Alternating Direction Method of Multipliers

a talk by

Vinícius and Prof. Daniel Palomar

The Hong Kong University of Science and Technology

ELEC5470/IEDA6100A - Convex Optimization

December, 2020

Contents

Why use optimization algorithms?

Motivations

- large-scale optimization
  - machine learning/statistics with huge datasets
  - computer vision
- descentralized optimization
  - entities/agents/threads coordinate to solve a large problem by passing small messages

Optimization Algorithms

- Gradient Descent
- Newton
- Interior Point Methods (IPM)
- Block Coordinate Descent (BCD)
- Majorization-Minimization (MM)
- Block Majorization-Minimization (BMM)
- Successive Convex Approximation (SCA)

Optimization Algorithms

- Gradient Descent
- Newton
- Interior Point Methods (IPM)
- Block Coordinate Descent (BCD)
- Majorization-Minimization (MM)
- Block Majorization-Minimization (BMM)
- Successive Convex Approximation (SCA)
- ...

Optimization Algorithms

- Gradient Descent
- Newton
- Interior Point Methods (IPM)
- Block Coordinate Descent (BCD)
- Majorization-Minimization (MM)
- Block Majorization-Minimization (BMM)
- Successive Convex Approximation (SCA)
- ...
- **Alternating Direction Method of Multipliers (ADMM)**

Reference

- Boyd *et al.* **Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers**. *Foundations and Trends in Machine Learning*. 2010.
- available online for **free**: https://web.stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf
- citations: 13519[1]
- Boyd's presentation: https://web.stanford.edu/class/ee364b/lectures/admm_slides.pdf
- Yuxin Chen's Princeton lecture notes ELE 522: Large-Scale Optimization for Data Science

[1] as of Nov. 24th 2020

Dual Problem

- convex equality constrained optimization problem

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

- Lagrangian: $L(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) + \boldsymbol{y}^\top(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$
- dual function: $g(\boldsymbol{y}) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y})$
- dual problem: $\underset{\boldsymbol{y}}{\text{maximize}} \; g(\boldsymbol{y})$
- recover: $\boldsymbol{x}^\star = \underset{\boldsymbol{x}}{\text{argmin}} \; L(\boldsymbol{x}, \boldsymbol{y}^\star)$

Dual Ascent

Dual Ascent

- gradient method for dual problem: $\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \rho^k \nabla g(\boldsymbol{y}^k)$
- $\nabla g(\boldsymbol{y}^k) = \boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}$, where $\boldsymbol{x}^{k+1} = \arg\min_{\boldsymbol{x}} \ L(\boldsymbol{x}, \boldsymbol{y}^k)$
- dual ascent method is

$$\begin{aligned} \boldsymbol{x}^{k+1} &:= \arg\min_{\boldsymbol{x}} \ L(\boldsymbol{x}, \boldsymbol{y}^k) \\ \boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho^k \left(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}\right) \end{aligned}$$

Dual Ascent

- gradient method for dual problem: $\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \rho^k \nabla g(\boldsymbol{y}^k)$
- $\nabla g(\boldsymbol{y}^k) = \boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}$, where $\boldsymbol{x}^{k+1} = \arg\min_{\boldsymbol{x}} \, L(\boldsymbol{x}, \boldsymbol{y}^k)$
- dual ascent method is

$$
\begin{aligned}
\boldsymbol{x}^{k+1} &:= \arg\min_{\boldsymbol{x}} \, L(\boldsymbol{x}, \boldsymbol{y}^k) \\
\boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho^k \left(\boldsymbol{A}\boldsymbol{x}^{k+1} - \boldsymbol{b}\right)
\end{aligned}
$$

- why?

Dual Decomposition

Dual Decomposition

- suppose f is separable:

$$f(\boldsymbol{x}) = f_1(x_1) + \cdots + f_n(x_n), \boldsymbol{x} = (x_1, \ldots, x_n)$$

- then the Lagrangian is separable in $\boldsymbol{x}$:

$$L_i(x_i, \boldsymbol{y}) = f_i(x_i) + \boldsymbol{y}^\top \boldsymbol{a}_{*,i} x_i$$

- $\boldsymbol{x}$-minimization splits into n separate minimizations

$$x_i^{k+1} := \operatorname*{arg\,min}_{x_i} L_i(x_i, \boldsymbol{y}^k), \boldsymbol{i} = 1, \ldots, n$$

which can be done in parallel and $\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \alpha^k \left(\sum_{i=1}^n \boldsymbol{a}_{*,i} x_i^{k+1} - \boldsymbol{b}\right)$

Optimization Problem

$$\begin{array}{ll} \underset{\boldsymbol{x},\, \boldsymbol{z}}{\text{minimize}} & f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c} \end{array} \tag{1}$$

- variables: $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathbb{R}^m$
- parameters: $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, $\boldsymbol{B} \in \mathbb{R}^{p \times m}$, and $\boldsymbol{c} \in \mathbb{R}^p$
- optimal value: $p^\star = \inf_{\boldsymbol{x},\boldsymbol{z}} \{f(\boldsymbol{x}) + g(\boldsymbol{z}) : \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c}\}$

Augmented Lagrangian Method

Augmented Lagrangian Method

- Augmented Lagrangian:

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = \underbrace{f(\boldsymbol{x}) + g(\boldsymbol{z}) + \langle \boldsymbol{y}, \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c} \rangle}_{\text{Lagrangian}} + \frac{\rho}{2} \|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\|_{\mathrm{F}}^2$$

- ALM consists of the iterations:

$$\begin{aligned} \boldsymbol{x}^{k+1}, \boldsymbol{z}^{k+1} &:= \operatorname*{arg\,min}_{\boldsymbol{x}, \boldsymbol{z}} \ L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}^k) \text{ // primal update} \\ \boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho \left(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c} \right) \text{ // dual update} \end{aligned}$$

- $\rho > 0$ is a penalty hyperparameter

Issues with Augmented Lagrangian Method

- the primal step is often expensive to solve – as expensive as solving the original problem
- minimization of $\boldsymbol{x}$ and $\boldsymbol{z}$ has to be done jointly

Alternating Direction Method of Multipliers

Alternating Direction Method of Multipliers

- Augmented Lagrangian:

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = \underbrace{f(\boldsymbol{x}) + g(\boldsymbol{z}) + \langle \boldsymbol{y}, \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c} \rangle}_{\text{Lagrangian}} + \frac{\rho}{2} \|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\|_\mathrm{F}^2$$

- ADMM consists of the iterations:

$$\begin{aligned}
\boldsymbol{x}^{k+1} &:= \operatorname*{arg\,min}_{\boldsymbol{x}} \; L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k) \\
\boldsymbol{z}^{k+1} &:= \operatorname*{arg\,min}_{\boldsymbol{z}} \; L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k) \\
\boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho \left(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c} \right)
\end{aligned}$$

- $\rho > 0$ is a penalty hyperparameter

Convergence and Stopping Criteria

- assume (very little!)
  - f, g are convex, closed, proper
  - L_0 has a saddle point
- then ADMM converges:
  - iterates approach feasibility: $\boldsymbol{A}\boldsymbol{x}^k + \boldsymbol{B}\boldsymbol{z}^k - \mathbf{c} \to \mathbf{0}$
  - objective approaches optimal value: $f(\boldsymbol{x}^k) + g(\boldsymbol{z}^k) \to p^\star$
- false (in general) statements: $\boldsymbol{x}$ converges, $\boldsymbol{z}$ converges
- true statement: $\boldsymbol{y}$ converges
- what matters: residual is small and near optimality in objective value

Convergence of ADMM in Practice

- ADMM is often slow to converge to high accuracy
- ADMM often converges to moderate accuracy within a few dozens of iterations, which is often sufficient for most practical purposes

Practical Examples

Robust PCA (Candes et al. '08)

- We would like to model a data matrix $\boldsymbol{M}$ as low-rank plus sparse components:

$$\begin{aligned} \underset{\boldsymbol{L},\,\boldsymbol{S}}{\text{minimize}} \quad & \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 \\ \text{subject to} \quad & \boldsymbol{L} + \boldsymbol{S} = \boldsymbol{M} \end{aligned}$$

- where $\|\boldsymbol{L}\|_* := \sum_{i=1}^{n} \sigma_i(\boldsymbol{L})$ is the nuclear norm
- and $\|\boldsymbol{S}\|_1 := \sum_{i,j} |S_{ij}|$ is the entrywise ℓ_1-norm

Robust PCA via ADMM

ADMM for solving robust PCA:

$$
\begin{aligned}
\boldsymbol{L}^{k+1} &= \underset{\boldsymbol{L}}{\arg\min}\ \ \|\boldsymbol{L}\|_* + \operatorname{tr}\left(\boldsymbol{Y}^{k\top}\boldsymbol{L}\right) + \frac{\rho}{2}\left\|\boldsymbol{L} + \boldsymbol{S}^k - \boldsymbol{M}\right\|_{\mathrm{F}}^2 \\
\boldsymbol{S}^{k+1} &= \underset{\boldsymbol{S}}{\arg\min}\ \lambda\|\boldsymbol{S}\|_1 + \operatorname{tr}\left(\boldsymbol{Y}^{k\top}\boldsymbol{S}\right) + \frac{\rho}{2}\left\|\boldsymbol{L}^{k+1} + \boldsymbol{S} - \boldsymbol{M}\right\|_{\mathrm{F}}^2 \\
\boldsymbol{Y}^{k+1} &= \boldsymbol{Y}^k + \rho\left(\boldsymbol{L}^{k+1} + \boldsymbol{S}^{k+1} - \boldsymbol{M}\right)
\end{aligned}
$$

Robust PCA via ADMM

$$\boldsymbol{L}^{k+1} = \mathsf{SVT}_{\rho^{-1}}\left(\boldsymbol{M} - \boldsymbol{S}^k - \frac{1}{\rho}\boldsymbol{Y}^k\right)$$
$$\boldsymbol{S}^{k+1} = \mathsf{ST}_{\lambda\rho^{-1}}\left(\boldsymbol{M} - \boldsymbol{L}^{k+1} - \frac{1}{\rho}\boldsymbol{Y}^k\right)$$
$$\boldsymbol{Y}^{k+1} = \boldsymbol{Y}^k + \rho\left(\boldsymbol{L}^{k+1} + \boldsymbol{S}^{k+1} - \boldsymbol{M}\right),$$

where for any $\boldsymbol{X}$ with SVD $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, $\boldsymbol{\Sigma} = \mathrm{diag}\left(\{\sigma_i\}\right)$, we have

$$\mathsf{SVT}_\tau\left(\boldsymbol{X}\right) = \boldsymbol{U}\mathrm{diag}\left(\left\{(\sigma_i - \tau)^+\right\}\right)\boldsymbol{V}^\top$$

and

$$\left(\mathsf{ST}_\tau\left(\boldsymbol{X}\right)\right)_{ij} = \begin{cases} X_{ij} - \tau, & \text{if } X_{ij} > \tau, \\ 0, & \text{if } |X_{ij}| \le \tau, \\ X_{ij} + \tau, & \text{if } X_{ij} < -\tau \end{cases}$$

Graphical Lasso

Precision matrix estimation from Gaussian samples:

$$\begin{aligned} \underset{\boldsymbol{\Theta}}{\text{minimize}} \quad & \underbrace{-\log\det\boldsymbol{\Theta} + \langle\boldsymbol{\Theta}, \boldsymbol{S}\rangle}_{\text{neg. log likelihood}} + \lambda\|\boldsymbol{\Theta}\|_1 \\ \text{subject to} \quad & \boldsymbol{\Theta} \succ \mathbf{0} \end{aligned}$$

Or equivalently, using a slack variable $\boldsymbol{\Psi} = \boldsymbol{\Theta}$

$$\begin{aligned} \underset{\boldsymbol{\Theta},\boldsymbol{\Psi}}{\text{minimize}} \quad & \underbrace{-\log\det\boldsymbol{\Theta} + \langle\boldsymbol{\Theta}, \boldsymbol{S}\rangle}_{\text{neg. log likelihood}} + \lambda\|\boldsymbol{\Psi}\|_1 \\ \text{subject to} \quad & \boldsymbol{\Theta} \succ \mathbf{0}, \boldsymbol{\Theta} = \boldsymbol{\Psi} \end{aligned}$$

Graphical Lasso via ADMM

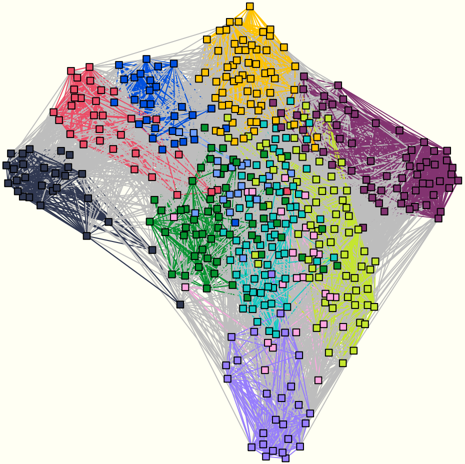
$$
\begin{aligned}
\boldsymbol{\Theta}^{k+1} &= \operatorname*{arg\,min}_{\boldsymbol{\Theta} \succ \mathbf{0}} \; -\log\det\boldsymbol{\Theta} + \langle \boldsymbol{\Theta}, \boldsymbol{S} + \boldsymbol{Y}^k \rangle + \frac{\rho}{2} \left\| \boldsymbol{\Theta} - \boldsymbol{\Psi}^k \right\|_{\mathrm{F}}^2 \\
\boldsymbol{\Psi}^{k+1} &= \operatorname*{arg\,min}_{\boldsymbol{\Psi}} \; \lambda \|\boldsymbol{\Psi}\|_1 - \langle \boldsymbol{\Psi}, \boldsymbol{Y}^k \rangle + \frac{\rho}{2} \left\| \boldsymbol{\Theta}^k - \boldsymbol{\Psi} \right\|_{\mathrm{F}}^2 \\
\boldsymbol{Y}^{k+1} &= \boldsymbol{Y}^k + \rho \left(\boldsymbol{\Theta}^{k+1} - \boldsymbol{\Psi}^{k+1} \right)
\end{aligned}
$$

Graphical Lasso via ADMM

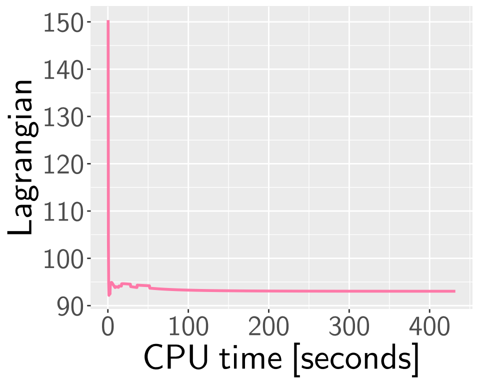
$$
\begin{aligned}
\boldsymbol{\Theta}^{k+1} &= \mathcal{F}_\rho\left(\boldsymbol{\Psi}^k - \frac{1}{\rho}\left(\boldsymbol{Y}^k + \boldsymbol{S}\right)\right) \\
\boldsymbol{\Psi}^{k+1} &= \mathsf{ST}_{\lambda\rho^{-1}}\left(\boldsymbol{\Theta}^{k+1} + \frac{1}{\rho}\boldsymbol{Y}^k\right) \\
\boldsymbol{Y}^{k+1} &= \boldsymbol{Y}^k + \rho\left(\boldsymbol{\Theta}^{k+1} - \boldsymbol{\Psi}^{k+1}\right)
\end{aligned}
$$

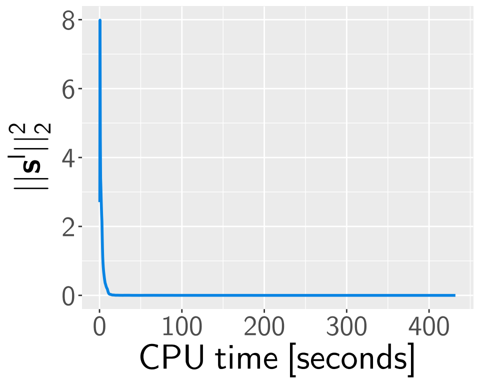
- where $\mathcal{F}_\rho(\boldsymbol{X}) := \frac{1}{2}\boldsymbol{U}\mathrm{diag}\left(\left\{\lambda_i + \sqrt{\lambda_i^2 + \frac{4}{\rho}}\right\}\right)\boldsymbol{U}^\top$, for $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$.

Network of stocks via Graphical Lasso



Network of stocks via Graphical Lasso





Conclusion

- ADMM is a versatile/flexible optimization framework
- may not be the best for a specific case, but often performs well in practice
- convergence often needs to be proved in a case-by-case scenario

Questions?