An Augmented ADMM Algorithm for Linearly Regularized Statistical Estimation Problems

Yunzhang Zhu

The Ohio State University

April 10, 2015

Outline

Introduction

- Optimization problem of interest (with f, g convex)

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{\theta}) + g(\boldsymbol{A}\boldsymbol{\theta})$$

- Motivation comes primarily from generalized lasso problem

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{\theta}) + \|\boldsymbol{A}\boldsymbol{\theta}\|_1,$$

  which includes fused lasso (Tibshirani et al., 2005), grouping pursuit (Shen and Huang, 2010; Zhu et al., 2013; Ke et al., 2013), OSCAR (Bondell and Reich, 2008), and trend filtering (Tibshirani 2014), among others.

- Another interesting example is convex clustering

$$\underset{\boldsymbol{u}_1,\cdots,\boldsymbol{u}_n\in\mathbb{R}^p}{\text{minimize}}\ \frac{1}{2}\sum_{i=1}^{n}\|\boldsymbol{u}_i - \boldsymbol{x}_i\|_2^2 + \lambda\sum_{i<j} w_{ij}\|\boldsymbol{u}_i - \boldsymbol{u}_j\|_q,$$

Existing literature

- Path following (homotopy) algorithms (Shen and Huang, 2010; Tibshirani and Taylor, 2011; Zhou and Wu, 2014; Arnold and Tibshirani, 2014)
- Fast first-order algorithms for composite functions (Becker et al., 2011; Beck and Teboulle, 2012)
- Alternating direction methods of multipliers (Boyd et al., 2011; Ramdas and Tibshirani, 2014)

Alternating direction methods of multipliers

- a method
  - with very good robustness
  - which can support parallelization
- proposed by Gabay, Mercier, Glowinski, Marrocco in 1976
- ADMM problem form (with f, g convex)

$$\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z} \end{array}$$

  - two sets of variables, with separable objective
- $L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{z}) + (\rho/2)\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{z}\|_2^2$
- ADMM:

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} \ L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k) \\ \boldsymbol{z}^{k+1} &= \arg\min_{\boldsymbol{z}} \ L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k) \\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \rho(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}) \end{aligned}$$

An example: Lasso

- lasso problem:

$$\text{minimize} \quad (1/2)\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1$$

- ADMM form:

$$\begin{aligned} \text{minimize} \quad & (1/2)\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\gamma}\|_1 \\ \text{subject to} \quad & \boldsymbol{\beta} = \boldsymbol{\gamma} \end{aligned}$$

- ADMM:

$$\begin{aligned} \boldsymbol{\beta}^{k+1} &= (\boldsymbol{X}^T\boldsymbol{X} + \rho\boldsymbol{I})^{-1}(\boldsymbol{X}^T\boldsymbol{Y} + \rho\boldsymbol{\gamma}^k - \boldsymbol{\alpha}^k) \\ \boldsymbol{\gamma}^{k+1} &= S_{\lambda/\rho}(\boldsymbol{\beta}^{k+1} + \boldsymbol{\alpha}^k/\rho) \\ \boldsymbol{\alpha}^{k+1} &= \boldsymbol{\alpha}^k + \rho(\boldsymbol{\beta}^{k+1} - \boldsymbol{\gamma}^{k+1}) \end{aligned}$$

Another example: distributed optimization

- Consider a problem with N objective terms

$$\text{minimize} \quad \sum_{i=1}^{N} f_i(\boldsymbol{\theta})$$

  - e.g., f_i is the loss function for ith block of training data

- ADMM form:

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{N} f_i(\boldsymbol{\theta}_i) \\ \text{subject to} & \boldsymbol{\theta}_i = \boldsymbol{\theta}, i = 1, \cdots, N \end{array}$$

  - $\boldsymbol{\theta}_i$ are *local* variables
  - $\boldsymbol{\theta}$ is the *global* variable
  - $\boldsymbol{\theta}_i - \boldsymbol{\theta} = 0$ are *consensus* constraints
  - can add regularization function $g(\boldsymbol{\theta})$

Another example: distributed optimization

- $L_\rho(\boldsymbol{\theta}_i, \boldsymbol{\alpha}_i, \boldsymbol{\theta}) = \sum_{i=1}^N \left(f_i(\boldsymbol{\theta}_i) + \boldsymbol{\alpha}_i^T(\boldsymbol{\theta}_i - \boldsymbol{\theta}) + (\rho/2)\|\boldsymbol{\theta}_i - \boldsymbol{\theta}\|_2^2 \right)$
- ADMM:

$$\boldsymbol{\theta}_i^{k+1} = \arg\min_{\boldsymbol{\theta}_i} \left(f_i(\boldsymbol{\theta}_i) + \langle \boldsymbol{\alpha}_i^k \, , \, \boldsymbol{\theta}_i - \bar{\theta}^k \rangle + (\rho/2)\|\boldsymbol{\theta}_i - \bar{\boldsymbol{\theta}}^k\|_2^2 \right)$$

$$\boldsymbol{\alpha}_i^{k+1} = \boldsymbol{\alpha}_i^k + \rho(\boldsymbol{\theta}_i^{k+1} - \bar{\boldsymbol{\theta}}^{k+1})$$

  where $\bar{\boldsymbol{\theta}}^k = (1/N)\sum_{i=1}^N \boldsymbol{\theta}_i^k$
- in each iteration
  - gather local estimator $\boldsymbol{\theta}^k$ and average to get $\bar{\boldsymbol{\theta}}^k$
  - scatter the "aggregated" estimator $\bar{\boldsymbol{\theta}}^k$ to processors
  - update $\boldsymbol{\alpha}_i^k$ locally (in parallel)
  - update $\boldsymbol{\theta}_i$ locally (in parallel)
- split data and use ADMM to reach consensus!

Standard ADMM

- Recall, we want to solve

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{\theta}) + g(\boldsymbol{A}\boldsymbol{\theta})$$

  Standard ADMM form:

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p,\boldsymbol{\gamma}\in\mathbb{R}^m}{\text{minimize}}\ f(\boldsymbol{\theta}) + g(\boldsymbol{\gamma}) \quad \text{subject to: } \boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\gamma} = \boldsymbol{0}$$

- Standard ADMM:

$$\begin{aligned}
\boldsymbol{\theta}^{k+1} &= \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min}\ \left(f(\boldsymbol{\theta}) + \frac{\rho}{2}\left\|\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\gamma}^k + \rho^{-1}\boldsymbol{\alpha}^k\right\|_2^2 \right), \\
\boldsymbol{\gamma}^{k+1} &= \underset{\boldsymbol{\gamma}\in\mathbb{R}^m}{\arg\min}\ \left(\frac{\rho}{2}\left\|\boldsymbol{A}\boldsymbol{\theta}^{k+1} - \boldsymbol{\gamma} + \rho^{-1}\boldsymbol{\alpha}^k\right\|_2^2 + g(\boldsymbol{\gamma}) \right), \\
\boldsymbol{\alpha}^{k+1} &= \boldsymbol{\alpha}^k + \rho(\boldsymbol{A}\boldsymbol{\theta}^{k+1} - \boldsymbol{\gamma}^{k+1}).
\end{aligned}$$

Generalized lasso problem

- Consider the generalized lasso problem

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\text{minimize}} \ \frac{1}{2n} \|\boldsymbol{Y} - \boldsymbol{X}\beta\|_2^2 + \lambda_1 \|\boldsymbol{A}\boldsymbol{\beta}\|_1$$

- Standard ADMM:

$$\boldsymbol{\beta}^{k+1} = \left(\rho \boldsymbol{A}^T \boldsymbol{A} + \boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \left(\rho \boldsymbol{A}^T \boldsymbol{A} \beta^k + \boldsymbol{X}^T \boldsymbol{Y} - \boldsymbol{A}^T (2\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})\right),$$
$$\boldsymbol{\alpha}^{k+1} = \mathcal{P}_{\left\{\|\boldsymbol{\alpha}\|_\infty \le \lambda\right\}} \left(\boldsymbol{\alpha}^k + \rho \boldsymbol{A} \boldsymbol{\beta}^{k+1}\right).$$

- Potential difficulty: the first update involves solving large linear system
- This motivates the augmented ADMM algorithm

The augmented ADMM

- The idea: we consider an "augmented" variable $(\gamma, \tilde{\gamma})$ by augmenting a new variable $\tilde{\gamma}$ to γ, where $\tilde{\gamma}$ relates to $\boldsymbol{\theta}$ through
$$\tilde{\gamma} = (\boldsymbol{D} - \boldsymbol{A}^T\boldsymbol{A})^{1/2}\boldsymbol{\theta}$$
with $\boldsymbol{D} \succeq \boldsymbol{A}^T\boldsymbol{A}$.
- Augmented ADMM form:
$$\begin{aligned} &\underset{\boldsymbol{\theta},\tilde{\boldsymbol{\gamma}}\in\mathbb{R}^p,\boldsymbol{\gamma}\in\mathbb{R}^m}{\text{minimize}} \; f(\boldsymbol{\theta}) + g(\gamma) \\ &\text{subject to: } \begin{pmatrix} \boldsymbol{A} \\ (\boldsymbol{D} - \boldsymbol{A}^T\boldsymbol{A})^{1/2} \end{pmatrix}\boldsymbol{\theta} - \begin{pmatrix} \gamma \\ \tilde{\gamma} \end{pmatrix} = \boldsymbol{0} \end{aligned}$$
- This seems to make the problem more difficult, but...

The augmented ADMM

- Applying standard ADMM to

$$\begin{aligned} &\underset{\boldsymbol{\theta},\tilde{\boldsymbol{\gamma}}\in\mathbb{R}^p,\boldsymbol{\gamma}\in\mathbb{R}^m}{\text{minimize}}\ f(\boldsymbol{\theta}) + g(\boldsymbol{\gamma}) \\ &\text{subject to: } \begin{pmatrix} \boldsymbol{A} \\ (\boldsymbol{D} - \boldsymbol{A}^T\boldsymbol{A})^{1/2} \end{pmatrix} \boldsymbol{\theta} - \begin{pmatrix} \boldsymbol{\gamma} \\ \tilde{\boldsymbol{\gamma}} \end{pmatrix} = \boldsymbol{0} \end{aligned}$$

  gives new ADMM:

$$\begin{aligned} \boldsymbol{\theta}^{k+1} &= \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min}\ \left(f(\boldsymbol{\theta}) + (2\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})^T \boldsymbol{A}\boldsymbol{\theta} + \frac{\rho}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^k)^T \boldsymbol{D}(\boldsymbol{\theta} - \boldsymbol{\theta}^k) \right) \\ \boldsymbol{\gamma}^{k+1} &= \underset{\boldsymbol{\gamma}\in\mathbb{R}^m}{\arg\min}\ \left(\frac{\rho}{2} \left\| \boldsymbol{A}\boldsymbol{\theta}^{k+1} - \boldsymbol{\gamma} + \rho^{-1}\boldsymbol{\alpha}^k \right\|_2^2 + g(\boldsymbol{\gamma}) \right) \\ \boldsymbol{\alpha}^{k+1} &= \boldsymbol{\alpha}^k + \rho(\boldsymbol{A}\boldsymbol{\theta}^{k+1} - \boldsymbol{\gamma}^{k+1}) \end{aligned}$$

- Remarkably, it does not involve the augmented variable $\tilde{\gamma}$!
- And more importantly, the $\boldsymbol{\theta}$-update is simplified!

The earlier example

- Generalized lasso problem

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\text{minimize}}\ \frac{1}{2n}\|\boldsymbol{Y}-\boldsymbol{X}\beta\|_2^2+\lambda\|\boldsymbol{A}\beta\|_1$$

- Standard ADMM:

$$\boldsymbol{\beta}^{k+1}=\left(\rho\boldsymbol{A}^T\boldsymbol{A}+\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\left(\rho\boldsymbol{A}^T\boldsymbol{A}\beta^k+\boldsymbol{X}^T\boldsymbol{Y}-\boldsymbol{A}^T(2\boldsymbol{\alpha}^k-\boldsymbol{\alpha}^{k-1})\right)$$
$$\boldsymbol{\alpha}^{k+1}=\mathcal{P}_{\left\{\|\boldsymbol{\alpha}\|_\infty\le\lambda\right\}}\left(\boldsymbol{\alpha}^k+\rho\boldsymbol{A}\beta^{k+1}\right)$$

- Augmented ADMM:

$$\boldsymbol{\beta}^{k+1}=\left(\rho\boldsymbol{D}+\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\left(\rho\boldsymbol{D}\boldsymbol{\beta}^k+\boldsymbol{X}^T\boldsymbol{Y}-\boldsymbol{A}^T(2\boldsymbol{\alpha}^k-\boldsymbol{\alpha}^{k-1})\right)$$
$$\boldsymbol{\alpha}^{k+1}=\mathcal{P}_{\left\{\|\boldsymbol{\alpha}\|_\infty\le\lambda\right\}}\left(\boldsymbol{\alpha}^k+\rho\boldsymbol{A}\beta^{k+1}\right)$$

- θ-update simplifies with special choice of $\boldsymbol{D}$:

$$(\rho\boldsymbol{D}+\boldsymbol{X}^T\boldsymbol{X})^{-1}=\rho^{-1}\boldsymbol{D}^{-1}-\rho^{-2}\boldsymbol{D}^{-1}\boldsymbol{X}^T(\boldsymbol{I}+\rho^{-1}\boldsymbol{X}\boldsymbol{D}^{-1}\boldsymbol{X}^T)^{-1}\boldsymbol{X}\boldsymbol{D}^{-1}$$

Termination criterion

- If conjugate functions of $f(\cdot)$ and $g(\cdot)$ are available, we stop the algorithm when the duality gap is small:

$$f(\boldsymbol{\theta}^k) + g(\boldsymbol{A}\boldsymbol{\theta}^k) + f^*(-\boldsymbol{A}^T\boldsymbol{\alpha}^k) + g^*(\boldsymbol{\alpha}^k) \leq \epsilon$$

- Otherwise, following termination criterion proposed in Boyd et al. (2011), we define

$$\boldsymbol{r}^{k+1} = \boldsymbol{A}\boldsymbol{\theta}^{k+1} - \boldsymbol{\gamma}^{k+1}, \quad \epsilon^{\text{pri}} = \sqrt{m}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\boldsymbol{A}\boldsymbol{\theta}^{k+1}\|_2$$
$$\boldsymbol{s}^{k+1} = \rho\boldsymbol{A}^T(\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^k), \quad \epsilon^{\text{dual}} = \sqrt{p}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\boldsymbol{A}^T\boldsymbol{\alpha}^{k+1}\|_2$$

and terminate when $\|\boldsymbol{r}^{k+1}\|_2 \leq \epsilon^{\text{pri}}, \quad \|\boldsymbol{s}^{k+1}\|_2 \leq \epsilon^{\text{dual}}$,

Acceleration

- Theoretical rate of convergence: $O(1/k)$ for a fixed penalty parameter $\rho > 0$ (c.f. He and Yuan (2012))
- In practice, the convergence speed depends heavily on ρ, and it is not clear how to choose ρ.
- A simple varying ρ strategy is given in Boyd et al. (2011), but is found to be unstable by Ramdas and Tibshirani (2014).
- We propose a more stable strategy: update ρ as follows

$$\rho = \begin{cases} 2\rho & \text{if } \|r^k\|_2/\epsilon^{\text{pri}} \geq 10\|s^k\|_2/\epsilon^{\text{dual}} \\ 2^{-1}\rho & \text{if } \|s^k\|_2/\epsilon^{\text{dual}} \geq 10\|r^k\|_2/\epsilon^{\text{pri}}, \end{cases}$$

for every k_i iterations with $\{k_i\}_{i=1}^{\infty}$ being an increasing sequence and $\lim_{i\to\infty} k_i = \infty$.

Simulation set-up

- $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$; $\boldsymbol{Y} \in \mathbb{R}^n$ is the response vector, $\boldsymbol{X}$ is a $n \times p$
- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ over covariates $X_1, \cdots, X_p$
- Let $\boldsymbol{C} \in \mathbb{R}^{m \times p}$ be its associated *oriented incidence matrix*, where $m = |\mathcal{E}|$ is the number of edges in graph $\mathcal{G}$.
- For fused graph,

$$\boldsymbol{C} = \begin{pmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{pmatrix}_{(p-1)\times p}$$

- Solve

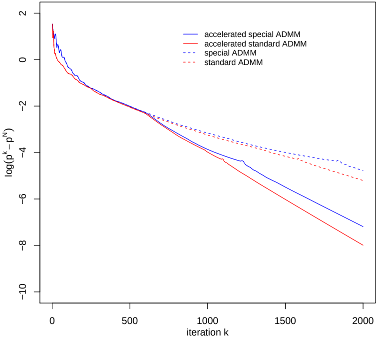
$$\underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\text{minimize}} \ \frac{1}{2n} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{C}\boldsymbol{\beta}\|_1$$

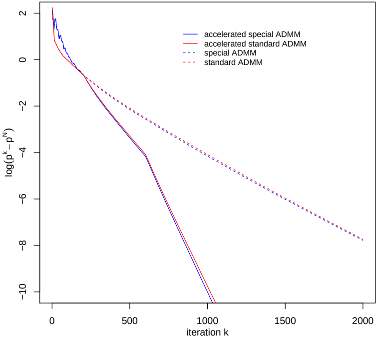
for 100 pairs (λ_1, λ_2).

Running time

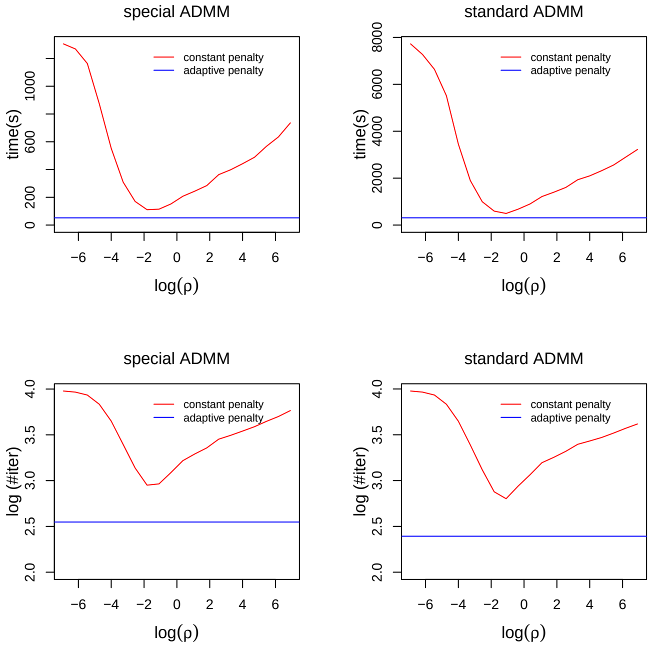
	$(n, p, m) = (200, 2200, 11440)$			$(n, p, m) = (300, 3300, 16940)$		
cor = .5	time(s)	#chol	#iter	time(s)	#chol	#iter
accelerated augmented ADMM	42	25	332	114	25	536
accelerated ADMM	286	26	227	803	25	285
augmented ADMM	87	100	668	256	100	1191
ADMM	531	100	282	1684	100	366
cor = .7	time(s)	#chol	#iter	time(s)	#chol	#iter
accelerated augmented ADMM	52	30	353	115	28	541
accelerated ADMM	309	30	247	972	25	316
augmented ADMM	115	100	784	286	100	1327
ADMM	661	100	408	2083	100	491
cor = .9	time(s)	#chol	#iter	time(s)	#chol	#iter
accelerated augmented ADMM	56	26	371	139	36	571
accelerated ADMM	330	28	234	1045	32	314
augmented ADMM	147	100	951	388	100	1443
ADMM	931	100	659	2623	100	692

Objective sub-optimality versus iterations





Acceleration



Summary

- ADMM algorithm is quite flexible for large-scale machine learning problems
- With some tricks, it gives simple single-processor algorithms that can be competitive with state-of-the-art
- Can be used to coordinate many processors, each solving a substantial problem, to solve a very large problem
- The proposal is closely connected to some algorithms used in imaging literature
- Other potential applications include isotonic regression, trend filtering, and convex clustering
- Hard to find diagonal matrix that dominates $\boldsymbol{A}^T\boldsymbol{A}$? Find a symmetric diagonal dominated (SDD) matrix instead, and resort to SDD linear system solvers (e.g., Koutis et al., 2014)
- Extensions to generalized linear models in which there are two linear operators