Proximal Methods

Tobia Boschi

Stat 540 - Spring 2108

Why Proximal Methods?

- **High dimensional convex problems**
  - non-differentiable
  - constrained
  - large-size and parallel implementations

- **Proximal methods to solve LASSO**

Key Idea

- avoid *gradient* and *hessian* computation
- evaluate instead the *proximal operator*:

$\longrightarrow$ small convex optimization problem

Definition

Let $f : R^n \to R$ be a closed proper *convex* function.
The proximal operator $\text{prox}_{\lambda f} : R^n \to R^n$ is defined by:

$$\text{prox}_{\lambda f}(x) = \text{argmin}_z \left(f(z) + \frac{1}{2\lambda} ||z - x||_2^2 \right)$$

It balances two goals:

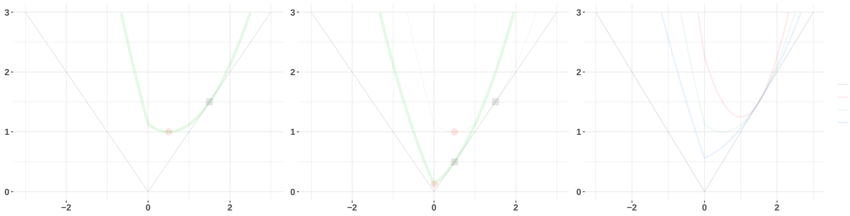
- minimizing f
- staying near x

Possible interpretations

- surrogate method: $f(x) \longrightarrow f(z) + \frac{1}{2\lambda} ||z - x||_2^2$
- gradient step for f: $\text{prox}_{\lambda f}(x) \cong x - \lambda \nabla f(x)$

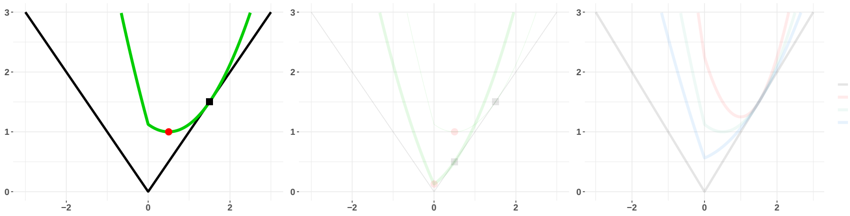
Simple Example

- $f(x) = |x|$
- $|z| + \frac{1}{2\lambda}(x - z)^2$
- $\text{prox}_{\lambda f}(x) = \text{sign}(x)(|x| - \lambda)_+$



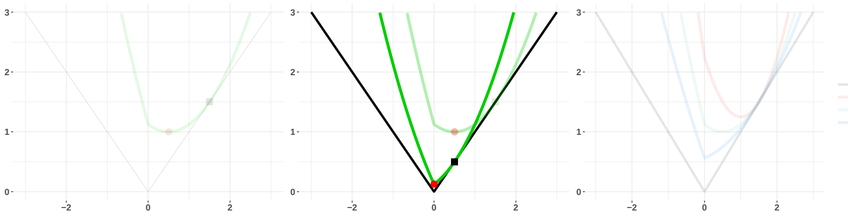
Simple Example

- $f(x) = |x|$
- $|z| + \frac{1}{2\lambda}(x - z)^2$
- $\text{prox}_{\lambda f}(x) = \text{sign}(x)(|x| - \lambda)_+$



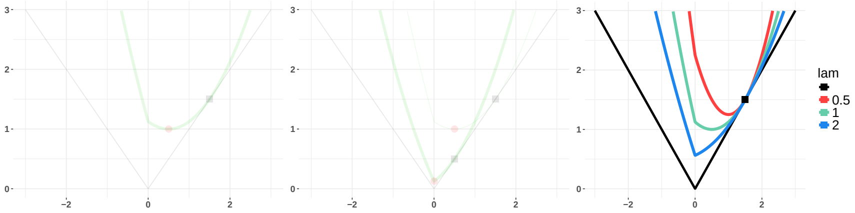
Simple Example

- $f(x) = |x|$
- $|z| + \frac{1}{2\lambda}(x - z)^2$
- $\mathrm{prox}_{\lambda f}(x) = \mathrm{sign}(x)(|x| - \lambda)_+$



Simple Example

- $f(x) = |x|$
- $|z| + \frac{1}{2\lambda}(x - z)^2$
- $\text{prox}_{\lambda f}(x) = \text{sign}(x)(|x| - \lambda)_+$



Proximal Methods and Regression

Let $x \in R^n, b \in R^m, A \in R^{m \times n}$. We want to minimize:

$$g(x) + h(x) = \frac{1}{2}||Ax - b||_2^2 + h(x)$$

- LASSO: $h(x) = \gamma ||x||_1$
- RIDGE: $h(x) = \frac{\gamma}{2}||x||_2^2$
- ELASTIC: $h(x) = \gamma_1 ||x||_1 + \frac{\gamma_2}{2}||x||_2^2$

Implemented Algorithms

- Proximal Gradient
- Proximal ADMM
  (alternating direction method of multipliers)

Proximal Methods and Regression

Let $x \in R^n, b \in R^m, A \in R^{m \times n}$. We want to minimize:

$$g(x) + h(x) = \frac{1}{2}||Ax - b||_2^2 + h(x)$$

- LASSO: $h(x) = \gamma ||x||_1$
- RIDGE: $h(x) = \frac{\gamma}{2}||x||_2^2$
- ELASTIC: $h(x) = \gamma_1 ||x||_1 + \frac{\gamma_2}{2}||x||_2^2$

Implemented Algorithms

- Proximal Gradient
- Proximal ADMM
  (alternating direction method of multipliers)

Proximal Gradient

Pseudo-code

- Gradient step

$$v^k = x^k - \lambda \nabla g(x^k)$$

- Proximal operator step

$$x^{k+1} = \mathrm{prox}_{\lambda h}(v^k)$$

Proximal Gradient

Pseudo-code

- Gradient step

$$v^k = x^k - \lambda \nabla \boldsymbol{g}(x^k)$$

- Proximal operator step

$$x^{k+1} = \mathrm{prox}_{\lambda \boldsymbol{h}}(v^k)$$

- $\nabla g(x) = A'Ax - A'b$
  - precompute $A'A$ and $A'b$
  - at each iteration evaluate $A'Ax$: $\mathcal{O}(n^2)$
- $\left(\mathrm{prox}_{\lambda\gamma \|\boldsymbol{x}\|_1}(x)\right)_i = \mathrm{prox}_{\lambda\gamma|x_i|}(x) = \mathrm{sign}(x)(|x| - \lambda\gamma)_+$

Proximal Gradient

- Proximal Gradient as an Majorization-Minimization algorithm

$$\hat{g}_\lambda(x, y) = g(y) + \nabla g(y)'(x - y) + \frac{1}{2\lambda}||x - y||_2^2 \geq g(x)$$

Majorization step

- compute $\hat{g}_\lambda(x, x^k) + h(x)$

Minimizaztion step

- $\min_x \left(\hat{g}_\lambda(x, x^k) + h(x)\right) = \mathrm{prox}_{\lambda \boldsymbol{h}}\left(x^k - \lambda \nabla \boldsymbol{g}(x^k)\right)$

$$= \mathrm{prox}_{\lambda \boldsymbol{h}}(v^k)$$

Proximal Gradient

- Proximal Gradient as an Majorization-Minimization algorithm

$$\hat{g}_\lambda(x, y) = g(y) + \nabla g(y)'(x - y) + \frac{1}{2\lambda}||x - y||_2^2 \geq g(x)$$

Majorization step

- compute $\hat{g}_\lambda(x, x^k) + h(x)$

Minimizaztion step

- $\min_x \left(\hat{g}_\lambda(x, x^k) + h(x)\right) = \mathrm{prox}_{\lambda \boldsymbol{h}}\left(x^k - \lambda \nabla \boldsymbol{g}(x^k)\right)$
  $= \mathrm{prox}_{\lambda \boldsymbol{h}}(v^k)$

- **Backtracking of λ**
  - $x = \mathrm{prox}_{\lambda \boldsymbol{h}}\left(x^k - \lambda \nabla \boldsymbol{g}(x^k)\right)$
  - *reduce λ since*: $g(x) \leq \hat{g}_\lambda(x, x^k)$

Proximal ADMM

Note: minimize $g(x) + h(x)$ is equivalent to minimize:

$$g(x) + h(z) \quad \textit{subject to} \quad x - z = 0$$

Proximal ADMM

Note: minimize $g(x) + h(x)$ is equivalent to minimize:

$$g(x) + h(z) \quad \textit{subject to} \quad x - z = 0$$

Pseudo-code

- ◎ $x^{k+1} = \text{prox}_{\lambda g}(z^k - u^k)$
- ◎ $z^{k+1} = \text{prox}_{\lambda h}(x^{k+1} + u^k)$
- ◎ $u^{k+1} = u^k + x^{k+1} - z^{k+1}$

- $g(x) = \frac{1}{2}||Ax - b||_2^2 = \frac{1}{2}(b - Ax)'(b - Ax)$
- $\text{prox}_{\lambda g}(x) = (I_n + \lambda A'A)^{-1}(x + \lambda A'b)$

Proximal ADMM

How to compute $(I_n + \lambda A'A)^{-1}$:

- $\boldsymbol{n > m}$
  - $C = \text{chol}(I_n + \lambda A'A)$
  - $(I_n + \lambda A'A)^{-1} = (C')^{-1}C^{-1}$
  - $\mathcal{O}(n^3)$

- $\boldsymbol{m > n}$
  - inversion lemma: $\quad (I_n + \lambda A'A)^{-1} = A'(I_m + \lambda AA')^{-1}A$
  - $C = \text{chol}(I_m + \lambda AA')$
  - $\mathcal{O}(m^3)$

Simulation 1

- A = random normal$(m \times n)$
- x_0 = random normal$(n \times 1)$
- $b = Ax_0 + \text{err}$
- sparsity = 0.95
- $\lambda = 1$
- $\gamma = 0.1\|A'b\|_\infty$

◎ $\boldsymbol{m = 500, \ n \uparrow}$

	sec			**obj**		
	CVX	grad	ADMM	CVX	err grad	err ADMM
$n = 10^2$	0.10966	0.00136	0.02259	0.4606	0	0
$n = 10^3$	11.10568	0.02289	0.02589	8.5773	0.0010	0.0003
$n = 10^4$	155.49445	3.97270	0.57923	86.8624	0.1288	0.00665
$n = 4 \cdot 10^4$	820.12007	111.32019	4.21288	305.0444	1.0231	0.08624

Simulation 1

- A = random normal$(m \times n)$
- x_0 = random normal$(n \times 1)$
- $b = Ax_0 + \text{err}$
- sparsity = 0.95
- $\lambda = 1$
- $\gamma = 0.1\|A'b\|_\infty$

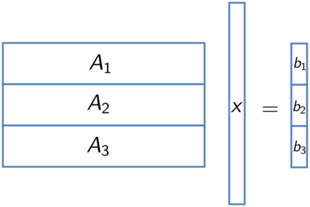
◎ $m = 500,\ n \uparrow$

	sec			obj		
	CVX	grad	ADMM	CVX	err grad	err ADMM
$n = 10^2$	0.10966	0.00136	0.02259	0.4606	0	0
$n = 10^3$	11.10568	0.02289	0.02589	8.5773	0.0010	0.0003
$n = 10^4$	155.49445	3.97270	0.57923	86.8624	0.1288	0.00665
$n = 4 \cdot 10^4$	820.12007	111.32019	4.21288	305.0444	1.0231	0.08624

$n >> m \Rightarrow$ ADMM $\mathcal{O}(m^3)$ > Gradient $\mathcal{O}(n^2)$

Distributed ADMM

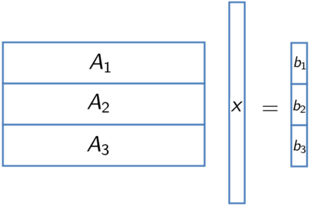
◎ reduce $m \longrightarrow$ divide A in S blocks



- $$\begin{aligned} g(x) &= \sum g_i(x) \\ &= \sum \tfrac{1}{2} \|A_i x - b_i\|_2^2 \end{aligned}$$
- $\mathcal{O}\big((m/S)^3\big)$

Distributed ADMM

- ◎ reduce m $\longrightarrow$ divide A in S blocks



- $g(x) = \sum g_i(x)$
  $= \sum \frac{1}{2}||A_i x - b_i||_2^2$
- $\mathcal{O}\left((m/S)^3\right)$

Pseudo-code

- ◎ $x_{\boldsymbol{i}}^{k+1} = \text{prox}_{\lambda \boldsymbol{g_i}}(z^k - u_{\boldsymbol{i}}^k)$
- ◎ $z^{k+1} = \text{prox}_{\frac{\lambda h}{\boldsymbol{S}}}(\bar{x}^{k+1} + \bar{u}^k)$
- ◎ $u_{\boldsymbol{i}}^{k+1} = u_{\boldsymbol{i}}^k + x_{\boldsymbol{i}}^{k+1} - z^{k+1}$

Simulation 2

- $m = 10.000, \; n = 50.000$
- $S = 10$

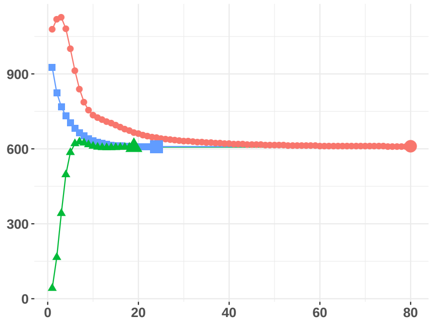
sec			obj		
grad	ADMM	distr	grad	ADMM	distr
3463	423	90	609.012	607.046	609.747

Simulation 2

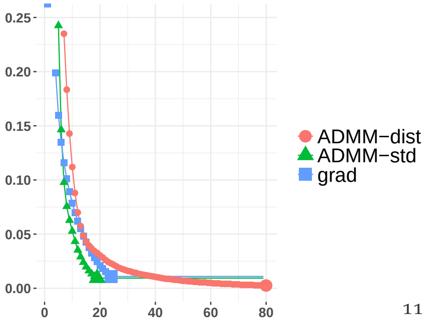
- $m = 10.000,\ n = 50.000$
- $S = 10$

sec			obj		
grad	ADMM	distr	grad	ADMM	distr
3463	423	90	609.012	607.046	609.747

objective



$\frac{||x^{k+1}-x^k||_2}{||x^k||_2}$



Conclusions

Pros

- non-smooth problem
- much faster
- easy to parallelize
- good approximations

Cons

- convex problems
- pointwise estimation
- approximations

More work...

- sensitivity study for λ and γ
- add performance indicators *(prediction error)*
- study constrained problems *(matrix decomposition)*