UIUC IE510 Applied Nonlinear Programming

Lecture 14: ADMM

Ruoyu Sun

Outline

Today

- Will introduce a family of algorithm called Alternating Direction Method of Multipliers (ADMM)
- Popular algorithm for large-scale machine learning, distributed computation, etc.
- Lots of applications
- Our discussion will be based on the following review paper: "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers", S. Boyd, N. Parikh, E. Chu, B. Peleato, J Eckstein, *Foundation and Trends in Machine Learning*, 2011

A Bit History of ADMM

- Considered to be first proposed in Glowinski and Marrocco'75 and Gabay and Mercier'75.
  - The original forms are very different from current ones.

- Few researchers have studied the algorithm between ~~1975-2005~~.
  - Two such research groups are Eckstein (1989 PhD Thesis, MIT; with Bertsekas); and Bingsheng He.

- After 2007 or so, some researchers in application field notice ADMM.

- Greatly popularized by the survey of Boyd et al.'11.

Why Popular Now?

- Why is BCD popular? Simple, fast, no stepsize tuning, etc.
- Why is SGD popular? Simple, fast, natural for ML problems with samples, etc.
- Why is BB popular? Best no-brainer stepsize; fast.
- Why is interior point method popular? Rich theory; state-of-art for LP/SDP

theory

- Why is ADMM popular?
  - Largely due to distributed optimization in ML
  - Almost as simple as CD/SGD
  - But nontrivial in theory, and in how to apply it

Introduction

In early 2015, 2k

Total citations Cited by 1907

Scholar articles Distributed optimization and statistical learning via the alternating direction method of multipliers
S Boyd, N Parikh, E Chu, B Peleato, J Eckstein - Foundations and Trends® in Machine Learning, 2011
Cited by 1907 - Related articles - All 31 versions

Total citations Cited by 1027

Scholar articles On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators
J Eckstein, DP Bertsekas - Mathematical Programming, 1992
Cited by 1027 - Related articles - All 14 versions

Introduction

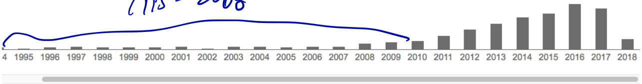
early 2018, 7k

Total citations Cited by 6901

Scholar articles Distributed optimization and statistical learning via the alternating direction method of multipliers
S Boyd, N Parikh, E Chu, B Peleato, J Eckstein - Foundations and Trends® in Machine learning, 2011
Cited by 6897 Related articles All 47 versions

Total citations Cited by 1813

1995 - 2008



Scholar articles On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators
J Eckstein, DP Bertsekas - Mathematical Programming, 1992
Cited by 1813 Related articles All 20 versions

The Problem to be Solved

The ADMM algorithm solves the following problem

$$\begin{aligned} \min \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c} \\ & \mathbf{x} \in X, \quad \mathbf{z} \in Z \end{aligned}$$

- $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^m$ are the variables
- f, g are two convex function, possible nonsmooth
- $\mathbf{A}$, $\mathbf{B}$ are two known matrices, $\mathbf{c}$ is a known vector

Why this Special Form?

Special form:

- Two blocks of variables
- separable in the objective
- coupled by a linear equation

Why consider this special form?

- The only case with clean proof
- Cover lots of applications

Widely accepted special form, s.t. even some optimizers don't realize it's special...

Recall: ALM

- Augmented Lagrangian

$$L_\rho(\mathbf{x}, \mathbf{z}; \lambda) = f(\mathbf{x}) + g(\mathbf{z}) + \langle \lambda, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}\|^2$$

- Method of multipliers or ALM:

$$\left(\mathbf{x}^{r+1}, \mathbf{z}^{r+1}\right) = \arg\min_{\mathbf{x},\mathbf{z}} L_\rho(\mathbf{x}, \mathbf{z}; \lambda^r)$$

$$\lambda^{r+1} = \lambda^r + \rho\left(\mathbf{A}\mathbf{x}^{r+1} + \mathbf{B}\mathbf{z}^{r+1} - \mathbf{c}\right)$$

Recall: ALM

- Augmented Lagrangian

$$L_\rho(\mathbf{x}, \mathbf{z}; \lambda) = f(\mathbf{x}) + g(\mathbf{z}) + \langle \lambda, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \frac{\rho}{2} \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}\|^2$$

- Method of multipliers or ALM:

$$\left(\mathbf{x}^{r+1}, \mathbf{z}^{r+1}\right) = \arg\min_{\mathbf{x},\mathbf{z}} L_\rho(\mathbf{x}, \mathbf{z}; \lambda^r)$$

$$\lambda^{r+1} = \lambda^r + \rho\left(\mathbf{A}\mathbf{x}^{r+1} + \mathbf{B}\mathbf{z}^{r+1} - \mathbf{c}\right)$$

The ADMM Algorithm

The steps of the ADMM Algorithm is given below

$$\mathbf{x}^{r+1} = \arg\min_{\mathbf{x}\in X} L_\rho(\mathbf{x}, \mathbf{z}^r; \lambda^r)$$
$$\mathbf{z}^{r+1} = \arg\min_{\mathbf{z}\in Z} L_\rho(\mathbf{x}^{r+1}, \mathbf{z}; \lambda^r)$$
$$\lambda^{r+1} = \lambda^r + \rho\left(\mathbf{A}\mathbf{x}^{r+1} + \mathbf{B}\mathbf{z}^{r+1} - \mathbf{c}\right)$$

- Divide and conquer: Optimize $\mathbf{x}$ and $\mathbf{z}$ once (coordinate descent on L_ρ), then update the dual variable
- The primal problem is no longer solved exactly (where the efficiency comes from)

General Form (careful...)

- Why assume objective $f(x) + g(z)$, not general form $f(x,z)$?
- Why assume two blocks x, z, not general n blocks?
- Why assume convexity?

Well, you can solve a general problem

$$\min_{x_1,\ldots,x_n} \quad f(x_1,\ldots,x_n), s.t. Ax = b, x_i \in X_i.$$

Multi-block ADMM:

For $r = 1, 2, \ldots,$

For $k = 1, \ldots, n$ (or some other orders),

$$x_k \leftarrow \arg \min_{x_k \in X_k} \quad L_\rho(x;\lambda)$$

$$\lambda \leftarrow \lambda + \rho\,(Ax - b)$$

Caution: Convergence Largely Unclear!

Interpretation of ADMM

- ALM with 5 iteration of BFGS
- ALM with 2 iteration of Newton method.

Common explanation: ADMM = BCD + ALM. (*)

– Idea: "solving ALM inexactly by one cycle of BCD"

– Multiple rounds of BCD give an inexact stationary point of L, and inexact ALM works. Hopefully one round of BCD also works

Issue 1: No classical proof is based on the idea

- The theory for multi-block and non-separable is still largely open and... strange

Issue 2: Historically, ADMM is not derived from the idea "solving ALM inexactly by BCD"

- As a comparison, interior point method is "solving barrier-subproblem inexactly by Newton method"
- GAN is "solving inner problem inexactly by a few SGD steps"

So...(*) helps you remember, but not the full story

Outline

Decomposition Method

For parallel / distributed computation

$\min_{x,y} f(x,y)$.

$(x,y) \leftarrow \arg\min_{(x,y)} f(x,y)$.

- Decomposable problem

$$\min_{x,y} f(x) + g(y).$$

$\min f(x) + g(y)$

$(x,y) \leftarrow \arg\min_{(x,y)} f(x)+g(y)$

$\Leftrightarrow \begin{cases} x \leftarrow \arg\min f(x) \\ y \leftarrow \arg\min g(y) \end{cases}$

CD: $\begin{cases} x^{+} \leftarrow \arg\min f(x,y) \\ y \leftarrow \arg\min f(x^{+},y) \end{cases}$ sequential.

Two subproblems can be solved independently (in parallel).

- Non-decomposable problem

$\|x-z\|^2 + \|y-z\|^2$.

$$(P1) \min_{x,y,z} f(x,z) + g(y,z). \quad (1)$$

$x_1 + x_2 + x_3 + x_4$.

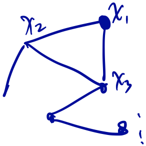
- Primal decomposition (BCD over (x,y) and z):

(x,y) update: $x \leftarrow \operatorname{argmin}_x f(x,z), \quad y \leftarrow \operatorname{argmin}_y g(y,z),$

z-update $z \leftarrow \operatorname{argmin}_z f(x,z) + g(y,z)$ or gradient update

Example: Graphical Model

SGD:



Example: MAP inference in graphical models.

$$\min_x \underbrace{\sum_i \theta(x_i)}_{\text{node}} + \underbrace{\sum_{(i,j) in E} \theta(x_i, x_j)}_{\text{edge}}.$$

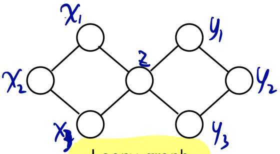
where E is the edge set, and θ is a certain function.

SGD: $\min_x x^T A x$

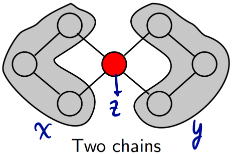
$A = \begin{bmatrix} \alpha_1 & & 0 \\ 0 & & \alpha_n \end{bmatrix}$

$\min_x \sum_i \alpha_i x_i^2$.

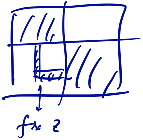
For instance, the following graph can be decomposed.



Loopy graph



Two chains



$$\underbrace{\theta(x_i, x_j) + \theta(x_i, z)}_{f(x,z)} + \underbrace{\theta(y_i, z) + \theta(y_i, y_j)}_{g(y,z)}$$

Review: Dual Ascent

- Primal problem (with equality constraints), assuming f, X convex

$$(P): \min \quad f(\mathbf{x}), \quad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \; \mathbf{x} \in X \tag{2}$$

- Consider the Lagrangian, λ is the dual variable (multiplier)

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \langle \lambda, \mathbf{A}\mathbf{x} - \mathbf{b} \rangle$$

- Dual function $d(\lambda) = \min_{\mathbf{x} \in X} L(\mathbf{x}, \lambda) \leq \min_{\mathbf{A}\mathbf{x}=\mathbf{b}, \mathbf{x} \in X} f(\mathbf{x})$
- Gradient ascent for the dual problem:

$$\lambda^{r+1} = \lambda^r + \alpha \nabla d(\lambda^r)$$

Let $\widetilde{\mathbf{x}} = \arg\min_{\mathbf{x} \in X} L(\mathbf{x}, \lambda^r)$, we know (can be verified)

$$\nabla d(\lambda^r) = \mathbf{A}\widetilde{\mathbf{x}} - \mathbf{b}$$

- **Dual Ascent Algorithm**

$$\mathbf{x}^{r+1} = \arg\min_{\mathbf{x} \in X} L(\mathbf{x}, \lambda^r)$$

$$\lambda^{r+1} = \lambda^r + \alpha \left(\mathbf{A}\mathbf{x}^{r+1} - \mathbf{b} \right)$$

Dual Decomposition: Special Form of DA

- Advantage of DA: sometimes lead to decentralized optimization
- If f is separable among variables

Linear programy.

$$(P): \min_{\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i = \mathbf{b}} \quad f(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) \tag{3}$$

(linear) coupled constraint

separable objective.

Special case: consensus least squares (8)

- Consider the Lagrangian

Q: min L over x, what happens?

$$L(\mathbf{x}, \lambda) = \sum_{i=1}^{N} L_i(\mathbf{x}_i, \lambda) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) + \langle \lambda, \mathbf{A}_i \mathbf{x}_i - \mathbf{b} \rangle \tag{4}$$

- Dual ascent algorithm (a.k.a. dual decomposition)

L is decomposable over x.

$$\mathbf{x}_i^{r+1} = \arg \min_{\mathbf{x}_i \in X_i} L_i(\mathbf{x}_i, \lambda^r), \ \forall\, i$$

$$\lambda^{r+1} = \lambda^r + \alpha \left(\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i^{r+1} - \mathbf{b} \right)$$

- Large-scale problem: Split variables, update $\mathbf{x}_i$ in parallel

Apply Dual Decomposition to (1)

Task. transform to constrained problem, then do dual decomposition.

Revisit the problem:

$$\min_{x,y,z} f(x,z) + g(y,z).$$

Copy-variable trick: reformulate as

$$\min_{x,y,z_1,z_2} f(x,z_1) + g(y,z_2), \quad \text{subject to } z_1 = z_2.$$

Lagrangian function (let $\mathrm{w} = (x, y, z_1, z_2)$)

$$L(\mathrm{w}, \lambda) = f(x, z_1) + g(y, z_2) + \lambda^T (z_1 - z_2).$$

Dual decomposition:

Apply Dual Decomposition to (1)

Revisit the problem:

$$\min_{x,y,z} f(x,z) + g(y,z).$$

Copy-variable trick: reformulate as

separable

linear constraint.

$$\min_{x,y,z_1,z_2} f(x,z_1) + g(y,z_2), \quad \text{subject to } z_1 = z_2.$$

Lagrangian function (let $\mathbf{w} = (x,y,z_1,z_2)$)

$$= \underbrace{f(x,z_1) + \lambda^T z_1}_{F(x,z_1)} + \underbrace{g(y,z_2) - \lambda^T z_2}_{G(y,z_2)}.$$

$$L(\mathbf{w},\lambda) = f(x,z_1) + g(y,z_2) + \lambda^T(z_1 - z_2).$$

Dual decomposition: (show how to "decompose").

$$w \leftarrow \arg\min_{w} L(w,\lambda)$$

or $(x,y,z_1,z_2) \leftarrow \arg\min_{x,y,z_1,z_2} L(x,y,z_1,z_2;\lambda)$.

$$\Longleftrightarrow \begin{cases} (x,z_1) \leftarrow \arg\min_{(x,z_1)} F(x,z_1) \\ (y,z_2) \leftarrow \arg\min_{(y,z_2)} G(y,z_2) \\ \lambda \leftarrow \lambda + \alpha(z_1 - z_2). \end{cases}$$

Convergence of Dual Decomposition

- Assumption for convergence
  - Dual function $d(\lambda)$ has Lipschitz continuous gradient (with constant L)
  - The stepsize $\alpha \leq \frac{1}{L}$
- (1) requires that problem (P) is strongly convex with constant at least $\frac{1}{L}$
- **One issue of DD**: not applicable to non-strongly convex objective
  - Remark: Can still use subgradient method, but not as good as dual gradient method.

Outline

Introduction to ADMM

Dual Ascent, Dual Decomposition and ADMM

ADMM: Improve Dual Decomposition

Applications

Motivation: Consensus Problem

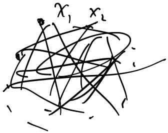
Consider the simple least square problem

$$(P2): \quad \min_{x \in \mathbb{R}^d} \quad \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \tag{5}$$

- There are m computing nodes; none of them can hold all data
- Suppose $\mathbf{A}$ is divided into m groups of rows $\mathbf{A}_1, \cdots, \mathbf{A}_N$ some samples
  - One row of $\mathbf{A}$ represents one data sample
- The problem can be re-written as

$$\min \quad \sum_{i=1}^{m} \|\mathbf{A}_i\mathbf{x} - \mathbf{b}_i\|^2 \quad = \sum_i f_i(x). \tag{6}$$

MapReduce



Cannot do primal decomposition if x_i's are connected.

Motivation: Consensus Problem

- Distributed computation?
- Copy-variable trick again: N variables $\mathbf{z}_1, \cdots, \mathbf{z}_n$
- Consider the following reformulation

$$\begin{aligned} \min_{x, z_1, \cdots, z_m} \quad & \sum_{i=1}^{m} \|\mathbf{A}_i \mathbf{z}_i - \mathbf{b}_i\|^2 \\ \text{subject to} \quad & \mathbf{z}_i = \mathbf{x}, \ \forall\, i \end{aligned} \tag{7}$$

- Each $\mathbf{z}_i$ can be handled by the i-th local agent (together with $\mathbf{A}_i$)!

How about $z_1 = z_2,\ z_2 = z_3,\ \cdots\ z_{n-1} = z_n$. ?

Depends on architecture of the network.

Apply Dual Decomposition to Consensus Problem?

Apply Dual Decomposition to

$$\begin{aligned} \min \quad & \sum_{i=1}^{N} \|\mathbf{A}_i \mathbf{z}_i - \mathbf{b}_i\|^2 \\ \text{subject to} \quad & \mathbf{z}_i = \mathbf{x}, \ \forall\, i \end{aligned} \tag{8}$$

The Lagrangian is $= \sum_i \underbrace{f_i(z_i) + \langle \lambda, z_i \rangle}_{F_i(z_i)} - \underbrace{m \sum_i \langle \lambda_i, x \rangle}_{G(x)}$.

$$L(\mathbf{x}, \mathbf{z}, \lambda) = \sum_{i=1}^{N} \left[f_i(\mathbf{z}_i) + \langle \lambda_i, \mathbf{z}_i - \mathbf{x} \rangle \right]. \tag{9}$$

What is the issue of Dual Decomposition?

parallel update
$z_i \leftarrow \arg\min_{z_i} F_i(z_i), \ i = 1, \ldots, m.$
$x \leftarrow \arg\min_{x} G(x)$ → linear function of x.
$\lambda_i \leftarrow \lambda_i + \alpha(z_i - x), \ \forall i.$

Revisit Simple Case: Another Copy-Variable Trick

Revisit the problem (P1):

$$\min_{x,y,z} f(x,z) + g(y,z).$$

Copy-variable trick: reformulate as

$$(P1b) \min_{x,y,z_1,z_2,z} f(x,z_1) + g(y,z_2), \quad \text{subject to } z_1 = z, z_2 = z. \tag{10}$$

Lagrangian function (let $\mathbf{w} = (x,y,z_1,z_2)$)

$$L(\mathbf{w},\lambda) = f(x,z_1) + g(y,z_2) + \lambda_1^T(z_1 - z) + \lambda_2^T(z_2 - z).$$

What is the issue of Dual Decomposition here?

Method of Multipliers (ALM)

- ALM can help resolve the non-strong convexity issue.
- Primal problem (with equality constraints)

$$\min_{\sum_{i=1}^{N}\mathbf{A}_i\mathbf{x}_i=\mathbf{b}} f(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) \tag{11}$$

- Consider the Augmented Lagrangian, λ is the dual variable

$$L(\mathbf{x},\lambda) = f(\mathbf{x}) + \langle \lambda, \mathbf{A}\mathbf{x}-\mathbf{b}\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$$
$$=$$

- **Method of Multipliers** (a.k.a. ALM)

$$\mathbf{x}^{r+1} = \arg\min_{\mathbf{x}\in X} L_{\rho}(\mathbf{x},\lambda^r)$$
$$\lambda^{r+1} = \lambda^r + \rho\left(\mathbf{A}\mathbf{x}^{r+1}-\mathbf{b}\right)$$

END OF PART I (Tuesday lecture)

Revisit Simple Case: Another Copy-Variable Trick

Revisit the problem (P1b) in (10)

$$\min_{x,y,z_1,z_2,z} f(x,z_1) + g(y,z_2), \quad \text{subject to } z_1 = z, z_2 = z.$$

Augmented Lagrangian function (let $\mathbf{w} = (x, y, z_1, z_2)$)

$$L(\mathbf{w}, \lambda) = f(x,z_1) + g(y,z_2) + \lambda_1^T(z_1 - z) + \lambda_2^T(z_2 - z) + \|z_1 - z\|^2 + \|z_2 - z\|^2.$$

What is the issue of ALM here?

ALM: Pros and Cons for Consensus Problem

- **Good**

  Weak Assumptions: f only needs to be convex

  Dual function has Lipschitz continuous gradient

- **Bad**

  Objective not separable anymore; no dual decomposition

How ADMM Resolves the Issue

- ADMM: apply the BCD trick for primal decomposition.
- Revisit the problem (P1b) in (10)

$$\min_{x,y,z_1,z_2,z} f(x,z_1)+g(y,z_2), \quad \text{subject to } z_1 = z, z_2 = z.$$

- Augmented Lagrangian function (let $\mathbf{w} = (x, y, z_1, z_2)$)

$$L(\mathbf{w},\lambda) = f(x,z_1)+g(y,z_2)+\lambda_1^T(z_1-z)+\lambda_2^T(z_2-z)+\|z_1-z\|^2+\|z_2-z\|^2.$$

- Split into two blocks $(x, y, z_1, z_2), z$.

Graph Summary: Motivation of ADMM from Dual Decomposition

Parallel SGD v.s. ADMM

The least square problem can be solved by GD, or parallel SGD.

$$\min_{x} \quad \sum_{i=1}^{N} \|\mathbf{A}_i \mathbf{x} - b_i\|^2$$

Each machine holds data A_i, and update x simultaneously or asynchronously

Parallel SGD v.s. ADMM

Advantage of ADMM v.s. GD (or parallel SGD):

- Not necessarily to have the "star network"

- Can handle nonsmooth $R(x)$, e.g. $\|x\|_1$

$$\min_x \quad \sum_i f_i(\mathbf{x}) + R(\mathbf{x}), \text{ subject to } x \in X \tag{12}$$

We can do the same procedure as before

$$\min \quad \sum_i f_i(\mathbf{z}_i) + R(\mathbf{x})$$
$$\text{subject to } \mathbf{x} \in X, \ \mathbf{z}_i = \mathbf{x}, \ \forall\, i$$

Outline

Application 1: Constrained ℓ_1 problem

$$\boxed{\begin{aligned} \min \quad & \|\mathbf{x}\|_1 \\ \text{s.t.} \quad & \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \leq \delta \end{aligned}}$$

Similar as the LASSO, but with explicit constrains on the noise magnitude

Formulate into the ADMM form?

Introduce a new variable $\mathbf{z}$ (**split**!), arrive at an equivalent formulation

$$\begin{aligned} \min \quad & \|\mathbf{x}\|_1 \\ \text{s.t.} \quad & \|\mathbf{z}\|^2 \leq \delta \\ & \mathbf{z} = \mathbf{A}\mathbf{x} - \mathbf{b} \end{aligned}$$

Mapping back to the original problem, $f(\mathbf{x}) = \|\mathbf{x}\|_1$, $g(\mathbf{z}) = 0$
$\mathbf{B} = -\mathbf{I}$, $Z := \{\mathbf{z} \mid \|\mathbf{z}\|^2 \leq \delta\}$, $X = \mathbb{R}^n$

Application 2: Multiple Regularizations

Consider the following sparse group LASSO problem

The problem is formulated as

$$\min \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 + \underbrace{\sum_{g=1}^{G}\lambda_g\|\mathbf{x}_g\|_2}_{\ell_1/\ell_2 \mathrm{norm}} + \underbrace{\lambda\|\mathbf{x}\|_1}_{\ell_1 \mathrm{norm}}$$

Select a few groups, and select a few variables within groups

How to deal with two different nonsmooth penalizations simultaneously?

Application 2: Multiple Regularizations

Same trick, **split and alternate**

Introduce two new variables $\mathbf{z}_1, \mathbf{z}_2$

Perform the following reformulation

$$\min \quad \underbrace{\sum_{g=1}^{G} \lambda_g \|\mathbf{x}_g\|_2}_{f(\mathbf{x})} + \underbrace{\frac{1}{2}\|\mathbf{A}\mathbf{z}_2 - \mathbf{b}\|^2 + \lambda \|\mathbf{z}_1\|_1}_{\mathrm{g}(\mathbf{z})}$$

$$\text{subject to} \quad \mathbf{z}_1 = \mathbf{x}, \quad \mathbf{z}_2 = \mathbf{x}$$

Reformulated in an ADMM form; deal with two norms separately?

Application 3: Intersection of Two Sets

Suppose we would like to solve the following problem

$$
\begin{aligned}
& FIND\ \mathbf{x} \\
\text{subject to} \quad & \|\mathbf{x}\|_1 \le \delta, \quad \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \le \epsilon
\end{aligned}
\tag{13}
$$

Finding the intersection of two convex sets

We can introduce two variables $\mathbf{z}_1, \mathbf{z}_2$

$$
\begin{aligned}
& FIND\ \mathbf{x} \\
\text{subject to} \quad & \|\mathbf{z}_1\|_1 \le \delta, \quad \|\mathbf{z}_2\|^2 \le \epsilon \\
& \mathbf{z}_1 = \mathbf{x}, \quad \mathbf{z}_2 = \mathbf{A}\mathbf{x} - \mathbf{b}
\end{aligned}
$$

First block $\mathbf{x}$, second block $(\mathbf{z}_1, \mathbf{z}_2)$; Fix $\mathbf{x}$, $\{\mathbf{z}_i\}$'s are independent!

Dealing with each constraint separately

Application 5: Sharing Problem

Consider the following sharing problem

$$\min \quad \sum_{i=1}^{N} f_i(\mathbf{x}_i) + g\left(\sum_{i=1}^{N} \mathbf{x}_i\right) \tag{15}$$

f_i local cost function; g shared objective

Important class of problem, applications for example in economics

Agents have selfish interests, but their behaviors are coupled together by some sort of “mutual interactions”

Application 5: Sharing Problem

First Approach Introducing a new variable $\mathbf{z}$

$$\min \quad \sum_{i=1}^{N} f_i(\mathbf{x}_i) + g(\mathbf{z})$$

$$\mathbf{z} = \sum_{i=1}^{N} x_i$$

Second Approach Introducing a set of new variable $\{\mathbf{z}_i\}$

$$\min \quad \sum_{i=1}^{N} f_i(\mathbf{x}_i) + g\left(\sum_{i=1}^{N} \mathbf{z}_i\right)$$

$$\mathbf{z}_i = \mathbf{x}_i$$

Outline