Optimization: SC42100 NDCS

Jiaxuan Zhang

January 13, 2022

Forword

Jiaxuan Zhang

January 13, 2022

Contents

Chapter 1

Chapter 1: Convex Optimization

Content

1. Convex Optimization and Some Definitions
2. First-Order Methods: Basic, Optimal and Subgradient Version
3. Duality Theory: Lagrangian, Dual Problem, Lower bound and Strong Duality

1.1 Basic Definition

Basic Definition about convex sets and convex functions is ignored here.

Proposition 1.1.1 (Affine Lower Bounds from Convexity).

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \tag{1.1}$$

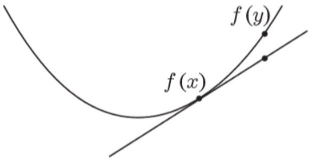


Figure 1.1: Affine Lower Bounds from Convexity

Definition 1.1.2 (Strong Convexity). *There exists a* ***quadratic lower bound*** *such that*

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{c}{2}\|y - x\|^2 \tag{1.2}$$

Definition 1.1.3 (Lipschitz-Continuous Gradient).

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \tag{1.3}$$

Notes: It is not equal to Lipschitz-Continuous-Function

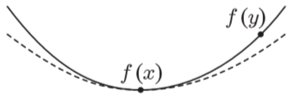


Figure 1.2: Strong Convexity

Proposition 1.1.4 (Quadratic Upper Bound). *Lipschitz-Continuous Gradient yields a **quadratic upper bound***

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2$$

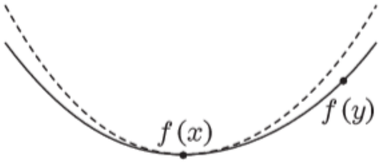


Figure 1.3: Lipschitz-Continuous Gradient

Theorem 1.1.5. *Strongly Convex Functions with Lipschitz Graident are bounded from **above and below by quadratic functions**.*

Definition 1.1.6 (Condition Number). ***condition number*** *$\kappa = L/C$ impatcs performance of first-order methods.*

Definition 1.1.7 (subgradient). *g is a subgradient for f at θ_0 if $g \in \partial f(\theta_0)$.*

$$\partial f(\theta_0) = \left\{ g \in \mathbb{R}^M \mid f(\theta) \geq f(\theta_0) + g^\top (\theta - \theta_0), \forall \theta \in \mathbb{R}^M \right\}$$

Definition 1.1.8 (ϵ-subgradient). *g is an ϵ -subgradient for f at θ_0 if $g \in \partial_\epsilon f(\theta_0)$.*

$$\partial_\epsilon f(\theta_0) = \left\{ g \in \mathbb{R}^M \mid f(\theta) \geq f(\theta_0) + g^\top (\theta - \theta_0) - \epsilon, \forall \theta \in \mathbb{R}^M \right\}$$

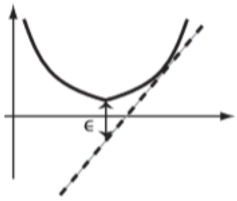


Figure 1.4: epsilon-subgradient

1.2 Convergence Rate Definition

1.3 Convex Optimization

1.3.1 General Problem Formulation

Definition 1.3.1 (General Problem Formulation of Convex Optimization Problems).

$$\begin{array}{ll} \underset{\theta}{\text{minimize}} & f(\theta) \\ \textit{subject to} & \theta \in \Theta \end{array} \tag{1.4}$$

- *$f : \mathbb{R}^M \to \mathbb{R}$ is a convex function, in general non-differentiable.*
- *$\Theta \subset \mathbb{R}^M$ is closed and convex*

1.4 First-Order Methods

1.4.1 Basic Gradient Method

Definition 1.4.1 (Basic Gradient Method).

$$\theta_{k+1} = \mathcal{P}_\Theta \left[\theta_k - \alpha_k \nabla f\left(\theta_k\right)\right] \tag{1.5}$$

Issues: evaluating the gradient, computing the projection $\mathcal{P}_\Theta$ (may get out from feasible set), setting the step-length

In offline-techniques, step-length is often decided by **line-search**.

Proposition 1.4.2. *If f is strictly convex, this converges for sufficiently small constant α.*

Method 1.4.3 (Step Choice). *For diminishing stepsizes $\sum_{t=0}^{\infty} \alpha^2(t) < \infty, \sum_{t=0}^{\infty} \alpha(t) = \infty$*

$$\lim_{T\to\infty} f(x(T)) = f^\star$$

1.4.2 Optimal First-Order Method

There are many kinds of optimal first-order method, we shown one case here

Definition 1.4.4.

$$\begin{array}{l} x(t+1) = y(t) - L^{-1}\nabla f(y(t)) \\ y(t+1) = x(t+1) + \frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}(x(t+1) - x(t)) \end{array} \tag{1.6}$$

1.4.3 Non-Smooth Version of Gradient Descent (Subgradient Methods)

If a convex optimization problem includes a non-differentiable $f(\cdot)$

$$\begin{array}{ll} \underset{\theta}{\text{minimize}} & f(\theta) \\ \text{subject to} & \theta \in \Theta \end{array} \tag{1.7}$$

then we can solve it with

Definition 1.4.5 (projected subgradient method).

$$\theta_{k+1} = \mathcal{P}_\Theta \left[\theta_k - \alpha_k g_k\right]$$

where g_k is a subgradient of f at θ_k.

The method converges if g is bounded and $\sum_k^\infty \alpha_k^2 < \infty, \sum_k^\infty \alpha_k = \infty$

Proposition 1.4.6 (Convergence). *The method converges if g is bounded and $\sum_k^\infty \alpha_k^2 < \infty, \sum_k^\infty \alpha_k = \infty$*

1.5 Brief Duality Theory

1.5.1 Lagrangian and Lagrange dual function

Definition 1.5.1 (primal problem). *standard form optimization problem minimize* $f_0(x)$

$$\textit{subject to } f_i(x) \leq 0, i = 1, \ldots, m$$

optimal value $p^\star$
domain D

Definition 1.5.2 (Lagrangian). *Lagrangian $L : \mathbf{R}^{n+m} \to \mathbf{R}$*

$$L(x, \lambda) = f_0(x) + \lambda_1 f_1(x) + \cdots + \lambda_m f_m(x)$$

- *λ_i called **Lagrange multipliers** or **dual variables***

Remark objective is augmented with weighted sum of constraint functions

Definition 1.5.3 ((Lagrange) Dual Function). *(Lagrange) dual function $g : \mathbf{R}^m \to \mathbf{R} \cup \{-\infty\}$*

$$\begin{aligned} g(\lambda) &= \inf_x L(x, \lambda) \\ &= \inf_x \left(f_0(x) + \lambda_1 f_1(x) + \cdots + \lambda_m f_m(x)\right) \end{aligned}$$

- *minimum of augmented cost as function of weights*
- *can be $-\infty$ for some λ*

Proposition 1.5.4. *g is concave (even if f_i not convex!)*

1.5.2 Lower bound proposition

Theorem 1.5.5 (Lower bound proposition). *if $\lambda \succeq 0$ and x is primal feasible, then*

$$g(\lambda) \leq f_0(x)$$

Proof. if $f_i(x) \leq 0$ and $\lambda_i \geq 0$,

$$\begin{aligned} f_0(x) &\geq f_0(x) + \sum_i \lambda_i f_i(x) \\ &\geq \inf_z \left(f_0(z) + \sum_i \lambda_i f_i(z)\right) \\ &= g(\lambda) \end{aligned}$$

$f_0(x) - g(\lambda)$ is called the *duality gap* of (primal feasible) x and $\lambda \succeq 0$ □

Definition 1.5.6 (dual feasible). - *$\lambda \in \mathbf{R}^m$ is **dual feasible** if $\lambda \succeq 0$ and $g(\lambda) > -\infty$*

- *dual feasible points yield **lower bounds** on optimal value!*

1.5.3 Lagrange dual problem

Definition 1.5.7 (Lagrange dual problem). *let's find best lower bound on $p^\star$:*

$$\begin{aligned} &\textit{maximize} \quad g(\lambda) \\ &\textit{subject to} \quad \lambda \succeq 0 \end{aligned} \tag{1.8}$$

Remark: The Lagrange dual problem is always a **convex problem**, even if primal isn't.

1.5.4 Strong Duality

Definition 1.5.8 (Strong Duality).

$$d^\star = p^\star$$

Proposition 1.5.9. - *for convex problems, we (usually) have strong duality*

- *strong duality does not hold, in general, for non-convex problems*
- *when strong duality holds, dual optimal λ^* serves as certificate of optimality for primal optimal point x^**

Theorem 1.5.10 (Slater's Condition). *if primal problem is strictly feasible (and convex), i.e., there exists $x \in$ relint D with*

$$f_i(x) < 0, i = 1, \ldots, m$$

then we have $p^\star = d^\star$

1.6 Summary

Chapter 2

Chapter 2: Decomposition

2.1 Content

1. Primal Decomposition
2. Dual Decomposition
3. Penalty Function Method

2.2 Decomposition

There are mainly three ways for decomposition: **primal decomposition**, **decomposition** and **Penalty-Function-Based Models**

In Primal Decomposition, master problem allocates amount of resources.

In Dual Decomposition, master problem sets the pricing strategy.

In Penalty-Function-Based Model, coupled constraints are moved to the augmented objective function.

2.2.1 Cases for Decomposition

Definition 2.2.1 (A Trivial Case). *Separable Objectives and Constraints*

$$\begin{array}{rl} \text{minimize} & \sum_i f_i\left(x_i\right) \\ \textit{subject to} & x_i \in X_i \end{array} \tag{2.1}$$

Definition 2.2.2 (Coupling Constraints).

$$\begin{array}{rl} \underset{y,\{\theta^i\}}{\text{minimize}} & \sum_i f^i\left(\theta^i\right) \\ \textit{subject to} & \theta^i \in \Theta^i, \forall i \\ & A^i\theta^i \leq y, \forall i \\ & y \in \mathcal{Y} \end{array} \tag{2.2}$$

Definition 2.2.3 (Complicating Objectives).

$$\begin{array}{rl} \underset{\{\theta'\}}{\text{minimize}} & \sum_i f^i\left(\theta^i\right) \\ \textit{subject to} & \theta^i \in \Theta^i, \forall i \\ & \sum_i h^i\left(\theta^i\right) \leq c \end{array} \tag{2.3}$$

2.2.2 Trivial Case Decomposition

Can be directly decomposed to:

$$\begin{array}{rl} \text{minimize} & f_i\left(x_i\right) \\ \text{subject to} & x_i \in X_i \end{array} \tag{2.4}$$

2.2.3 Primal Decomposition

Can be decomposed in two levels:

Theorem 2.2.4 (lower level)**.** *with **fixed y**:*

$$\begin{gathered} \underset{\theta^i}{\operatorname{minimize}} f^i\left(\theta^i\right) \\ \textit{subject to } \theta^i \in \Theta^i, \\ A^i \theta^i \leq y \end{gathered} \tag{2.5}$$

Theorem 2.2.5 (higher level)**.** *updating y:*

$$\begin{gathered} \underset{y}{\operatorname{minimize}} \textstyle\sum_i f^{i*}(y) \\ \textit{subject to } y \in \mathcal{Y} \end{gathered} \tag{2.6}$$

2.2.4 Dual Decomposition

Can be decomposed in two levels:

Theorem 2.2.6 (lower level)**.** *We have the subproblems, one for each i over θ^i*

$$\underset{\theta^i}{\operatorname{minimize}} f^i\left(\theta^i\right)+\lambda^{\top} h^i\left(\theta^i\right)$$

subject to $\theta^i \in \Theta^i$

Theorem 2.2.7 (higher level)**.** *We have the master dual problem, updating the dual variable λ :*

$$\begin{gathered} \underset{\lambda}{\operatorname{maximize}} \quad d(\lambda)=\sum_i d^i(\lambda)-\lambda^{\top} c \\ \textit{subject to } \lambda \geq 0 \end{gathered} \tag{2.7}$$

where $d(\lambda)$ is the dual function obtained as the maximum value of the Lagrangian for a given λ.

Remark: The approach will only give appropriate results if **strong duality holds**

2.2.5 Penalty Function Based Model

Theorem 2.2.8 (lower level)**.** *with **fixed** λ*

$$\begin{array}{rl} \underset{\theta^i}{\operatorname{minimize}} & f^i\left(\theta^i\right)+\lambda^{\top} h^i\left(\theta^i\right) \\ \textit{subject to} & \theta^i \in \Theta^i \end{array} \tag{2.8}$$

Theorem 2.2.9 (higher level). *updating dual variable* λ

$$\begin{array}{ll} \underset{\lambda}{\operatorname{maximize}} & d(\lambda)=\sum_i d^i(\lambda)-\lambda^\top c \\ \textit{subject to} & \lambda \geq 0 \end{array} \tag{2.9}$$

2.3 Examples

2.3.1 Coupled Constraints

Example 2.3.1 (Example of Coupled Constraints)**.**

$$\begin{array}{ll} \underset{\theta}{\operatorname{minimize}} & \phi^i\left(\theta^i\right)+\phi^j\left(\theta^j\right) \\ \textit{subject to} & \theta^i=\theta^j \end{array} \tag{2.10}$$

Solution:
use **dual decomposition** The Lagrangian of the problem is

$$\begin{aligned} L\left(\theta, \mu^{(i,j)}\right) &= \phi^i\left(\theta^i\right)+\phi^j\left(\theta^j\right)+\mu^{(i,j)}\left(\theta^i-\theta^j\right) \\ &= \phi^i\left(\theta^i\right)+\mu^{(i,j)}\theta^i+\phi^j\left(\theta^j\right)-\mu^{(i,j)}\theta^j \end{aligned} \tag{2.11}$$

which leads to a separable dual function

$$\begin{aligned} d\left(\mu^{(i,j)}\right) &= \inf_{\theta^i}\left[\phi^i\left(\theta^i\right)+\mu^{(i,j)}\theta^i\right]+\inf_{\theta^j}\left[\phi^j\left(\theta^j\right)-\mu^{(i,j)}\theta^j\right] \\ &:= d^i\left(\mu^{(i,j)}\right)+d^j\left(\mu^{(i,j)}\right) \end{aligned} \tag{2.12}$$

computation steps:

1. Node i fixes $\mu^{(i,j)}$, announces to node j
2. Node j returns subgradient $\theta^{j*}\left(\mu^{(i,j)}\right)$ of d^j at $\mu^{(i,j)}$
3. Node i updates Lagrange multiplier according to (**this update method based on the objective function at the higher level**)

$$\mu_{k+1}^{(i,j)}=\mu_k^{(i,j)}+\alpha\left[\theta^{i*}\left(\mu_k^{(i,j)}\right)-\theta^{j*}\left(\mu_k^{(i,j)}\right)\right] \tag{2.13}$$

2.3.2 Example of Complicating Constraints

Example 2.3.2 (Example of Complicating Constraints)**.**

$$\begin{array}{ll} \underset{\theta}{\operatorname{minimize}} & f^i\left(\theta^i\right)+f^j\left(\theta^j\right) \\ \textit{subject to} & \theta^i+\theta^j \leq 1 \end{array} \tag{2.14}$$

Solution:
[1. With Dual Decomposition]
First use Lagrange Multiplier: (higher level objective function)

$$
\begin{aligned}
L\left(\theta, \lambda^{(i,j)}\right) &= f^i\left(\theta^i\right) + f^j\left(\theta^j\right) + \lambda^{(i,j)}\left(\theta^i + \theta^j - 1\right) \\
d(\lambda^{i,j}) &= \inf_{\theta^i}\left[f^i\left(\theta^i\right) + \lambda^{(i,j)}\theta^i\right] + \inf_{\theta^j}\left[f^j\left(\theta^j\right) - \lambda^{(i,j)}\theta^j\right] - \lambda^{(i,j)}
\end{aligned}
\tag{2.15}
$$

[2. With Primal Decomposition]

We can rewrite the function as:

$$
\begin{aligned}
\underset{\theta}{\text{minimize}} \quad & f^i\left(\theta^i\right) + f^j\left(\theta^j\right) \\
\text{subject to} \quad & \theta^i \leq r^i \\
& \theta^j \leq 1 - r^i
\end{aligned}
\tag{2.16}
$$

Then it can be divided in two lower level problem:

$$
\begin{aligned}
\phi^i\left(r^i\right) &= \inf_{\theta^i} f^i\left(\theta^i\right): \quad \theta^i \leq r^i \\
\phi^j\left(r^i\right) &= \inf_{\theta^j} f^j\left(\theta^j\right): \quad \theta^j \leq 1 - r^i
\end{aligned}
\tag{2.17}
$$

Comparison:

For Dual Composition, when in iteration, the constraints may not hold, but finally, it will hold when convergence happen

For Primal Composition, the constrains will always hold when iteration.

2.4 Discussion

2.4.1 Why We Need Distributed Decomposition

2.4.2 Key Point When Decomposing

How to keep each question become independable?

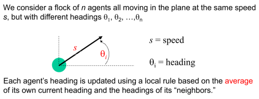
Chapter 3

Chapter 3: Consensus Problem

Content

1. Background: Flocking Problem
2. Discrete-Time model for Consensus (fixed neighbor and switching neighbor)
3. Discrete-Time agreement for Consensus (fixed neighbor and switching neighbor)
4. Continuous-Time model for Consensus (fixed neighbor and switching neighbor)
5. Continuous-Time agreement for Consensus (fixed neighbor and switching neighbor)

3.1 Background: Flocking Problem



We consider a flock of *n* agents all moving in the plane at the same speed *s*, but with different headings $\theta_1, \theta_2, \ldots, \theta_n$

Each agent's heading is updated using a local rule based on the average of its own current heading and the headings of its "neighbors."

Figure 3.1: Flocking Problem

Viscek et al 1995: Simulations demonstrate that these averaging rules **can cause all agents to eventually move in the same direction** despite the absence of centralized coordination and despite the fact that each agent's set of neighbors changes with time

3.2 Discrete-Time model for Consensus

At each step, each agent updates its heading to the weighted average of its own current heading and the headings of its "neighbors"

$$\theta_i(t+1) = \sum_{j \in N_i} a_{ij}(t)\theta_j(t), \quad t = 0, 1, 2, \cdots \tag{3.1}$$

where N_i is the set of neighbors of agent i and $i \in N_i . a_{ij}(t) > 0$ if j is a neighbor of i, and $a_{ij}(t) = 0$ otherwise, and $\sum_{j=1}^{n} a_{ij}(t) = 1$, for all $i = 1, \ldots, n$

3.2.1 General Model

Definition 3.2.1 (Discrete-Time Consensus Model). *Let $\theta(t) = [\theta_1(t), \theta_2(t), \ldots, \theta_n(t)]^T$. , then:*

$$\theta(t+1) = A(t)\theta(t)$$

where $A(t) = (a_{ij}(t))_{n\times n}$. $A(t)$ is a stochastic matrix.

Definition 3.2.2 (Stochastic Matrix). *A matrix $P = (p_{ij})_{n\times n}$ is called a* ***stochastic matrix****, if $p_{ij} \geq 0$ and $\sum_{j=1}^{n} p_{ij} = 1$, for all $i = 1, \ldots, n$.*

For a stochastic matrix P, two properties hold

Proposition 3.2.3.

- *1 is an eigenvalue of P.*
- *$[1, \cdots, 1]^T$ is a right eigenvector associated to 1*

Proof. Trivial, because of $\sum_{j=1}^{n} p_{ij} = 1$, for all $i = 1, \ldots, n$ □

3.2.2 Time-Invariant Model

Definition 3.2.4 (DTCM with fixed neighbor relationship).

$$\begin{aligned} \theta(t+1) &= A\theta(t) \\ \theta(t) &= A^t\theta(0) \end{aligned} \tag{3.2}$$

3.2.3 Agreement Theorem

Definition 3.2.5 (Spanning Tree). *A graph G is said to contain a* ***spanning tree*** *if we can find a vertex i of G such that there is a directed path from i to ever other vertex j*

Theorem 3.2.6 (agreement for fixed neighbor relationship). *If the graph G describing the neighbor relationships of agents* ***contains a spanning tree****, then $A^t \to 1c^T$, as t $\to \infty$, which implies $\theta(t) \to \theta_{ss}$ 1, as t $\to \infty$. Thus* ***all the agents reach an agreement asymptotically****.*

Note: This part is quite similar to the convergence of **Markov Chain**

Definition 3.2.7 (union of graphs). *The union of a collection of several graphs, $\{G_1, G_2, \ldots, G_m\}$, each with vertex set $V = \{1, \ldots, n\}$, is meant the graph with vertex set V and edge set equaling the union of the edge sets of all of the graphs in the collection.*

Theorem 3.2.8 (agreement for switching neighbor relationship). *If there exists an* ***infinite sequence*** *of contiguous, nonempty, uniformly bounded, time-intervals $[t_i, t_{i+1})$, starting at $t_0 = 0$, with the proposition that across each such interval, the* ***union*** *of the collection of the graphs $\{G(t_i), G(t_i+1), \ldots, G(t_{i+1}-1)\}$* ***contains a spanning tree****,*

then $\theta(t) \to \theta_{ss}1$, as $t \to \infty$

Note: This theorem means if there is a sequence of small period and each small period, the graph is connected then it will reach agreement

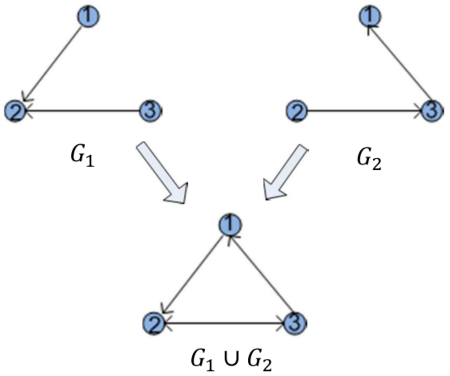
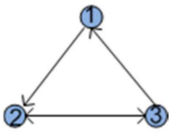


Figure 3.2: Union of Graphs



$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$L = D - A = \begin{bmatrix} 1 & 0 & -1 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

Figure 3.3: example laplacian matrix

3.3 Continuous-Time model for Consensus

3.3.1 General Model

Definition 3.3.1 (Continuous-Time Model for Consensus).

$$\dot{\theta}_i(t) = \sum_{j \in N_i} a_{ij}(t) \left(\theta_j(t) - \theta_i(t)\right) \tag{3.3}$$

where N_j is the set of neighbors of agent $i.a_{ii}(t) = 0$ and $a_{ij}(t) > 0$ if j is a neighbor of i, *and $a_{ij}(t) = 0$ otherwise.*

Definition 3.3.2 (In-degree Matrix). $d_i(t) = \sum_{j=1, j \neq i}^{n} a_{ij}(t)$ *is the in-degree of node i. The diagonal matrix $D = \operatorname{diag}\left(d_1(t), \ldots, d_n(t)\right)$ is the **in-degree matrix**.*

Definition 3.3.3 (Laplacian Matrix). *$L(t) = D(t) - A(t)$ is called the Laplacian matrix of the graph $G(t)$ where $A(t)$ is the adjacent matrix, $D(t)$ is the in-degree matrix*

Proposition 3.3.4. *Since $d_i(t) = \sum_{j=1, j \neq i}^{n} a_{ij}(t), 0$ is an eigenvalue of L with an eigenvector* $[1, \ldots, 1]^T$

$$L \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.4}$$

With the help of laplacian model, the continuous model can be rewritten to

Definition 3.3.5. $\dot{\theta}(t) = -L(t)\theta(t)$

3.3.2 fixed neighbor relationship model

Definition 3.3.6 (continuous-time consensus model with fixed neighbor)**.**

$$\dot{\theta}(t) = -L\theta(t)$$

The solution to the system is

$$\theta(t) = e^{-Lt}\theta(0)$$

Theorem 3.3.7 (agreement for fixed neighbor relationship)**.** *If the graph G contains a spanning tree, then $\theta(t) \rightarrow \theta_{ss}1$ as $t \rightarrow \infty$ Thus all the agents reach an agreement asymptotically. .*

3.3.3 switching neighbor relationship

3.3.4 switching neighbor relationship model

Definition 3.3.8 (continuous-time consensus model with fixed neighbor)**.**

$$\dot{\theta}(t) = -L\left(t_i\right)\theta(t), \quad t \in [t_i, t_i + \tau_i)$$

where $t_0 = 0$ is the initial time, $\tau_i = t_{i+1} - t_i$, and $t_1, t_2, \ldots$ is an infinite time sequence at which the interaction graph changes, resulting in a change in $L(t)$

Theorem 3.3.9 (agreement for switching neighbor relationship)**.** *agreement for switching neighbor relationship*

Let $t_1, t_2, \ldots$ be an infinite time sequence at which the interaction graph switches and $a_{ij}(t) \in [a_L, a_M]$, where a_L and a_M are arbitrary positive numbers satisfying $a_L < a_M$

If there exists an ***infinite sequence of uniformly bounded time intervals*** *$\left[t_{i_j}, t_{i_{j+1}}\right), j = 1, 2, \ldots$, starting at $t_{i_1} = t_0$, with the proposition that* ***the union of the graphs across each interval*** *$\left[t_{i_j}, t_{i_{j+1}}\right)$* ***contains a spanning tree****, then the states of all the agents reach an agreement, that is $\theta(t) \rightarrow \theta_{Ss}1$, as* t → ∞

3.4 summary

Chapter 4

Chapter 4: ADMM and Incremental Subgradient Methods

4.1 ADMM

4.1.1 Dulaity Theory Model

Definition 4.1.1.

$$\begin{aligned} &\textit{minimize} \quad f(x) \\ &\quad s.t. \quad Ax = b \end{aligned} \tag{4.1}$$

- *Lagrangian:* $L(x,y) = f(x) + y^T(Ax - b)$
- *Dual function:* $g(y) = \inf_x L(x,y)$
- *Dual problem: maximize* $g(y) \rightarrow y^\star$
- *Primal optimizer:* $x^\star = \arg\min_x L(x, y^\star)$

4.1.2 Normal Method: Dual Ascent Method

Method 4.1.2 (Dual Ascent Method)**.**

$$\begin{aligned} &y^{k+1} = y^k + \alpha^k \nabla g\left(y^k\right) \\ &\textit{where } \nabla g\left(y^k\right) = A\tilde{x} - b \textit{ with } \tilde{x} = \arg\min_x L\left(x, y^k\right) \end{aligned} \tag{4.2}$$

which can be executed by following two steps:

$$\begin{aligned} x^{k+1} &:= \arg\min_x L\left(x, y^k\right) \quad x\textit{ -minimization} \\ y^{k+1} &:= y^k + \alpha^k\left(Ax^{k+1} - b\right) \quad \textit{dual update} \end{aligned} \tag{4.3}$$

Proposition 4.1.3. *1. often slow 2. Need Strong Assumptions for convergence*

4.1.3 Dual Ascent Method with Dual Decomposition

$$\begin{aligned} f(x) &= f_1(x_1) + \cdots + f_N(x_N), \quad x = (x_1, \cdots, x_N) \\ L(x,y) &= L_1(x_1, y) + \cdots + L_N(x_N, y) - y^T b, \\ L_i(x_i, y) &= f_i(x_i) + y^T A_i x_i \end{aligned} \tag{4.4}$$

Method 4.1.4 (Dual Ascent Method with Dual Composition).

$$\begin{aligned} x^{k+1} &:= \arg\min_x L_i\left(x_i, y^k\right) \quad i = 1, \ldots, N \\ y^{k+1} &:= y^k + \alpha^k \left(\textstyle\sum_{i=1}^N A_i x_i^{k+1} - b\right) \end{aligned} \tag{4.5}$$

Remark: Main steps for implementation:

1. broadcast y^k dual variable
2. update x_i in parallel
3. collect $A_i x_i^{k+1}$

Proposition 4.1.5. *1. often slow 2. Need Strong Assumptions for convergence*

4.1.4 Method of Multipliers

Definition 4.1.6 (Augmented Lagrangian (with regularization)).

$$L_\rho(x, y) = f(x) + y^T(Ax - b) + \frac{\rho}{2}\|Ax - b\|_2^2 \tag{4.6}$$

Method 4.1.7 (Dual Ascent Method in Method of Multipliers).

$$\begin{aligned} x^{k+1} &:= \arg\min_x L_\rho\left(x, y^k\right) \\ y^{k+1} &:= y^k + \rho\left(A x_i^{k+1} - b\right) \end{aligned} \tag{4.7}$$

Proposition 4.1.8.
- *Converges under **much more relaxed conditions** compared to dual decomposition*
- *Quadratic penalty on constraint violation makes the **augmented Lagrangian non-separable***

4.1.5 Alternating Direction Method of Multipliers (ADMM)

Assume two sets of variables with a separable objective function where f and g are convex:

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

and the following augmented Lagrangian

$$L_\rho(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2 \tag{4.8}$$

Method 4.1.9 (ADMM).

$$\begin{aligned} x^{k+1} &:= \arg\min_x L_\rho\left(x, z^k, y^k\right) \quad \textit{x-minimization} \\ z^{k+1} &:= \arg\min_z L_\rho\left(x^{k+1}, z, y^k\right) \quad \textit{z-minimization} \\ y^{k+1} &:= y^k + \rho\left(Ax^{k+1} + Bz^{k+1} - c\right) \quad \textit{dual update} \end{aligned} \tag{4.9}$$

4.1.6 ADMM with Scaled Dual Variables

Definition 4.1.10 (ADMM with Scaled Dual Variables).

$$\begin{aligned} L_\rho(x,z,y) &= f(x) + g(z) + y^T(Ax+Bz-c) + (\rho/2)\|Ax+Bz-c\|_2^2 \\ &= f(x) + g(z) + (\rho/2)\|Ax+Bz-c+u\|_2^2 + \textit{ const.} \end{aligned} \tag{4.10}$$

Method 4.1.11.

$$\begin{aligned} & x^{k+1} := \arg\min_x \left(f(x) + \tfrac{\rho}{2} \left\|Ax + Bz^k - c + u^k\right\|_2^2 \right) \\ & z^{k+1} := \arg\min_z \left(g(z) + \tfrac{\rho}{2} \left\|Ax^{k+1} + Bz - c + u^k\right\|_2^2 \right) \\ & u^{k+1} := u^k + \left(Ax^{k+1} + Bz^{k+1} - c\right) \end{aligned} \tag{4.11}$$

Proposition 4.1.12.

1. *Very few asusmptions are need for convergence: f, g convex, closed, proper, L_0 has saddle point*
2. *Primal feasibility and optimality are achieved asymptotically*
3. *Good results in a few tens of iterations, although slow to converge to very high accuracy: worst case complexity is $\mathcal{O}\left(1/\epsilon^2\right)$*

Definition 4.1.13 (meta-algorithm). *Specifically, a meta-algorithm, in the context of learning theory, is an algorithm that* ***decides how to take a set of other (typically, though not necessarily non-meta) "algorithms"*** *(which might be as dumb as a constant output, for example), and constructs a new algorithm out of those, often by combining or weighting the outputs of the component algorithms*

Note: ADMM is a **meta-algorithm** that coordinates existing solvers to solve problems of arbitrary size. **The update steps can be implemented using an analytical solution, Newton's method, Conjugate Gradient, first-order methods, custom methods**

4.1.7 Special Cases for ADMM

4.1.8 Example for ADMM

Example 4.1.14 (ADMM for Constrained Convex Optimization). *Consider the generic problem minimize $f(x)$ subject to $x \in \mathcal{C}$ In order to translate this to ADMM form, take g as the indicator function of $\mathcal{C}$*

$$\textit{minimize} \quad f(x) + g(z)$$

subject to $x - z = 0$

Solution The ADMM algorithm then becomes

$$\begin{aligned} & x^{k+1} := \arg\min_x \left(f(x) + \tfrac{\rho}{2} \left\|x - z^k + u^k\right\|_2^2 \right) \\ & z^{k+1} := \Pi_{\mathcal{C}}\left(x^{k+1} + u^k\right) \\ & u^{k+1} := u^k + x^{k+1} - z^{k+1} \end{aligned}$$

4.1.9 ADMM for Consensus Optimization

The problem to be solved contains N objective terms:

$$\text{minimize} \sum_{i=1}^{N} f_i(x)$$

This can be expressed in ADMM form as:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} f_i\left(x_i\right) \\ \text{subject to} & x_i - z = 0 \end{array}$$

The augmented Lagrangian becomes:

$$L_\rho(x, z, y) = \sum_{i=1}^{N}\left(f_i\left(x_i\right) + y_i^T\left(x_i - z\right) + \frac{\rho}{2}\left\|x_i - z\right\|_2^2\right)$$

Method 4.1.15 (ADMM for Consensus Optimization)**.**

$$\begin{aligned} x_i^{k+1} &:= \arg\min_{x_i}\left(f_i\left(x_i\right) + y_i^{kT}\left(x_i - z^k\right) + \tfrac{\rho}{2}\left\|x_i - z^k\right\|_2^2\right) \\ z^{k+1} &:= \tfrac{1}{N}\textstyle\sum_{i=1}^{N}\left(x_i^{k+1} + \tfrac{1}{\rho}y_i^k\right) \\ y_i^{k+1} &:= y_i^k + \rho\left(x_i^{k+1} - z^{k+1}\right) \end{aligned} \tag{4.12}$$

proof of z update.

$$\begin{aligned} &\textstyle\sum \frac{\rho}{2}\left\|x_i - z\right\|_2^2 + y_i^\top\left(x_i - z\right) \\ &\Rightarrow \Sigma - \rho\left(x_i - z\right) - y_i^\top = 0 \\ &\Rightarrow \quad z = \tfrac{1}{N}\Sigma\tfrac{1}{\rho}y_i^\top + x_2^k \end{aligned} \tag{4.13}$$

□

The ADMM for Consensus Optimization can be optimized

Method 4.1.16 (ADMM for consensus (optimized))**.**

$$\begin{aligned} x_i^{k+1} &:= \arg\min_{x_i}\left(f_i\left(x_i\right) + y_i^{kT}\left(x_i - \bar{x}^k\right) + \tfrac{\rho}{2}\left\|x_i - \bar{x}^k\right\|_2^2\right) \\ y_i^{k+1} &:= y_i^k + \rho\left(x_i^{k+1} - \bar{x}^{k+1}\right) \end{aligned} \tag{4.14}$$

where $\bar{x}^k = \frac{1}{N}\sum_{i=1}^{N} x_i^k$

Remark: The dual variables are separately updated to drive the variables into consensus

quadratic regularization helps pull the variables toward their average value while still attempting to minimize each local f_i

4.2 Incremental Subgradient Methods

Method 4.2.1.

$$\begin{aligned} &\vartheta_k^0 = \theta_k, && \textit{(initialize)} \\ &\vartheta_k^i = \mathcal{P}_\Theta\left[\vartheta_k^{i-1} - \alpha_k g_k^i\right], && \textit{(iterate)} \\ &g_k^i \in \partial f^i\left(\vartheta_k^{i-1}\right), i = 1, \ldots, N, && \\ &\theta_{k+1} = \vartheta_k^N, && \textit{(update)} \end{aligned} \tag{4.15}$$

Proposition 4.2.2.

- *Convergence can be faster than standard method: small but more steps, more on-time adjustment and more exploration*
- *Calculations can be performed in a distributed way through* ***sequential*** *updates to* θ*.*

Example 4.2.3 (Finite-Time Optimal Rendezvous (FTOR)). *Consider N dynamic agents*

$$\begin{aligned} x_{t+1}^i &= A^i x_t^i + B^i u_t^i \\ y_t^i &= C^i x_t^i \end{aligned}$$

Polyhedral constraints

$$x_t^i \in \mathcal{X}^i, \quad u_t^i \in \mathcal{U}^i, \quad t \geq 0$$

Target: Rendezvous

$$\begin{aligned} y_{T+k}^i &= \theta, \quad \forall k \geq 0, \quad i = 1, \ldots, N, \\ u_{T+k}^i &= u_T^i, \quad \forall k \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{4.16}$$

Each agent has nontrivial, constrained LTI dynamics. And the rendezvous point $\theta \in \Theta$ is ***not fixed a priori****, but chosen optimally*

Cost function:

$$\begin{aligned} V^i\left(x_k^i, u_k^i, \theta\right) = &\left(x_k^i - x_e^i(\theta)\right)^\top Q^i \left(x_k^i - x_e^i(\theta)\right) \\ &+ \left(u_k^i - u_e^i(\theta)\right)^\top R^i \left(u_k^i - u_e^i(\theta)\right) \end{aligned} \tag{4.17}$$

Solution: The formulation optimization problem of FTOR will be :

$$\begin{array}{ll} \min_{U,\theta} & \sum_{i=1}^N \sum_{k=0}^{T-1} V^i\left(x_k^i, u_k^i, \theta\right) \\ \text{subject to} & x_{k+1}^i = A^i x_k^i + B^i u_k^i, \\ & y_k^i = C^i x_k^i, \\ & x_k^i \in \mathcal{X}^i, \quad k = 1, \ldots, T, \\ & u_k^i \in \mathcal{U}^i, \quad k = 0, \ldots, T-1, \\ & y_T^i = \theta, x_T^i = x_e^i(\theta), \\ & x_0^i = x^i(0), \quad i = 1, \ldots, N \\ & \theta \in \Theta \end{array} \tag{4.18}$$

For simplicity, first, eliminate control inputs $u^i = k^i\left(x^i, \theta\right)$

$$\begin{gathered} f^i\left(x^i, \theta\right) = \min_{U^i} \sum_{k=0}^{T-1} V^i\left(x_k^i, u_k^i, \theta\right) \\ \text{subject to constraints, } k = 1, \ldots, T-1 \end{gathered} \tag{4.19}$$

Then, the problem can be rewritten to:

$$\begin{gathered} f^*(x) = \min_\theta \sum_{i=1}^N f^i\left(x^i, \theta\right) \\ \text{subject to } \theta \in \Theta \end{gathered} \tag{4.20}$$

Then, primal decomposition can be used. And then, cyclic incremental subgradient method can be used

1. Initialize θ_0 and α_0,set $k = 0, h = 1$

2. $\alpha_h = \frac{\alpha_0}{h}$

3. for $i = 1$ to N do

(a) Compute a subgradient λ_k^i for $f^i(\theta_k)$

(b) $\theta_k = \mathcal{P}_\Theta\left[\theta_k - \alpha_h \lambda_k^i\right]$

(c) $k = k + 1$

4. $h = h + 1$ go to step 2

4.3 More Generalized Problem Form

Definition 4.3.1.

$$\begin{aligned} &\underset{\theta}{\text{minimize}} \quad f(\theta) = \textstyle\sum_{i=1}^N f^i(\theta) \\ &\textit{subject to } \theta \in \Theta \end{aligned} \tag{4.21}$$

- *$f^i : \mathbb{R}^M \to \mathbb{R}$ nondifferentiable convex functions.*
- *$\Theta \subseteq \mathbb{R}^M$ nonempty, closed, and convex set.*
- *Computations should be performed in a distributed fashion.*
- *Information exchange is* ***only allowed through edges*** *of an N -node undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.*

Methods learnt until now all have their issues:

- Primal Decomposition:Needs a **coordinator**, hard to implement in this form, because hard to find a node directly connect to all other nodes
- Dual Decomposition: Needs one **consistency price** per edge
- Incremental Subgradient: not peer-to-peer, they need to pass an estimate in a ring, or to a random neighbor

4.3.1 Model Transformation use Consensus Methods

Method 4.3.2 (Model Transformation use Consensus Methods)**.** *Problem with format*

$$\begin{aligned} &\underset{\theta}{\text{minimize}} \textstyle\sum_{i=1}^N f^i(\theta) \\ &\textit{subject to } \theta \in \Theta \end{aligned} \tag{4.22}$$

can be transformed to:

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \textstyle\sum_{i=1}^N f^i\left(\theta^i\right) \\ \textit{subject to} \quad & \theta^i \in \Theta \\ & \theta^i = \theta^j, \forall (i,j) \in \mathcal{E} \end{aligned} \tag{4.23}$$

After transformation:

- Each node has local view of global decision variables
- Coordinates with neighbors to achieve consistency (using consensus iterations)

4.3.2 Modified Subgradient Iterations (Combined Consensus)

Our goal is to use agreement protocols to relax communication constraints in distributed optimization schemes.

Method 4.3.3.

$$\theta_{k+1}^{i}=\mathcal{P}_{\Theta}\left[\sum_{j=1}^{N}\left[W^{\varphi}\right]_{ij}\left(\theta_{k}^{j}-\alpha_{k}g^{j}\left(\theta_{k}^{j}\right)\right)\right] \tag{4.24}$$

ith $W=I-\varepsilon L(\mathcal{G})$ Perron matrix corresponding to the communication graph.

Remark: [Main Interations of the Algorithm]

1. Perform **local subgradient update** on local variable x^i.
2. Do ϕ **consensus iterations** with neighbors. (Can be interpreted as enforcing approximate equality constraints with neighboring variables.) (when ϕ goes to infinity, it will not be approximate)
3. Repeat

Note: When understanding the Modified Subgradient Iterations, imagine there is a **buffer** for each agent. The local update will be stored in the buffer, and will be transmitted among agents

Theorem 4.3.4 (Convergence Result)**.** *Under appropriate assumptions, the sequence $\left\{\theta_k^1,\ldots,\theta_k^N\right\}_{k=0}^{\infty}$ generated by the combined SG/consensus update with φ consensu: iterations and $\left\|\theta_0^i-\bar{\theta}_0\right\|\leq\beta$, converges to a neighborhood of the optimal point:*

$$\liminf_{k\rightarrow\infty} f\left(\theta_k^i\right)\leq f^{\star}+\frac{\alpha NC^2}{2}+3NC\beta, \forall i=1,\ldots,N$$

Remark:

- We can get $\liminf_{k\rightarrow\infty} f\left(\theta_k^i\right)$ to be arbitrarily close to $f^{\star}$, by choosing the constants α and β arbitrarily small.
- Note that the number of required consensus negotiations, φ, to reach a fixed β is fixed, i.e., **independent of** k.

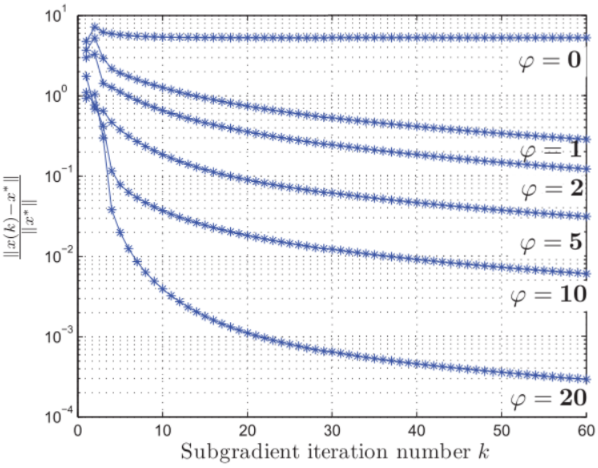


Figure 4.1: Numerical Example Modified Subgradient Method

4.3.3 summary