Lecture 10 (Sep/28)

Scribe? HW due tomorrow.

Last time

- Classroom Chaos
- Proof lower bound

Today

- Review of smooth optimization
- Motivating Problems
- Proximal operator

Summary of guarantees for smooth optimazation.

Method	Generic rate (L-smooth)	Quadratic growth
Gradient Descent (for nonconvex f)	$\frac{1}{T}\sum_{k=0}^{T-1} \lVert \nabla f(x_k) \rVert^2 \leq \Theta\left(\frac{1}{T}\right)$	$f(x_T) - f(x^*) \leq \Theta\left(\left(1 - \frac{\mu^2}{4L^2}\right)^T\right)$ (Local rate for $\nabla f(x^*) > 0$)
Gradient Descent (for convex f)	$f(x_T) - \min f \leq \Theta\left(\frac{1}{T}\right)$	$f(x_T) - \min f \leq \Theta\left(\left(\frac{\kappa - 1}{\kappa + 1}\right)^{2T}\right)$ (μ-strongly convex)
Accelerated Gradient (for convex f)	$f(y_T) - \min f \leq \Theta\left(\frac{1}{T^2}\right)$	$f(x_T) - \min f \leq \Theta\left(\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^{2T}\right)$ (μ-strongly convex)

Optimal

HW2 P3 (Also optimal)

What's next? Structured nonsmooth optimization

1. Motivating problems
2. The proximal operator
3. Proximal gradient method
4. Constraints and projections
5. Acceleration
6. More proximal methods.

Motivating problems

Several optimization problems are non-smooth. One common way in which nonsmoothness arise is by promoting structure.

Sparsity

Imagine we wished to solve a linear system

$$Ax = b,$$

This could be solved using least-squares

$$\min \tfrac{1}{2}\|Ax - b\|^2.$$

which works well when $A =$; more constraints than variables. But often in science we have more variables than constraints $A =$. Thus, we have mutiple solutions. Which one to pick?

- This a common problem stats (Regression). A common approach is to pick one with few nonzero entries. ← Good for interpretability

This motivated Rob Tibshirani to propose LASSO

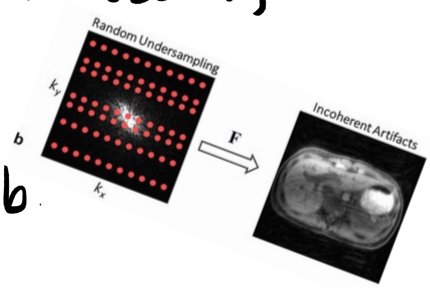
$$\min \ \tfrac{1}{2}\|Ax-b\|^2 + \lambda\|x\|_1 \quad \leftarrow \text{Promotes sparsity}$$

Nonsmooth

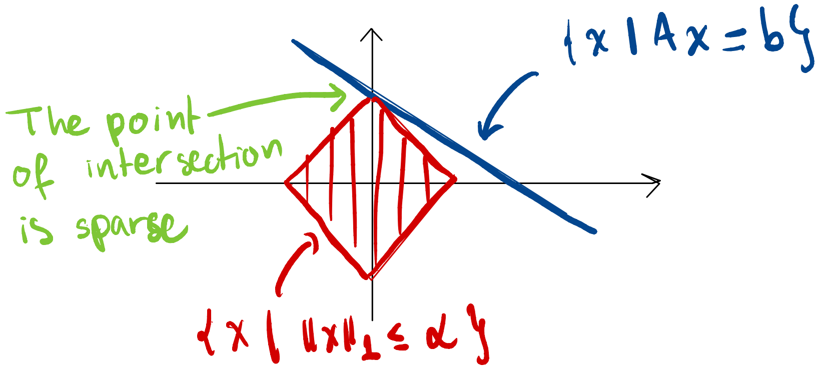
- This is also a common problem in signal processing (inverse problems) when you are trying to recover a sparse signal.

Donoho (2004), Candes, Romberg, Tao (2004) proposed compressed sensing

$$\min_{x \in \mathbb{R}^d} \|x\|_1 \quad \text{s.t.} \quad Ax = b.$$



Intuition



Low - Rankness

Sometimes researchers are interested in recovering a matrix $X \in \mathbb{R}^{d_1 \times d_2}$ satisfying a linear system

$$\underbrace{\mathcal{A}}_{\text{Linear map}}(X) = b \qquad \mathcal{A} : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}^m$$

but $d_1 \times d_2 \gg m$ (less constraints than variables).

Examples arise in

- Signal processing

  The seminal problem of phase retrieval aims to recover a rank 1 matrix X. Other examples include blind deconvolution.

- Recommendation systems

$$\begin{array}{cc} & \text{movies} \\ \text{users} & \begin{bmatrix} 5 & & 3 & & & \cdots \\ 5 & & 1 & & 1 & \\ & 1 & & 2 & & 5 \\ & & \vdots & & & \end{bmatrix} \end{array}$$

The matrix completion problem aims to recover a matrix X from entries (a linear map).

X is assumed to be low-rank (similar people like similar movies).

To solve this problems Fazel (2002) proposed to solve

$$\min \frac{1}{2}\|A(X)-b\|^2 + \lambda \|X\|_*$$

nuclear norm

$$\|X\|_* = \sum_{i=1}^{d_1 \wedge d_2} \sigma_i(X).$$

A class of problems

These examples have the form

$$\min_{x \in \mathbb{R}^d} \underbrace{f(x)}_{\text{smooth}} + \underbrace{h(x)}_{\text{convex (and nicely decomposable)}}.$$

In the next few lectures we will study how to solve optimization problems of this form.

Proximal operator

How do we come up with algorithms?

Approximations!

We saw before that gradient descent can be written as

$$x_{t+1} = \operatorname{argmin} \left\{ f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{1}{2\alpha_t}\|x - x_t\|^2 \right\}.$$

This strategy goes well beyond GD. Given a closed convex function $\Psi : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$.

We define the proximal operator

$$\operatorname{prox}_{\alpha\Psi}(x) = \operatorname*{argmin}_z \left\{ \Psi(z) + \frac{1}{2\alpha} \|z - x\|^2 \right\}.$$

Lemma: The $\operatorname{prox}_{\alpha\Psi} : \mathbb{R}^d \to \mathbb{R}^d$ is well-defined.

Proof: The function $z \mapsto \Psi(z) + \frac{1}{2\alpha} \|z - x\|^2$ is strongly convex. By HW2 it has a unique minimizer. $\square$

Lemma: Let $\Psi : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ be a closed convex function and $f : \mathbb{R}^d \to \mathbb{R}$ be a smooth function. Let $x^* \in \operatorname{argmin} f(x) + \Psi(x)$, then

$$-\nabla f(x^*) \in \partial \Psi(x^*).$$

Proof: Let $x \in \mathbb{R}^d$ and $t \in [0, 1]$

$$\Rightarrow \quad f(x^*) + \Psi(x^*) \le f(\overbrace{x^* + t(x - x^*)}^{x_t}) + \Psi(x^* + t(x - x^*))$$
$$\le f(x_t) + (1 - t)\Psi(x^*) + t\Psi(x)$$

$$\Rightarrow \quad f(x^*) - f(x_t) \le t(\Psi(x) - \Psi(x^*)) \quad (:)$$

By definition of the gradient:

$$\langle -\nabla f(x^*), x - x^* \rangle = \lim_{t \downarrow 0} \frac{f(x^*) - f(x^* + t(x - x^*))}{t}$$

$$\overset{(\ddot\smile)}{\leq} \Psi(x) - \Psi(x^*).$$

$$\Rightarrow -\nabla f(x^*) \in \partial \Psi(x^*).$$

$\square$

Lemma: Let $\Psi : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ be a closed convex function and $f : \mathbb{R}^d \to \mathbb{R}$ be a convex smooth function. Then

$$x^* \in \operatorname{argmin} \Psi(x) + f(x) \Leftrightarrow -\nabla f(x^*) \in \partial \Psi(x^*).$$

Proof: "$\Rightarrow$" ✓

"$\Leftarrow$" For any $x \in \mathbb{R}^d$

$$f(x^*) + \Psi(x^*) \leq f(x) + \langle \nabla f(x^*), x^* - x \rangle + \Psi(x) - \langle \nabla f(x), x^* - x \rangle$$

$$\leq f(x) + \Psi(x).$$

$\square$

Proposition ♡: The point $x^+ = \operatorname{prox}_{\alpha\Psi}(x)$ iff

$$\frac{1}{\alpha}(x - x^+) \in \partial \Psi(x^+).$$

Proof. Follows directly from the previous Lemma. $\square$

The update $x_{k+1} \leftarrow \text{prox}_{\alpha\Psi}(x)$ is usually called an implicit (or backward) step because

$g_k \in \partial\Psi(x_{k+1})$.

$$x_{k+1} = x_k - \alpha g_k$$

That is like gradient descent with the gradient evaluated at the future iterate x_{k+1}.

The proximal operator gives a natural templates to design algorithms:

Loop $k \geq 0$:

Define approximation Ψ_k of f near x_k

Update $x_{k+1} \leftarrow \text{prox}_{\alpha_k \Psi_k}(x_k)$.

Two examples:

Gradient descent

$$\Psi_k(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle$$

Proximal point method

$$\Psi_k(x) = f(x)$$

Each iteration might be just as hard as original problem.

Forward - Backward Method.

When we have a sum $\underset{\text{smooth}}{f} + \underset{\text{convex}}{h}$. we have

a natural approximation

$$\Psi_k(x) = \underbrace{f(x^*) + \langle \nabla f(x^*), x - x^* \rangle}_{\text{linear approximation}} + \underbrace{h(x)}_{\text{perfect approx.}}$$

Then, at each iteration we update

$$x_{k+1} \leftarrow \underset{x}{\operatorname{argmin}} \Big\{ \underbrace{h(x)}_{\text{convex}} + \underbrace{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2}_{\text{smooth part}} \Big\}$$

By Lemma ★

$$\frac{1}{\alpha_k}(x_k - \alpha_k \nabla f(x_k) - x_{k+1}) \in \partial h(x)$$

By Proposition ♡, this is equivalent to

$$x_{k+1} = \underbrace{\operatorname{prox}_{\alpha_k h}}_{\text{Backward step}} (\underbrace{x_k - \alpha_k f(x_k)}_{\text{Forward step}}).$$

Thus, this method works well for