G^2}{2}$$

summing from $t=1$ to $t=T$ and dividing by T :

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T (f_t(x_t) - f_t(x_*)) &\leq \frac{1}{2\gamma T} (\|x_1 - x_*\|^2 - \|x_{T+1} - x_*\|^2) + \frac{\gamma G^2}{2} \\ &\leq \frac{1}{2\gamma T} \|x_1 - x_*\|^2 + \frac{\gamma G^2}{2} \end{aligned}$$

if we let $R = \|x_1 - x_*\|$, we obtain:

$$\frac{1}{T} \sum_{t=1}^T (f_t(x_t) - f_t(x_*)) \leq \frac{R^2}{2\gamma T} + \frac{\gamma G^2}{2}$$

very similar to previous GD result.

Note: if we optimize over γ for fixed T , we obtain

$$\gamma = \frac{R}{G\sqrt{T}} \quad \text{and this leads to:}$$

$$\frac{1}{T} \sum_{t=1}^T (f_t(x_t) - f_t(x_*)) \leq \frac{RG}{\sqrt{T}}$$

if we don't know R or G , we can just use $\gamma = \frac{1}{\sqrt{T}}$ and:

$$\frac{1}{T} \sum_{t=1}^T (f_t(x_t) - f_t(x_*)) \leq \frac{R^2 + G^2}{2\sqrt{T}} \propto \frac{1}{\sqrt{T}}$$

Interpretation: if we only have knowledge that gradients are bounded, we can get convergence error to $\propto \frac{1}{\sqrt{T}}$ with a fixed stepsize. Other relevant results (we will not prove these in class).

★ by using a diminishing stepsize $\gamma_t = \frac{1}{\sqrt{t}}$, error $\propto \frac{1}{\sqrt{T}}$

★ if f has Lipschitz gradients, i.e.

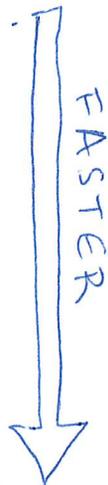
$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \text{ so no sharp corners, then we can prove that}$$

error $\propto \frac{1}{T}$

★ if f is strongly convex, i.e. $\left\{ \begin{array}{l} \text{"accelerated" methods} \\ \text{can achieve error } \propto \frac{1}{T^2} \end{array} \right\}$

$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\ell}{2}\|y-x\|^2$; lower-bounded by a quadratic; never flat, can prove:

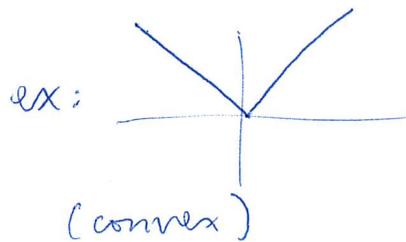
error $\propto \rho^T$
for some $0 < \rho < 1$



Observations

7

★ why can't we make error go to zero in bounded gradient case? Turns out we shouldn't be able to make such a guarantee.

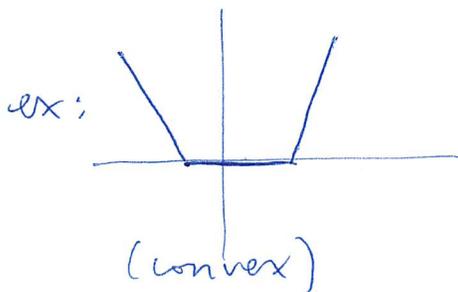


★ gradient is ± 1 depending on whether $x > 0$ or $x < 0$.

★ $x_{k+1} = x_k \pm \gamma$. we will never reach zero!

★ need diminishing stepsize.

★ why are the bounds on function error $|f(x_k) - f(x_*)| \rightarrow 0$ instead of iterate error $\|x_k - x_*\| \rightarrow 0$? Again, we can't guarantee iterates converge; only function values.



★ many possible choices for x_* , only one f_* .

★ GD will converge to one x_* , depends on the initial point x_0 .

★ under stronger assumptions about f , we can prove faster convergence rates or stronger results, e.g. exact convergence or convergence in $x_k \rightarrow x_*$.