Gradient Descent and Subgradient

Jianyong Sun

Xi'an Jiaotong University

jy.sun@xjtu.edu.cn

Gradient Descent

Consider unconstrained, smooth convex optimization

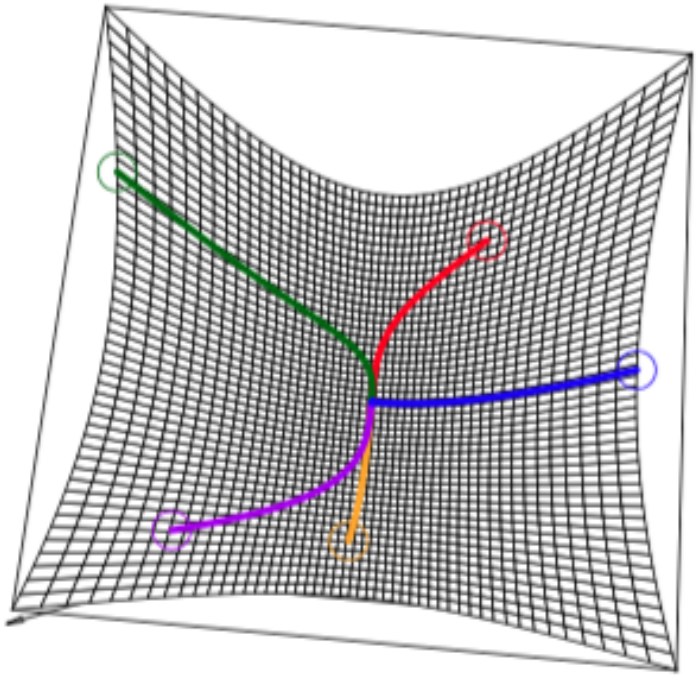
$$\min_x f(x)$$

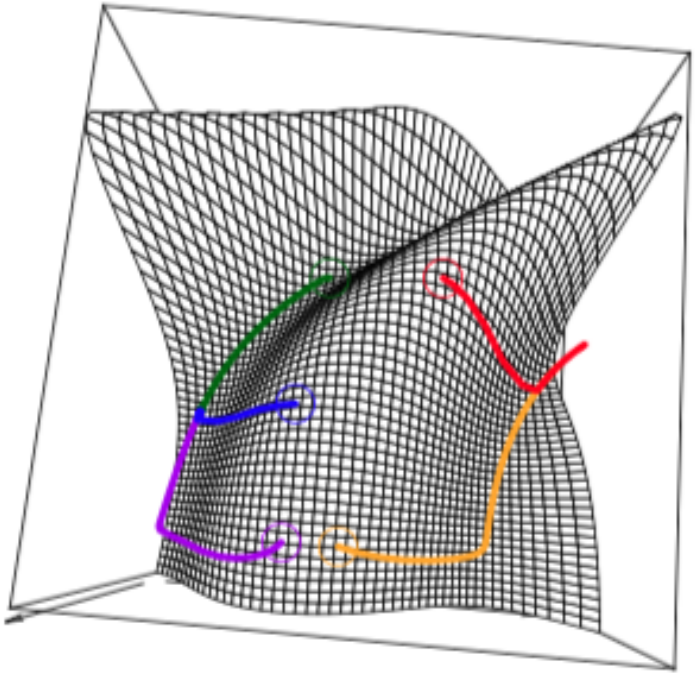
i.e. f is convex and differentiable with $\mathrm{dom}(f) = \mathbb{R}^n$. Denote the optimal criterion value by $f^\star = \min_x f(x)$, and a solution by $x^\star$.

Gradient Descent: choose initial point $x^{(0)} \in \mathbb{R}^n$, repeat

$$x^{(k)} = x^{(k-1)} - t_k \nabla f(x^{(k-1)}), k = 1, 2, \cdots$$

stop at some point





Gradient descent interpretation

At each iteration, consider the expansion

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$

Quadratic approximation: replacing usual Hessian matrix $\nabla^2 f(x)$ by $\frac{1}{t} I$

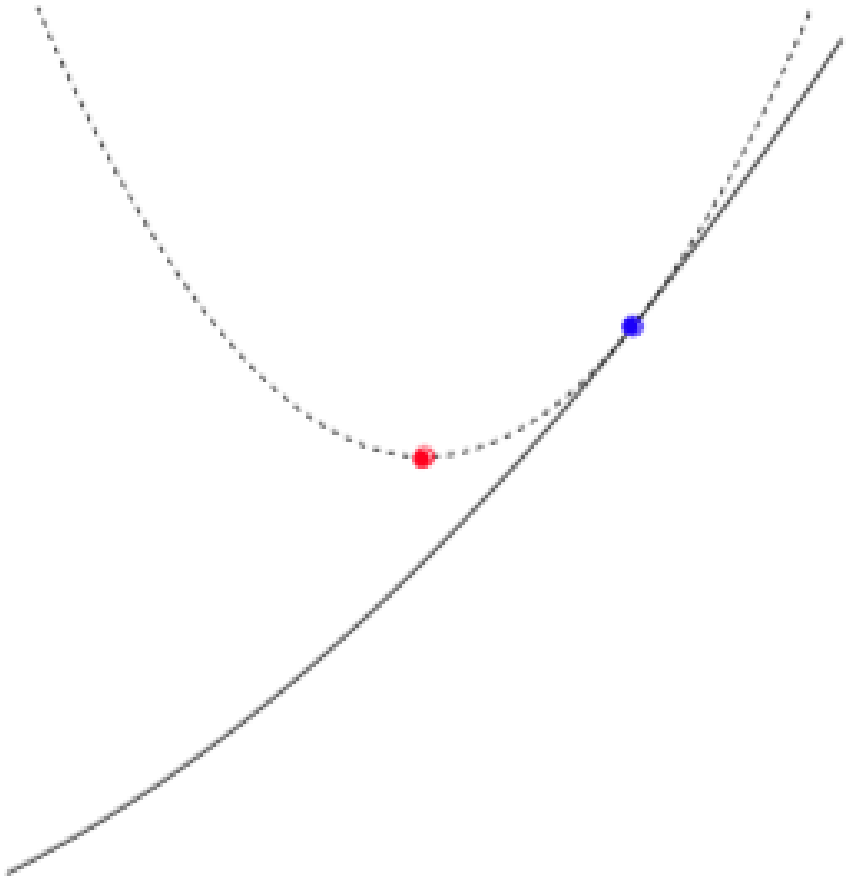
$f(x) + \nabla f(x)^T (y - x)$	linear approximation to f
$\frac{1}{2t}\|y - x\|_2^2$	proximity term to x with weight $1/(2t)$

Choose next point $y = x^+$ to minimize the quadratic approximation

$$x^+ = x - t\nabla f(x)$$

i.e.

$$x^+ = \arg\min f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$



Blue point is x, red point is

$$x^+ = \operatorname*{argmin}_y f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$

Outline

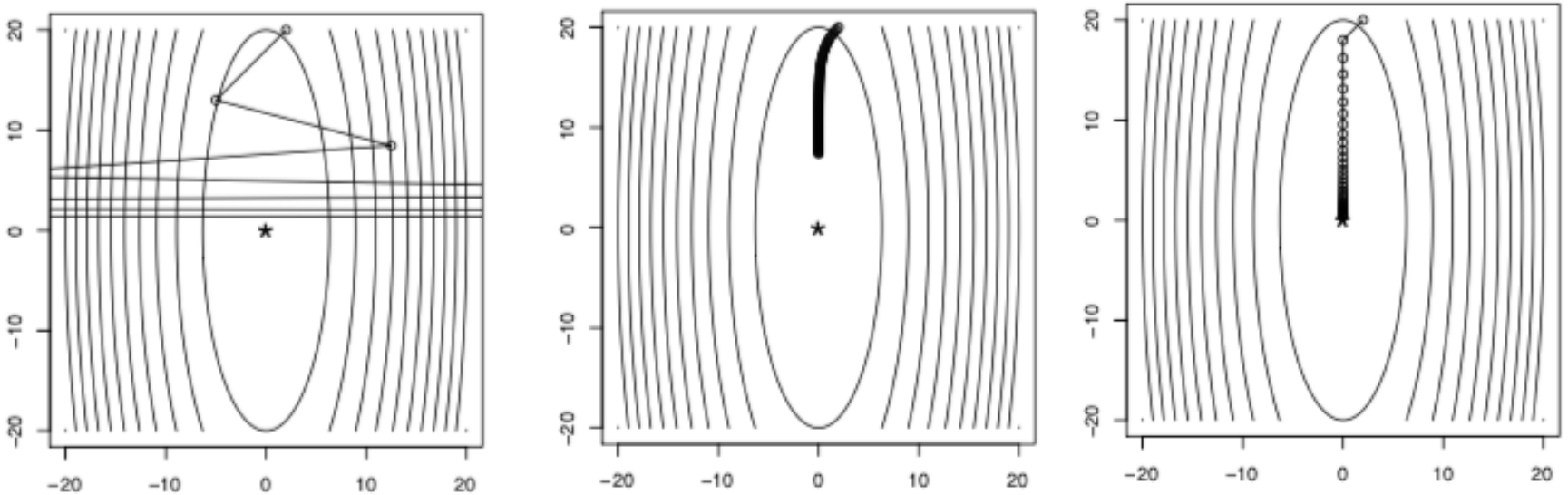
- How to choose step sizes
- Convergence analysis
- Gradient Boosting
- Stochastic Gradient descent

Fixed step size

Simply take $t_k = t$ for all $k = 1, 2, \cdots$, can diverge if t is too big and can be slow if t is too small, but convergence nicely if t is "just right".

Consider $f(x) = (10x_1^2 + x_2^2)/2$ for different step size.

Convergence analysis will give us a precise idea of "just right".



Backtracking line search

One way to adaptively choose the step size is to use backtracking line search:

- First fix parameters $0 < \beta < 1$ and $0 < \alpha \leq 1/2$
- At each iteration, start with $t = t_{\text{init}}$, and while

$$f(x - t\nabla f(x)) > f(x) - \alpha t \|\nabla f(x)\|_2^2$$

shrink $t = \beta t$. Else perform gradient descent update

$$x^+ = x - t\nabla f(x)$$

Simply and tends to work well in practice (further simplification, just take $\alpha = \frac{1}{2}$. Try backtracking line search with $\alpha = \beta = 0.5$ for the example function.

Backtracking interpretation

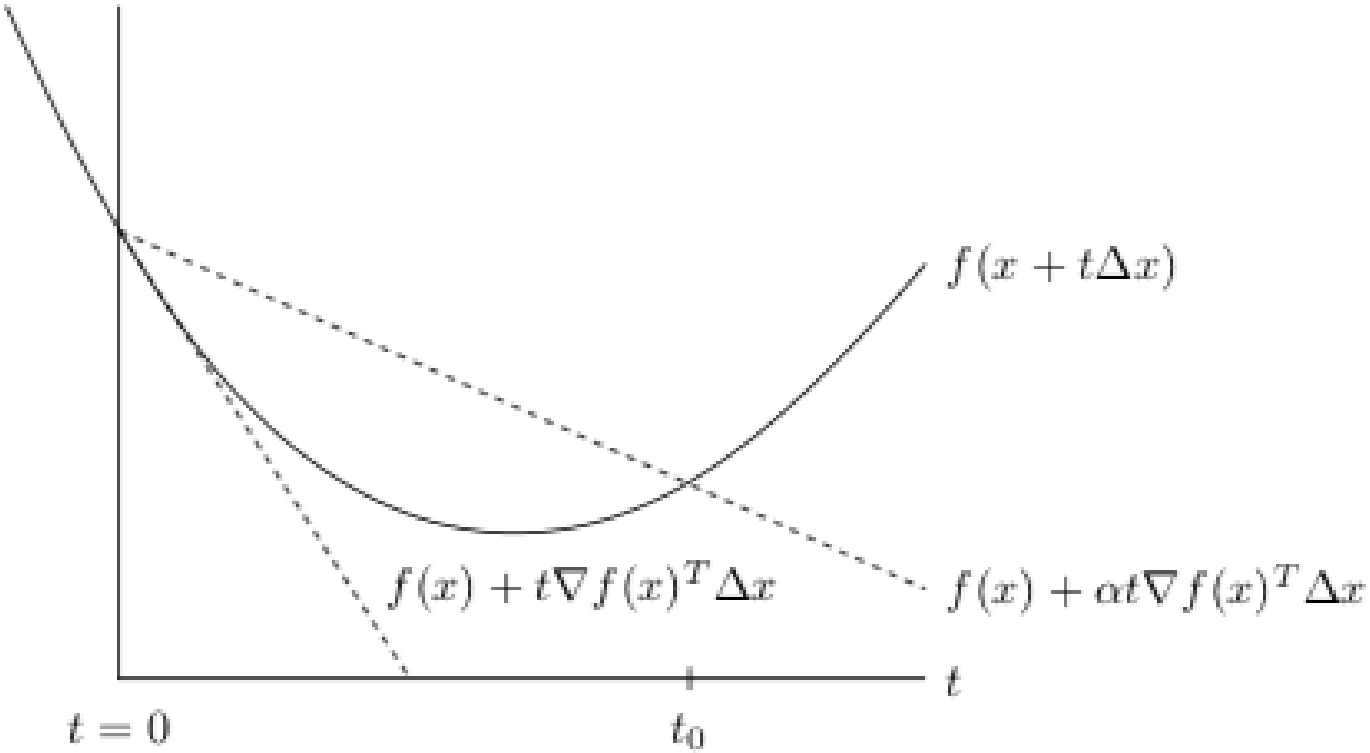


Figure: The curve shows f, restricted to the line over which we search. The lower dashed line shows the linear extrapolation of f, and the upper dashed line has a slope a factor of α smaller. The backtracking condition is that f lies below the upper dashed line, i.e.,$0 \leq t \leq t_0$. For us $\Delta(x) = -\nabla f(x)$

Exact line search

Could also choose step to do the best we can along direction of negative gradient, called exact line search

$$t = \arg\min_{s \geq 0} f(x - s\nabla f(x))$$

Usually not possible to do this minimization exactly

Approximations to exact line search are often not as efficient as backtracking, and it's usually not worth it.

Convergence analysis

Assume that f convex and differentiable, with $\mathrm{dom}(f) = \mathbb{R}^n$, and additionally

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2 \text{ for any } \quad x, y$$

i.e. ∇f is Lipschitz continuous with constant $L > 0$

Theorem

Gradient descent with fixed size $t \leq 1/L$ satisfies

$$f(x^{(k)}) - f^\star \leq \frac{\|x^{(0)} - x^\star\|_2^2}{2tk}$$

We say gradient descent has convergence rate $O(1/k)$, i.e. to get $f(x^{(k)}) - f^\star \leq \epsilon$ we need $O(1/\epsilon)$ iterations.

Proof

Key steps:

- ∇f Lipschitz with constant $L \Rightarrow$

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{L}{2}\|y - x\|_2^2, \forall x, y$$

- Plugging in $y = x^+ = x - t\nabla f(x)$

$$f(x^+) \leq f(x) - \left(1 - \frac{Lt}{2}\right) t\|\nabla f(x)\|_2^2$$

- Taking $0 < t \leq 1/L$, using convexity of f, $f^\star \geq f(x) + \nabla f(x)^T (x^\star - x)$,

$$\begin{aligned} f(x^+) &\leq f^\star + \nabla f(x)^T (x - x^\star) - \frac{t}{2}\|\nabla f(x)\|_2^2 \\ &= f^\star + \frac{1}{2t}(\|x - x^\star\|_2^2 - \|x^+ - x^\star\|_2^2) \end{aligned}$$

Proof

- Summing over all iterations till k

$$\begin{aligned}\sum_{i=1}^{k} f(x^{(i)}) - f(x^\star) &\leq \frac{1}{2t}\sum(\|x^{(i-1)} - x^\star\|_2^2 - \|x^{(i)} - x^\star\|_2^2) \\ &\leq \frac{1}{2t}\|x^{(0)} - x^\star\|_2^2\end{aligned}$$

- Since $f(x^{(k)})$ is non-increasing (i.e. $f(x^{(0)}) \leq f(x^{(1)}) \leq \cdots$)

$$f(x^{(k)}) - f^\star \leq \frac{1}{k}\sum_{i=1}^{k} f(x^{(i)}) - f^\star \leq \frac{\|x^{(0)} - x^\star\|_2^2}{2tk}$$

Convergence analysis for backtracking

Same assumptions, f is convex and differentiable, $\mathrm{dom}(f) = \mathbb{R}^n$ and ∇f is Lipschitz continuous with constant $L > 0$

Same rate for a step size chosen by backtracking search

Theorem

Gradient descent with backtracking line search satisfies

$$f(x^{(k)}) - f^\star \leq \frac{\|x^{(0)} - x^\star\|_2^2}{2t_{\min} k}$$

where $t_{\min} = \min\{1, \beta/L\}$

If β is not too small, then we don't lose much compared to fixed step size (β/L vs $1/L$)

Convergence analysis under strong convexity

Reminder: strong convexity of f means $f(x) - \frac{m}{2}\|x\|_2^2$ is convex for some $m > 0$. If f is twice differentiable, then this is equivalent to

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2}\|y - x\|_2^2, \forall x, y$$

Under Lipschitz assumption as before, and also strong convexity

Theorem

Gradient descent with fixed step size $t \leq 2/(m + L)$ or with backtracking line search satisfies

$$f(x^{(k)}) - f^\star \leq c^k \frac{L}{2}\|x^{(0)} - x^\star\|_2^2$$

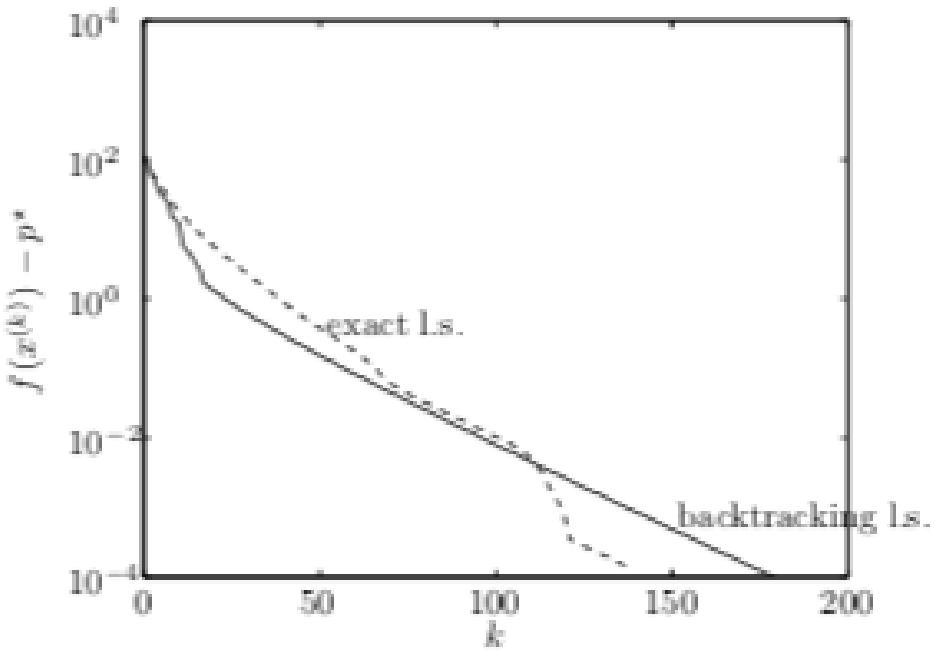
where $0 < c < 1$

Convergence analysis under strong convexity

I.e. rate with strong convexity is $O(c^k)$, exponentially fast!

I.e. to get $f(x^{(k)}) - f^\star \leq \epsilon$, need $O(\log(1/\epsilon))$ iterations, called linear convergence

Constant c depends adversely on condition number L/m (higher condition number $\rightarrow$ slow rate)



A look at the conditions

Look at the conditions for a simple problem $f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2$

Lipschitz continuity of ∇f:

- This means $\nabla^2 f(x) \preceq L\mathbb{I}$
- As $\nabla^2 f(\beta) = X^T X$, we have $L = \sigma_{\max}^2(X)$

Strong convexity of f

- This means $\nabla^2 f(x) \succeq m\mathbb{I}$
- As $\nabla^2 f(\beta) = X^T X$, we have $m = \sigma_{\min}^2(X)$
- If X is wide—i.e. X is $n \times p$ with $p > n$—then $\sigma_{\min}(X) = 0$ and f cannot be strongly convex
- Even if $\sigma_{\min}(X) > 0$, can have a very large condition number $L/m = \sigma_{\max}(X)/\sigma_{\min}(X)$

A function f having Lipschitz gradient and being strongly convex satisfies

$$mI \preceq \nabla^2 f(x) \preceq LI \text{ for all } \ x \in \mathbb{R}^n$$

for constants $L > m > 0$

Think of f being sandwiched between two quadratics

May seem like a strong condition to hold globally (for all $x \in \mathbb{R}^n$). But a careful look at proofs show that we only need Lipschitz gradients/strong convexity over the sublevel set

$$S = \{x : f(x) \leq f(x^{(0)})\}$$

which is less restrictive.

Can we do better?

Gradient descent has $O(1/\epsilon)$ convergence rate over problem class of convex, differential functions with Lipschitz gradients

First-order method, iterative method, update $x^{(k)}$ in

$$x^{(0)} + \text{span}\{\nabla f(x^{(0)}), \nabla f(x^{(1)}), \cdots, \nabla f(x^{(k-1)})\}$$

Theorem

For any $k \leq (n-1)/2$ and any starting point $x^{(0)}$, there is a function f in the problem class such that any first-order method satisfies

$$f(x^{(k)}) - f^\star \geq \frac{3L\|x^{(0)} - x^\star\|}{32(k+1)^2}$$

Curtsey to Nesterov.

Can we obtain rate $O(1/k^2)$ or $O(1/\sqrt{\epsilon})$?

Practicalities

Stopping rule: stop when $\|\nabla f(x)\|_2$ is small

- Recall $\nabla f(x^\star) = 0$ at solution $x^\star$
- If f is strong convex with parameter m, then

$$\|\nabla f(x)\|_2 \leq \sqrt{2m\epsilon} \Rightarrow f(x) - f(x^\star) \leq \epsilon$$

Pros and cons of gradient descent

- Pros: simple, each iteration is cheap (usually)
- fast for well-conditioned, strongly convex problems
- Cons: often be slow, because many interesting problems aren't strongly convex or well-conditioned
- cannot handle nondifferentiable functions

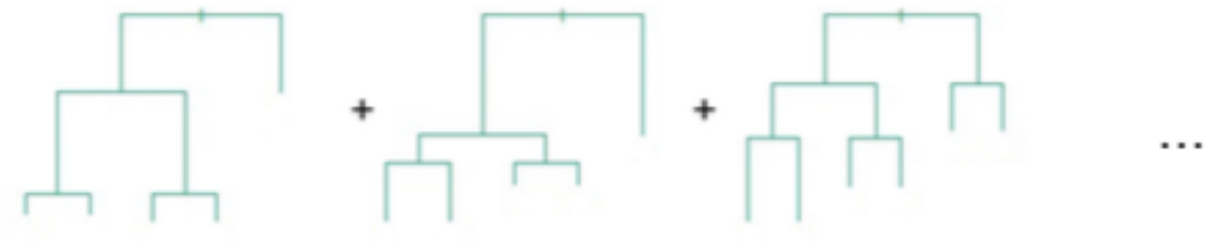
Gradient Boosting

Given observations $y = (y_1, y_2, \cdots, y_n) \in \mathbb{R}^n$, predicator measurements $x_i \in \mathbb{R}^p, i = 1, \cdots, n$

Want to construct a flexible (nonlinear) model for outcome based on predicators. Weighted sum of trees

$$u_i = \sum_{j=1}^{m} \beta_j T_j(x_i), i = 1, \cdots, n$$

Each tree T_j inputs predicator measurements x_i, output prediction. Trees are typically grow pretty short.



Pick a loss function L that reflects setting, e.g. for continuous y, could take $L(y_i, u_i) = (y_i - u_i)^2$

Want to solve

$$\min_{\beta} \sum_{i=1}^{n} L\left(y_i, \sum_{j=1}^{M} \beta_j T_j(x_i)\right)$$

Indexes all trees of a fixed size (e.g. depth $= 5$), so M is huge.

Space is simply too big to compute.

Gradient boosting: basically a version of gradient descent that is forced to work with trees.

First think of optimization as $\min_u f(u)$ over predicted values u_i, subject to u coming from trees.

Start with initial model, e.g. fit a single tree $u^{(0)} = T_0$, Repeat:

- Compute negative gradient d at latest prediction $u^{(k-1)}$:

$$d_i = -\left[\frac{\partial L(y_i, u_i)}{\partial u_i}\right]\Bigg|_{u_i = u_i^{(k-1)}}, i = 1, 2, \cdots, n$$

- Find a tree T_k that is close to d_i, i.e. according to

$$\min_{\text{tree}\, T} \sum_{i=1}^{n} (d_i - T(x_i))^2$$

  Not hard to (approximately) solve for a single tree

- Compute step size α_k and update our predictor

$$u^{(k)} = u^{(k-1)} + \alpha_k T_k$$

Note: predictions are weighted sum of trees as desired.

Stochastic gradient descent (SGD)

Stochastic Gradient Descent

Consider minimizing a sum of functions

$$\min_x \sum_{i=1}^m f_i(x)$$

As $\nabla \sum f_i(x) = \sum \nabla f_i(x)$, gradient descent would repeat

$$x^{(k)} = x^{(k-1)} - t_k \cdot \sum_{i=1}^m \nabla f_i(x^{(k-1)}), k = 1, 2, \cdots$$

In comparison, stochastic gradient descent or SGD (or incremental gradient descent) repeats

$$x^{(k)} = x^{(k-1)} - t_k g_{i_k}^{(k-1)}, k = 1, 2, \cdots$$

where $i_k \in \{1, \cdots, m\}$ is some chosen index at iteration k

Two rules for choosing index i_k at iteration k:

- Cyclic rule: choose $i_k = 1, 2, \cdots, m, 1, 2, \cdots, m, \cdots$
- Randomized rule: choose $i_k \in \{1, 2, \cdots, m\}$ uniformly at random

Randomized rule is more common in practice

What's the difference between stochastic and batch method? Computationally, m stochastic steps $\approx$ one batch step. But what about progress?

- Cyclic rule: m steps: $x^{(k+m)} = x^{(k)} - t\sum_{i=1}^{m} \nabla f_i(x^{(k+i-1)})$
- Batch method: one step: $x^{(k+1)} = x^{(k)} - t\sum_{i=1}^{m} \nabla f_i(x^{(k)})$
- Difference in directions: $\sum_{i=1}^{m} \nabla f_i(x^{(k+i-1)}) - \nabla f_i(x^{(k)})$

So SGD should converge if each $\nabla f_i(x)$ does not vary widely at x

Rule of thumb: SGD thrives far from optimum, struggles close to optimum

...

SGD and beyond

More on Stochastic gradient descent, Momentum, Adagrad, RSProp, Adaptive moment estimation (ADAM), etc.

Subgradient

Subgradients

If $\nabla f(x)$ is Lipschitz, gradient descent as convergence rate $O(1/\epsilon)$, but GD requires f is differentiable and it can be slow to converge

- Subgradients
- Examples
- Subgradient rules
- Optimality characterizations

Subgradients

Recall that for convex and differentiable f

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \text{ for all } x, y$$

I.e. linear approximation always underestimates f

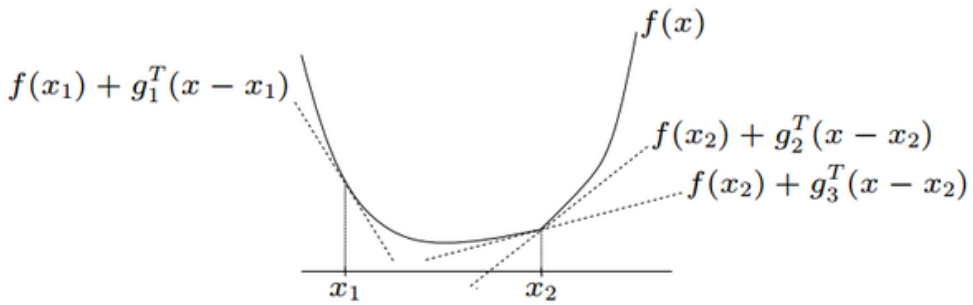
A subgradient of a convex function f at x is any $g \in \mathbb{R}^n$ such that

$$f(y) \geq f(x) + g^T (y - x) \text{ for all } \quad y$$

- Always exists
- If f differentiable at x, then $g = \nabla f(x)$ uniquely
- Actually, same definition works for nonconvex f (however, subgradient needs not exist)

Examples

Consider $f : \mathbb{R} \to \mathbb{R}$



Consider $f(x) = |x|$, its subgradient

- for $x \neq 0$, unique subgradient $g = \text{sign}(x)$
- for $x = 0$, subgradient g is any element of $[-1, 1]$

Consider $f(x) = \|x\|_2$, its subgradient

- For $x \neq 0$, unique subgradient $g = x/\|x\|_2$
- For $x = 0$, subgradient g is any element of $\{x : \|x\|_2 \leq 1\}$

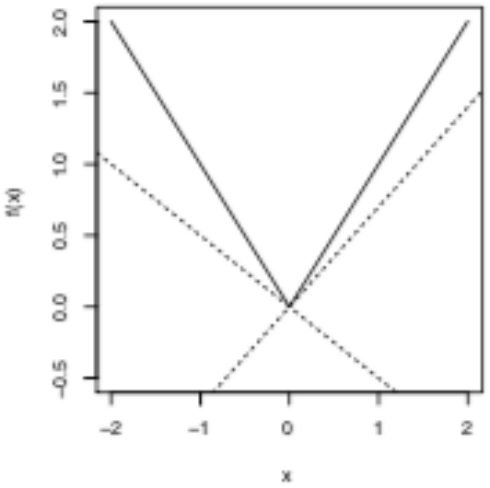
Consider $f(x) = \|x\|_1, x \in \mathbb{R}^n$, its subgradient

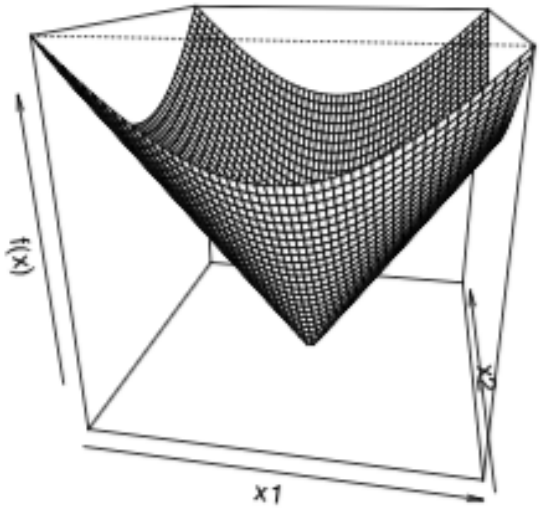
- for $x_i \neq 0$, unique subgradient $g_i = \text{sign}(x)$
- for $x_i = 0$, subgradient g_i is any element of $[-1, 1]$

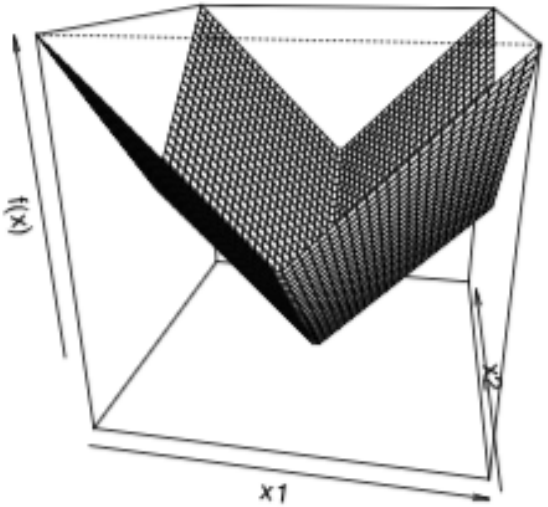
Let $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ be convex and differentiable, and consider $f(x) = \max\{f_1(x), f_2(x)\}$

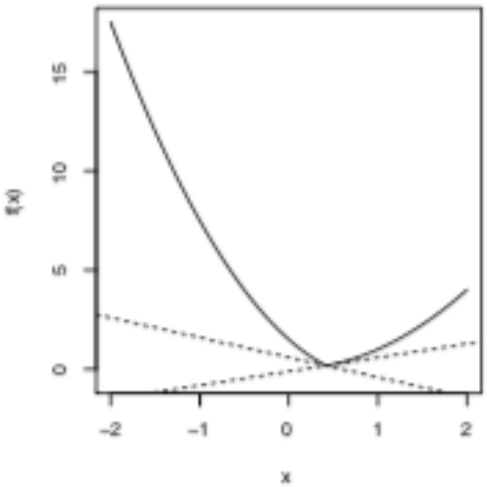
- For $f_1(x) > f_2(x)$, unique subgradient $\nabla f_1(x)$
- For $f_2(x) > f_1(x)$, unique subgradient $\nabla f_2(x)$
- For $f_1(x) = f_2(x)$, subgradient g is any point on the line segment between $\nabla f_1(x)$ and $\nabla f_2(x)$

$$g = \{\alpha \nabla f_1(x) + (1 - \alpha)\nabla f_2(x), \forall \alpha \in [0, 1]\}$$


f(x)
2.0
1.5
1.0
0.5
0.0
−0.5
−2
−1
0
1
2
x


f(x)
x1
x2


f(x)
x1
x2


f(x)
15
10
5
0
−2
−1
0
1
2
x

Subdifferential

Set of all subgradients of convex f is called the Subdifferential

$$\partial f(x) = \{g \in \mathbb{R}^n : g \text{ is a subgradient of } f \text{ at } x\}$$

- $\partial f(x)$ is convex and closed, even for nonconvex function
- Nonempty (can be empty for nonconvex f)
- If f is differentiable at x, then $\partial f(x) = \nabla f(x)$
- If $\partial f(x) = \{g\}$, then f is differentiable at x and $\nabla f(x) = g$

Connection to convex geometry

Consider convex set $C \subseteq \mathbb{R}^n$, consider indicator function, $I_C : \mathbb{R}^n \to \mathbb{R}$

$$I_C = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{if } x \notin C \end{cases}$$

For $x \in C, \partial I_C(x) = \mathcal{N}_C(x)$, the normal cone of C at x, that is

$$\mathcal{N}_C(x) = \left\{ g \in \mathbb{R}^n : g^T x \geq g^T y \text{ for all } y \in C \right\}$$

Why? By definition of subgradient g

$$I_C(y) \geq I_C(x) + g^T(y - x) \text{ for all } y$$

That is

- For $y \notin C$, $I_C(y) = \infty$
- For $y \in C$, this means $0 \geq g^T(y - x)$

Subgradient calculus

Basic rules for convex functions

- Scaling: $\partial(af) = a\partial(f)$ provided $a > 0$
- Addition: $\partial(f_1 + f_2) = \partial(f_1) + \partial(f_2)$
- Affine composition: if $g(x) = f(Ax + b)$, then

$$\partial g(x) = A^T \partial f(A^T x + b)$$

- Finite pointwise maximization: if $f(x) = \max_{i=1,\cdots,m} f_i(x)$, then

$$\partial f(x) = \text{conv}\left(\bigcup_{i:f_i(x)=f(x)} \partial f_i(x)\right)$$

convex hull of union of subgradiential of all active functions at x

Subgradient calculus

- General pointwise maximization: if $f(x) = \max_{i \in S} f_i(x)$, then

$$\partial f(x) \subset \text{cl} \left\{ \text{conv} \left(\bigcup_{i: f_i(x) = f(x)} \partial f_i(x) \right) \right\}$$

  under some regularity conditions (on S and f_i), we can get an equality above.

- Norms: $f(x) = \|x\|_p$, let q be such that $1/p + 1/q = 1$, then

$$\|x\|_p = \max_{\|z\|_q \leq 1} z^T x$$

  Hence

$$\partial f(x) = \operatorname*{argmax}_{\|z\|_q \leq 1} z^T x$$

Why subgradients?

Subgradients are important for two reasons:

- Convex analysis: optimality characterization via subgradients, monotonicity, relationship to duality
- Convex optimization: if you can compute subgradients, then you can minimize (almost) any convex function.

Optimality condition

For any f (convex or not),

$$f(x^\star) = \min_x f(x) \Leftrightarrow 0 \in \partial f(x)$$

i.e. $x^\star$ is a minimizer if and only if 0 is a subgradient of f at $x^\star$. This is called the subgradient optimality condition.

Why? $g = 0$ being a subgradient means that for all y:

$$f(y) \geq f(x^\star) + 0^T(y - x) = f(x^\star)$$

Note the implication for a convex and differentiable function f with $\partial f(x) = \{\nabla f(x)\}$

Derivation of first-order optimality condition

Recall that for f convex and differentiable, the problem

$$\min f(x) \text{ subject to } x \in C$$

is solved at x if and only if

$$\nabla f(x)^T (y - x) \geq 0 \text{ for all } y \in C$$

Intuitively says that gradient increase as we move away from x. How to see this? First recast problem as

$$\min_x f(x) + I_C(x)$$

Now apply subgradient optimality

$$0 \in \partial(f(x) + I_C(x))$$

Derivation of first-order optimality condition

But

$$\begin{aligned} 0 \in \partial(f(x) + I_C(x)) &\iff 0 \in \{\nabla f(x) + \mathcal{N}_C(x)\} \\ &\iff -\nabla f(x) \in \mathcal{N}_C(x) \\ &\iff -\nabla f(x)^T x \geq -\nabla f(x)^T y \text{ for all } \ y \in C \\ &\iff \nabla f(x)^T (y - x) \geq 0 \text{ for all } \ y \in C \end{aligned}$$

as desired.

Note: the condition $0 \in \partial(f(x) + I_C(x))$ is a fully general condition for optimality in a convex problem. But this is not always easy to work with KKT conditions.

Example: lasso optimality condition

Given $y \in \mathbb{R}^n, X \in \mathbb{R}^{n \times p}$, lasso problem can be paramterized as

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

where $\lambda \geq 0$. Subgradient optimality says

$$\begin{aligned} 0 \in \partial\left(\frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1\right) &\Leftrightarrow 0 \in -X^T(y - X\beta) + \lambda\partial\|\beta\|_1 \\ &\Leftrightarrow X^T(y - X\beta) = \lambda v \end{aligned}$$

for some $v \in \partial\|\beta\|_1$, i.e.

$$v_i \in \begin{cases} \{1\} & \text{if } \beta_i > 0 \\ \{-1\} & \text{if } \beta_i < 0, i = 1, \cdots, p \\ [-1, 1] & \text{if } \beta_i = 0 \end{cases}$$

Example: lasso optimality condition

Write $X_1, \cdots, X_p$ for columns of X. Then subgradient optimality reads

$$\begin{cases} X_i^T(y - X\beta) = \lambda \cdot \text{sign}(\beta_i) & \text{if } \beta_i \neq 0 \\ |X_i^T(y - X\beta)| \leq \lambda & \text{if } \beta_i = 0 \end{cases}$$

Note that the subgradient optimality condition do not directly lead to an expression for a lasso solution,....,however they do provide a way to check lasso optimality.

They are also helpful to in understanding the lasso estimator, e.g. if $|X_i^T(y - X\beta)| \leq \lambda$, then $\beta_i = 0$.

Example: soft-thresholding

Simplified lasso problem with $X = I$,

$$\min_{\beta} \frac{1}{2}\|y - \beta\|_2^2 + \lambda\|\beta\|_1$$

This can be solved directly using subgradient optimality. Solution is $\beta = S_\lambda(y)$, where S_λ is the soft-thresholding operator

$$[S_\lambda(y)]_i = \begin{cases} y_i - \lambda & \text{if } \ y_i > \lambda \\ 0 & \text{if } \ -\lambda \leq y_i \leq \lambda \\ y_i + \lambda & \text{if } \ y_i < -\lambda \end{cases}$$

Check: from last slide, subgradient optimality conditions are

$$\begin{cases} y_i - \beta_i = \lambda \cdot \text{sign}(\beta_i) & \text{if } \ \beta_i \neq 0 \\ |y_i - \beta_i| \leq \lambda & \text{if } \ \beta_i = 0 \end{cases}$$

Now plug in $\beta = S_\lambda(y)$ and check these are satisfied

- When $y_i > \lambda$, $\beta_i = y_i - \lambda > 0$, so $y_i - \beta_i = \lambda = \lambda \cdot 1$ since $\text{sign}(\beta_i) = 1$
- When $y_i < -\lambda$, argument is similar
- When $|y_i| \leq \lambda$, $\beta_i = 0$, and $|y_i - \beta_i| = |y_i| \leq \lambda$

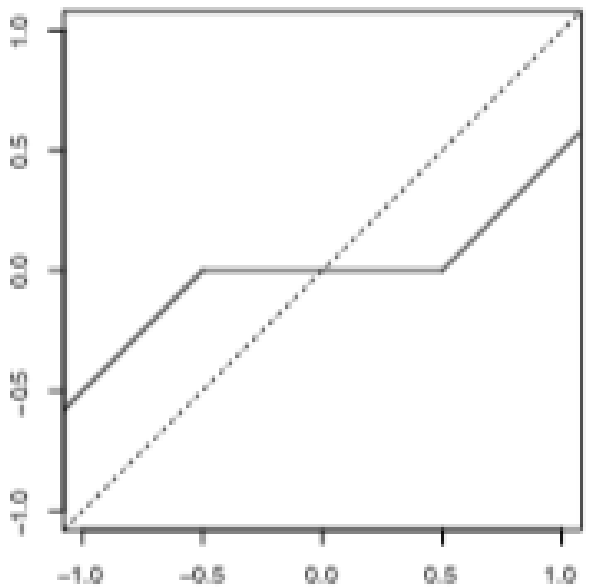


Figure: soft-thresholding in one variable

Example: distance to a convex set

Recall the distance function to a closed, convex set C

$$\text{dist}(x, C) = \min_{y \in C} \|y - x\|_2$$

This is a convex function, what are its subgradients?

Write $\text{dist}(x, C) = \|x - P_C(x)\|_2$, where $P_C(x)$ is the projection of x onto C:

$$P_C(x) = \operatorname*{argmin}_{y \in C} \|y - x\|_2$$

It turns out that when $\text{dist}(x, C) > 0$

$$\partial \text{dist}(x, C) = \left\{ \frac{x - P_C(x)}{\|x - P_C(x)\|_2} \right\}$$

Only has one element, so in fact $\text{dist}(x, C)$ is differentiable and this is its gradient.

Write $u = P_C(x)$, by first-order optimality condition for a projection

$$(x-u)^T(y-u) \leq 0 \text{ for all } \quad y \in C$$

Hence

$$C \subseteq H = \{y : (x-u)^T(y-u) \leq 0\}$$

Claim:

$$\text{dist}(x, C) \geq \frac{(x-u)^T(y-u)}{\|x-u\|_2} \text{ for all } \quad y$$

Check: first for $y \in H$, the RHS is ≤ 0
Now for $y \notin H$, we have $(x-u)^T(y-u) = \|x-u\|_2\|y-u\|_2 \cos\theta$ where θ is the angle between $x-u$ and $y-u$. Thus

$$\frac{(x-u)^T(y-u)}{\|x-u\|_2} = \|y-u\|_2 \cos\theta = \text{dist}(y, H) \leq \text{dist}(y, C)$$

as desired.

Under the claim, we have for any y,

$$\begin{aligned} \text{dist}(x, C) &\geq \frac{(x-u)^T(y-x+x-u)}{\|x-u\|_2} \\ &= \|x-u\|_2 + \left(\frac{x-u}{\|x-u\|_2}(y-x)\right)^T \end{aligned}$$

Hence $g = (x-u)/\|x-u\|_2$ is a subgradient of $\text{dist}(x, C)$ at x

Subgradient method

Now consider f convex with $\mathrm{dom}(f) = \mathbb{R}^n$, but not necessarily differentiable

Subgradient method: like gradient descent, but replacing gradients with subgradients, i.e. initialize $x^{(0)}$, repeat

$$x^{(k)} = x^{(k-1)} - t_k g^{(k-1)}, k = 1, 2, \cdots$$

where $g^{(k-1)} \in \partial f(x^{(k-1)})$, any subgradient of f at $x^{(k-1)}$

Subgradient method is not necessarily a descent method, so we keep track of the best iterate $x_{\mathrm{best}}^{(k)}$ among $x^{(0)}, \cdots, x^{(k)}$ so far, i.e.

$$f(x_{\mathrm{best}}^{(k)}) = \min_{i=0,\cdots,k} f(x^{(i)})$$

Step size choices

- Fixed step sizes $t_k = t$ for all $k = 1, 2, 3, \cdots$
- Diminishing step size: choose to meet conditions

$$\sum_{k=1}^{\infty} t_k^2 < \infty, \sum_{t=1}^{\infty} = \infty$$

  i.e. square summable but not summable
- Other options, but important difference to gradient descent: step sizes are typically pre-specified, not adaptively computed.

Convergence analysis

Assume that f convex and $\mathrm{dom}(f) = \mathbb{R}^n$, and also that f is Lipschitz continuous with constant $G > 0$, i.e.

$$|f(x) - f(y)| \leq G\|x - y\|_2 \text{ for all } x, y$$

Theorem

For a fixed size t, subgradient method satisfies

$$\lim_{k\to\infty} f(x_{best}^{(k)}) \leq f^\star + G^2 t/2$$

Theorem

For diminishing step sizes, subgradient method satisfies

$$\lim_{k\to\infty} f(x_{best}^{(k)}) = f^\star$$

Basic inequality

Can prove both results from the same basic inequalities. Key steps:

- Using definition of subgradient

$$\|x^{(k)} - x^\star\|_2^2 \leq \|x^{(k-1)} - x^\star\|_2^2 - 2t_k(f(x^{(k-1)}) - f(x^\star)) + t_k^2\|g^{(k-1)}\|_2^2$$

- Iterating last inequality

$$\|x^{(k)} - x^\star\|_2^2 \leq \|x^{(0)} - x^\star\|_2^2 - 2\sum_{i=1}^{t} t_i(f(x^{(i-1)}) - f(x^\star)) + \sum_{i=1}^{t} t_i^2\|g^{(i-1)}\|_2^2$$

Basic inequality

- Using $\|x^{(k)} - x^\star\|_2^2 \geq 0$, and letting $R = \|x^0 - x^\star\|_2^2$

$$0 \leq R^2 - 2\sum_{i=1}^{t} t_i(f(x^{(i-1)}) - f(x^\star)) + G^2 \sum_{i=1}^{t} t_i^2$$

- Introducing $f(x_{\text{best}}^{(k)}) = \min_{i=0,\cdots,k} f(x^{(i)})$, and rearraning

$$f(x_{\text{best}}^{(k)}) - f(x^\star) \leq \frac{R^2 + G^2 \sum_{i=1}^{k} t_i^2}{2\sum_{i=1}^{k} t_i}$$

  We call this basic inequality

For different step size choices, convergence results can be directly obtained from this bound.

Convergence rate

The basic inequality tells us that after k steps, we have

$$f(x_{\text{best}}^{(k)}) - f^\star \leq \frac{R^2 + G^2 \sum_{i=1}^k t_i^2}{2 \sum_{i=1}^k t_i}$$

With fixed step size t, this gives

$$f(x_{\text{best}}^{(k)}) - f^\star \leq \frac{R^2}{2kt} + \frac{G^2 t}{2}$$

For this to be $\leq \epsilon$, let's make each term $\leq \epsilon/2$. Therefore, choose $t = \epsilon/G^2$ and $k = R^2/t \cdot 1/\epsilon = R^2 G^2/\epsilon$

i.e. subgradient method has convergence rate $O(1/\epsilon^2)$, compare this to $O(1/\epsilon)$ rate of gradient descent.

Example: regularized logistic regression

Given $(x_i, y_i) \in \mathbb{R}^p \times \{0, 1\}$ for $i = 1, \cdots, n$. consider the logistic regression loss

$$f(\beta) = \sum_{i=1}^{n} \left(-y_i x_i^T \beta + \log(1 + \exp(x_i^T \beta)) \right)$$

This is a smooth and convex with

$$\nabla f(\beta) = \sum_{i=1}^{n} (y_i - p_i(\beta)) x_i$$

where

$$p_i(\beta) = \frac{\exp(x_i^T \beta)}{1 + \exp(x_i^T \beta)}, i = 1, \cdots, n$$

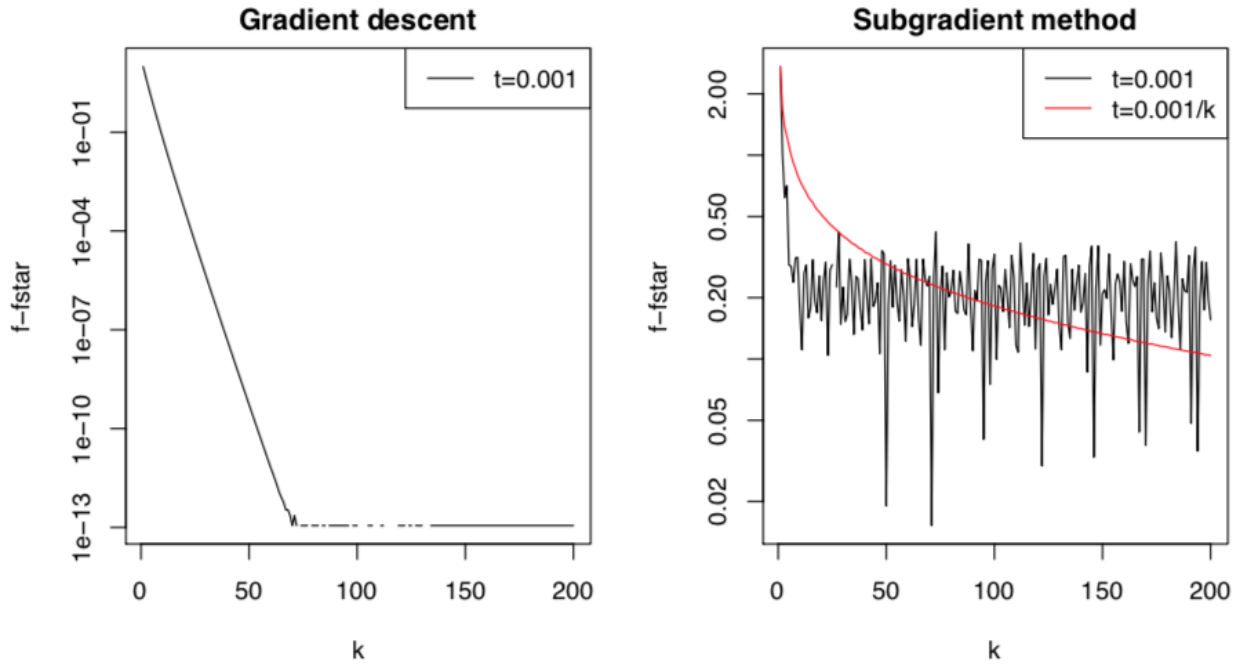
We will consider the regularized problem

$$\min_{\beta} f(\beta) + \lambda \cdot P(\beta)$$

where $P(\beta) = \|\beta\|_2^2$ (ridge penalty) or $\|\beta\|_1$ (lasso penalty)

Example: regularized logistic regression

Ridge problem: use gradients; lasso problem: use subgradients.
Data example with $n = 1000, p = 20$



Step sizes hand-tuned to be favorable for each method (comparison is apparently not perfect)

Polyak step sizes

Polyak step sizes: when the optimal value $f^\star$ is known, take

$$t_k = \frac{f(x^{(k-1)}) - f^\star}{\|g^{(k-1)}\|_2^2}, k = 1, 2, \cdots$$

can be motivated from the first step in subgradient proof

$$\|x^{(k)} - x^\star\|_2^2 \leq \|x^{(k-1)} - x^\star\|_2^2 - 2t_k(f(x^{(k-1)}) - f^\star) + t_k^2\|g^{(k-1)}\|_2^2$$

Polyak step sizes minimizes the RHS.

With Polyak step sizes, can show subgradient method converges to optimal value, but the rate is still $O(1/\epsilon^2)$

Can we do better?

Pros: broad applicability, Cons: convergence rate $O(1/\epsilon^2)$ over problem classes of convex, Lipschitz function is really slow.

Non-smooth first-order methods: iterative methods updating $x^{(k)}$ in

$$x^{(0)} + \text{span}\{g^{(0)}, g^{(1)}, \cdots, g^{(k-1)}\}$$

where subgradients $g^{(0)}, g^{(1)}, \cdots, g^{(k-1)}$ come from weak oracle.

Theorem

For any $k \leq n-1$ and starting point $x^{(0)}$, there is a function in the problem class such that any non-smooth first-order method satisfies

$$f\left(x^{(k)}\right) - f^\star \geq \frac{RG}{2(1+\sqrt{k+1})}$$

Example: intersection of sets

Suppose we want to find $x^\star \in C_1 \cap \cdots \cap C_m$, i.e. in intersection of closed, convex sets $C_1, \cdots, C_m$

First define

$$\begin{aligned} f_i(x) &= \text{dist}(x, C_i), i = 1, \cdots, m \\ f(x) &= \max_{i=1,2,\cdots,m} f_i(x) \end{aligned}$$

and now solve

$$\min_x f(x)$$

Note that $f^\star = 0 \Rightarrow x^\star \in C_1 \cap \cdots \cap C_m$. Check: is this problem convex?

Recall the gradient of the distance function $\text{dist}(x, C) = \min_{y \in C} \|y - x\|_2$ is

$$\nabla \text{dist}(x, C) = \frac{x - P_C(x)}{\|x - P_C(x)\|_2}$$

where $P_C(x)$ is the projection of x onto C

Recall that the subgradient rule: if $f(x) = \max_{i=1,2,\cdots,m} f_i(x)$, then

$$\partial f(x) = \text{conv}\left(\bigcup_{i: f_i(x) = f(x)} \partial f_i(x) \right)$$

so if $f_i(x) = f(x)$ and $g_i \in \partial f_i(x)$, then $g_i \in \partial f(x)$

Put these two facts together for intersection of sets problem, with $f_i(x) = \text{dist}(x, C_i)$: if C_i is furthest set from x (so $f_i(x) = f(x)$) and

$$g_i = \nabla f_i(x) = \frac{x - P_C(x)}{\|x - P_C(x)\_2}$$

then $g_i \in \partial f(x)$

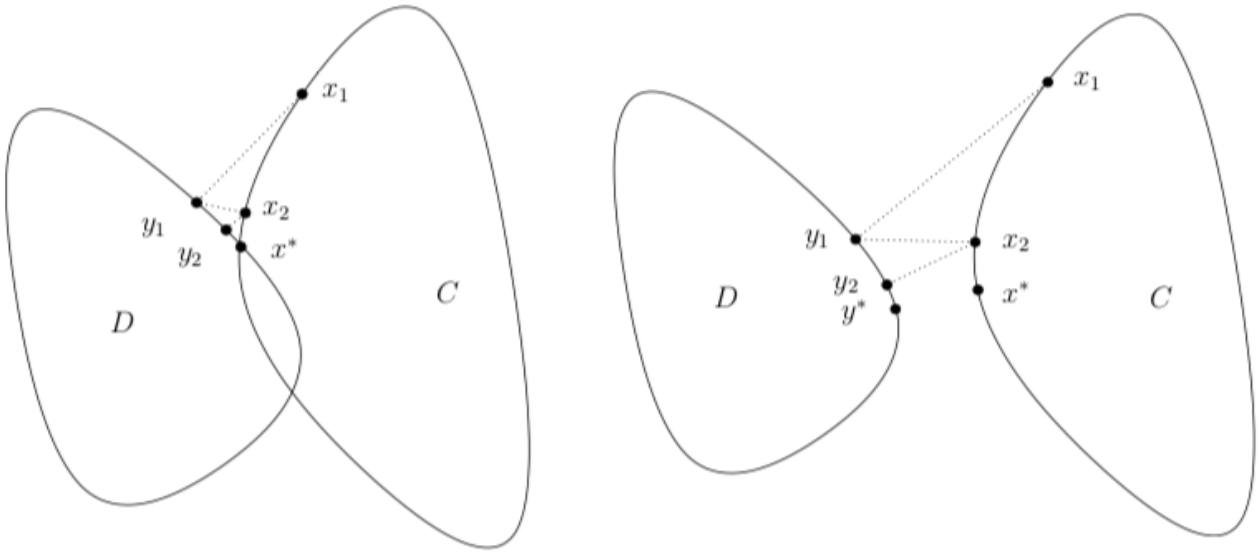
Now apply subgradient method, with Polyak size $t_k = f(x^{(k-1)})$. At iteration k, with C_i furthest from $x^{(k-1)}$, we perform update

$$\begin{aligned} x^{(k)} &= x^{(k-1)} - f(x^{(k-1)}) \frac{x^{(k-1)} - P_{C_i}(x^{(k-1)})}{\|x^{(k-1)} - P_{C_i}(x^{(k-1)})\|_2} \\ &= P_{C_i}(x^{(k-1)}) \end{aligned}$$

Here $f(x^{(k-1)}) = \text{dist}(x^{(k-1)}, C_i) = \|x^{(k-1)} - P_{C_i}(x^{(k-1)})\|_2$

Alternating Projection

For two sets, this is the famous alternating projection algorithm, i.e. just keep projecting back and forth



Projected subgradient method

To optimize a convex function f over a convex set C

$$\min_{x \in C} f(x)$$

we can use the projected subgradient method. Just like the usual subgradient method, except we project onto C at each iteration

$$x^{(k)} = P_C\left(x^{(k-1)} - t_k g^{(k-1)}\right), k = 1, 2, \cdots$$

Assuming we can do this projection, we get the same convergence guarantee as the usual subgradient method, with the same step size choices.

Projected subgradient method

What sets C are easy to project onto? Lots, e.g.

- Affine images: $\{Ax + b : x \in \mathbb{R}^n\}$
- Solution set of linear system: $\{x : Ax = b\}$
- Nonnegative orthant: $\mathbb{R}^n_+ = \{x : x \geq 0\}$
- Some norm balls: $\{x : \|x\|_p \leq 1\}$ for $p = 1, 2, \infty$
- Some simple polyhedra and simple cones

Warning: it is easy to write down seemingly simple set C, and P_C can turn out to be very hard! E.g. generally hard to project onto arbitrary polyhedron $C = \{x : Ax \leq b\}$

Note: projected gradient descent works too.

Stochastic subgradient method

Similar to our setup for stochastic gradient descent. Consider sum of convex functions

$$\min_x \sum_{i=1}^m f_i(x)$$

Stochastic subgradient method repeats

$$x^{(k)} = x^{(k-1)} - t_k \cdot g_{i_k}^{(k-1)}, k = 1, 2, 3, \cdots$$

where $i_k \in \{2, \cdots, m\}$ is some chosen index at iteration k, chosen by either the random or cyclic rule. and $g_{i_k}^{(k-1)} \in \partial f_i(x^{(k-1)})$ (this update direction is used in place of the usual $\sum_i g_i^{(k-1)}$

Note that when each $f_i, i = 1, \cdots, m$ is differentiable, this reduces to stochastic gradient descent.

Convergence of stochastic methods

Assume each $f_i, i = 1, \cdots, m$ is convex and Lipschitz with constant $G > 0$

For fixed step sizes $t_k = t, k = 1, 2, \cdots$,, cyclic and randomized stochastic subgradient methods both satisfy

$$\lim_{k \to \infty} f(x_{\text{best}}^{(k)}) \leq f^\star + 5m^2 G^2 t/2$$

Note: mG can be considered as Lipschitz constant for whole function $\sum_i f_i$, so this comparable to batch bound.

For diminishing step sizes, cyclic and randomized methods satisfy

$$\lim_{k \to \infty} f(x_{\text{best}}^{(k)}) = f^\star$$

Convergence of stochastic methods

How about convergence rate?

Looking back carefully, the batch subgradient method rate was $O(G_{\text{batch}}^2/\epsilon^2)$, where Lipschitz constant G_{batch}^2 is for the whole function

- Cyclic rule: iteration complexity is $O(m^3G^2/\epsilon^2)$, therefore the number of cycles needed is $O(m^2G^2/\epsilon^2)$, comparable to batch
- Randomized rule[1]: iteration complexity $O(m^2G^2/\epsilon^2)$. Thus the number of random cycles needed is $O(mG^2/\epsilon^2)$, reduced by a factor of m!

This is a convincing reason to use randomized stochastic methods, for problems where m is big.

[1]For randomized rule, result holds in expectation, i.e. bound is on expected number of iterations

Example: stochastic logistic regression

Back to the logistic regression problem (now we are talking SGD)

$$\min_{\beta} f(\beta) = \sum_{i=1}^{n} \underbrace{\left(- y_i x_i^T \beta + \log(1 + \exp(x_i^T \beta)) \right)}_{f_i(\beta)}$$

The gradient computation $\nabla f(\beta) = \sum_i \big(y_i - p_i(\beta)\big) x_i$ is doable when n is moderate, but not when $n \approx 500$ millon. Recall

- One batch update costs $O(np)$, p is the number of features
- One stochastic update costs $O(p)$

So clearly, e.g. 10K stochastic steps are much more affordable.

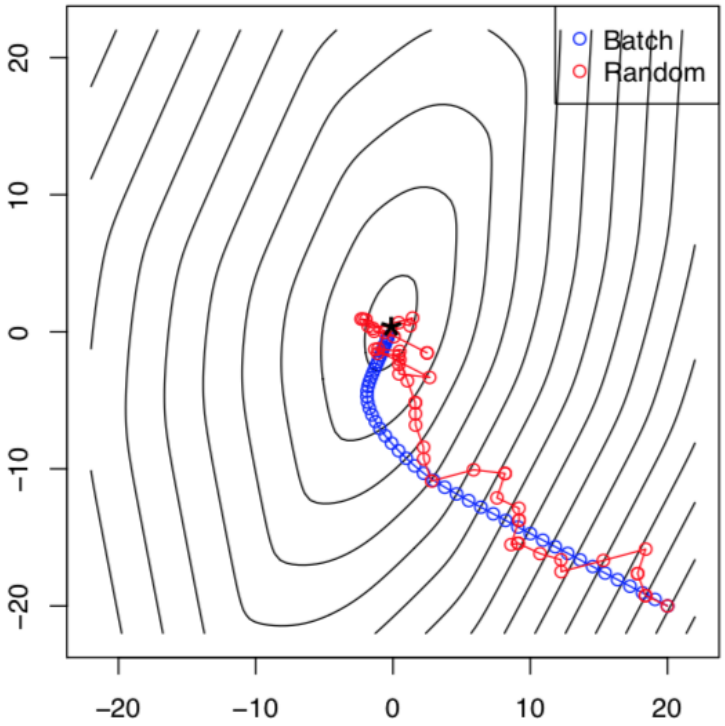


Figure: Blue: batch steps $O(np)$, Red: stochastic steps $O(p)$

Rule of thumb for stochastic methods

- generally thrive far from optimum
- generally struggle close to optimum.

Improving on the subgradient method

In words, we cannot do better than the $O(1/\epsilon^2)$ rate of subgradient method (unless we go beyond nonsmooth first-order methods)

So instead of trying to improve across the board, we will focus on minimizing composite functions of the form

$$f(x) = g(x) + h(x)$$

where g is convex and differentiable, h is convex and nonsmooth but "simple".

For a lot of problems (i.e. function h), we can recover the $O(1/\epsilon)$ rate of gradient descent with a simple algorithm, having important practical consequences.

Questions?