[1] Опис лиценци Креативне заједнице доступан је на адреси creativecommons.org.rs/?page_id=74.

UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCES
DEPARTMENT OF MATHEMATICS
AND INFORMATICS

Dušan Jakovetić **Aleksandar Armacki**

Distributed Optimization with Applications

Novi Sad, 2020

Distributed optimization with applications
First edition

Authors
Dr. Dušan Jakovetić
Assistant Professor, Univ. of Novi Sad Faculty of Sciences

Aleksandar Armacki
Researcher, Univ. of Novi Sad Faculty of Sciences

Reviewers
Academician Prof. Dr. Gradimir Milovanović
Serbian Academy of Sciences and Arts, Mathematical Institute

Prof. Dr. Nataša Krejić
Full Prof. University of Novi Sad Faculty of Sciences

Dr. Nataša Krklec Jerinkić
Associate Prof. University of Novi Sad Faculty of Sciences

Publisher
University of Novi Sad Faculty of Sciences

Editor-in-Chief
Prof. Dr. Milica Pavkov-Hrvojević, Dean of University of Novi Sad Faculty of Sciences

Publication of this textbook is approved by the Council of Faculty of Sciences on March 12, 2020.
ISBN: 978-86-7031-477-1

CIP - Каталогизација у публикацији
Библиотеке Матице српске, Нови Сад

51(0758)

JAKOVETIĆ, Dušan, 1983-
Distributed optimization with applications [Elektronski izvor] / Dušan Jakovetić, Aleksandar Armacki. - 1st ed. - Novi Sad : Faculty of Sciences, 2020

Način pristupa (URL):
https://www.pmf.uns.ac.rs/studije/epublikacije/matinf/jakovetic_armacki_distributed_optimization_applications.pdf. - Opis zasnovan na stanju na dan 18.3.2020. - Nasl. s naslovnog ekrana. - Bibliografija.

ISBN 978-86-7031-477-1

1. Armacki, Aleksandar, 1993-
a) Примењена математика

COBISS.SR-ID 333387527

Contents

1 Convex functions **3**
- 1.1 Introduction . . . 3
- 1.2 Rules for recognizing convex functions . . . 7

2 Subgradient methods **11**
- 2.1 Subgradients and ϵ-subgradients . . . 11
- 2.2 Subgradient calculus . . . 13
- 2.3 Subgradient and inexact subgradient methods . . . 15
  - 2.3.1 Optimality conditions for unconstrained minimization 16
  - 2.3.2 Analysis of exact and inexact subgradient methods . . 16

3 Duality **19**
- 3.1 Introduction . . . 19
- 3.2 Weak Duality . . . 22
- 3.3 Strong Duality . . . 23
- 3.4 Dual subgradient method . . . 25

4 Dual distributed methods **27**
- 4.1 Introduction . . . 27
- 4.2 Computation model . . . 28
- 4.3 Communication models . . . 29
- 4.4 Application examples . . . 30
- 4.5 Dual distributed methods . . . 34

5 Primal distributed subgradient methods **39**
- 5.1 Communication and computation model . . . 39
- 5.2 The weight matrix . . . 40
- 5.3 The algorithm . . . 41
- 5.4 Convergence analysis . . . 41

6 An advanced topic: Primal distributed Nesterov-like gradient methods **49**
6.1 Algorithm D–NG 49
6.1.1 Model and notational preliminaries 49
6.1.2 D–NG algorithm 50
6.2 Analysis of algorithm D–NG 51
6.2.1 Assumptions and setup 51
6.2.2 Clone functions Ψ_k 53
6.3 Convergence analysis of projected mD–NC 55
6.4 Projected mD–NC method: Constrained optimization 62
6.4.1 Model and algorithm 62
6.4.2 Framework of Inexact Nesterov gradient method . . . 63
6.4.3 Convergence rate analysis 65
6.5 Proof of Lemma 5 (1) 69

7 Conclusion **75**

Foreword

This educational-purpose manuscript represents a supporting (auxiliary) textbook for the course "Distributed optimization with applications" within the master program "Applied mathematics – Data science" at the University of Novi Sad, Faculty of Sciences. The related course assumes that students possess solid background in standard concepts from numerical optimization, gained through the course "Fundamentals of numerical optimization" within the same master program. The manuscript covers a major part of the course curriculum, focusing on distributed methods for convex, non-smooth optimization, and is aimed to assist students in acquiring solid understanding of the concepts thought in class. The manuscript represents a supporting material, while a more elaborated treatment of most of the topics can be found in the references provided. In more detail, Chapter 1 of the manuscript starts with reviewing relevant concepts and properties of convex functions. Chapter 2 reviews elements of subgradient calculus and subgradient methods. Chapter 3 is concerned with duality theory. The material in Chapters 1-3 provides a required background for understanding of design and analysis of parallel and distributed optimization methods for convex (non)smooth problems. Chapter 4 introduces some common communication and computational (optimization) models and provides several application examples. The chapter also considers dual distributed methods, while Chapter 5 considers primal (sub)gradient methods. Chapter 6 considers an advanced topic and is recommended for students that would like to understand the materials beyond the nominal curriculum. The Chapter is concerned with the primal distributed methods based on the Nesterov gradient method. Finally, we conclude in Chapter 7.

Novi Sad, February 2020 Dušan Jakovetić, Aleksandar Armacki

Chapter 1

Convex functions

This chapter gives the definition of a convex function and provides some examples of convex functions. It also describes some commonly used rules on how to characterize convex functions. This is standard material, see, e.g., [1, 2, 3, ?], for which we primarily closely follow [1] in Section 1.2. We also closely follow [4] throughout the Chapter.

1.1 Introduction

A commonly studied class of functions found in optimization problems is the class of *convex* functions. Informally, a function is convex, if the function value of a convex combination of any two points is less than or equal to the convex combination of the function values at those two points. More formally, a convex function is defined as follows.

Definition 1.1. A function $f : \mathbb{R}^d \mapsto \mathbb{R}$ is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y), \tag{1.1}$$

for all $\theta \in [0,1]$ and all $x, y \in \mathbb{R}^d$. If strict inequality holds for all x, y and θ, we say that the function is strictly convex.

If a function is twice differentiable, a necessary and sufficient condition for convexity can be stated ([1], Section 3.1.4) as follows.

Proposition 1.1. *A twice differentiable function $f : \mathbb{R}^d \mapsto \mathbb{R}$ is convex if and only if*

$$\nabla^2 f(x) \succeq 0, \tag{1.2}$$

for all $x \in \mathbb{R}^d$. If a strict inequality holds for all x, we say that the function is strictly convex.

Here, notation $A \succeq 0$ means that matrix A is positive semi-definite.

Remark. Equation (1.2) states that for a function to be convex, it suffices that its Hessian matrix is positive semi-definite. Recall that a (symmetric) matrix $M \in \mathbb{R}^{d \times d}$ is positive semi-definite if one of the following (equivalent) conditions holds:

- $x^T M x \geq 0$, for all $x \in \mathbb{R}^d$,
- all eigenvalues of M are nonnegative.

Example 1.1.1. Affine function:

We consider the function $f : \mathbb{R}^d \mapsto \mathbb{R}$, given by $f(x) = a^T x + b$, where $a \in \mathbb{R}^d$, $b \in \mathbb{R}$. We would like to show that $f(x)$ is a convex function.

In order to do so, pick any two points $x, y \in \mathbb{R}^d$ and an arbitrary $\theta \in [0, 1]$. We want to show that in the case of our f, (1.1) holds for x, y and θ.

$$\begin{aligned} f(\theta x + (1-\theta)y) &= a^T(\theta x + (1-\theta)y)) + b = \theta a^T x + (1-\theta)a^T y) + (1 \pm \theta)b \\ &= \theta(a^T x + b) + (1-\theta)(a^T y + b) = \theta f(x) + (1-\theta) f(y) \end{aligned}$$

We have shown that $f(\theta x + (1-\theta)y) = \theta f(x) + (1-\theta) f(y)$, for all $x, y \in \mathbb{R}^d$ and $\theta \in [0, 1]$. Therefore, f is convex. Additionally, it can be shown that $f(x)$ is also *concave*.

To do so, first recall that a function f is concave if $g(x) = -f(x)$ is convex. If we define $\hat{a} = -a$ and $\hat{b} = -b$, then based on what we have just proven, we can conclude that $g(x) = \hat{a}^T x + \hat{b} = -f(x)$ is convex, therefore showing that $f(x)$ is concave.

Alternatively, to show that f is convex, we can prove that its Hessian is positive semi-definite. Recall that we compute the gradient of a function $f : \mathbb{R}^d \mapsto \mathbb{R}$ through its partial derivatives as follows.

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x) \\ \frac{\partial f}{\partial x_2}(x) \\ \vdots \\ \frac{\partial f}{\partial x_d}(x) \end{bmatrix}. \tag{1.3}$$

Knowing that we can rewrite our $f(x) = a^T x + b$ as

$$f(x) = a_1 x_1 + a_2 x_2 + \ldots + a_d x_d + b,$$

it is not hard to see that in our particular example, the gradient of f is given

by

$$\nabla f(x) = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_d \end{bmatrix}.$$

Recalling that the Hessian is computed as

$$[\nabla^2 f(x)]_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}(x),$$

it is not hard to see that the Hessian evaluates to the zero matrix (exercise). Therefore, we have shown that $\nabla^2 f(x) \succeq 0$, for all $x \in \mathbb{R}^d$, hence showing that f is convex ∎

Example 1.1.2. Quadratic function:

We consider the function $f : \mathbb{R}^d \mapsto \mathbb{R}$, given by $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, where $A \in \mathbb{R}^{d \times d}$ is a symmetric positive semi-definite matrix, $b \in \mathbb{R}^d$ is a vector, and $c \in \mathbb{R}$ is a scalar. We want to show that $f(x)$ is convex.

To do so, we start by rewriting the function f as

$$f(x) = \frac{1}{2}\sum_{i,j=1}^{d} A_{ij}x_i x_j + \sum_{i=1}^{d} b_i x_i + c.$$

Recalling how we compute the gradient (1.3), it is not hard to see that

$$[\nabla f(x)]_i = \frac{\partial f}{\partial x_i}(x) = A_{i1}x_1 + A_{i2}x_2 + \ldots + A_{id}x_d + b_i. \tag{1.4}$$

Writing matrix A in terms of its columns, i.e.

$$A = \left[a_1 | a_2 | \ldots | a_d\right],$$

we get to an equivalent representation of (1.4),

$$[\nabla f(x)]_i = a_i^T x + b_i.$$

From the expression above, we can conclude that the full gradient is of the

form

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_d}(x) \end{bmatrix} = \begin{bmatrix} a_1^T x + b_1 \\ \vdots \\ a_d^T x + b_d \end{bmatrix} = \begin{bmatrix} a_1^T \\ \vdots \\ a_d^T \end{bmatrix}_{d\times d} \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}_{d\times 1} + \begin{bmatrix} b_1 \\ \vdots \\ b_d \end{bmatrix}_{d\times 1}. \tag{1.5}$$

Noting that the first term on the left-hand side of (1.5) is actually the transposed matrix A, and recalling that the matrix A is symmetric, we can write the gradient of f as

$$\nabla f(x) = Ax + b. \tag{1.6}$$

Recalling the partial derivatives from (1.4), we can calculate the second-order partial derivatives as

$$\begin{aligned} \frac{\partial^2 f}{\partial x_i \partial x_j}(x) &= A_{ij} \\ \frac{\partial^2 f}{\partial^2 x_i}(x) &= A_{ii} \end{aligned},$$

which implies that

$$\nabla^2 f(x) = A. \tag{1.7}$$

From (1.7) and recalling that $A \succeq 0$, we can conclude that f is convex.

In the case where $A \preceq 0$, f becomes a concave function. To show this, we first recall that a (symmetric) matrix $M \in \mathbb{R}^{d\times d}$ is negative semi-definite if one of the following (equivalent) conditions holds:

- $x^T M x \leq 0$, for all $x \in \mathbb{R}^d$,
- all eigenvalues of M are nonpositive,
- $-M$ is positive semi-definite.

Next, we use the fact that, for a function f to be concave, it suffices to show that $-f$ is convex.

Consider $g(x) = -f(x) = \frac{1}{2}x^T(-A)x + (-b)^T x + (-c)$. Denote $\hat{A} = -A$, $\hat{b} = -b$ and $\hat{c} = -c$. Because $A \preceq 0$, we know that $\hat{A} \succeq 0$ (exercise). From the previous proof, we can conclude that $g(x) = \frac{1}{2}x^T\hat{A}x + x^T\hat{b} + \hat{c}$ is convex. Since $f(x) = -g(x)$, it implies that f is concave, which is what we wanted to show.

In the case of $A = 0$, f evaluates to an affine function (as in the first example) and so it is both convex and concave.

Another interesting case to consider is that of A being neither positive nor negative semi-definite. In this case, f is neither convex nor concave. To illustrate, consider a quadratic function $f : \mathbb{R}^2 \mapsto \mathbb{R}$, given by $f(x) = x^T Ax$, where

$$A = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

In this case, we can write f as: $f(x) = x_1^2 - x_2^2$. To see that f is not convex, observe its behaviour along $(0, x_2)$, whereas to show that f is not concave, observe its behaviour along $(x_1, 0)$.

Note that in all the cases so far, we assumed A to be a symmetric matrix. We now consider the case when A is not symmetric. First, note that the quadratic form of a matrix and its transpose is the same (exercise). Therefore, we can rewrite function f as $f(x) = \frac{1}{2}x^T(\frac{A+A^T}{2})x + b^T x + c$, that ensures the matrix that corresponds to the quadratic term is symmetric.

1.2 Rules for recognizing convex functions

There are a few standard rules that guarantee a function is convex. Here, we list some frequently used rules, e.g., [1]:

1. **Concatenation of an affine function and a convex function:**

   Let $h : \mathbb{R}^d \mapsto \mathbb{R}^m$ be an affine function, given by $h(x) = A^T x + b$, where $A \in \mathbb{R}^{d \times m}$ and $b \in \mathbb{R}^m$, and let $g : \mathbb{R}^m \mapsto \mathbb{R}$ be a convex function. Then, the function $f : \mathbb{R}^d \mapsto \mathbb{R}$, given by $f(x) = g(A^T x + b) = g(h(x))$ is a convex function.

2. **Pointwise supremum of convex functions:**

   Let $\{f_i\}_{i \in A}$, $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ be a family of functions, where $A \neq \varnothing$. Define $\phi : \mathbb{R}^d \mapsto \mathbb{R}$, as $\phi(x) = \sup_{i \in A} f_i(x)$. If f_i is convex for all $i \in A$, then ϕ is convex as well.

3. **Infimum rule:**

   Let $f : \mathbb{R}^m \times \mathbb{R}^n \mapsto \mathbb{R}$, and define $\phi : \mathbb{R}^m \mapsto \mathbb{R}$, as $\phi(x) = \inf_{y \in C} f(x, y)$, where C is a non-empty set. If C is a convex set and f is a convex function, then ϕ is convex.

4. **Non-negative weighted sum:**

   Let $\{f_i\}_{i=1}^m$, $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, f_i convex, for all $i = 1, \ldots, m$. Let $\{\alpha_i\}_{i=1}^m$ be a finite sequence of non-negative real numbers. Then, for $f : \mathbb{R}^d \mapsto \mathbb{R}$, defined by $f(x) = \alpha_1 f_1(x) + \ldots + \alpha_m f_m(x)$ it holds that f is convex.

Example 1.2.1. Norms:

Recall that a norm is any function $\|\cdot\| : \mathbb{R}^d \mapsto \mathbb{R}$, that satisfies:

1. $\|ax\| = |a|\|x\|$, for all $a \in \mathbb{R}$, for all $x \in \mathbb{R}^d$,
2. $\|x + y\| \leq \|x\| + \|y\|$, for all $x, y \in \mathbb{R}^d$,
3. $\|x\| \geq 0$, for all $x \in \mathbb{R}^d$,
4. $\|x\| = 0 \iff x = 0$, for all $x \in \mathbb{R}^d$.

In optimization, norms are used as cost functions (e.g. when minimizing the distance from a target position in target localization, or when minimizing the difference between the predicted and the true values in machine learning), or often times as regularizers (e.g., introducing L1 or L2 regularization to induce (group) sparsity in the solution). Some of commonly used norms include:

- L1 norm - $\|x\|_1 = |x_1| + |x_2| + \ldots + |x_d|$,
- L2 norm - $\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_d^2}$,
- L_∞ norm - $\|x\|_\infty = \max_{i=1,\ldots,d} |x_i|$.

Any norm on $\mathbb{R}^d$ is convex. In order to show this, pick any $x, y \in \mathbb{R}^d$ and any $\theta \in [0, 1]$. We want to prove that (1.1) holds for $f \equiv \|\cdot\|$. Starting from the LHS of (1.1), we get

$$\|\theta x + (1-\theta)y\| \leq \|\theta x\| + \|(1-\theta)y\| = |\theta|\|x\| + |(1-\theta)|\|y\| = \theta\|x\| + (1-\theta)\|y\|,$$

where the first inequality stems from the triangle inequality, while the second and the third stem from the properties of norms and the fact that $\theta \in [0, 1]$ ∎

Example 1.2.2. Least squares problem:

Consider function $f : \mathbb{R}^d \mapsto \mathbb{R}$, $f(x) = \|Ax - b\|_2^2$, where $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$. We want to show that $f(x) = \|Ax - b\|_2^2$ is convex.

The L2 norm can be written as

$$\|x\|_2 = \sqrt{x^T x}. \tag{1.8}$$

Proceeding from (1.8), we can rewrite the least squares problem as

$$\begin{aligned} f(x) = (Ax-b)^T(Ax-b) &= (Ax)^T Ax - 2b^T Ax + b^T b \\ &= x^T A^T Ax - 2b^T Ax + b^b. \end{aligned}$$

Noting that $f(x)$ is actually a quadratic function, it suffices to show that (1.2) holds. The Hessian of $f(x)$ is given by

$$\nabla^2 f(x) = A^T A,$$

which is a positive semi-definite matrix (exercise), therefore satisfying (1.2). This completes the proof.

Alternatively, we can use the rules of convexity to show that the least squares problem is a concatenation of an affine and a convex function. We start by defining $g : \mathbb{R}^d \mapsto \mathbb{R}^m$ as $g(x) = Ax + b$, for $A \in \mathbb{R}^{m\times d}$ and $b \in \mathbb{R}^m$. Obviously, g is an affine function. Next, we analyze the function $f : \mathbb{R}^m \mapsto \mathbb{R}$, given by $f(x) = \|x\|_2^2$. Using (1.8) once more, we can see that f is of the form

$$f(x) = x^T x.$$

It is obvious that f is a quadratic function and its Hessian is the identity matrix, I (exercise). Since $I \succeq 0$, we know that f is a convex function. Therefore, our original function $h(x) = \|Ax + b\|_2^2$ can be expressed as a concatenation of an affine ($g(x)$) and a convex function ($f(x)$). By the previously defined rules, this makes $h(x) = f(g(x))$ a convex function ∎

Example 1.2.3. Maximal eigenvalue of symmetric matrices:

Define $f : \mathbb{S}^n \mapsto \mathbb{R}$, as $f(X) = \lambda_{max}(X)$, where $\mathbb{S}^n$ represents the set of symmetric $n \times n$ matrices, and $\lambda_{max}(\cdot)$ denotes the maximal eigenvalue of a given matrix.

Recall that symmetric matrices have real eigenvalues and are diagonalizable (exercise). Therefore, it can be shown that we can compute the maximal eigenvalue of a symmetric matrix as (see, e.g., [5])

$$\lambda_{max}(X) = \max_{Q=\{q\in\mathbb{R}^n:\|q\|^2=1\}} q^T Xq. \tag{1.9}$$

Consider the function $\varphi_q : \mathbb{S}^n \mapsto \mathbb{R}$, defined as $\varphi_q(X) = q^T Xq = \sum_{i,j=1}^n X_{ij} q_i q_j$, where $q \in \mathbb{R}^n$. We can note that φ_q is affine in X. Moreover, $\lambda_{max}(X)$ can be represented as

$$\lambda_{max}(X) = \max_Q \varphi_q(X).$$

Using the fact that $\varphi_q(X)$ is affine (and therefore convex) in X, and going back to the rules of convexity, we can conclude that $\lambda_{max}(X)$ is a convex function, as it represents a pointwise maximum of convex functions ∎

The main sources used for writing this chapter are [1, 4].

Chapter 2

Subgradient methods

This chapter addresses how to develop first order optimization algorithms for convex functions that are not differentiable. The chapter introduces the concept of subgradient, subdifferential, a basic subgradient method, and its inexact variant. This is mainly standard material covered in, e.g., [7, 8, 9, 10], complemented with inexact methods [11]. Here, we primarily closely follow [9, 10].

2.1 Subgradients and ϵ-subgradients

Gradient-like methods are lauded for their simplicity and efficiency, which is well-established, especially for convex, differentiable functions. A typical gradient-like iterative method for solving problems of the type

$$\min_{x \in \mathbb{R}^d} f(x), \tag{2.1}$$

has the following form

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k), \tag{2.2}$$

where x^k represents the current estimate, x^{k+1} the next estimate, α^k the chosen step-size and k the iteration counter.

It is of interest to develop gradient-like methods for functions that are convex, but not necessarily differentiable. As it turns out, it is possible to generalize the concept of gradients to convex, non-differentiable functions. This concept is called *subgradient*, see for example, [9, 3, 2], and it is used as a search direction in methods of the type (2.2).

To motivate the definition of subgradients, recall the convexity condition

(1.1). For differentiable functions, a condition equivalent to (1.1) is given by

$$f(y) \geq f(x) + \nabla^T f(x)(y - x), \text{ for all } x \in \mathbb{R}^d, \tag{2.3}$$

for any fixed $y \in \mathbb{R}^d$. Using (2.3), we define the subgradient as follows [9].

Definition 2.1. Consider a convex, not necessarily differentiable function $f : \mathbb{R}^d \mapsto \mathbb{R}$. We say that $g^f(x) \in \mathbb{R}^d$ is a subgradient of f at point x, if

$$f(y) \geq f(x) + (g^f(x))^T(y - x), \text{ for all } y \in \mathbb{R}^d. \tag{2.4}$$

A notion tied with subgradient is that of *subdifferential*. Formally, a subdifferential is defined as follows.

Definition 2.2. The subdifferential $\partial f(x)$ of function f at point x is the set of all subgradients of f at x.

Before further characterizing the subdifferential, let us recall the following properties of sets.

Definition 2.3. A set $S \subset \mathbb{R}^d$ is said to be *open* if for any point $x \in S$, there exists a ball $B_\epsilon(x) = \{y \in \mathbb{R}^d : \|y - x\| < \epsilon\}$, such that $B_\epsilon(x) \subset S$.

Definition 2.4. A set $S \subset \mathbb{R}^d$ is said to be *closed* if its complement, $\bar{S} = \mathbb{R}^d \setminus S$, is open.

Definition 2.5. A set $S \subset \mathbb{R}^d$ is said to be *convex* if

$$\theta x + (1 - \theta)y \in S,$$

for any $\theta \in [0, 1]$ and any $x, y \in S$.

We state the following two propositions without proof; see for example [9].

Proposition 2.1. *For any convex function $f : \mathbb{R}^d \mapsto \mathbb{R}$ the subdifferential ∂f at any point $x \in \mathbb{R}^d$ is a non-empty, closed, convex set.*

Proposition 2.2. *For any convex function $f : \mathbb{R}^d \mapsto \mathbb{R}$ and any point $x \in \mathbb{R}^d$, the following holds:*

1. *If f is differentiable at x, then $\partial f(x) = \{\nabla f(x)\}$, i.e., the subdifferential is a singleton set with the unique element $\nabla f(x)$.*
2. *If $\partial f(x) = \{g^f(x)\}$, for some $g^f(x) \in \mathbb{R}^d$, then f is differentiable at x and $\nabla f(x) = g^f(x)$.*

Example 2.1.1. Absolute value:

Consider a function $f : \mathbb{R} \mapsto \mathbb{R}$, given by $f(x) = |x|$.

Recall that f is differentiable everywhere, except at 0. Therefore, we can conclude that:

$$\partial f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases}$$

Using (2.5), it can be shown that $0, \pm 1 \in \partial f(0)$ (exercise). Moreover, it can be shown that $\partial f(0) \in [-1, 1]$.

With numerical optimization methods for nondifferentiable convex functions, a useful concept is that of ϵ-subgradients (inexact subgradients), e.g., [11].

Definition 2.6. Consider a convex, not necessarily differentiable function $f : \mathbb{R}^d \mapsto \mathbb{R}$, and let ϵ be a positive scalar. We say that $g^f(x) \in \mathbb{R}^d$ is an ϵ-subgradient of f at point x, if

$$f(y) \geq f(x) + (g^f(x))^T(y - x) - \epsilon, \text{ for all } y \in \mathbb{R}^d. \tag{2.5}$$

2.2 Subgradient calculus

Informally, subgradient calculus represents a set of rules on how one can calculate subgradients and subdifferentials. The subgradient calculus may be divided into the following two categories [9]:

- **Weak calculus**, that corresponds to calculation of a (single) subgradient $g^f(x)$ of a function f at point x.
- **Strong calculus**, that corresponds to calculation of the (full) subdifferential $\partial f(x)$ of a function f at point x.

Remark. Recall the gradient-type algorithms of the form (2.2). In the case of non-differentiable functions, the update rule (2.2) changes to

$$x^{k+1} = x^k - \alpha^k g^f(x^k),$$

where $g^f(x_k)$ is any subgradient of f at x^k. Since any subgradient from the subdifferential set can be used here, the weak calculus can be considered to be more relevant in terms of numerical algorithms definition and implementation.

We next introduce some properties of subgradients.

1. **Scaling**: Let $f : \mathbb{R}^d \mapsto \mathbb{R}$ and $\alpha \geq 0$, then

$$\partial[\alpha f(x)] = \alpha \partial f(x) = \{\alpha g : g \in \partial f(x)\}.$$

2. **Summation**: Let $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, 2$, f_1, f_2 convex. Then

$$\partial[f_1 + f_2](x) = \partial f_1(x) + \partial f_2(x) = \{g + h : g \in \partial f_1(x),\ h \in \partial f_2(x)\}.$$

We state and prove a "weaker" version of the summation rule above.

Claim 2.3. *If $g \in \partial f_1(x)$ and $h \in \partial f_2(x)$, $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, 2$, then*

$$g + h \in \partial[f_1 + f_2](x),$$

for any $x \in \mathbb{R}^d$.

Proof. Fix a point $x \in \mathbb{R}^d$. Because $g \in \partial f_1(x)$, we know that (2.5) holds for g and f_1 at x. For the same reasons, (2.5) holds for h and f_2 at x. Summing up the inequalities, we obtain

$$f_1(y) + f_2(y) \geq f_1(x) + f_2(x) + (g + h)^T (y - x), \tag{2.6}$$

for all $y \in \mathbb{R}^d$. From (2.6) it follows that $g + h \in \partial[f_1 + f_2](x)$, which is exactly what we wanted to show □

The summation property of subgradients and subdifferentials can also be generalized for m functions, instead of just two.

3. **Pointwise maximum**: Let $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, \ldots, m$, f_i, $i = 1, \ldots, m$ convex. Define $f(x) = \max_{i=1,\ldots,m} f_i(x)$. It is easy to note that $f(x)$ is a convex function, therefore subgradients of f exist at any point $x \in \mathbb{R}^d$. We will apply the weak and strong calculus to compute the subgradients and subdifferentials of f.

Weak calculus for pointwise maximum: A subgradient g^f of f at point x can be computed as any subgradient $g^{f_i}(x)$ of function f_i at x, such that $f(x) = f_i(x)$. In other words, for an arbitrary point x, we first find the "active function" as $i = \arg\max_{i=1,\ldots,m} f_i(x)$ and any subgradient of f_i at x is simultaneously a subgradient of f at x.

Example 2.2.1. Absolute value:

Recall the absolute value function from example (2.1.1). The absolute value function can equivalently be represented as

$$f(x) = \max\{f_1(x), f_2(x)\},$$

where $f_1(x) = x$, $f_2(x) = -x$. Note that both f_i, $i = 1, 2$ are convex (and differentiable), therefore (sub)gradients are defined everywhere. We can use the weak calculus for pointwise maximum to compute subgradients of the absolute value function f.

For example, at the point $x = 1$, $f_1(1) = 1$, $f_2(1) = -1$, so we can conclude that f_1 is the active function. Using the weak calculus, a (sub)gradient of f at $x = 1$ is a (sub)gradient of f_1 at $x = 1$. Since f_1 is differentiable, it is easy to note that, $g^{f_1}(1) = 1$. Therefore, $g^f(1) = 1$. Similarly, at $x = 0$ we have $f_1(0) = f_2(0) = 0$. Since both f_1 and f_2 are active at $x = 0$, as per the weak calculus rule, one can take both the (sub)gradient of f_1, being $g^{f_1}(0) = 1$, or the (sub)gradient of f_2, being $g^{f_2}(0) = -1$, as the subgradient of f at $x = 0$ ∎

Some further rules on weak and strong subgradient calculus can be found, e.g., in [9].

2.3 Subgradient and inexact subgradient methods

In the case of convex, not necessarily differentiable f, one can utilise subgradient-based iterative methods of the form [10]

$$x^{k+1} = x^k - \alpha^k g^k, \tag{2.7}$$

$k = 0, 1, \ldots$, where $\alpha_k \geq 0$ is the step-size and g^k is any subgradient of f at x_k. The iterative algorithm is usually initialized at an arbitrary point $x^0 \in \mathbb{R}^d$. We will focus mainly on the constant step-size choice, $\alpha_k = \alpha > 0$. See, e.g., [10], for several alternative step-size choices. In (2.7), an (exact) subgradient can be replaced with an ϵ-subgradient (an inexact subgradient), yielding an inexact subgradient method. Under certain conditions, inexact subgradient methods are close in performance to exact subgradient methods, as described ahead. As considered in Chapter 5, inexact (centralized) subgradient methods are very useful in the analysis of distributed subgradient

methods.

In the following subsections, we will state the conditions and assumptions needed for algorithms of type (2.7) to converge, as well as perform the convergence analysis for various step-size choices.

2.3.1 Optimality conditions for unconstrained minimization

Recall the problem (2.1). For a convex, differentiable f, it is a well known fact, e.g., [8], that, for a solvable problem of unconstrained minimization of function f, the following holds.

$$\nabla f(x^*) = 0 \iff \inf_{x \in \mathbb{R}^d} f(x) = f(x^*). \tag{2.8}$$

When f is convex, but not necessarily differentiable, the equivalence (2.8) becomes

$$0 \in \partial f(x^*) \iff \inf_{x \in \mathbb{R}^d} f(x) = f(x^*). \tag{2.9}$$

The goal of iterative algorithms of type (2.7) is to converge to a point x^* (or its neighborhood) that satisfies (2.9).

2.3.2 Analysis of exact and inexact subgradient methods

A set of assumptions commonly used for the convergence of subgradient-based methods is as follows [10]:

1. **Existence of a minimum**:

$$f^* := \inf_{x \in \mathbb{R}^d} f(x) > -\infty.$$

2. **Existence of a minimizer**:

$$f^* = f(x^*),$$

for some $x^* \in \mathbb{R}^d$.

3. **Bounded subgradients**:

   There exists a constant $G \in [0, +\infty)$ such that

$$\|g^f(x)\| \leq G, \text{ for all } g^f(x) \in \partial f(x),$$

for all $x \in \mathbb{R}^d$.

4. **Bounded distance from the solution set**:

$$\|x^0 - x^*\| \leq R, \text{ for all } x^* \in X^* = \{y^* : f(y^*) = f^*\},$$

for some constant $R \in [0, +\infty)$.

Define f_{best}^k as

$$f_{best}^k = \min_{i=0,\dots,k} f(x^i), \tag{2.10}$$

and let $\bar{f} \coloneqq \lim_{k\to\infty} f_{best}^k$. It can be shown that (see [10] for the derivation):

$$\|x^{k+1} - x^*\|^2 \leq \|x^k - x^*\|^2 + 2\alpha^k(f^* - f(x^k)) + \|\alpha^k g^k\|^2, \tag{2.11}$$

with x^* being any solution, and $f^* = f(x^*)$. The latter inequality can be shown to yield:

$$f_{best}^k - f^* \leq \frac{\|x^0 - x^*\|^2 + \sum_{i=0}^k (\alpha^i)^2 \|g^i\|^2}{2\sum_{i=0}^k \alpha^i}. \tag{2.12}$$

Next, consider the constant step size $\alpha^k = \alpha > 0$. Using the bounded subgradients and distance from the solution set assumptions in (2.12), we get

$$f_{best}^k - f^* \leq \frac{R^2 + \alpha^2 G^2 (k+1)}{2\alpha(k+1)} = \frac{R^2}{2\alpha(k+1)} + \frac{\alpha G^2}{2}. \tag{2.13}$$

From (2.13), we can conclude that

$$\lim_{k\to\infty} (f_{best}^k - f^*) \leq \frac{\alpha G^2}{2}, \tag{2.14}$$

or that, using a fixed step-size, the subgradient method converges to a fixed neighborhood of the solution.

Remark. Recall the inequality (2.11). We can interpret it as follows: assuming the step-size α^k is small, a crude approximation of (2.11) is the following

$$\|x^{k+1} - x^*\|^2 \leq \|x^k - x^*\|^2 + 2\alpha^k(f^* - f(x^k)).$$

This roughly means that, if we are sufficiently away from the solution set, the distance of the iterate to any solution is decreasing.

Remark. Note that inequality (2.14)depends on α. It is possible to converge to an arbitrary neighborhood of a solution, by choosing a sufficiently small

α. However, this comes at the cost of slower convergence of the algorithm towards its plateau.

Finally, when we replace the exact subgradient with an ϵ-subgradient in the constant step size subgradient method, it can be shown that (2.13) changes into (see, e.g., [11], for a related analysis):

$$f_{best}^k - f^* \leq \frac{R^2}{2\alpha(k+1)} + \frac{\alpha G^2}{2} + O(\epsilon), \tag{2.15}$$

where $O(\cdot)$ stands for the "big-O" notation.

The main sources used for writing this chapter are [9, 10, 11].

Chapter 3

Duality

This Chapter introduces basic concepts from the (Lagrangian) duality theory, including some standard weak and strong duality results. The topic is covered, e.g., in [3, 2, 12, 4]. Here, we follow closely [4].

3.1 Introduction

The (Lagrangian) duality theory is very useful in developing dual decomposition and dual distributed methods, as studied in Chapter 4. In this Chapter, we provide a review of basic concepts from the duality theory.

We first review basic concepts regarding formulation of constrained optimization problems. Consider a problem of the form

$$\begin{array}{l} \min f(x) \\ h_i(x) = 0,\ i = 1, \dots, p \\ g_j(x) \leq 0,\ j = 1, \dots, m \\ x \in \mathcal{X} \end{array}, \tag{3.1}$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}$ is called the *objective (cost)* function, $h_i : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, \dots, p$ are called the *equality constraints*, $g_j : \mathbb{R}^d \mapsto \mathbb{R}$, $j = 1, \dots, m$ are called the *inequality constraints*, while the set $\mathcal{X}$ is a non-empty, closed set. We define

$$\begin{aligned} h(x) &:= \big(h_1(x), \dots, h_p(x)\big)^T \\ g(x) &:= \big(g_1(x), \dots, g_m(x)\big)^T \end{aligned}.$$

It is known that the optimization problem (3.1) is convex if the objective

function f is convex, and the constraint set

$$\mathcal{S} = \left\{x \in \mathbb{R}^d : h_i(x) = 0,\ i = 1, \ldots, p,\ g_j(x) \leq 0,\ j = 1, \ldots, m,\ x \in \mathcal{X}\right\}$$

is a convex set [1]. Note that $\mathcal{S}$ can also be written as

$$\mathcal{S} = \bigcap_{i=1}^{p} \left\{x \in \mathbb{R}^d : h_i(x) = 0\right\} \bigcap \left(\bigcap_{j=1}^{m} \left\{x \in \mathbb{R}^d : g_j(x) \leq 0\right\}\right) \bigcap \mathcal{X}.$$

Recall that the set $\mathcal{S}$ is called the *feasible set*. We next analyze the convexity of $\mathcal{S}$.

To begin with, we analyze the set $\{x \in \mathbb{R}^d : g(x) \leq 0\}$, where g is an arbitrary convex function. Take any two points $x, y \in \mathbb{R}^d$, such that $g(x) \leq$ and $g(y) \leq 0$, and an arbitrary constant $\theta \in [0, 1]$. Then

$$g(\theta x + (1-\theta)y) \leq \theta g(x) + (1-\theta)g(y) \leq \big(\theta + (1-\theta)\big)0 = 0, \tag{3.2}$$

where the first inequality comes from the convexity of g, while the second comes from the fact that both x and y satisfy $g(\cdot) \leq 0$. From (3.2), we can conclude that $\theta x + (1-\theta)y \in \{x \in \mathbb{R}^d : g(x) \leq 0\}$, therefore, the said set is convex, for a convex g.

Next, consider the set $\left\{x \in \mathbb{R}^d : h(x) = 0\right\}$. For example, choosing h to be $h(x) = \|x\|^2 - 1$, one can show that the set $\left\{x \in \mathbb{R}^d : h(x) = 0\right\}$ does not have to be convex, even if h is a convex function (exercise). The proof of the following proposition is left for exercise.

Proposition 3.1. *Consider the set* $\mathcal{H} = \left\{x \in \mathbb{R}^d : h(x) = 0\right\}$. *Then, if* h *is an affine function,* $\mathcal{H}$ *is a convex set.*

From the analysis above, we can conclude that the problem (3.1) is a convex problem if:

1. f is a convex function,
2. h_i is an affine function, for all i,
3. g_j is a convex function, for all j,
4. $\mathcal{X}$ is a convex set.

In terms of the duality theory, problems of the form (3.1) are called *primal* problems. Consider a primal problem of the form (3.1), not necessarily convex. We define the *Lagrangian function* in the following way.

Definition 3.1. For a primal problem of the form (3.1), the Lagrangian function $\mathcal{L} : \mathbb{R}^d \times \mathbb{R}^p \times \mathbb{R}^m \mapsto \mathbb{R}$ is given by

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu^T g(x). \tag{3.3}$$

Using the Lagrangian function (3.3) of the primal problem (3.1), one can define the *dual function* as follows.

Definition 3.2. Consider the Lagrangian function (3.3) of the problem (3.1). The dual function $\mathcal{D} : \mathbb{R}^p \times \mathbb{R}^m \mapsto \mathbb{R}$ is given by

$$\mathcal{D}(\lambda, \mu) = \inf_{x \in \mathcal{X}} \mathcal{L}(x, \lambda, \mu). \tag{3.4}$$

Using the dual function (3.4), one can formulate the *dual problem* of (3.1), as follows.

Definition 3.3. Given the primal problem (3.1), one can formulate the dual problem as

$$\begin{array}{c} \max\limits_{\lambda \in \mathbb{R}^p, \mu \in \mathbb{R}^m} \mathcal{D}(\lambda, \mu) \\ \text{s.t.} \quad \mu \geq 0 \end{array}, \tag{3.5}$$

where the inequality $\mu \geq 0$ is point-wise ($\mu_i \geq 0$, for all $i = 1, \ldots, m$).

The following proposition states the concavity property of the dual function (3.4).

Proposition 3.2. *The dual function (3.4) is concave.*

Proof. Recall that the dual function is given by

$$\mathcal{D}(\lambda, \mu) = \inf_{x \in \mathcal{X}} \mathcal{L}(x, \lambda, \mu) = \inf_{x \in \mathcal{X}} \left\{ f(x) + \lambda^T h(x) + \mu^T g(x) \right\}.$$

Define $\phi_x(\lambda, \mu)$ as

$$\phi_x(\lambda, \mu) = f(x) + \lambda^T h(x) + \mu^T g(x).$$

It is easy to observe that $\phi_x(\lambda, \mu)$ is affine in (λ, μ). Therefore, the dual function

$$\mathcal{D}(\lambda, \mu) = \inf_{x \in \mathcal{X}} \left\{ \phi_x(\lambda, \mu) \right\}$$

represents a point-wise infimum of affine functions. Recalling the pointwise supremum rule for convexity – see, e.g., [9], one can conclude that $\mathcal{D}$ is a concave function □

Remark. From proposition 3.2 and the definition of the dual problem (3.5), we can conclude that the dual problem is always convex, regardless of convexity of (3.1).

3.2 Weak Duality

Recall the primal and dual problems, (3.1) and (3.5) respectively. Weak duality characterizes the relationship between the primal and the dual problems. To begin with, we define *primal* and *dual* feasibility.

Definition 3.4. For a point $x \in \mathbb{R}^d$, we say it is primaly feasible with respect to (3.1), if it satisfies the imposed constraints, i.e. it holds that $h(x) = 0$, $g(x) \leq 0$, $x \in \mathcal{X}$.

Definition 3.5. For a point $\mu \in \mathbb{R}^m$, we say it is dualy feasible with respect to (3.5), if it satisfies the imposed constraints, i.e. it holds that $\mu \geq 0$.

Theorem 3.3. *For any $x \in \mathbb{R}^d$ that is primal feasible ($h(x) = 0$, $g(x) \leq 0$, $x \in \mathcal{X}$) and for any $\lambda \in \mathbb{R}^p$ and any $\mu \in \mathbb{R}^m$ that are dual feasible ($\mu \geq 0$), it holds:*

1. $\mathcal{D}(\lambda, \mu) \leq f(x)$,
2. $d^* = \sup_{\{(\lambda,\mu)\in\mathbb{R}^p\times\mathbb{R}^m:\mu\geq 0\}} \mathcal{D}(\lambda, \mu) \leq p^* = \inf_{\{x\in\mathbb{R}^d: h(x)=0,\, g(x)=0,\, x\in\mathcal{X}\}} f(x)$.

Proof. We first prove the statement *1.* Pick any dual feasible point (λ, μ). We have

$$\mathcal{D}(\lambda, \mu) = \inf_{x\in\mathcal{X}} \left\{ f(x) + \lambda^T h(x) + \mu^T g(x) \right\} \leq f(x) + \lambda^T h(x) + \mu^T g(x). \quad (3.6)$$

Recall that x is a primal feasible point. From there, it directly follows that $h(x) = 0$ and $g(x) \leq 0$. Combining this fact with the fact that (λ, μ) is dual feasible, it follows that $\mu^T g(x) = \mu_1 g_1(x) + \ldots \mu_m g_m(x) \leq 0$, as each component is a product of a positive and a negative number. Using these facts in (3.6), we can conclude that

$$\mathcal{D}(\lambda, \mu) \leq f(x), \quad (3.7)$$

which is exactly what we wanted to prove. For the statement *2.*, we use the claim that was just proved. Observe (3.7), and take the supremum over all

the dual feasible points over the left-hand side as well as the infimum over all the primal feasible points over the right-hand side. We get

$$\sup_{\{(\lambda,\mu)\in\mathbb{R}^p\times\mathbb{R}^m:\mu\geq 0\}} \mathcal{D}(\lambda,\mu) \leq \inf_{\{x\in\mathbb{R}^d: h(x)=0,\, g(x)=0,\, x\in\mathcal{X}\}} f(x),$$

which is exactly what we wanted to prove □

Remark. Note from the previous proposition that $p^* - d^* \geq 0$. We call the quantity $p^* - d^*$ the *duality gap*. The duality gap is a measure of how well the dual problem "approximate" the primal problem. If the duality gap is zero, we can use the dual problem to obtain the exact optimal value of the primal problem.

3.3 Strong Duality

Informally, strong duality specifies a set of conditions for which we can recover the solution of the primal problem (3.1) by solving the dual problem (3.5). Recall the duality gap, $p^* - d^*$. If the duality gap is zero, and some additional conditions are satisfied, the primal solution can be reconstructed by solving the dual problem. In what follows, the conditions for which strong duality holds will be characterized more formally.

We consider convex problems of a general form, where the affine eqaulity constraints are subsumed within the overall constraint set, under the inequality constraints (obviously, one can always represent an equality by a non-strict inequality). The problems that we consider in this section are given by

$$\begin{aligned} &\min f(x) \\ &Ax \leq b \quad , \\ &x \in \mathcal{X} \end{aligned} \tag{3.8}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a convex function, $\mathcal{X}$ is a closed, convex set, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ and the inequality is element-wise. The Lagrangian of (3.8) is given by

$$\mathcal{L}(x,\mu) = f(x) + \mu^T(Ax - b),$$

while the dual function is given by

$$\mathcal{D}(\mu) = \min_{x\in\mathcal{X}} f(x) + \mu^T(Ax - b).$$

We can therefore formulate the dual problem as

$$\max_{\{\mu \in \mathbb{R}^m : \mu \geq 0\}} \mathcal{D}(\mu). \tag{3.9}$$

Before we formally state the conditions for strong duality, and the properties of thereof, some definitions are in order.

Definition 3.6. For a point $x \in \mathbb{R}^d$, a ball centered at x, with a radius $\epsilon > 0$ is given by

$$B(x, \epsilon) = \{y \in \mathbb{R}^d : \|x - y\| < \epsilon\},$$

where $\|\cdot\|$ is any norm on $\mathbb{R}^d$.

Remark. In general, we will work with L2-defined balls, unless stated otherwise.

Definition 3.7. An *affine hull* of a given set $S \subseteq \mathbb{R}^d$ is the set

$$\text{aff}(S) = \left\{\alpha_1 x_1 + \ldots + \alpha_r x_r : \alpha_i \in \mathbb{R},\ x_i \in S,\ i = 1, \ldots, r,\ \sum_{i=1}^{r} \alpha_i = 1,\ r = 1, 2, \ldots\right\}. \tag{3.10}$$

Definition 3.8. The *relative interior* of a set $S \subseteq \mathbb{R}^d$ is the set

$$rel.int.(S) = \{x \in S : \exists \epsilon > 0 \ s.t.\ B(x, \epsilon) \bigcap aff(S) \subseteq S\}.$$

A key condition for strong duality to hold is the *Slater's condition*. Recall the problem (3.8). The condition is stated as follows.

Slater's condition. There exists a point $x_0 \in rel.int.(\mathcal{X})$ such that $Ax_0 \leq b$.

Remark. When $\mathcal{X} \equiv \mathbb{R}^n$, it holds that $rel.int.(\mathcal{X}) \equiv \mathbb{R}^n$ (exercise). In this case, the Slater's condition reduces to the following: there exists a point $x_0 \in \mathbb{R}^n$ such that $Ax_0 \leq b$. In other words, the Slater's condition requires the existance of (at least one) feasible point in this case.

We are now ready to state the strong duality claim, e.g., [4]. .

Claim 3.4. *Consider the problem (3.8). Assume that the Slater's condition holds. Then, the following is true:*

1. $p^* = d^*$ *(zero duality gap)*
2. *Moreover, if* $p^* = d^* \in (-\infty, +\infty)$ *then the dual problem (3.9) is solvable, that is, there exists a* $\mu^* \geq 0$*, such that* $d^* = \sup_{\mu \geq 0} \mathcal{D}(\mu) = \mathcal{D}(\mu^*)$.

Recall the (strict) convexity definition (1.1). If a function is strictly convex, the following claim holds, e.g., [4].

Claim 3.5. *If, in addition to the conditions from the previous claim, f is strictly convex, then*

$$x^* = \arg\min_{x \in \mathcal{X}} \left\{ f(x) + (\mu^*)^T (Ax - b) \right\},$$

where $\mu^* \geq 0$ is dual solution.

3.4 Dual subgradient method

Consider the problem

$$\begin{aligned} &\min f(x) \\ &Ax = b \quad , \\ &x \in \mathcal{X} \end{aligned} \tag{3.11}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a strictly convex function, $\mathcal{X}$ is a compact, convex set, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and the strong duality is assumed to hold true. We can define the Lagrangian of (3.11) as

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^T (Ax - b),$$

the dual function as

$$\mathcal{D}(\lambda) = \min_{x \in \mathcal{X}} \left\{ f(x) + \lambda^T (Ax - b) \right\},$$

and state the dual problem as

$$\max_{\lambda \in \mathbb{R}^m} \mathcal{D}(\lambda). \tag{3.12}$$

Recall that $\mathcal{D}$ is a concave function, and that the problem (3.12) is equivalent to

$$\min_{\lambda \in \mathbb{R}^m} -\mathcal{D}(\lambda). \tag{3.13}$$

Since $-\mathcal{D}$ is a convex function, we can apply the subgradient method (2.7) to solve the problem (3.13). Denoting $-g^k \in \partial(-\mathcal{D}(\lambda^k))$, we get the following rule for solving (3.13) (and consequently (3.12) as well)

$$\lambda^{k+1} = \lambda^k + \alpha g^k.$$

We are left with the question of how to calculate a subgradient of $-\mathcal{D}$. For a fixed λ^k we need to find an "active" function, that is

$$x^*(\lambda^k) = x^{k+1} = \arg\min_{x \in \mathcal{X}} \left\{ f(x) + (\lambda^k)^T (Ax - b) \right\}.$$

Then, we can calculate the subgradient as $g^k = Ax^{k+1} - b$.

In summary, the dual subgradient method works as follows. For $k = 0, 1, 2, \ldots$

$$\begin{aligned} x^{k+1} &= \arg\min_{x \in \mathcal{X}} \left\{ f(x) + (\lambda^k)^T (Ax - b) \right\} \\ \lambda^{k+1} &= \lambda^k + \alpha (Ax^{k+1} - b) \end{aligned} . \tag{3.14}$$

In many cases, it is easier to apply the dual subgradient method to solve (3.12), rather than solve (3.11) directly.

The main sources for writing this chapter are [4, 1, 2].

Chapter 4

Dual distributed methods

This Chapter introduces a commonly studied distributed optimization model, as well as some commonly studied corresponding communication models, and it presents a dual subgradient method for the studied setup. The Chapter considers commonly used models, techniques, and results, see, e.g., [12, 14, 15, 16, 17], for related more advanced materials. We also provide some application examples for distributed optimization in Section 4.4. This Section is mainly taken practically unaltered from the PhD thesis [18].

4.1 Introduction

So far, we assumed that all the components of the optimization problem (e.g., objective functions, constraints) are available at a single location. For instance, consider the least squares example (1.2.2). In terms of the problem components, we have the matrix A and the target vector b. In a standard machine learning setting, the matrix A would be the data matrix, whereas the vector b would be the vector of target values, i.e., the values that we want the model to learn.

In the era of Big data, it is fairly common for the data to be gathered and stored at multiple points (e.g. sensors, computers), rather than having a single storage and computation point. Various examples where this scenario is relevant include large scale machine learning [19], [20], [21], distributed inference in sensor networks [22], as well as distributed target localization, spectrum cartography in cognitive radio (CR) networks, flow control in communication networks [23], to name a few. This, as well as the structure of problems (e.g. in large-scale machine learning) has given rise to distributed computational models, where distributed optimization plays an important role. It is worth noting that distributed computation and optimization has

a long history; see, e.g., [12].

4.2 Computation model

Recall that a standard optimization model is of the form

$$\begin{aligned} &\min f(x) \\ &h_i(x) = 0,\ i = 1, \dots, p \\ &g_j(x) \le 0,\ j = 1, \dots, q \\ &x \in \mathcal{X} \end{aligned}.$$

A problem that is suitable for decomposition, i.e. distributed computing, has the following form

$$\min_{x \in \mathbb{R}^d} f_1(x) + f_2(x) + \ldots + f_N(x), \tag{4.1}$$

where $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, f_i strictly convex, for all $i = 1, \dots, N$.

Remark. A condition that is milder can be introduced, while the analysis here would continue to hold. Instead of requiring that all f_i's be strictly convex, it suffices that all f_i's are convex, while at least one of them is strictly convex.

The cost function of the form (4.1) appears in many applications (many of them listed above), e.g., in machine learning.

Example 4.2.1. Least squares problem:

Recall the least squares example (1.2.2). Assume that the data matrix $A \in \mathbb{R}^{Nm \times d}$ has the following structure

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_N \end{bmatrix},$$

where each $A_i \in \mathbb{R}^m$, $i = 1, \dots, N$, and similarly, the target vector $b \in \mathbb{R}^{Nm}$ has the following structure

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix},$$

where each $b_i \in \mathbb{R}^m$. Denote the cost function as $f : \mathbb{R}^d \mapsto \mathbb{R}$, and recall its form, $f(x) = \|Ax - b\|^2$.

Using the specific structure of the data matrix and the target vector, we can write the cost function as follows

$$f(x) = \|Ax - b\|^2 = \| \begin{bmatrix} A_1 x - b_1 \\ A_2 x - b_2 \\ \vdots \\ A_N x - b_2 \end{bmatrix} \|^2 = \sum_{i=1}^{N} \|A_i x - b_i\|^2, \tag{4.2}$$

where the third equality stems from the structure of the data matrix and target vector, and the definition of the L2 norm (exercise). If we define $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ as $f_i(x) = \|A_i x - b_i\|^2$, it is clear that we can represent (4.2) as a sum of N functions:

$$f(x) = \|Ax - b\|^2 = \sum_{i=1}^{N} \|A_i x - b_i\|^2 = \sum_{i=1}^{N} f_i(x) \blacksquare$$

4.3 Communication models

In order to capitalize on the specific structure of the data, as well as the cost function, communication models that define how distributed computation and information propagation is performed need to be introduced.

To begin with, in distributed optimization, a communication model is often represented as a graph[1] $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices (representing nodes or agents in our model), while $\mathcal{E}$ is the set of edges (representing communication links in our model). An obvious requirement for all the communication models is that the underlying graph is connected. This condition stems, e.g., from the information propagation and consensus requirements, to be discussed soon.

Some common communication models in distributed optimization settings are the following:

- **Master-worker model**. The master-worker framework is represented by an underlying star graph. In short, the *master node* is the central node, connected to all the other nodes. It usually holds a copy of the solution to the problem of interest, performs updates, and in some cases

[1]Depending on the settings, the underlying graph can be both directed and undirected. In this book, we will consider only the communication models modeled as undirected graphs; fore more on distributed optimization over directed graphs, the reader is refered to [24], [25], [26] and the references therein.

has a chunk of the data and performs a part of the computation itself. *Worker nodes* are the nodes where the data is stored. At each iteration, they receive the current solution estimate from the master node, and their main goal is to perform computation (e.g., compute the gradient) based on the solution estimate, and send the information back to the master node, where the updates are performed.

- **Fully distributed model**. This communication model is represented by a generic connected graph, such that usually no vertex is connected to all the other vertices (i.e., no agent can communicate with all the other agents in the network). Each agent can communicate only with its direct neighbors. Each agent contains a chunk of the data. The goal of each agent is to iteratively minimize the global cost function based on its local data, while simultaneously achieving consensus on the solution estimate with other agents in the network. The graph may be undirected or directed, while here we study the undirected graphs case.

4.4 Application examples

We now give several examples of problem (6.1) relevant in applications.

Example 1: Consensus. We explain consensus in the context of sensor networks, but many other contexts, e.g., social networks, are possible. Let N be deployed in a field; each sensor acquires a scalar measurement d_i, e.g., temperature at its location. The goal is for each sensor to compute the average temperature in the field: $\frac{1}{N}\sum_{i=1}^{N} d_i$. Consensus can be cast as (6.1) by setting $f_i(x) = \frac{1}{2}(x - d_i)^2$.

Example 2: Distributed learning: linear classifier. For concreteness, we focus on linear classification, but other distributed learning problems fit naturally (6.1). Training data (e.g., data about patients, as illustrated in [15]) is distributed across agents in the network (different hospitals); each agent has N_s data samples, $\{a_{ij}, b_{ij}\}_{j=1}^{N_s}$, where $a_{ij} \in \mathbb{R}^m$ is a feature vector (patient signature – blood pressure, etc) and $b_{ij} \in \{-1, +1\}$ is the class label of the vector a_{ij} (patient healthy or ill). For the purpose of future feature vector classifications, each agent wants to learn the linear classifier $a \mapsto \operatorname{sign}(a^\top x' + x'')$, i.e., to determine a vector $x' \in \mathbb{R}^m$ and a scalar $x'' \in \mathbb{R}$, based on all agents' data samples, that makes the best classification in a certain sense. Specifically, we seek $x' \in \mathbb{R}^m$ and $x'' \in \mathbb{R}$ that minimize a convex

surrogate loss with respect to $x = ((x')^\top, x'')^\top$:

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \lambda R(x') \ . \tag{4.3}$$

Here $\phi(z)$ is a surrogate loss function, e.g., logistic, exponential, or hinge loss [27], $\lambda > 0$ is a parameter, and $R(x')$ is the regularization, e.g., l_1 norm or quadratic. Problem (4.3) fits (6.1), with $f_i(x) := \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \frac{\lambda}{N} R(x')$.

Example 3: Acoustic source localization in sensor networks. A sensor network instruments the environment where an acoustic source is positioned at an unknown location $\theta \in \mathbb{R}^2$, e.g. [28]. The source emits signal isotropically. Each agent (sensor) i measures the received signal energy:

$$y_i = \frac{A}{\|\theta - r_i\|^\beta} + \zeta_i.$$

Here $r_i \in \mathbb{R}^2$ is agent i's location, known to agent i, $A > 0$ and $\beta > 0$ are constants known to all agents, and ζ_i is zero-mean additive noise. The goal is for each agent to estimate the source's position θ. A straightforward approach is to find the nonlinear least squares estimate $\overline{\theta} = x^\star$ by minimizing the following cost function (of the variable x):

$$\text{minimize} \quad \sum_{i=1}^{N} \left(y_i - \frac{A}{\|x - r_i\|^\beta}\right)^2 \ . \tag{4.4}$$

Problem (4.4) is nonconvex and hence difficult; still, it is possible to efficiently obtain a good estimator $\widehat{\theta}$ based on the data y_i, $i = 1, ..., N$, by solving the following *convex* problem:

$$\text{minimize} \quad \frac{1}{2} \sum_{i=1}^{N} \text{dist}^2\,(x, C_i) \ , \tag{4.5}$$

where C_i is the disk $C_i = \left\{x \in \mathbb{R}^2 : \|x - r_i\| \le \left(\frac{A}{y_i}\right)^{1/\beta}\right\}$, and $\text{dist}(x, C) = \inf_{y \in C} \|x - y\|$ is the distance from x to the set C. In words, (4.6) finds a point $\widehat{\theta}$ that has the minimal total squared distance from disks C_i, $i = 1, ..., N$. Problem (4.6) fits our framework (6.1) with $f_i(x) = \frac{1}{2}\text{dist}^2\,(x, C_i)$.

Example 4: Resource allocation in cognitive radio networks. Consider a set of secondary users (cognitive radios CRs), connected in a generic network topology, that want to allocate their resources (e.g., frequency subchannels) by minimizing the interference towards the primary, licenced users. Each CR i wants to allocate its resource $x_i \in \mathbb{R}^d$ (e.g., $d = 1$ and x_i is the frequency of a subchannel), so that the following three criteria

are met: (1) minimize the interference that it experiences; (2) avoid collisions in the resource domain with the neighboring CRs; and (3) the overall system should avoid excessive spread of the total resources used by all CRs. The resource allocation problem can thus be modeled as the following optimization problem:

$$\text{minimize} \quad \sum_{i=1}^{N} J_i(x_i) + \sum_{\{i,j\}\in E} a_{ij}(x_i - x_j)^2 - \sum_{\{i,j\}\in E} r_{ij} \log\left((x_i - x_j)^2\right) . \tag{4.6}$$

The term $J_i(x_i)$ is the overall interference power perceived by CR i; the term $-r_{ij}\log\left((x_i - x_j)^2\right)$ (with $r_{ij} > 0$) is the repulsion term that penalizes collision in the resource domain among the neighboring CRs i and j; and $a_{ij}(x_i - x_j)^2$ ($a_{ij} > 0$) is the attraction term that prevents from an excessive spread of resources (too distant x_i and x_j.) The described resource allocation mechanism has been proposed in [29] and is inspired by animal swarms. Denote by $x = (x_1, ..., x_N)^\top$. Problem (4.6) fits our framework with $f_i(x) = J_i(x_i) + \sum_{j\in O_i} \frac{a_{ij}}{2}(x_i - x_j)^2 - \frac{r_{ij}}{2}\log\left((x_i - x_j)^2\right)$, though the problem is nonconvex.

Example 5: Spectrum sensing for cognitive radio networks. Consider N secondary users (CRs) connected by a generic network. The CRs sense the power spectral density (PSD) to reconstruct the PSD map of primary users (PUs), i.e., the CRs want to determine at what physical locations the PUs are present, and what frequencies they use; this example is studied in [30]. The model assumes N_p potential locations (a grid) of PUs; each "potential" PU p has a power spectral density (PSD) expressed as:

$$\Phi_p(f) = \sum_{b=1}^{N_b} \theta_{bp}\Psi_b(f),$$

where f is the frequency, $\Psi_b(f)$ is rectangle over interval b and θ_{bp} is a coefficient that says how much PSD is generated by the pth (potential) PU in the frequency range b. The PSD at CR i is modeled as a superposition of all potential PU's PSDs:

$$\Phi_i(f) = \sum_{p=1}^{N_p} g_{ip}\Phi_p(f) = \sum_{p=1}^{N_p} g_{ip} \sum_{b=1}^{N_b} \theta_{bp}\Psi_b(f), \tag{4.7}$$

where g_{ip} is the channel gain between PU p and CR i. Denote by θ the vector that stacks all the θ_{bp}'s. Each CR collects samples at frequencies f_l,

$l = 1, ..., L$, modeled as:

$$y_{i,l} = \Phi_i(f_l) + \zeta_{i,l} = h_{i,l}(\theta) + \zeta_{i,l},$$

where $\zeta_{i,l}$ is a zero-mean additive noise, and $h_{i,l}(\theta)$ is a linear function of θ. Reference [30] assumes that most of the coefficients θ_{bp} are zero, i.e., the vector θ is sparse. Hence, the spectrum sensing problem of determining the vector $x^\star$ – an estimate of θ – is:

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{l=1}^{L} (h_{i,l}(x) - y_{i,l})^2 + \lambda \|x\|_1. \tag{4.8}$$

In framework (6.1), we now have $f_i(x) = \sum_{l=1}^{L} (h_{i,l}(x) - y_{i,l})^2 + \frac{\lambda}{N} \|x\|_1$.

Example 6: Distributed detection in sensor networks. Consider N agents that face the binary hypothesis test H_1 versus H_0, e.g., [31]. Each agent i, at each time k, acquires a sample $y_i(k)$, where, for $i = 1, 2, ..., N$:

$$H_1 : \; y_i(k) = m_i + \zeta_i(k), \qquad H_0 : \; y_i(k) = \zeta_i(k). \tag{4.9}$$

Here, m_i is a deterministic, time invariant signal, and $\zeta_i(k)$ is a spatio-temporally i.i.d. zero-mean noise. Consider a hypothetical fusion center that collects the measurements $y_i(k)$ from all agents i, at all times k. The optimal centralized detector (that minimizes the Bayes error probability) is the log-likelihood ratio test:

$$D(k) := \frac{1}{N\,k} \sum_{t=1}^{k} \sum_{i=1}^{N} L_i(t) \underset{H_0}{\overset{H_1}{\gtrless}} \gamma_k, \tag{4.10}$$

where γ_k is the test threshold, and $L_i(t)$ is the log likelihood ratio based on the agent i's sample $y_i(t)$ at time t. We now give an optimization perspective in the sense of (6.1) to describe the detection problem. The recursive update of the centralized detector's decision variable $D(k)$ can be written as:

$$D(k+1) = D(k) - \frac{1}{k+1} \left(\frac{1}{N} \sum_{i=1}^{N} (D(k) - L_i(k+1)) \right), \tag{4.11}$$

which is a stochastic gradient algorithm with step-size $1/(k+1)$ to solve the following unconstrained quadratic stochastic optimization problem:

$$\text{minimize} \;\; f(x) := \frac{1}{2} \sum_{i=1}^{N} \mathbb{E}\left[(x - L_i(k))^2 | H_l \right]. \tag{4.12}$$

Namely, the term $\left(\frac{1}{N}\sum_{i=1}^{N}(x - L_i(k+1))\right)$ is an instantaneous, stochastic approximation of the gradient $\nabla f(x) = \sum_{i=1}^{N}\mathbb{E}[x - L_i(k)|H_l]$. Indeed, as is well-known, under H_l, $l = 0, 1$, $D(k)$ converges almost surely to $x^\star := \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[L_i(k)|H_l]$ – the solution to (4.12). Note that (4.12) is an instance of (6.1), with $f_i(x) := \frac{1}{2}\mathbb{E}[(x - L_i(k))^2|H_l]$ (assuming that hypothesis H_l is true.)

The major part of this Section is taken practically unaltered from the PhD thesis [18].

4.5 Dual distributed methods

In this section we analyze distributed methods for the problems of the form (4.1), for the previously introduced communication models.

We start with the master-worker framework. Each node $i \in \{1, \ldots, N\}$ is assigned exactly one f_i. First, the problem (4.1) is reformulated as follows; see, e.g., [12, 4]. Each agent is assigned its own variable, x_i, $i = 1, \ldots, N$. We define $x = (x_1^T, \ldots, x_N^T)^T \in \mathbb{R}^{Nd}$ and $f : \mathbb{R}^{Nd} \mapsto \mathbb{R}$, given by $f(x) = \sum_{i=1}^{N} f_i(x_i)$. The problem can then be stated as

$$\min_{x_1=\ldots=x_N} \sum_{i=1}^{N} f_i(x_i). \tag{4.13}$$

Note that the feasible set of (4.13) is $S = \{(x^T, \ldots, x^T)^T : x \in \mathbb{R}^d\}$. This constraint is the *consensus* constraint, where we require all the agents to agree on a solution, since the goal of the distributed system is that agents collaboratively solve the joint problem.

Also note that the problems (4.1) and (4.13) are equivalent in the following sense. If $x^* \in \mathbb{R}^d$ is the solution to (4.1), then $(x^{*T}, \ldots, x^{*T})^T$ is the solution to (4.13). Conversely, if $\hat{x} = (\hat{x}_1^T, \ldots, \hat{x}_N^T)^T \in \mathbb{R}^{Nd}$ is the solution to (4.13), then $\hat{x}_1 = \ldots = \hat{x}_N$, and $\hat{x}_1$ is the solution to (4.1). We next reformulate (4.13) to an equivalent form, given by

$$\begin{array}{l} \min \sum_{i=1}^{N} f_i(x_i) \\ x_1 = x_2 \\ x_1 = x_3 \\ \quad \vdots \\ x_1 = x_N \end{array}, \tag{4.14}$$

and proceed to dualize the constraints, by associating the dual variable λ_{1j} with the constraint $x_1 = x_j$, $j = 2, \ldots, N$. Note that the problem (4.14) is convex, as we assumed f_i's are strictly convex, and the equality constraints are affine. Since the Slater's condition is satisfied (exercise), strong duality holds as well.

Next, we form the Lagrangian of (4.14). The Lagrangian is formed as

$$\mathcal{L}(x_1, \ldots, x_N, \lambda_{12}, \ldots, \lambda_{1N}) = \sum_{i=1}^{N} f_i(x_i) + \lambda_{12}^T(x_1 - x_2) + \ldots + \lambda_{1N}^T(x_1 - x_N). \quad (4.15)$$

The dual subgradient method (3.14) can now be applied:

- Primal update:

$$x^{k+1} = (x_1^{k+1}, \ldots, x_N^{k+1}) = \underset{(x_1, \ldots, x_N)}{\arg\min}\, \mathcal{L}(x_1, \ldots, x_N, \lambda_{12}, \ldots, \lambda_{1N}).$$

- Dual update:

$$\lambda^{k+1} = (\lambda_{12}^{k+1}, \ldots, \lambda_{1N}^{k+1}) = \lambda^k + \alpha \nabla_\lambda \mathcal{L}(x^{k+1}, \lambda^k),$$

  where the subgradient with respect to λ is calculated at point λ^k, when x is fixed, and equal to x^{k+1}.

Grouping the variables in (4.15), we can reformulate it as

$$\mathcal{L}(x, \lambda) = \left(f_1(x_1) + (\lambda_{12} + \lambda_{13} + \ldots + \lambda_{1N})^T x_1\right) + \left(f_2(x_2) - \lambda_{12}^T x_2\right) + \ldots + \left(f_N(x_N) - \lambda_{1N}^T x_N\right). \quad (4.16)$$

Define next

$$\begin{aligned} \phi_1(x_1) &= f_1(x_1) + (\lambda_{12} + \lambda_{13} + \ldots + \lambda_{1N})^T x_1 \\ \phi_i(x_i) &= f_i(x_i) - \lambda_{1i}^T x_i \end{aligned},$$

$i = 2, \ldots, N$, then the primal update is

$$(x_1^{k+1}, \ldots, x_N^{k+1}) = \underset{(x_1, \ldots, x_N)}{\arg\min} [\phi_1(x_1) + \phi_2(x_2) + \ldots + \phi_N(x_N)]. \quad (4.17)$$

Note that (4.17) is equivalent to

$$\begin{aligned} x_1^{k+1} &= \arg\min_{x_1} \phi_1(x_1) \\ x_2^{k+1} &= \arg\min_{x_2} \phi_2(x_2) \\ &\vdots \\ x_N^{k+1} &= \arg\min_{x_N} \phi_N(x_N) \end{aligned} \quad . \tag{4.18}$$

It is obvious that all the arg min operations in (4.18) can be done in parallel. Similarly, using the formulation (4.17), we can perform the dual update in parallel by noting

$$\begin{aligned} \lambda_{12}^{k+1} &= \lambda_{12}^k + \alpha(x_1^{k+1} - x_2^{k+1}) \\ \lambda_{13}^{k+1} &= \lambda_{13}^k + \alpha(x_1^{k+1} - x_3^{k+1}) \\ &\vdots \\ \lambda_{1N}^{k+1} &= \lambda_{1N}^k + \alpha(x_1^{k+1} - x_N^{k+1}) \end{aligned} \quad . \tag{4.19}$$

Observe that, while we have N primal variables, we only have $N-1$ dual variables. Also note that the construction of $\phi_1(x_1)$ (and consequently the update of x_1) requires the knowledge of all the dual variables, whereas $\phi_i(x_i)$ (and consequently the update of x_i), $i = 2, \ldots, N$ only require the knowledge of the "local" dual variables. Therefore, in the master-worker framework, the master is assigned x_1, while each agent contains x_i as well as λ_{1i}, $i = 2, \ldots, N$. The resulting master-worker dual method is summarized in algorithm 1.

Algorithm 1: A master-worker dual method

initialize $x_i^0 \in \mathbb{R}^d$, $i = 1, \ldots, N$, $\lambda_{1j}^0 \in \mathbb{R}^d$, $j = 2, \ldots, N$, $\alpha > 0$;
for *k = 0,1,2,...* **do**
  Agents $j = 2, \ldots, N$ in parallel: send λ_{1j}^k to master;
  All in parallel: perform (4.18);
  Master: send x_1^{k+1} to agents $j = 2, \ldots, N$;
  Agents $j = 2, \ldots, N$ in parallel: perform (4.19);
end

We next consider the dual subgradient method from section 3.4, applied to the computation model (4.1), with the underlying communication network given by the fully distributed model.

The goal is to derive a fully distributed iterative method to solve (4.1), based on the dual subgradient method, where nodes perform computations

in parallel, and exchange information only across communication links (i.e., with their direct neighbors). To begin with, we first define a neighborhood of agent.

Definition 4.1. The set of neighbors of agent i, in the fully distributed communication model, is given by

$$\mathcal{N}_i = \big\{ j \in \mathcal{V} : \{i,j\} \in \mathcal{E} \big\},$$

where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent the set of vertices and edges of the underlying graph of the communication model.

Similarly to the master-worker framework, we use the reformulation (4.13) of the original problem. However, unlike (4.14), we define the consensus constraints as follows, e.g., [12, ?]:

$$\begin{aligned} &\min \sum_{i=1}^{N} f_i(x_i) \\ &x_i = x_j, \ \text{ for all } \{i,j\} \in \mathcal{E} \end{aligned} . \tag{4.20}$$

Note that (4.13) and (4.20) are equivalent. First, the feasible set of (4.20) is given by $S = \{(x_1^T, \ldots, x_N^T)^T \in \mathbb{R}^{Nd} : x_1 = x_2 = \ldots = x_N\}$. It is easy to see that if $x^* \in \mathbb{R}^d$ is a solution to (4.13), then $(x^{*T}, \ldots, x^{*T})^T \in \mathbb{R}^{Nd}$ is a solution to (4.20). Conversely, if $(\hat{x}_1^T, \ldots, \hat{x}^T)^T \in \mathbb{R}^{Nd}$ is a solution of (4.20), then $\hat{x}_1 \in \mathbb{R}^d$ is a solution of (4.13).

The Lagrangian of the problem (4.20) is given by

$$\mathcal{L}(x, \lambda) = \sum_{i=1}^{N} f_i(x_i) + \sum_{\{i,j\} \in \mathcal{E}, i<j} \lambda_{ij}^T (x_i - x_j), \tag{4.21}$$

where $x = (x_1^T, \ldots, x_N^T) \in \mathbb{R}^{Nd}$, $\lambda = (\ldots \lambda_{ij} \ldots) \in \mathbb{R}^{Md}$ and $M = |\mathcal{E}|$. The notation $|\cdot|$ refers to the *set cardinality function*. In general, we can write (4.21) as

$$\mathcal{L}(x, \lambda) = \sum_{i=1}^{N} \phi_i(x_i), \tag{4.22}$$

where $\phi_i(x_i)$ is given by

$$\phi_i(x_i) = f_i(x_i) + x_i^T \Big(\sum_{j \in \mathcal{N}_i} \lambda_{ij} \operatorname{sign}(i-j) \Big). \tag{4.23}$$

Here, $\text{sign} : \mathbb{R} \mapsto \mathbb{R}$ refers to the sign function, given by

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases}.$$

Recall the primal and dual updates (3.14) of the dual subgradient method. We want to derive the dual subgradient based primal-dual updates for (4.22). From (4.22) and (4.23) it can be seen that for a fixed x^{k+1}, the gradient of $\mathcal{L}$ with respect to λ at λ^k is given by (exercise)

$$\nabla_\lambda \mathcal{L}(x^{k+1}, \lambda^k) = \begin{bmatrix} \vdots \\ x_i^{k+1} - x_j^{k+1} \\ \vdots \end{bmatrix},$$

and therefore, the dual update for λ_{ij} becomes

$$\lambda_{ij}^{k+1} = \lambda_{ij}^{k+1} + \alpha(x_i^{k+1} - x_j^{k+1}), \tag{4.24}$$

for all $\{i, j\} \in \mathcal{E}$. For the primal update, recalling (4.22) and (4.23), we can conclude that each agent in parallel performs

$$x_i^{k+1} = \arg\min_{x_i} f_i(x_i) + \Big(\sum_{j \in \mathcal{N}_i} \lambda_{ij}^k \, \text{sign}(j - i) \Big)^T x_i \tag{4.25}$$

similarly to the master-worker framework. The fully distributed dual method is summarized in algorithm 2. For analysis of related more advanced methods, see, e.g., [16, 17].

Algorithm 2: A fully distributed dual subgradient method

initialize $x_i^0 \in \mathbb{R}^d$, $\lambda_{ij}^0 \in \mathbb{R}^d$, $j \in \mathcal{N}_i$, $i = 1, \ldots, N$, $\alpha > 0$;
Agents $i = 1, \ldots, N$ execute in parallel;
for *k = 0,1,2,...* **do**
  Perform (4.25);
  Send x_i^{k+1} to neighbors $j \in \mathcal{N}_i$;
  Perform (4.24);
end

Chapter 5

Primal distributed subgradient methods

This chapter considers a primal distributed subgradient algorithm in [14], see also [12], and provides an analysis of the algorithm under a simplified setting with respect to reference [14].

5.1 Communication and computation model

To begin with, we specify the communication and computation models and some basic assumptions.

The communication model used in [14] is the fully distributed model introduced in section 4.3. The underlying network is represented by an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. As was stated before, $\mathcal{N}_i$ is the neighborhood set of agent i, $i = 1, \ldots, N$. For each agent i, we define the set

$$\bar{\mathcal{N}}_i = \mathcal{N}_i \cup \{i\},$$

which is the set of neighbors of i including i itself.

The assumed computational model is as follows:

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^{N} f_i(x). \tag{5.1}$$

Each agent i has knowledge only of its local function $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, which is assumed to be convex.

5.2 The weight matrix

Many distributed, consensus-based algorithms, e.g., [31], use a weight (consensus) matrix. For a network of N agents, it represents a $N \times N$ matrix, whose key feature is to follow the network sparsity pattern.

More precisely, for a given network of N agents, a weight matrix W is a $N \times N$ square matrix, satisfying:

1. $w_{ij} > 0$, $\{i,j\} \in \mathcal{E}$,
2. $w_{ij} = 0$, $\{i,j\} \notin \mathcal{E}$, $i \neq j$.
3. $w_{ii} = 1 - \sum_{j \neq i} w_{ij} > 0$, $i = 1, ..., N$.

Here, $\mathcal{E}$ is the set of edges of the underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. We note that not all of the above conditions are necessary, but we impose them all to simplify exposition.

Informally, the weight matrix reflects the trust of agent i in its own, as well as its neighbors solutions. We make the following standard assumptions about the weight matrix, used in convergence analysis.

Assumption 1. *The weight matrix W is a symmetric, doubly stochastic matrix. In other words, $W^T = W$ and $W1 = 1$, where $1 \in \mathbb{R}^N$*[2].

Some facts about the weight matrix follow.

- $\|W\| = 1$.
- Define the $N \times N$ matrix J by

$$J = \frac{1}{N} 1 1^T = \begin{bmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{bmatrix}.$$

  Then, $\|W - J\| < 1$.

The intuition behind the second claim stems from the fact that J can be considered as the "perfect" averaging matrix. However, since the graph is sparse, we do not have access to such a weight assignment. If the graph $\mathcal{G}$ is connected, and the assumptions above about W hold, then the second claim is true and may be understood as follows: W is a "good enough" approximation of J.

[2] While we simultaneously use 1 to denote both the scalar 1, as well as the vector of all ones, it will be clear from the context which object is in force.

5.3 The algorithm

We are now ready to describe the algorithm. Let x_i^k be agent i's solution estimate at iteration k. The next estimate is then computed as

$$x_i^{k+1} = \sum_{j \in \mathcal{N}_i} w_{ij} x_j^k - \alpha g_i^k, \tag{5.2}$$

where g_i^k is a (sub)gradient of f_i evaluated at x_i^k, $i = 1, \ldots, N$. A more complete algorithm representation is presented in algorithm 3.

Algorithm 3: Distributed subgradient method

initialize $x_i^0 \in \mathbb{R}^d$, $i = 1, \ldots, N$, $\alpha > 0$;
Agents $i = 1, \ldots, N$ execute in parallel;
for $k = 0,1,2,\ldots$ **do**
  Send the current solution estimate x_i^k to neighbors $j \in \mathcal{N}_i$;
  Receive the neighbors solution estimates x_j^k, $j \in \mathcal{N}_i$;
  Compute a subgradient g_i^k of f_i at x_i^k;
  Perform (5.2);
end

5.4 Convergence analysis

Let $x^k = (x_1^k, \ldots, x_N^k)^T \in \mathbb{R}^{Nd}$. We can state the update rule (5.2) in a more compact way as

$$x^{k+1} = \mathbf{W} x^k - \alpha g^k, \tag{5.3}$$

where $g^k = (g_1^k, \ldots, g_N^k)^T$, $\mathbf{W} = W \otimes I \in \mathbb{R}^{Nd \times Nd}$, $I \in \mathbb{R}^{d \times d}$ is the identity matrix, and the symbol $\otimes$ represents the *Kronecker product.* [3]

Without loss of generality, we can assume that $d = 1$, while the following analysis can be applied to an arbitrary $d \in \mathbb{N}$. We define $\hat{W} = W - J$. From the analysis in section 5.2, we know that $\|\hat{W}\| < 1$.

Claim 5.1. *For the matrices W, I, J and $\hat{W}$, it holds that*

$$(I - J)W = \hat{W}(I - J). \tag{5.4}$$

[3] Recall that for given matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$, the Kronecker product is defined as

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \ldots & A_{1n}B \\ \vdots & \vdots & & \vdots \\ A_{m1}B & A_{m2}B & \ldots & A_{mn}B \end{bmatrix}.$$

Proof. First, we analyze the left-hand side (LHS) of (5.4).

$$(I - J)W = W - JW = W - \frac{1}{N}11^T W = W - \frac{1}{N}11^T = W - J = \hat{W},$$

where the third equality is due to the weight matrix being doubly stochastic. Next, we analyze the right-hand side of (5.4).

$$\begin{aligned}\hat{W}(I - J) &= \hat{W} - \hat{W}J = \hat{W} - (W - J)J = \hat{W} - WJ + JJ \\ &= \hat{W} - J + \frac{1}{N}11^T\frac{1}{N}11^T = \hat{W} - J + \frac{1}{N^2}11^T11^T \\ &= \hat{W} - J + \frac{1}{N^2}N11^T = \hat{W} - J + \frac{1}{N}11^T = \hat{W}.\end{aligned}$$

This completes the proof □

Next, we define the following sequence

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

Multiplying (5.3) by 1^T from the left and dividing it by N, we get

$$\bar{x}^{k+1} = \bar{x}^k - \frac{\alpha}{N}\sum_{i=1}^{N} g_i^k. \tag{5.5}$$

The update (5.5) resembles the update of a centralized subgradient method to solve (4.1), given by

$$y^{k+1} = y^k - \frac{\alpha}{N}\sum_{i=1}^{N} g_i^k, \tag{5.6}$$

where g_i^k is a subgradient of f_i evaluated at y^k.

Note the key difference between the updates (5.5) and (5.6) - the points at which we evaluate the gradients. If (5.5) was a centralized method, the gradients of f_i would have been evaluated at $\bar{x}^k$, instead of x_i^k.

However, we will show that x_i^k is close to $\bar{x}^k$, for all $i = 1, \ldots, N$, and therefore, the method (5.5) behaves similarly to the centralized subgradient method.

We make the following assumptions on the problem of interest (5.1).

Assumption 2. *Each f_i in (5.1) is convex, not necessarily differentiable.*

Assumption 3. *The problem (5.1) is solvable.*

Assumption 4. *For all* $i = 1, \dots, N$, $\|g_i(x)\| \leq G$, $G > 0$, *for all subgradients* g_i *of* f_i *evaluated at* x, *for all* $x \in \mathbb{R}^d$.

Assumption 5. $\|x_i^0 - x^\star\| \leq R$, *for all* $i = 1, \dots, N$ *and for all solutions* $x^\star$ *of the problem (5.1).*

Additionally, define the following sequence

$$\hat{x}_i^k = x_i^k - \bar{x}^k,\ i = 1, \dots, N,$$

and $\hat{x}^k = (\hat{x}_1^k, \dots, \hat{x}_N^k)^T \in \mathbb{R}^{Nd}$. It is easy to note that $\hat{x}^k = (I - J)x^k$ (exercise).

We will study the evolution of the two sequences introduced next.

1. The "disagreement" sequence: $\hat{x}_i^k = x_k^i - \bar{x}^k$, $i = 1, ..., N$.
2. The evolution of the average sequence $\bar{x}^k$.

From the definition of $\hat{x}^k$ and (5.3), we have

$$\begin{aligned}\hat{x}^{k+1} = (I - J)x^{k+1} = (I - J)\mathbf{W}x^k - \alpha(I - J)g^k &= \hat{W}(I - J)x^k - \alpha(I - J)g^k \\ &= \hat{W}\hat{x}^k - \alpha(I - J)g^k,\end{aligned} \tag{5.7}$$

where we used calim 5.1 for the third equality. Applying the L2 norm on (5.7) and recalling that $\|\hat{W}\| < 1$, we get

$$\|\hat{x}^{k+1}\| = \|\hat{W}\hat{x}^k - \alpha(I - J)g^k\| \leq \|\hat{W}\|\|\hat{x}^k\| + \alpha\|I - J\|\|g^k\| \leq \delta\|\hat{x}^k\| + \alpha\|I - J\|\|g^k\|, \tag{5.8}$$

where we denote $\|\hat{W}\| \coloneqq \delta < 1$. Next, recall assumption 4. It can be shown that $\|g^k\| \leq \sqrt{N}G$ (exercise). For $\|I - J\|$ it holds that $\|I - J\| \leq \|I\| + \|J\| \leq 2$. It can actually be shown that $\|I - J\| = 1$ (exercise). Using these facts in 5.8, we get

$$\|\hat{x}^{k+1}\| \leq \delta\|\hat{x}^k\| + \alpha\sqrt{N}G, \tag{5.9}$$

for all $k = 0, 1, 2 \dots$.

Additionally, we make the following assumption.

Assumption 6. *Let the initialization of* $x_i^0 \in \mathbb{R}^d$, $i = 1, \dots, N$ *be such that* $\hat{x}^0 = 0$, *where* $0 \in \mathbb{R}^d$.

Remark. Note that assumption 6 is not hard to satisfy. For example, initializing all x_i^0 to zero vectors, $i = 1, \dots, N$ (a standard initialization in optimization algorithms) satisfies the assumption.

Applying 5.9 for $k = 1$ with assumption 6 in place, we get

$$\|\hat{x}^1\| \leq \alpha\sqrt{N}G.$$

Similarly, for $k = 2, 3$ we get

$$\|\hat{x}^2\| \leq \delta\|\hat{x}^1\| + \alpha\sqrt{N}G \leq (1+\delta)\alpha\sqrt{N}G,$$
$$\|\hat{x}^3\| \leq \delta\|\hat{x}^2\| + \alpha\sqrt{N}G \leq (1+\delta+\delta^2)\alpha\sqrt{N}G.$$

Applying the ineqaulity all the way up to k, we get

$$\|\hat{x}^k\| \leq (1+\delta+\ldots+\delta^{k-1})\alpha\sqrt{N}G \implies \|\hat{x}^k\| \leq \frac{\alpha\sqrt{N}G}{(1-\delta)}, \tag{5.10}$$

which gives us the disagreement bound, completing the first part of the analysis.

Next, we will analyze the evolution of $\bar{x}^k$. Recall assumption 2. For each f_i it holds

$$f_i(y) \geq f_i(x_i^k) + (y - x_i^k)^T g_i^k, \tag{5.11}$$

for all $y \in \mathbb{R}^d$. Summing (5.11) over all $i = 1, \ldots, N$, we get

$$\sum_{i=1}^{N} f_i(y) \geq \sum_{i=1}^{N} f_i(x_i^k) + \sum_{i=1}^{N} (y - x_i^k)^T g_i^k. \tag{5.12}$$

Denote $f(x) = \sum_{i=1}^{N} f_i(x_i)$. From (5.12) we get

$$\begin{aligned} f(y) &\geq \sum_{i=1}^{N} \left(f_i(x_i^k) \pm f_i(\bar{x}^k)\right) + \sum_{i=1}^{N} (y - x_i^k \pm \bar{x}^k)^T g_i^k \\ &= f(\bar{x}^k) - \sum_{i=1}^{N} \left(f_i(\bar{x}^k) - f_i(x_i^k)\right) + (y - \bar{x}^k)^T \sum_{i=1}^{N} g_i^k - \sum_{i=1}^{N} (x_i^k - \bar{x}^k)^T g_i^k. \end{aligned} \tag{5.13}$$

Next, recall that, by the Cauchy-Schwarz inequality we have $x^T y \leq \|x\|\|y\|$. Applying this fact in (5.13), we get

$$f(y) \geq f(\bar{x}^k) - \sum_{i=1}^{N} \left(f_i(\bar{x}^k) - f_i(x_i^k)\right) + (y - \bar{x}^k)^T \sum_{i=1}^{N} g_i^k - \sum_{i=1}^{N} \|x_i^k - \bar{x}^k\|\|g_i^k\|. \tag{5.14}$$

By the convexity of f_i, we have

$$\begin{aligned} & f_i({x_i}^k) \geq f_i(\bar{x}^k) + (x_i^k - \bar{x}^k)^T g^{f_i}(\bar{x}^k) \\ \implies & -\big(f_i(\bar{x}^k) - f_i(x_i)\big) \geq -(\bar{x}^k - x_i^k)^T g^{f_i}(\bar{x}^k) \geq -\|g^{f_i}(\bar{x}^k)\|\|\bar{x}^k - x_i^k\| \\ \implies & -\big(f_i(\bar{x}^k) - f_i(x_i)\big) \geq -G\|\bar{x}^k - x_i^k\|, \end{aligned} \tag{5.15}$$

where we used assumption 4 in the last inequality. Substituting (5.15) and appying assumption 4 in (5.14), we get

$$f(y) \geq f(\bar{x}^k) - G\sum_{i=1}^N \|\bar{x}^k - x_i^k\| + (y - \bar{x}^k)^T \sum_{i=1}^N g_i^k - G\sum_{i=1}^N \|\bar{x}^k - x_i^k\|. \tag{5.16}$$

Noting that $\|\hat{x}_i^k\| = \|\bar{x}^k - x_i^k\|$ and denoting $\epsilon^k = 2G\sum_{i=1}^N \|\hat{x}_i^k\|$, we get

$$f(y) \geq f(\bar{x}^k) + (y - \bar{x}^k)^T \sum_{i=1}^N g_i^k - \epsilon^k. \tag{5.17}$$

Next, we get

$$\begin{aligned} \left(\sum_{i=1}^N \|\hat{x}_i^k\|\right)^2 = \left(\frac{N}{N}\sum_{i=1}^N \|\hat{x}_i^k\|\right)^2 = N^2\left(\frac{1}{N}\sum_{i=1}^N \|\hat{x}_i^k\|\right)^2 \leq N^2\left(\frac{1}{N}\sum_{i=1}^N \|\hat{x}_i^k\|^2\right) \\ = N\left(\sum_{i=1}^N \|\hat{x}_i^k\|^2\right) = N\|\hat{x}^k\|^2. \end{aligned} \tag{5.18}$$

Using (5.18) in the definition of ϵ^k, and recalling (5.10), we get

$$\epsilon^k = 2G\sum_{i=1}^N \|\hat{x}_i^k\| \leq 2G\sqrt{N}\frac{\alpha\sqrt{N}G}{1-\delta} = \frac{2\alpha N G^2}{1-\delta} = \epsilon. \tag{5.19}$$

Substituting (5.19) in (5.17), we get

$$f(y) \geq f(\bar{x}^k) + (y - \bar{x}^k)^T \sum_{i=1}^N g_i^k - \epsilon. \tag{5.20}$$

Recall (5.5). Define $\bar{g}^k = \sum_{i=1}^N g_i^k$, and let x^* be a minimizer of $f(x) = \sum_{i=1}^N f_i(x)$. We have

$$\|\bar{x}^{k+1} - x^*\|^2 \leq \|\bar{x}^k - \frac{\alpha}{N}\bar{g}^k - x^*\|^2 \leq \|\bar{x}^k - x^*\|^2 - \frac{2\alpha}{N}(\bar{x}^k - x^*)^T\bar{g}^k + \frac{\alpha^2}{N^2}\|\bar{g}^k\|^2. \tag{5.21}$$

Using (5.20), we have

$$(\bar{x}^k - x^*)^T \bar{g}^k \leq -(f(\bar{x}^k) - f(x^*)) + \epsilon. \tag{5.22}$$

Substituting (5.22) in (5.21), we get

$$\begin{aligned} \|\bar{x}^{k+1} - x^*\|^2 &\leq \|\bar{x}^k - x^*\|^2 - \frac{2\alpha}{N}(f(\bar{x}^k) - f(x^*)) + \frac{2\alpha}{N}\epsilon + \frac{\alpha^2}{N^2}\|\bar{g}^k\|^2 \\ &\leq \|\bar{x}^k - x^*\|^2 - \frac{2\alpha}{N}(f(\bar{x}^k) - f(x^*)) + \frac{2\alpha\epsilon}{N} + \alpha^2 G^2, \end{aligned} \tag{5.23}$$

where we used $\|\bar{g}^k\| = N\|g_i^k\|$ (exercise), and assumption 4 in the second inequality. Rearranging (5.23), we get

$$\frac{2\alpha}{N}(f(\bar{x}^k) - f(x^*)) \leq \|\bar{x}^k - x^*\|^2 - \|\bar{x}^{k+1} - x^*\|^2 + \frac{2\alpha\epsilon}{N} + \alpha^2 G^2. \tag{5.24}$$

Applying (5.24) telescopically all the way to $k = 0$ and summing all the equations up, we get

$$\sum_{t=0}^{k-1}(f(\bar{x}^t) - f(x^*)) \leq \frac{N}{2\alpha}\|\bar{x}^0 - x^*\|^2 + k\epsilon + \frac{k\alpha N G^2}{2}. \tag{5.25}$$

Define the running average $\bar{x}_{ra}^k = \frac{1}{k}\sum_{t=0}^{k-1}\bar{x}^t$. Dividing (5.25) by k, we get

$$\frac{1}{k}\sum_{t=0}^{k-1} f(\bar{x}^t) - f(x^*) \leq \frac{N}{2\alpha k}\|\bar{x}^0 - x^*\|^2 + \epsilon + \frac{\alpha N G^2}{2}. \tag{5.26}$$

Due to convexity of f, we know that $f(\bar{x}_{ra}^k) = f(\frac{1}{k}\sum_{t=0}^{k-1}\bar{x}^t) \leq \frac{1}{k}\sum_{t=0}^{k-1} f(\bar{x}^t)$ (exercise). Therefore, we get

$$f(\bar{x}_{ra}^k) - f(x^*) \leq \frac{N\|\bar{x}^0 - x^*\|^2}{2\alpha k} + \epsilon + \frac{\alpha N G^2}{2}. \tag{5.27}$$

Applying assumption 5, we get

$$f(\bar{x}_{ra}^k) - f(x^*) \leq \frac{N R^2}{2\alpha k} + \epsilon + \frac{\alpha N G^2}{2}, \tag{5.28}$$

which shows that the running average converges to a neighborhood the solution, as $k \to \infty$. If we define $x_{i,ra}^k = \frac{1}{k}\sum_{t=0}^{k-1} x_i^t$, we have

$$f(x_{i,ra}^k) - f(x^*) = \big(f(x_{i,ra}) - f(\bar{x}_{ra}^k)\big) + \big(f(\bar{x}_{ra}^k) - f(x^*)\big). \tag{5.29}$$

While can bound the second term on the right-hand side of (5.28) using (5.26), we can use convexity of f to bound the first term on the right-hand side as follows.

$$\begin{aligned} & f(\bar{x}_{ra}^k) \geq f(x_{i,ra}^k) + g^f(x_{i,ra}^k)^T(\bar{x}_{ra}^k - x_{i,ra}^k) \\ & \implies f(x_{i,ra}^k) - f(\hat{x}_{ra}^k) \leq g^f(x_{i,ra}^k)^T(x_{i,ra}^k - \bar{x}_{ra}^k) \leq \|g^f(x_{i,ra}^k)\|\|(x_{i,ra}^k - \bar{x}_{ra}^k)\| \\ & \quad \leq G\|\frac{1}{k}\sum_{t=0}^{k-1}(x_i^t - \bar{x}^t)\| \leq G\frac{1}{k}\sum_{t=0}^{k-1}\|\hat{x}_i^t\| \leq NG\frac{1}{k}\sum_{t=0}^{k-1}\|\hat{x}^t\| \leq \frac{\alpha N^{\frac{3}{2}}G^2}{1-\delta}. \end{aligned} \tag{5.30}$$

Finally, using (5.29), (5.28) and (5.30), we get

$$f(x_{i,ra}^k) - f(x^*) \leq \frac{\alpha N^{\frac{3}{2}}G^2}{1-\delta} + \frac{NR^2}{2\alpha k} + \frac{1}{2}\alpha NG^2 + \frac{2\alpha NG^2}{1-\delta}, \tag{5.31}$$

which shows that the running average of each individual agent's estimate converges to the neighborhood of the solution.

Remark. With respect to the centralized subgradient method (see (2.12)), the right-hand side of (5.28) is of a similar form. It is only deteriorated by the additive term ϵ. Note also that with the centralized subgradient method, we kept track of the best point so far. Here, we kept track of the "time" average. Among other reasons, this is because, in the distributed settings, keeping track of the best point so far is hard (no agent has access to the full objective f, only its local f_i).

Remark. Note the difference between (5.28) and (5.31). Compared to (5.28), (5.31) is only deteriorated by $\frac{\alpha N^{\frac{3}{2}}G^2}{1-\delta}$, which is due to the disagreement between the local solution and the average network solution.

Chapter 6

An advanced topic: Primal distributed Nesterov-like gradient methods

This chapter considers distributed gradient methods based on the (centralized) Nesterov gradient method, e.g., [32], for smooth convex costs. The methods have been proposed in [33, 34], while here we present the analysis of these methods from Chapter 4 of the PhD thesis [18]. Specifically, we consider here the methods D–NG [33] and mD–NC [34]. The main purpose of the chapter is to show an example of a mechanism for distributed algorithms acceleration (Nesterov gradient-based acceleration), as well as describe the main steps in the analysis of an accelerated distributed method. A major part of the Chapter is taken practically unaltered from the PhD thesis [18].

6.1 Algorithm D–NG

6.1.1 Model and notational preliminaries

Notation. We denote by: $\mathbb{R}^d$ the d-dimensional real coordinate space, $d \geq 1$; A_{ij} the entry in the i-th row and j-th column of a matrix A; a_i the i-th entry of a vector a; $(\cdot)^\intercal$ the transpose; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument (We note that $\|\cdot\|$ also denotes the modulus of a scalar throughout); $\lambda_i(\cdot)$ the i-th smallest eigenvalue; $|\cdot|$ the cardinality of a set; $\nabla\mathcal{J}(y)$ the gradient evaluated at y of a function $\mathcal{J} : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$. Finally, notation $r(k) = O(q(k))$ means existence of a $K > 0$ such that $r(k) \leq \mu\, q(k)$, for some $\mu > 0$, for all $k \geq K$.

Distributed optimization model. The nodes solve the unconstrained problem:

$$\text{minimize} \sum_{i=1}^{N} f_i(x) =: f(x). \tag{6.1}$$

The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only to node i. We impose Assumptions 7 and 8.

Assumption 7 (Solvability; Lipschitz continuous gradient). *1. There exists a solution $x^\star \in \mathbb{R}^d$ with $f(x^\star) = \inf_{x\in\mathbb{R}^d} f(x) =: f^\star$.*

2. for all i, f_i is convex, differentiable, with Lipschitz continuous derivative with constant $L \in [0,\infty)$: $\|\nabla f_i(x) - \nabla f_i(y)\| \le L\|x-y\|$, for all $x, y \in \mathbb{R}^d$.

Assumption 8 (Bounded gradients). *There exists a constant $G \in [0,\infty)$ such that, for all i, $\|\nabla f_i(x)\| \le G$, for all $x \in \mathbb{R}^d$.*

6.1.2 D–NG algorithm

We now describe the algorithm D–NG proposed in [33]. We continue to assume a generic, undirected, connected network, as in Chapter 5, with an associated doubly stochastic weight matrix W. Algorithm D–NG generates the sequence $(x_i(k), y_i(k))$, $k = 0, 1, 2, ...$, at each node i. Here, $y_i(k)$ is an auxiliary variable. D–NG is initialized by $x_i(0) = y_i(0) \in \mathbb{R}^d$, for all i. The update at node i and $k = 1, 2, ...$ is given by the following:

$$x_i(k) = \sum_{j\in O_i} W_{ij}\, y_j(k-1) - \alpha_{k-1}\nabla f_i(y_i(k-1)) \tag{6.2}$$

$$y_i(k) = x_i(k) + \beta_{k-1}\left(x_i(k) - x_i(k-1)\right). \tag{6.3}$$

Here, W_{ij} are the averaging weights (the entries of the $N \times N$ matrix W), and O_i is the neighborhood set of node i (including i). The step-size α_k and the sequence β_k are:

$$\alpha_k = \frac{c}{k+1}, \quad c > 0; \quad \beta_k = \frac{k}{k+3}, \quad k = 0, 1, ... \tag{6.4}$$

Vector form. We can also present D–NG in vector format. Introduce $x(k) = (x_1(k)^\top, x_2(k)^\top, ..., x_N(k)^\top)^\top$, $y(k) = (y_1(k)^\top, y_2(k)^\top, ..., y_N(k)^\top)^\top$, and define $F : \mathbb{R}^{Nd} \to \mathbb{R}^N$ as: $F(x) = F(x_1, x_2, ..., x_N) = (f_1(x_1), f_2(x_2), ..., f_N(x_N))^\top$. Then, given $x(0) = y(0)$, D–NG in vector form becomes:

$$x(k) = (W \otimes I)y(k-1) - \alpha_{k-1}\nabla F(y(k-1)) \quad (6.5)$$
$$y(k) = x(k) + \beta_{k-1}\left(x(k) - x(k-1)\right), \; k = 1, 2, ..., \quad (6.6)$$

6.2 Analysis of algorithm D–NG

In this Section, we consider D–NG for unconstrained optimization, when the f_i's satisfy a growth condition. Subsection 6.2.1 details the Assumptions and setup, Subsection 6.2.2 defines clone functions Ψ and gives their properties, and Subsection 6.3 performs convergence rate analysis. Throughout the current section, we consider static networks.

6.2.1 Assumptions and setup

We impose the following two Assumptions on the f_i's.

Assumption 9. *For all i, $f_i : \mathbb{R}^d \to \mathbb{R}$ is convex, differentiable, and has Lipschitz continuous derivative with constant L, i.e., for all i:*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \quad \text{for all } x, y \in \mathbb{R}^d.$$

Assumption 10 (Growth assumption)**.** *There exist two positive scalars b and B, such that, for all i, $f_i(x) \geq b\|x\|$ whenever $\|x\| \geq B$.*

Assumption 9 is standard in the analysis of gradient methods. Assumption 10 says that the function grows at least as $b\|x\|$ when $\|x\|$ is sufficiently large. The two Assumptions hold with many costs, e.g., quadratics with positive Hessians, and source localization costs. Under Assumption 10, each f_i is coercive[4], and so is $f := \sum_{i=1}^N f_i$. Thus, there exist $x_i^\star$, $i = 1, ..., N$, and $x^\star$, such that $f_i^\star := \inf_{x\in\mathbb{R}^d} f_i(x) = f_i(x_i^\star)$, and $f^\star := \inf_{x\in\mathbb{R}^d} f(x) = f(x^\star)$. Without loss of generality (w.l.o.g.), we choose the constant B in Assumption 10 such that:

$$f_i^\star < b\,B, \quad \text{for all } i = 1, ..., N, \quad (6.7)$$
$$f^\star < N\,b\,B. \quad (6.8)$$

Hence, any minimizer $x_i^\star$ of f_i, for all i, and any minimizer $x^\star$ of f, belongs to the closed ball $\{x \in \mathbb{R}^d : \|x\| \leq B\}$.

[4]Coercive means that $f_i(x) \to +\infty$ whenever $\|x\| \to +\infty$.

Introduce the function $F : \mathbb{R}^N \to \mathbb{R}$, as:

$$F(x) = F(x_1, ..., x_N) = f_1(x_1) + ... + f_N(x_N). \tag{6.9}$$

For future reference, we introduce the set:

$$\mathcal{S} := \left\{ x \in \mathbb{R}^N : F(x) \leq N\, b\, B \right\}, \tag{6.10}$$

and the following two constants:

$$\begin{aligned} \mathcal{D} &:= \sup_{x \in \mathcal{S}} \|x\| < \infty \\ G &:= \sup_{x \in \mathcal{S}} \|\nabla F(x)\| < \infty. \end{aligned} \tag{6.11}$$

The set $\mathcal{S}$ is compact, because the function F is coercive by Assumption 10. The two suprema in (6.11) are attained at some points and are finite, because the set $\mathcal{S}$ is compact, and the functions $\|\cdot\|$ and $\|\nabla F(\cdot)\|$ are continuous (The latter function is continuous due to continuity of the gradients of the f_i's – see Assumption 9.)

We consider the D–NG algorithm. Each node i updates its solution estimate $x_i(k)$ and an auxiliary variable $y_i(k)$ over iterations k as follows:

$$x_i(k) = (1 - \ell_i\, w)\, y_i(k-1) + w \sum_{j \in O_i - \{i\}} y_j(k-1) - \alpha \nabla f_i(y_i(k-1)) \tag{6.12}$$

$$y_i(k) = x_i(k) + \beta_{k-1} \left(x_i(k) - x_i(k-1) \right), \quad k = 1, 2, ..., \tag{6.13}$$

with $x_i(0) = y_i(0) \in \mathbb{R}^d$. The step-size α_k and the sequence β_k are:

$$\alpha_k = \frac{c}{k+1}, \quad \beta_k = \frac{k}{k+3}, \quad k = 0, 1, ... \tag{6.14}$$

We choose c and w as:

$$\begin{aligned} c &\leq \frac{1}{\rho\, \lambda_N(\mathcal{L}) + L} \\ w &= c\, \rho. \end{aligned} \tag{6.15}$$

Here, ρ is a positive constant; e.g., it can be set to $\rho = 1$. To satisfy (6.15), we require that nodes know beforehand (upper bounds on) ρ, $\lambda_N(\mathcal{L})$, and L. We can set $\rho = 1$. Further, it can be shown that $\lambda_N(\mathcal{L}) \leq 2 \max_{i=1,...,N} \ell_i$. Finally, if ∇f_i has a Lipschitz constant L_i, known by node i, we can take L as $L := \max_{i=1,...,N} L_i$. Hence, requirement (6.15) is accomplished beforehand through two distributed maximum computations – in $O(N)$ per-node scalar

communications.

[5] Here, and throughout the whole Chapter, to avoid notational clutter, we assume equal initialization $\overline{x}(0) := x_i(0) = y_i(0) = x_j(0) = y_j(0)$, for all i, j, and we let $d = 1$, but the results extend for a generic d as well.

6.2.2 Clone functions Ψ_k

For a real number $k \geq 0$, consider the following (clone) unconstrained optimization problem over $x = (x_1, x_2, ..., x_N) \in \mathbb{R}^N$:

$$\text{minimize} \quad \Psi_k(x) := \sum_{i=1}^N f_i(x_i) + \frac{k\rho}{2} x^\top \mathcal{L} x, \tag{6.16}$$

where $\rho > 0$ is a parameter. In (6.16), recall that $\mathcal{L}$ is the graph Laplacian matrix. Consider the step-size $\alpha_{k-1} = c/k$, and let $w = c\rho$. Introduce compact notation for nodes' estimates $x(k) := (x_1(k), ..., x_N(k))^\top$, and $y(k) := (y_1(k), ..., y_N(k))^\top$. Then, it is easy to verify that algorithm (6.5) can be re-written as:

$$\begin{aligned} x(k) &= y(k-1) - \alpha_{k-1} \nabla \Psi_k(y(k-1)) \\ y(k) &= x(k) + \beta_{k-1} \left(x(k) - x(k-1) \right), \; k = 1, 2, ... \end{aligned} \tag{6.17}$$

with α_k and β_k in (6.14) and the initialization is $x(0) = y(0) = \overline{x}(0)\mathbf{1}$. Hence, at iteration k, algorithm (6.5) performs the (exact) Nesterov gradient step with respect to the clone function Ψ_k.

We impose that the step-size satisfies:

$$\alpha_{k-1} = c/k \leq 1/L_{\Psi_k},$$

where L_{Ψ_k} is a Lipschitz constant of the gradient of Ψ_k, which we can take as:

$$L_{\Psi_k} = k \left(\rho \, \lambda_N(\mathcal{L}) + L \right).$$

Thus, we can choose:

Properties of the clone functions Ψ_k

Next Lemma states certain properties of the clone functions Ψ_k's and problem (6.16). The Lemma also relates (6.16) with the original problem (6.1).

Lemma 1 (Properties of (6.16))**.** *Consider* (6.16)*. Then, there holds:*

[5] We assign equal weights w_0 to all neighbors; generalization to unequal weights is straightforward.

1. *There exists a solution* $x^{\mathrm{c}}(k) = (x_1^{\mathrm{c}}(k), ..., x_N^{\mathrm{c}}(k))$ *to* (6.16) *that satisfies* $\inf_{x\in\mathbb{R}^N} \Psi_k(x) = \Psi(x^{\mathrm{c}}(k)) =: \Psi_k^\star > -\infty$. *Further, the corresponding solution set is compact.*

2. $f^\star \geq \Psi_k^\star \geq \Psi_t^\star$ *for all* k, t, $k > t$.

3. *For any* $x^{\mathrm{c}}(k)$, *there holds:* $\|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))\| \leq L\mathcal{D} + G =: G^\star$, *for all* i, k, *where* $x_{\mathrm{avg}}^{\mathrm{c}}(k) = \frac{1}{N}\sum_{i=1}^N x_i^{\mathrm{c}}(k)$.

4. *For any* $k > 0$, $\Psi_k^\star \geq f^\star - \frac{N(G^\star)^2}{2\rho k \lambda_2(\mathcal{L})}$.

For part (a), note that the function Ψ_k is coercive. Also, by Assumption 7, the function Ψ_k is closed and convex. Hence, as Ψ_k is closed, convex, and coercive, problem (6.16) is solvable, the solution set is compact, and $\Psi_k^\star > -\infty$ (see, e.g., [35]).

We prove part (b). Fix some k, and note that:

$$f^\star = \Psi_k(x^\star \mathbf{1}) = \sum_{i=1}^N f_i(x^\star) \geq \Psi_k(x^{\mathrm{c}}(k)) = \Psi_k^\star,$$

and thus $f^\star \geq \Psi_k^\star$. Next, fix k, t, $k > t$, and note that:

$$\begin{aligned} \Psi_t^\star &= \Psi_t(x^{\mathrm{c}}(t)) \leq \Psi_t(x^{\mathrm{c}}(k)) = \sum_{i=1}^N f_i(x_i^{\mathrm{c}}(k)) + \frac{\rho t}{2} x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) \\ &\leq \sum_{i=1}^N f_i(x_i^{\mathrm{c}}(k)) + \frac{\rho k}{2} x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) = \Psi_k^\star, \end{aligned}$$

and so $\Psi_t^\star \leq \Psi_k^\star$ whenever $t < k$, which completes the proof of part (b).

We now prove part (c). From part (b):

$$\sum_{i=1}^N f_i(x_i^{\mathrm{c}}(k)) \leq \Psi_k^\star \leq f^\star \leq N\, b\, B, \quad \text{for all } k.$$

Thus, $x^{\mathrm{c}}(k)$ belongs to set $\mathcal{S}$, and then, in view of (6.11), we have: $\|x_i^{\mathrm{c}}(k)\| \leq \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}$, for all i, for all k. Further:

$$\|x_{\mathrm{avg}}^{\mathrm{c}}(k)\| = \left\| \frac{1}{N} \sum_{i=1}^N x_i^{\mathrm{c}}(k) \right\| \leq \frac{1}{N} N \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}.$$

We now upper bound $\|\nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k))\|$:

$$\begin{aligned}
\|\nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k))\| &= \|\nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k)) - \nabla f_i(x^{\mathrm{c}}_i(k)) + \nabla f_i(x^{\mathrm{c}}_i(k))\| \\
&\leq \|\nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k)) - \nabla f_i(x^{\mathrm{c}}_i(k))\| + \|\nabla f_i(x^{\mathrm{c}}_i(k))\| \\
&\leq L\|x^{\mathrm{c}}_{\mathrm{avg}}(k) - x^{\mathrm{c}}_i(k)\| + \|\nabla f_i(x^{\mathrm{c}}_i(k))\| \\
&\leq L\left(\|x^{\mathrm{c}}_{\mathrm{avg}}(k)\| + \|x^{\mathrm{c}}_i(k)\|\right) + \|\nabla f_i(x^{\mathrm{c}}_i(k))\| \\
&\leq 2L\mathcal{D} + G =: G^\star, \quad \text{for all } k,
\end{aligned}$$

where we use $\|x^{\mathrm{c}}_i(k)\| \leq \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}$ (as $x^{\mathrm{c}}(k)$ belongs to set $\mathcal{S}$), (6.11), and the Lipschitz continuity of the gradient ∇f_i (see Assumption 9.) Thus, the result in part (c). We now prove part (d). We have:

$$\begin{aligned}
\Psi^\star_k &= \sum_{i=1}^N f_i(x^{\mathrm{c}}_i(k)) + \frac{\rho}{2} k x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) \\
&\geq \sum_{i=1}^N \left(f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k)) + \nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k))(x^{\mathrm{c}}_i(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k))\right) \qquad (6.18) \\
&+ \frac{\rho}{2} k (x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1})^\top \mathcal{L}(x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1}) \\
&\geq f(x^{\mathrm{c}}_{\mathrm{avg}}(k)) + \sum_{i=1}^N \left(-G^\star \|x^{\mathrm{c}}_i(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\| + \frac{\rho}{2} k \lambda_2(\mathcal{L}) \|x^{\mathrm{c}}_i(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\|^2\right) \qquad (6.19) \\
&\geq f^\star - \frac{N(G^\star)^2}{2\rho k \lambda_2(\mathcal{L})},
\end{aligned}$$

after (separately) minimizing each summand in (6.19) over $\epsilon := \|x^{\mathrm{c}}_i(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\| \in \mathbb{R}$. Inequality (6.18) used convexity of the f_i's and the fact that $\mathcal{L}(x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1}) = 0$. Inequality (6.19) used the bound on the gradients, given by $\|\nabla f_i(x^{\mathrm{c}}_{\mathrm{avg}}(k))\| \leq G^\star$, and the variational characterization of the eigenvalues to show $(x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1})^\top \mathcal{L}(x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1}) \geq \lambda_2(\mathcal{L})\|x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1}\|^2$, as $(x^{\mathrm{c}}(k) - x^{\mathrm{c}}_{\mathrm{avg}}(k)\mathbf{1})$ is orthogonal to $q_1 = \frac{1}{\sqrt{N}}\mathbf{1}$–the eigenvector of $\mathcal{L}$ that corresponds to $\lambda_1(\mathcal{L}) = 0$. Thus, the result in part (d).

6.3 Convergence analysis of projected mD–NC

We briefly summarize the analysis. We first show that $\sum_{i=1}^N f_i(x_i(k)) = O(\log k)$, thus showing that $\sum_{i=1}^N f_i(x_i(k))$ does not grow fast with k. Then, using the growth assumption on the f_i's in Assumption 10, we show that $\|\nabla f_i(y_i(k))\| = O(\log k)$. This then gives us that $f(x_i(k)) - f^\star = O(\log^3 k/k)$,

and, as a corollary, that $f(x_i(k))$ is uniformly bounded, for all i, k. (Lemma 4.) Finally, we explain how to boost the bound to $O(\log k/k)$.

Bounding the function values by $O(\log k)$

We now show that $\sum_{i=1}^N f_i(x_i(k))$ is $O(\log k)$.

Lemma 2. *Consider algorithm (6.12)–(6.15). Further, denote by* $R := \|\overline{x}(0) - x^\star\|$. *Then, for all* $k = 1, 2, \ldots$:

$$\sum_{i=1}^N f_i(x_i(k)) \leq \frac{2\,N\,R^2}{c} + f(\overline{x}(0)) + 3\left(b\,N\,B - \sum_{i=1}^N f_i^\star\right) + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})} S_k,$$

where

$$S_k = 1 + \sum_{t=1}^{k-1} \frac{(t+1)^2}{t^3} + \sum_{t=2}^{k-1} \frac{1}{t\lfloor t/2 \rfloor} = O(\log k). \tag{6.20}$$

We prove Lemma 2 using the interpretation (6.17) that the iteration k of our algorithm (6.5) is a Nesterov gradient step with respect to Ψ_k. We use it here to estimate the progress in one iteration with respect to Ψ_k. More precisely, denote by $v(k) = \frac{y(k) - (1-\theta_k)x(k)}{\theta_k}$ and recall that $\theta_k = 2/(k+2)$. Applying Lemma 5 with $f \equiv \Psi_k$, $x^\bullet \equiv x^\star \mathbf{1}$, and $L_k = 1/\alpha_k = k/c$ (Note that here $\delta_k = 0$; also, we do not choose $x^\bullet$ to be an optimizer of Ψ_k, which is a valid choice):

$$\begin{aligned} \frac{(k+1)^2}{k}\left(\Psi_k(x(k)) - \Psi_k(x^\star\mathbf{1})\right) + (2/c)\|v(k) - x^\star\mathbf{1}\|^2 \\ \leq \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star\mathbf{1})\right) + (2/c)\|v(k-1) - x^\star\mathbf{1}\|^2. \end{aligned} \tag{6.21}$$

Next, note that the term $\frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star\mathbf{1})\right)$ on the right hand side of (6.21) is, for $k = 1, 2, \ldots$, upper bounded as (because $\Psi_k(x^\star\mathbf{1}) \geq \Psi_k^\star$):

$$\begin{aligned} \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star\mathbf{1})\right) &\leq \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k^\star\right) \\ &\leq k\left(\Psi_k(x(k-1)) - \Psi_k^\star\right), \end{aligned} \tag{6.22}$$

where the last inequality follows because $\Psi_k(x(k-1)) \geq \Psi_k^\star$. Further, the term $\frac{(k+1)^2}{k}\left(\Psi_k(x(k)) - \Psi_k(x^\star\mathbf{1})\right)$ on the left hand side of (6.21) is, for $k = 1, 2, \ldots$,

lower bounded as:

$$\frac{(k+1)^2}{k}(\Psi_k(x(k)) - \Psi_k(x^\star \mathbf{1})) = \frac{(k+1)^2}{k}(\Psi_k(x(k)) - \Psi_k^\star) - \frac{(k+1)^2}{k}(\Psi_k(x^\star \mathbf{1}) - \Psi_k^\star) \tag{6.23}$$

$$\geq (k+2)(\Psi_k(x(k)) - \Psi_k^\star) - \frac{(k+1)^2}{k}(\Psi_k(x^\star \mathbf{1}) - \Psi_k^\star) \tag{6.24}$$

$$\geq (k+2)(\Psi_k(x(k)) - \Psi_k^\star) - \frac{(k+1)^2}{k}\frac{N(G^\star)^2}{2\rho k \lambda_2(\mathcal{L})}, \tag{6.25}$$

Here, (6.24) uses the fact that $\Psi_k(x(k)) - \Psi_k^\star \geq 0$, so that $\frac{k^2+2k+1}{k}(\Psi_k(x(k)) - \Psi_k^\star) \geq \frac{k^2+2k}{k}(\Psi_k(x(k)) - \Psi_k^\star)$; and (6.25) uses inequality $f^\star - \Psi_k^\star = \Psi_k(x^\star \mathbf{1}) - \Psi_k^\star \leq \frac{N(G^\star)^2}{2\rho k \lambda_2(\mathcal{L})}$ from Lemma 1, part (d). Using (6.22) and (6.25), dividing (6.21) by k, and rearranging the terms:

$$\begin{aligned} &\left(1 + \frac{2}{k}\right)(\Psi_k(x(k)) - \Psi_k^\star) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \\ \leq\ &(\Psi_k(x(k-1)) - \Psi_k^\star) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 + \frac{(k+1)^2}{k^3}\frac{N(G^\star)^2}{2\rho\lambda_2(\mathcal{L})}, \end{aligned}$$

or, equivalently:

$$\begin{aligned} (\Psi_k(x(k)) - \Psi_k^\star) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \leq &(\Psi_k(x(k-1)) - \Psi_k^\star) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 \\ &+ \frac{(k+1)^2}{k^3}\frac{N(G^\star)^2}{2\rho\lambda_2(\mathcal{L})} - \frac{2}{k}(\Psi_k(x(k)) - \Psi_k^\star). \end{aligned} \tag{6.26}$$

We next replace the term $\left(\Psi_k(x(k)) - \Psi_k^\star\right)$ on the left hand side in (6.26) with its lower bound that involves $\left(\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star\right)$. Using the definition of the functions Ψ_k and Ψ_{k+1}, adding and subtracting $\Psi_{k+1}^\star + \frac{\rho}{2}x(k)^\top \mathcal{L} x(k)$, and using the relation $\Psi_{k+1}^\star \geq \Psi_k^\star$ (see Lemma 1, part (b)):

$$\begin{aligned} (\Psi_k(x(k)) - \Psi_k^\star) &= \sum_{i=1}^N f_i(x_i(k)) + \frac{\rho k}{2}x(k)^\top \mathcal{L} x(k) - \Psi_k^\star + \Psi_{k+1}^\star - \Psi_{k+1}^\star \\ &+ \frac{\rho}{2}x(k)^\top \mathcal{L} x(k) - \frac{\rho}{2}x(k)^\top \mathcal{L} x(k) \\ &= (\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star) - \frac{\rho}{2}x(k)^\top \mathcal{L} x(k) + (\Psi_{k+1}^\star - \Psi_k^\star) \\ &\geq (\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star) - \frac{\rho}{2}x(k)^\top \mathcal{L} x(k). \end{aligned} \tag{6.27}$$

Further, we replace the term $-\frac{2}{k}\left(\Psi_k(x(k)) - \Psi_k^\star\right)$ on the right hand side of (6.26) by an upper bound as follows. We express $\left(\Psi_k(x(k)) - \Psi_k^\star\right)$ in terms of $\left(\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi_{\lfloor k/2\rfloor}^\star\right)$ as follows:

$$\begin{aligned}(\Psi_k(x(k)) - \Psi_k^\star) &= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{\rho k}{2} x(k)^\top \mathcal{L} x(k) - \Psi_k^\star \\ &= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{\rho \lfloor k/2\rfloor}{2} x(k)^\top \mathcal{L} x(k) - \Psi_{\lfloor k/2\rfloor}^\star + \Psi_{\lfloor k/2\rfloor}^\star \\ &\quad - \Psi_k^\star + \frac{\rho(k - \lfloor k/2\rfloor)}{2} x(k)^\top \mathcal{L} x(k) \\ &= (\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi_{\lfloor k/2\rfloor}^\star) - (\Psi_k^\star - \Psi_{\lfloor k/2\rfloor}^\star) + \frac{\rho(k - \lfloor k/2\rfloor)}{2} x(k)^\top \mathcal{L} x(k).\end{aligned}$$

Thus, using $(\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi_{\lfloor k/2\rfloor}^\star) \geq 0$, and $k - \lfloor k/2\rfloor \geq k/2$, the term $\left(\Psi_k(x(k)) - \Psi_k^\star\right)$ is bounded from above as:

$$(\Psi_k(x(k)) - \Psi_k^\star) \geq -(\Psi_k^\star - \Psi_{\lfloor k/2\rfloor}^\star) + \rho \frac{(k/2)}{2} x(k)^\top \mathcal{L} x(k),$$

or, equivalently:

$$-\frac{2}{k}(\Psi_k(x(k)) - \Psi_k^\star) \leq \frac{2}{k}(\Psi_k^\star - \Psi_{\lfloor k/2\rfloor}^\star) - \frac{\rho}{2} x(k)^\top \mathcal{L} x(k).$$

Next, by Lemma 1, parts (c) and (d), the term :

$$(\Psi_k^\star - \Psi_{\lfloor k/2\rfloor}^\star) = (\Psi_k^\star - f^\star) + (f^\star - \Psi_{\lfloor k/2\rfloor}^\star) \leq \frac{N(G^\star)^2}{2\rho\lambda_2(\mathcal{L})\lfloor k/2\rfloor}, \; k = 2, 3, ...,$$

which finally gives:

$$\begin{aligned}-\frac{2}{k}(\Psi_k(x(k)) - \Psi_k^\star) &\leq \left(\Psi_k^\star - \Psi_{\lfloor k/2\rfloor}^\star\right) - \frac{\rho}{2} x(k)^\top \mathcal{L} x(k) \\ &\leq \frac{N(G^\star)^2}{\rho\lambda_2(\mathcal{L}) k \lfloor k/2\rfloor} - \frac{\rho}{2} x(k)^\top \mathcal{L} x(k), \; k = 2, 3, ...\end{aligned} \tag{6.28}$$

Note that, for $k = 1$:

$$-\frac{2}{k}(\Psi_k(x(k)) - \Psi_k^\star) \leq 2(\Psi_1^\star - \Psi_0^\star) - \frac{\rho}{2} x(1)^\top \mathcal{L} x(1). \tag{6.29}$$

Now, applying the bounds (6.27) and (6.28) to (6.26), for $k = 2, 3, ...$:

$$(\Psi_{k+1}(x(k)) - \Psi^\star_{k+1}) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \le (\Psi_k(x(k-1)) - \Psi^\star_k) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 + \frac{N(G^\star)^2}{2\lambda_2(\mathcal{L})}\left(\frac{(k+1)^2}{2\,k^3} + \frac{1}{k\lfloor k/2\rfloor}\right),$$

which gives:

$$\begin{aligned}(\Psi_{k+1}(x(k)) - \Psi^\star_{k+1}) \;&\le\; (\Psi_k(x(k-1)) - \Psi^\star_k) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 - \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \\ &+\; \left[\frac{(k+1)^2}{2\,k^3} + \frac{1}{k\lfloor k/2\rfloor}\right]\frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})},\; k = 2, 3, ... \end{aligned} \tag{6.30}$$

Also, for $k = 1$:

$$\begin{aligned}(\Psi_2(x(1)) - \Psi^\star_2) \;&\le\; (\Psi_1(x(0)) - \Psi^\star_1) + \frac{2}{c}\|v(0) - x^\star \mathbf{1}\|^2 - \frac{2}{c}\|v(1) - x^\star \mathbf{1}\|^2 \\ &+\; \frac{(1+1)^2}{2\cdot 1^3}\frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})} + 2(\Psi^\star_1 - \Psi^\star_0). \end{aligned} \tag{6.31}$$

Finally, by telescoping (6.30) and (6.31), and using the definition of S_{k+1}:

$$\begin{aligned}(\Psi_{k+1}(x(k)) - \Psi^\star_{k+1}) \le \Psi_1(x(0)) - \Psi^\star_1 + (2/c)\|v(0) - x^\star \mathbf{1}\|^2 \\ + \frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})}\,[S_{k+1}] + 2(\Psi^\star_1 - \Psi^\star_0). \end{aligned} \tag{6.32}$$

Use equality $x(0) = v(0) = \overline{x}(0)\mathbf{1}$; $\Psi^\star_1 \le f^\star$; $\Psi^\star_0 \ge \sum_{i=1}^N f^\star_i$; $\Psi^\star_1 \ge \sum_{i=1}^N f^\star_i$; and $\Psi_1(\overline{x}(0)\mathbf{1}) = f(\overline{x}(0))$. Substituting the latter findings in (6.32), we get the desired result.

Bounding gradients by $O(\log k)$

We now use Lemma 2 to show that the gradients $\|\nabla f_i(y_i(k))\| = O(\log K)$, $k = 1, ..., K$. Denote by:

$$C_f := \left(f^\star - \min_{i=1,...,N}\sum_{j\neq i} f^\star_j\right) + 3\left(b\,N\,B - \sum_{i=1}^N f^\star_i\right) + f(\overline{x}(0)) + \frac{2\,N\,R^2}{c} + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})}. \tag{6.33}$$

Lemma 3. *Consider algorithm (6.12)–(6.15), and denote by:*

$$C_{\text{grad}} := 3L \max\left\{B, \frac{1}{b}C_f\right\} + L\|\overline{x}(0)\| + \max_{i=1,\dots,N} \|\nabla f_i(y_i(\overline{x}(0)))\|. \tag{6.34}$$

Then, for all $k = 0, 1, ..., K$:

$$\|\nabla f_i(y_i(k))\| \leq C_{\text{grad}}\, S_K.$$

Fix arbitrary node $i \in \{1, 2, ..., N\}$. By Lemma 2, and using $f_j(x_i(k)) \geq f_j^\star$, $j \neq i$:

$$\begin{aligned} f_i(x_i(k)) &\leq \left(f^\star - \sum_{j\neq i}^{N} f_j^\star\right) + \frac{2NR^2}{c} + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})} S_k + 3\left(bNB - \sum_{i=1}^{N} f_i^\star\right) \\ &\leq C_f\, S_k \leq C_f\, S_K, \end{aligned}$$

because $S_k \geq 1$ for all $k = 1, ..., K$, and $S_k \leq S_K$, for all $k = 1, ..., K$. Next, using Assumption 10:

$$\|x_i(k)\| \leq \max\{B, (1/b)C_f\}\; S_K.$$

which, because $\|y(k)\| = \|x(k) + \frac{k-1}{k+2}(x(k) - x(k-1))\| \leq 2\|x(k)\| + \|x(k-1)\|$, gives:

$$\|y_i(k)\| \leq 3\, \max\{B, (1/b)C_f\}\; S_K, \tag{6.35}$$

Now, using the Lipschitz continuity of ∇f_i and the triangle inequality:

$$\begin{aligned} \|\nabla f_i(y_i(k))\| &= \|\nabla f_i(y_i(k)) - \nabla f_i(y_i(0)) + \nabla f_i(y_i(0))\| \\ &\leq \|\nabla f_i(y_i(k)) - \nabla f_i(y_i(0))\| + \|\nabla f_i(y_i(0))\| \\ &\leq L\|y_i(k) - y_i(0)\| + \|\nabla f_i(y_i(0))\| \\ &\leq L\|y_i(k)\| + L\|y_i(0)\| + \|\nabla f_i(y_i(0))\|. \end{aligned}$$

The latter gives the desired result using the bound (6.57), the inequalities $\|y_i(0)\| = \|x_i(0)\| = \|\overline{x}(0)\|$, $\|\nabla f_i(y_i(0))\| \leq \max_{i=1,\dots,N} \|\nabla f_i(\overline{x}(0))\|$, and using $S_K \geq 1$.

Optimality gap $O(\log^3 k/k)$: Bounding the function values by $O(1)$

We are now ready to prove the $O(\log^3 k/k)$ rate of convergence, as well as the bounded gradients result.

Theorem 6.1 (The $O(\log^3 k/k)$ rate of convergence under the growth as-

sumption). *Consider algorithm (6.12)–(6.15). Then, for all nodes i, the optimality gap $\frac{1}{N}(\, f(x_i(k)) \leq f^\star\,) = O(\log^3 k/k)$; more precisely:*

$$\begin{aligned} & \frac{1}{N}\left(f(x_i(k)) - f^\star\right) \qquad (6.36) \\ \leq \quad & \frac{2\,R^2}{c} + 16\,LC_{\text{cons}}^2 C_{\text{grad}}^2\,\frac{S_k^2}{k}\sum_{t=1}^{k}\frac{(t+1)^2}{t^3} + C_{\text{grad}}^2\,C_{\text{cons}}\,\frac{S_k^2}{k},\ k=1,2,... \end{aligned}$$

Lemma 4. *Consider algorithm (6.12)–(6.15). Then, for all nodes i, the optimality gap $f(x_i(k)) \leq f^\star = O(\log^3 k/k)$; more precisely:*

$$f(x_i(k)) \leq f^\star + \frac{2NR^2}{c} + a_1\,LC_{\text{cons}}^2 C_{\text{grad}}^2 + a_2\,NC_{\text{grad}}^2 C_{\text{cons}},$$

where a_1, a_2 are universal constants independent of system parameters.

[Proof of Theorem 6.1]

Recall that, to establish the optimality gap at iteration k, the proof of this Theorem actually required only that the gradients $\|\nabla f_i(y_i(t))\|$ be bounded, for all $t = 0, 1, ..., k$. Hence, for a fixed k, we can replace the unform bound on the gradients G with a bound G_k that satisfies: $\|\nabla f_i(y_i(t))\| \leq G_k$, for all $t = 0, 1, ..., k$. We can use $G_k = C_{\text{grad}}\,S_k$, with S_k in (6.20) and C_{grad} in (6.34). Thus, Theorem 6.1 follows.

[Proof of Lemma 4] Lemma 4 follows after maximizing the right hand side in (6.36) over $k \geq 1$, i.e., after calculating that:

$$16\,\max_{k\geq 1}\left\{\frac{S_k^2}{k}\sum_{t=1}^{k}\frac{(t+1)^2}{t^3}\right\} \leq 2000, \quad \max_{k\geq 1}\left\{\frac{S_k^2}{k}\right\} \leq 50.$$

Improving convergence rate to $O(\log k/k)$

It is clear that we can now improve convergence rate to $O(\log k/k)$. As the function values $f(x_i(k))$ are uniformly bounded by a constant for all k, we proceed like in the proof of Lemma 3, and conclude that the gradients $\nabla f_i(y_i(k))$ are uniformly bounded by a constant, i.e., it holds that: $\|\nabla f_i(y_i(k))\| \leq C'_{\text{grad}}$, for all i, for all k, for a certain constant C'_{grad}. Hence, we obtain the $O(\log k/k)$ convergence rate, as desired.

6.4 Projected mD–NC method: Constrained optimization

6.4.1 Model and algorithm

We consider constrained optimization problem:

$$\text{minimize } \sum_{i=1}^{N} f_i(x) =: f(x) \text{ subject to } x \in \mathcal{X}, \tag{6.37}$$

and let the f_i's and $\mathcal{X}$ obey the following.

Assumption 11. *1. The set $\mathcal{X}$ is nonempty, convex, and compact with $\|x\| \leq B$, for all $x \in \mathcal{X}$ for some $B \in (0, \infty)$.*

2. For all i, $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ is convex, continuously differentiable, with Lipschitz continuous gradient with constant L on the set $\mathcal{X}' := \{x \in \mathbb{R}^d : \|x\| \leq 3B\}$:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \quad \text{for all } x, y \in \mathcal{X}'.$$

By Assumption 11, there exists a solution $x^\star \in \mathcal{X}$ with $f(x^\star) = f^\star = \inf_{x \in \mathcal{X}} f(x)$, and the solution set $\{x^\star \in \mathcal{X} : f(x^\star) - f^\star\}$ is compact. Also, the gradient $\nabla f_i(x)$ is bounded over the set $\mathcal{X}'$, i.e., there exists a constant $G \in [0, \infty)$ such that $\|\nabla f_i(x)\| \leq G$, for all $x \in \mathcal{X}'$. Assumption 11 encompasses many costs f_i's; e.g., any f_i that is twice continuously differentiable on $\mathbb{R}^d$ obeys Assumption 11 (b) with constant $L = \max_{x \in \mathcal{X}'} \|\nabla^2 f_i(x)\|$.

The projected mD–NC algorithms operates in two time scales, i.e., it has inner iterations s and outer iterations k. There are τ_k inner iterations at the outer iteration s, with τ_k specified further ahead. We capture the communication pattern at (k, s) by the random matrix $W(k, s)$ that obeys the following.

Assumption 12. *The matrices $W(k, s)$ are:*

1. Mutually independent and identically distributed;

2. Stochastic, symmetric, with positive diagonals, almost surely;

3. For all $i, j = 1, ..., N$, almost surely, $W_{ij}(k, s) \in \{0\} \cup [\underline{w}, 1]$;

4. The graph is connected on average, i.e., $\|\mathbb{E}[W(k, s)] - J\| < 1$.

Denote by $\overline{\mu} := \left(\|\mathbb{E}\left[W(k,s)^2\right] - J\|\right)^{1/2}$, and introduce, for future reference, the following matrices:

$$\mathcal{W}(k) = W(k,\tau_k)W(k,\tau_k-1)...W(k,1) \quad \text{and} \quad \widetilde{\mathcal{W}}(k) := \mathcal{W}(k) - J. \tag{6.38}$$

Projected mD–NC algorithm

The projected mD–NC algorithm is summarized in Algorithm 4. See also [34] for an unconstrained variant of the method. The step-size $\alpha \le 1/(2L)$.

Algorithm 4: The projected mD–NC algorithm

1: Initialization: Node i sets: $x_i(0) = y_i(0) \in \mathbb{R}^d$; and $k = 1$.

2: Node i calculates: $x_i^{(a)}(k) = y_i(k-1) - \alpha\nabla f_i(y_i(k-1))$.

3: (Consensus) Nodes run average consensus on a $2d \times 1$ variable $\chi_i(s,k)$, initialized by $\chi_i(s=0,k) = (x_i^{(a)}(k)^\top, x_i(k-1)^\top)^\top$:

$$\chi_i(s,k) = \sum_{j\in O_i(k)} W_{ij}(k,s)\,\chi_j(s-1,k),\ s = 1,...,\tau_k, \quad \tau_k = \left\lceil \frac{4\log k + \log N}{-\log\bar{\mu}} \right\rceil, \tag{6.39}$$

and set $x_i^{(c)}(k) := [\chi_i(s=\tau_k,k)]_{1:d}$ and $x_i^{(b)}(k-1) := [\chi_i(s=\tau_k,k)]_{d+1:2d}$. (Here $[a]_{l:m}$ is a selection of l-th, $l+1$-th, ..., m-th entries of vector a.)

4: Node i calculates:

$$x_i(k) := P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}.$$

5: Node i calculates:

$$y_i(k) = (1+\beta_{k-1})x_i(k) - \beta_{k-1}\,x_i^{(b)}(k-1).$$

6: Set $k \mapsto k+1$ and go to step 2.

6.4.2 Framework of Inexact Nesterov gradient method

Throughout this Subsection, we consider the (centralized) constrained minimization of a function $f(x)$ subject to $x \in \mathcal{X}$, where $f : \mathbb{R}^d \to \mathbb{R}$ is convex, and $\mathcal{X} \subset \mathbb{R}^d$ is a nonempty, closed, convex set.

Definition 6.1 (Inexact oracle)**.** Consider a convex function $f : \mathbb{R}^d \to \mathbb{R}$ and a nonempty, closed, convex set $\mathcal{X}$. We say that a pair $\left(\widehat{f}_y, \widehat{g}_y\right) \in \mathbb{R} \times \mathbb{R}^d$ is a

(L_y, δ_y) inexact oracle of f at point $y \in \mathbb{R}^d$ over the set $\mathcal{X}$ if:

$$f(x) \geq \widehat{f}_y + \widehat{g}_y^\top (x - y), \quad \text{for all } x \in \mathcal{X} \tag{6.40}$$

$$f(x) \leq \widehat{f}_y + \widehat{g}_y^\top (x - y) + \frac{L_y}{2}\|x - y\|^2 + \delta_y, \quad \text{for all } x \in \mathcal{X}. \tag{6.41}$$

We give a couple of remarks with respect to Definition 6.1. First, in Definition 6.1, we require that x belongs to $\mathcal{X}$, while y may lie outside $\mathcal{X}$. Second, throughout we just use the wording "inexact oracle at y" rather than "inexact oracle of f at y over $\mathcal{X}$," as the set $\mathcal{X}$ and the function f are understood from context. Finally, if $\left(\widehat{f}_y, \widehat{g}_y\right)$ is a (L_y, δ_y) inexact oracle at y, then it is also a $\left(L'_y, \delta_y\right)$ inexact oracle at y, with $L'_y \geq L_y$.

We next give the definition of an inexact projection. First, denote the exact (Euclidean) projection of $y \in \mathbb{R}^d$ on $\mathcal{X}$ by $P_{\mathcal{X}}\{y\} = \arg\min_{z \in \mathcal{X}} \|z - y\|$.

Definition 6.2 (Inexact projection)**.** We say that $x \in \mathbb{R}^d$ is a ζ-inexact projection of $y \in \mathbb{R}^d$ on $\mathcal{X}$ if: 1) $x \in \mathcal{X}$; and 2) $\|x - P_{\mathcal{X}}\{y\}\| \leq \zeta$.

Inexact projected Nesterov gradient. We consider the following inexact iteration of the Nesterov gradient method to minimize $f(x)$ over $\mathcal{X}$. For a given point $(\overline{x}(k-1), \overline{y}(k-1)) \in \mathcal{X} \times \mathbb{R}^d$, let $\left(\widehat{f}_{k-1}, \widehat{g}_{k-1}\right)$ be a (L_{k-1}, δ_{k-1}) inexact oracle at $\overline{y}(k-1)$; further, let $\widehat{\mathcal{P}}_k\left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\}$ be a ζ_{k-1}-inexact projection of $\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}$. Construct $\overline{x}(k), \overline{y}(k)$ as:

$$\overline{x}(k) = \widehat{\mathcal{P}}_k\left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\}, \quad \overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right) \tag{6.42}$$

With respect to (6.42), we are interested in two choices of $\mathcal{X}$: (a) compact constraint set $\mathcal{X}$, in which case ζ_{k-1} may be non-zero; and (b) unconstrained optimization $\mathcal{X} = \mathbb{R}^d$, in which case we assume $\zeta_{k-1} = 0$.

Lemma 5 (Progress per iteration)**.** *Consider* (6.42) *for some* $k = 1, 2, \ldots$ *and let* $x^\bullet$ *be arbitrary point in* $\mathcal{X}$*. Then:*

1. ***Compact constraint set.*** *If* $\mathcal{X}$ *is compact with* $\|x\| \leq B$, for all $x \in \mathcal{X}$, *we have:*

$$\begin{aligned} & (k+1)^2\left(f(\overline{x}(k)) - f(x^\bullet)\right) + 2L_{k-1}\|\overline{v}(k) - x^\bullet\|^2 && (6.43)\\ \leq\ & (k^2-1)\left(f(\overline{x}(k-1)) - f(x^\bullet)\right) + 2L_{k-1}\|\overline{v}(k-1) - x^\bullet\|^2 && (6.44)\\ +\ & (k+1)^2\delta_{k-1} + (k+1)^2\eta_{k-1}, \end{aligned}$$

where $\theta_k = 2/(k+2)$ *and*

$$\overline{v}(k-1) = \frac{\overline{y}(k-1) - (1-\theta_{k-1})\overline{x}(k-1)}{\theta_{k-1}} \tag{6.45}$$

$$\eta_{k-1} = L_{k-1}\zeta_{k-1}^2 + L_{k-1}\left(6B + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}}\right)\zeta_{k-1} \tag{6.46}$$

2. ***Unconstrained case.*** *If* $\mathcal{X} = \mathbb{R}^d$ *and* $\zeta_{k-1} = 0$*, then* (6.43) *holds with* $\eta_{k-1} = 0$*.*

6.4.3 Convergence rate analysis

Inexact oracle framework

To analyze the convergence rate of the projected mD–NC algorithm, we use the framework of inexact projected Nesterov gradient method [32]. We consider the global average $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^N x_i(k)$, the disagreement at node i: $\widetilde{x}_i(k) := x_i(k) - \overline{x}(k)$, and the aggregate quantities $x(k) := (x_1(k)^\top, ..., x_N(k)^\top)^\top$ and $\widetilde{x}(k) := (x_1(k)^\top, ..., x_N(k)^\top)^\top$. We also consider the counterparts for $y_i(k)$, $x_i^{(a)}(k)$, $x_i^{(b)}(k)$, and $x_i^{(c)}(k)$, defined analogously.

We next derive the update equation for $(\overline{x}(k), \overline{y}(k))$. From Algorithm 4, steps 2 and 3, we have that $\overline{x}^{(a)}(k) = \overline{x}^{(c)}(k) = \overline{y}(k) - \frac{\alpha}{N}\sum_{i=1}^N \nabla f_i(y_i(k-1))$; from the latter and steps 4 and 5:

$$\overline{x}(k) = \widehat{\mathcal{P}}_k\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^N \nabla f_i(y_i(k-1))\right\} \tag{6.47}$$

$$\overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right), \tag{6.48}$$

where we define the inexact projection $\widehat{\mathcal{P}}_k$ by:

$$\widehat{\mathcal{P}}_k\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^N \nabla f_i(y_i(k-1))\right\} = \widehat{\mathcal{P}}_k\left\{\overline{x}^{(c)}(k)\right\} := \frac{1}{N}\sum_{i=1}^N P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}. \tag{6.49}$$

As with mD–NC for unconstrained optimization, algorithm (6.47)–(6.48) can be viewed as an inexact projected Nesterov gradient algorithm. Both the "gradient direction" and the projection step are inexact. With respect to "gradient direction" inexactness, it can be shown that we have that the "amount of inexactness" is $\delta_{k-1} := L\|\widetilde{y}(k-1)\|^2$. The next Lemma quantifies the projection inexactness.

Lemma 6. *Consider the projected mD–NC algorithm with step size $\alpha \leq 1/(2L)$. Then, $\overline{x}(k)$ is ζ_{k-1}–inexact projection of $\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))$, with*

$$\|\zeta_{k-1}\| \leq \frac{1}{\sqrt{N}}\|\widetilde{x}^{(c)}(k)\|. \tag{6.50}$$

That is: 1) $\overline{x}(k) \in \mathcal{X}$, and 2) $\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))\right\}\| \leq \zeta_{k-1}$.

We first prove claim 1 ($\overline{x}(k) \in \mathcal{X}$). Note that $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^{N} P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$, and so it belongs to $\mathcal{X}$ as a convex combination of the points that belong to $\mathcal{X}$. We next prove claim 2. Using $\overline{x}^{(c)}(k) = \overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))$, equations (6.47) and (6.49), and expressing $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^{N} P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$:

$$\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))\right\}\| \quad \leq \tag{6.51}$$

$$\frac{1}{N}\sum_{i=1}^{N}\|P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\} - P_{\mathcal{X}}\left\{\overline{x}^{(c)}(k)\right\}\|. \tag{6.52}$$

Consider the right hand side in (6.51). Expressing $x_i^{(c)}(k) = \overline{x}^{(c)}(k) + \widetilde{x}_i^{(c)}(k)$, and using the non-expansiveness property of projection: $\|P_{\mathcal{X}}\{u\} - P_{\mathcal{X}}\{v\}\| \leq \|u - v\|$, for all $u, v \in \mathbb{R}^d$, obtain:

$$\left\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))\right\}\right\| \leq \frac{1}{N}\sum_{i=1}^{N}\left\|\widetilde{x}_i^{(c)}(k)\right\| \tag{6.53}$$

$$\leq \frac{1}{\sqrt{N}}\left\|\widetilde{x}^{(c)}(k)\right\|, \tag{6.54}$$

where the last inequality follows by convexity of $u \mapsto u^2$: $\left(\frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_i^{(c)}(k)\|\right)^2 \leq \frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_i^{(c)}(k)\|^2 = \frac{1}{N}\|\widetilde{x}^{(c)}\|^2$.

Disagreement estimate

We next find the bounds on $\|\widetilde{y}(k)\|$ and $\|\widetilde{x}^{(c)}(k)\|$, in order to characterize the oracle inexactness δ_k and the projection inexactness ζ_k.

Lemma 7. *Consider the projected mD–NC algorithm , and the step size*

$\alpha \le 1/(2L)$. *Then, for all* $k = 1, 2, ...$:

$$\left(\mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|\right]\right)^2 \le \mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|^2\right] \le \frac{N\,(3B+\alpha G)^2}{k^8} \tag{6.55}$$

$$\left(\mathbb{E}\left[\|\widetilde{y}(k)\|\right]\right)^2 \le \mathbb{E}\left[\|\widetilde{y}(k)\|^2\right] \le \frac{5\,N\,(3B+\alpha G)^2}{k^8}. \tag{6.56}$$

The left inequalities in (6.55) and (6.56) follow, e.g., from the Jensen inequality $h\,(\mathbb{E}[Z]) \le \mathbb{E}[h(Z)]$, with $h(z) = z^2$.

We now prove the two right inequalities. We conduct the proof for $d = 1$, while the extension to generic d is straightforward.

The proof has four steps. In Step 1, we upper bound $\|y(k)\|$. In Step 2, we prove (6.55). In Step 3, we upper bound $\mathbb{E}\left[\|\widetilde{x}(k)\|^2\right]$. Finally, in Step 4, we prove (6.56).

Step 1: Bounding $\|y(k)\|$. We first prove a bound for $\|\widetilde{y}(k)\|$. Consider step 3 in Algorithm 4 and fix arbitrary node i. Note that $x_i^{(b)}(k-1)$ belongs to $\mathcal{X}$, because it is a convex combination of the points $x_j(k-1)$, $j = 1, ..., N$, that belong to the set $\mathcal{X}$. Next, using $|\beta_{k-1}| \le 1$:

$$\|y_i(k)\| \le 2\|x_i(k)\| + \|x_i^{(b)}(k-1)\| \le 3B,$$

as $x_i(k), x_i^{(b)}(k-1) \in \mathcal{X}$, for all k. Thus, we obtain the desired bound:

$$\|y(k)\| \le 3\sqrt{N}B, \text{ for all } k. \tag{6.57}$$

Step 2: Proof of (6.55). Recall the definition of $\mathcal{W}(k)$ in (6.38). and $\widetilde{\mathcal{W}}(k) = \mathcal{W}(k) - J$. From steps 2 and 3 in Algorithm 4, note that $\widetilde{x}^{(c)}(k)$ can be written as:

$$\widetilde{x}^{(c)}(k) = \widetilde{\mathcal{W}}(k)(I-J)\left(y(k-1) - \alpha\nabla F(y(k-1))\right).$$

Take the norm, use the sub-multiplicative and sub-additive properties of norms, and square the obtained inequality. Further, use $\|I - J\| = 1$, inequality (6.57), $\|\nabla F(y(k-1))\| \le \sqrt{N}G$, and the Jensen inequality, to obtain (6.55).

Step 3: Upper bounding $\mathbb{E}\left[\|\widetilde{x}(k)\|^2\right]$. For $\widetilde{x}_i(k) := x_i(k) - \overline{x}(k)$, using

$x_i(k) = P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$ and $\overline{x}(k) = \frac{1}{N}\sum_{j=1}^{N} P_{\mathcal{X}}\{x_j(k)\}$, we have:

$$\begin{aligned}
\|\widetilde{x}_i(k)\| &= \left\|P_{\mathcal{X}}\{x_i^{(c)}(k)\} - \frac{1}{N}\sum_{j=1}^{N} P_{\mathcal{X}}\{x_j^{(c)}(k)\}\right\| && (6.58)\\
&= \left\|\frac{1}{N}\sum_{j=1}^{N}\left(P_{\mathcal{X}}\{x_i^{(c)}(k)\} - P_{\mathcal{X}}\{x_j^{(c)}(k)\}\right)\right\| \\
&\leq \frac{1}{N}\sum_{j=1}^{N}\left\|P_{\mathcal{X}}\{x_i^{(c)}(k)\} - P_{\mathcal{X}}\{x_j^{(c)}(k)\}\right\| && (6.59)\\
&\leq \frac{1}{N}\sum_{j=1}^{N}\left\|x_i^{(c)}(k) - x_j^{(c)}(k)\right\| && (6.60)\\
&\leq \frac{1}{N}\sum_{j=1}^{N}\left(\left\|\widetilde{x}_i^{(c)}(k)\right\| + \left\|\widetilde{x}_j^{(c)}(k)\right\|\right) && (6.61)\\
&\leq \left\|\widetilde{x}_i^{(c)}(k)\right\| + \frac{1}{\sqrt{N}}\left\|\widetilde{x}^{(c)}(k)\right\|. && (6.62)
\end{aligned}$$

The left inequality in (6.59) is by convexity of norms, while the right inequality is by the non-expansiveness of the Euclidean projection: $\|P_{\mathcal{X}}\{a\} - P_{\mathcal{X}}\{b\}\| \leq \|a-b\|$, for all $a, b \in \mathbb{R}^d$. The left inequality in (6.61) is by expressing $\|x_i^{(c)}(k) - x_j^{(c)}(k)\| = \|(x_i^{(c)}(k) - \overline{x}^{(c)}(k)) + (\overline{x}^{(c)}(k) - x_j^{(c)})\| \leq \|x_i^{(c)}(k) - \overline{x}^{(c)}(k)\| + \|\overline{x}^{(c)}(k) - x_j^{(c)})\|$; and the right inequality in (6.61) is by $\left(\frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_j^{(c)}(k)\|\right)^2 \leq \frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_j^{(c)}(k)\|^2 = \frac{1}{N}\|\widetilde{x}^{(c)}(k)\|^2$. Summing the squared right inequalities in (6.61) over $i = 1, ..., N$, and using

$$\left(\left\|\widetilde{x}_i^{(c)}(k)\right\|^2 + \frac{1}{\sqrt{N}}\left\|\widetilde{x}^{(c)}(k)\right\|\right)^2 \leq 2\left\|\widetilde{x}_i^{(c)}(k)\right\|^2 + \frac{2}{N}\left\|\widetilde{x}^{(c)}(k)\right\|^2,$$

we obtain:

$$\|\widetilde{x}(k)\|^2 \leq 4\|\widetilde{x}^{(c)}(k)\|^2.$$

Thus, from (6.55), we obtain the desired bound:

$$\left(\mathbb{E}\left[\|\widetilde{x}(k)\|\right]\right)^2 \leq \mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|^2\right] \leq \frac{4N(3B + \alpha G)^2}{k^8}. \qquad (6.63)$$

Step 4: Proof of (6.56). From step 5 in Algorithm 4, we have:

$$\begin{aligned}
\widetilde{y}(k) &= (1+\beta_{k-1})\widetilde{x}(k) - \beta_{k-1}\widetilde{x}^{(b)}(k-1) \\
&= (1+\beta_{k-1})\widetilde{x}(k) - \beta_{k-1}\widetilde{\mathcal{W}}(k)\,(I-J)\,x(k-1).
\end{aligned}$$

Thus, using $\|\beta_{k-1}\| \leq 1$:

$$\|\widetilde{y}(k)\| \quad \leq \quad 2\,\|\widetilde{x}(k)\| + \|\widetilde{\mathcal{W}}(k)\|\,\|\widetilde{x}(k-1)\|.$$

Squaring the latter inequality, using $(a+b)^2 \leq 2a^2 + 2b^2$, (6.63), and $\|(I - J)x(k-1)\| \leq \sqrt{N}B$, we obtain:

$$\|\widetilde{y}(k)\|^2 \leq 4\,\|\widetilde{x}(k)\|^2 + 2\|\widetilde{\mathcal{W}}(k)\|^2\,\|\widetilde{x}(k-1)\|^2.$$

Taking expectations, using (6.63), and using Jensen's inequality, we finally obtain (6.56).

Convergence rate

We are now ready to state the convergence rate result for the projected mD–NC algorithm.

Theorem 6.2. *Consider the projected mD–NC algorithm, with the constant step size $\alpha \leq 1/(2L)$. Let $\|\overline{x}(0) - x^\star\| \leq R$, $R \geq 0$. Then, after*

$$\mathcal{K} = \sum_{t=1}^{k} \tau_t \leq \frac{1}{-\log \bar{\mu}}\,(4(k+1)\log(k+1) + (k+1)\log N)$$

communication rounds, i.e., after k outer iterations, we have, at any node i:

$$\frac{\mathbb{E}\,[f(x_i(k)) - f^\star]}{N} \leq \frac{1}{k^2}\left(\frac{2}{\alpha}R^2 + a_1' LB^2 + a_2' L(6B + \alpha G)^2 + \alpha G^2\right), \quad (6.64)$$

$$k = 1, 2, ..., \quad (6.65)$$

where a_1' and a_2' are universal constants independent of system parameters.

[Proof outline] We apply Lemma 5 (a) with $x^\bullet \equiv x^\star$. Further, as $\delta_k \leq L\|\widetilde{y}(k)\|^2$. we have, by Lemma 6, and Lemma 7: $\delta_k \leq L\|\widetilde{y}(k)\|^2 = \frac{9NLB^2}{k^8}$, $\zeta_k \leq \frac{\|\widetilde{x}^{(b)}(k)\|}{\sqrt{N}} \leq \frac{3B+\alpha G}{k^4}$. Finally, set $L_{k-1} \equiv N/\alpha$, and note that $\|\widehat{g}_k\| = \|\sum_{i=1}^N \nabla f_i(y_i(k))\| \leq NG$, for all k, as $y_i(x) \in \mathcal{X}'$, for all k. We now have all the relevant quantities set, and the proof proceeds by applying Lemma 5 (1).

6.5 Proof of Lemma 5 (1)

We perform the proof in three steps.

Step 1. We first prove the following auxiliary equality:

$$\theta_{k-1}\overline{v}(k) = \overline{x}(k) - (1-\theta_{k-1})\overline{x}(k-1). \tag{6.66}$$

Using the definition of $\overline{v}(k)$ in (6.45), $\theta_k = 2/(k+2)$, and $\beta_{k-1} = (k-1)/(k+2)$:

$$\begin{aligned}\overline{v}(k) &= \frac{k+2}{2}\left(\overline{x}(k) + \frac{k-1}{k+2}\overline{x}(k) - \frac{k-1}{k+2}\overline{x}(k-1) - \frac{k}{k+2}\overline{x}(k)\right)\\ &= \frac{k+1}{2}\overline{x}(k) - \frac{k-1}{2}\overline{x}(k-1).\end{aligned}$$

Multiplying the expression on the right hand side of the last equality by $\theta_{k-1} = 2/(k+1)$, the result follows.

Step 2. We prove the following relation:

$$f(\overline{x}(k)) \le f(z) + L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^\top (z - \overline{x}(k)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 \tag{6.67}$$

$$+ \delta_{k-1} + \eta_{k-1}, \quad \text{for all } z \in \mathcal{X}. \tag{6.68}$$

Because $\overline{x}(k) \in \mathcal{X}$ (by construction), we have, using (6.41):

$$f(\overline{x}(k)) \le \widehat{f}_{k-1} + \widehat{g}_{k-1}^\top (\overline{x}(k) - \overline{y}(k-1)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1}. \tag{6.69}$$

Denote by $p := P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\}$. We next upper bound the term

$$\Pi(z) := L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - \overline{x}(k)\right)^\top (\overline{x}(k) - z),$$

for arbitrary $z \in \mathcal{X}$. Adding and subtracting p in the second and third factors of $\Pi(z)$, obtain:

$$\begin{aligned}\Pi(z) &= L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\right)^\top (p - z)\\ &+ L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\right)^\top (\widehat{x}(k) - p) \qquad (6.70)\\ &+ L_{k-1}(p - \widehat{x}(k))^\top (p - z) - L_{k-1}\|p - \widehat{x}(k)\|^2\\ &\ge -L_{k-1}\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|\,\|\widehat{x}(k) - p\| \qquad (6.71)\\ &- L_{k-1}\|p - \widehat{x}(k)\|\,\|p - z\| - L_{k-1}\|p - \widehat{x}(k)\|^2. \qquad (6.72)\end{aligned}$$

The inequality follows by: 1) upper bounding the last three summands of

$\Pi(z)$ via $u^\top v \geq -\|u\|\,\|v\|$, for all $u, v \in \mathbb{R}^d$; and 2) using the fact that the first summand is nonnegative by the following projection property: $(w - P_{\mathcal{X}}\{w\})^\top (p_{\mathcal{X}}\{w\} - z) \geq 0$, for all $z \in \mathcal{X}$. We next upper bound $\|\overline{x}(k) - p\|$, $\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|$, and $\|p - z\|$. Upper bound $\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|$ using the sub-additive property of norms:

$$\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\| \leq \|\overline{y}(k-1)\| + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}} + \|p\|.$$

Next, note that $|\beta_{k-1}| \leq 1$, and because $\overline{x}(k-1), \overline{x}(k) \in \mathcal{X}$: $\|\overline{y}(k-1)\| \leq 3B$. Also, because $\overline{x}(k), p \in \mathcal{X}$, we have $\|\overline{x}(k)\| \leq B$ and $\|p\| \leq B$. Using the latter bounds on $\|\overline{y}(k-1)\|$, $\|\overline{x}(k)\|$, and $\|p\|$:

$$\|\overline{x}(k) - p\| \leq \zeta_{k-1}, \;\; \|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\| \leq 4B + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}}, \;\; \|z - p\| \leq 2B, \quad (6.73)$$

where the bound on $\|\overline{x}(k) - p\|$ is by the algorithm construction. Applying (6.73) to (6.71), obtain:

$$0 \leq \Pi(z) + \eta_{k-1} = L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - \overline{x}(k)\right)^\top (\overline{x}(k) - z) + \eta_{k-1}, \quad (6.74)$$

where η_{k-1} is given in (6.46). From property (6.40): $\widehat{f}_{k-1} \leq f(z) + \widehat{g}_{k-1}^\top(\overline{y}(k-1) - z)$, and so, using the last equation and adding (6.69) and (6.74), the claim (6.67) follows.

Step 3. We finalize the proof of Lemma 5 by proving (6.43). We start by using relation (6.67). Namely: 1) setting $z = \overline{x}(k-1)$ in (6.67) and multiplying inequality (6.67) by $1 - \theta_{k-1}$; 2) setting $z = x^\bullet$ in (6.67) and multiplying inequality (6.67) by θ_{k-1}; and 3) adding the corresponding two

inequalities:

$$\begin{aligned}
& \theta_{k-1}\{f(\overline{x}(k)) - f(x^\bullet)\} + (1-\theta_{k-1})\{f(\overline{x}(k)) - f(\overline{x}(k-1))\} \\
= \; & \{f(\overline{x}(k)) - f(x^\bullet)\} - (1-\theta_{k-1})\{f(\overline{x}(k-1)) - f(x^\bullet)\} \\
\leq \; & \theta_{k-1}L_{k-1}\,(\overline{x}(k) - \overline{y}(k-1))^\top(x^\bullet - \overline{x}(k)) \qquad (6.75) \\
+ \; & (1-\theta_{k-1})L_{k-1}\,(\overline{x}(k) - \overline{y}(k-1))^\top(\overline{x}(k-1) - \overline{x}(k)) \\
+ \; & \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1} + \eta_{k-1} \\
= \; & L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^\top(\theta_{k-1}x^\bullet + (1-\theta_{k-1})\overline{x}(k-1) - \overline{x}(k)) \qquad (6.76) \\
+ \; & \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1} + \eta_{k-1} \\
= \; & \frac{L_{k-1}}{2}(2(\overline{x}(k) - \overline{y}(k-1))^\top(\theta_{k-1}x^\bullet + (1-\theta_{k-1})\overline{x}(k-1) - \overline{x}(k)) \\
+ \; & \|\overline{x}(k) - \overline{y}(k-1)\|^2) + \delta_{k-1} + \eta_{k-1}. \qquad (6.77)
\end{aligned}$$

Denote by:

$$\mathcal{M}_{k-1} = (2(\overline{x}(k)-\overline{y}(k-1))^\top(\theta_{k-1}x^\bullet+(1-\theta_{k-1})\overline{x}(k-1)-\overline{x}(k))+\|\overline{x}(k)-\overline{y}(k-1)\|^2).$$

Then, inequality (6.77) is written simply as:

$$\{f(\overline{x}(k)) - f(x^\bullet)\} - (1-\theta_{k-1})\{f(\overline{x}(k-1)) - f(x^\bullet)\} \leq \frac{L_{k-1}}{2}\mathcal{M}_{k-1} + \delta_{k-1} + \eta_{k-1}. \qquad (6.78)$$

Now, we simplify the expression for $\mathcal{M}_{k-1}$ as follows. Using the identity:

$$\|\overline{x}(k) - \overline{y}(k-1)\|^2 = 2(\overline{x}(k) - \overline{y}(k-1))^\top\overline{x}(k) + \|\overline{y}(k-1)\|^2 - \|\overline{x}(k)\|^2,$$

we have:

$$\begin{aligned}
\mathcal{M}_{k-1} &= 2(\overline{x}(k) - \overline{y}(k-1))^\top(\theta_{k-1}x^\bullet + (1-\theta_{k-1})\overline{x}(k-1)) - \|\overline{x}(k)\|^2 + \|\overline{y}(k-1)\|^2 \\
&= \|\overline{y}(k-1) - ((1-\theta_{k-1})\overline{x}(k-1) + \theta_{k-1}x^\bullet)\|^2 \qquad (6.79) \\
&- \|\overline{x}(k) - ((1-\theta_{k-1})\overline{x}(k-1) + \theta_{k-1}x^\bullet)\|^2 \\
&= \theta_{k-1}^2\|\overline{v}(k-1) - x^\bullet\|^2 - \theta_{k-1}^2\|\overline{v}(k) - x^\bullet\|^2, \qquad (6.80)
\end{aligned}$$

where the last equality follows by the definition of $\overline{v}(k-1)$ in (6.45) and by the identity (6.66). Now, combining (6.78) and (6.80):

$$\begin{aligned}
(f(\overline{x}(k)) - f(x^\bullet)) \; - \; & (1-\theta_{k-1})(f(\overline{x}(k-1)) - f(x^\bullet)) \\
\leq \; & \frac{L_{k-1}\theta_{k-1}^2}{2}\left(\|\overline{v}(k-1) - x^\bullet\|^2 - \|\overline{v}(k) - x^\bullet\|^2\right) + \delta_{k-1} + \eta_{k-1}.
\end{aligned}$$

Finally, multiplying the last equation by $\frac{4}{\theta_{k-1}^2}$, and using $\theta_{k-1} = 2/(k+1)$, we get the result.

The major part of this chapter is taken practically unaltered from the PhD thesis [18].

Chapter 7

Conclusion

This manuscript covered relevant concepts to help gain understanding of optimization methods for distributed non-smooth optimization. Chapter 1 reviewed relevant concepts and properties of convex functions. Chapter 2 considered elements of subgradient calculus and introduced subgradient methods. Chapter 3 was concerned with duality theory. The material in Chapters 1-3 provided a required background for understanding of design and analysis of parallel and distributed optimization methods for convex (non)smooth problems. Chapter 4 introduced some common communication and computational (optimization) models and provided several application examples, and it considered dual distributed methods. Chapter 5 considered primal (sub)gradient methods. Finally, Chapter 6 was concerned with the primal distributed methods based on the Nesterov gradient method.

Bibliography

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] J. Hiriart-Urruty and C. Lemarechal. *Fundamentals of Convex Analysis*. Springer, 2001.

[3] J. Hiriart-Urruty and C. Lemarechal. *Convex Analysis and Minimization Algorithms I and II*. Springer, New York, 1993.

[4] J. Xavier. Nonlinear optimization. 2018. Lecture slides, PhD course, Carnegie Mellon University and Instituto Superior Tecnico.

[5] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[6] A. Friedlander, N. Krejic, and N. Krklec Jerinkic. *Lectures on Fundamentals of Numerical Optimization*. University of Novi Sad Faculty of Sciences, 2019.

[7] Jonathan M. Borwein and Adrian S. Lewis. Convex analysis and nonlinear optimization, theory and examples, 2000.

[8] D.P. Bertsekas, A. Nedić, and A.E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific optimization and computation series. Athena Scientific, 2003.

[9] S. Boyd, J. Duchi, and L. Vandenberghe. *Subgradients*. Notes for EE364b, Stanford University, Spring 2014-2015, April 2018.

[10] S. Boyd and J. Park. *Subgradient Methods*. Notes for EE364b, Stanford University, Spring 2013-2014, May 2014.

[11] A. Nedic and D. Bertsekas. The effect of deterministic noise in subgradient methods. *Mathematical Programming*, 125:75–99, 2010.

[12] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs, 1989.

[13] D.P. Bertsekas. *Convex Optimization Theory*. Athena Scientific optimization and computation series. Athena Scientific, 2009.

[14] A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, Jan 2009.

[15] P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11:1663–1707, 2010.

[16] Qing Ling, Wei Shi, Gang Wu, and Alejandro Ribeiro. DLM: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15).

[17] Kun Yuan, Bicheng Ying, Xiaochuan Zhao, and Ali H. Sayed. Exact diffusion for distributed optimization and learning — Part I: Algorithm development. *IEEE Trans. Sig. Proc.*, 67(3):708–723, 2019.

[18] D. Jakovetic. Distributed optimization with applications. 2013. PhD Thesis, Carnegie Mellon University, Instituto Superior Tecnico.

[19] R. Bekkerman, M. Bilenko, and J. Langford. *Scaling up Machine Learning: Parallel and Distributed Approaches*. Cambridge University Press, 2011.

[20] V. Cevher, S. Becker, and M. Schmidt. Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics. *IEEE Signal Processing Magazine*, 31(5):32–43, Sep. 2014.

[21] K. I. Tsianos, S. Lawlor, and M. G. Rabbat. Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1543–1550, Oct 2012.

[22] S. Kar, J. M. F. Moura, and K. Ramanan. Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication. *IEEE Transactions on Information Theory*, 58(6):3575–3605, June 2012.

[23] P. D. Lorenzo and G. Scutari. Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136, June 2016.

[24] K. I. Tsianos, S. Lawlor, and M. G. Rabbat. Push-sum distributed dual averaging for convex optimization. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5453–5458, Dec 2012.

[25] A. Nedić and A. Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, March 2015.

[26] C. Xi and U. A. Khan. Dextra: A fast algorithm for optimization over directed graphs. *IEEE Transactions on Automatic Control*, 62(10):4980–4993, Oct 2017.

[27] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[28] D. Blatt and A. O. Hero. Energy based sensor network source localization via projection onto convex sets (pocs). *IEEE Trans. on Signal Processing*, 54(9):3614–3619, 2006.

[29] P. Di Lorenzo and S. Barbarossa. A bio-inspired swarming algorithm for decentralized access in cognitive radio. *IEEE Trans. on Signal Processing*, 2011.

[30] J. A. Bazerque and G. B. Giannakis. Distributed spectrum sensing for cognitive radio networks by exploiting sparsity. *IEEE Transactions on Signal Processing*, 58(3):1847–1862, March 2010.

[31] D. Bajovic, D. Jakovetic, J. M. F. Moura, J. Xavier, and B. Sinopoli. Large deviations performance of consensus+innovations detection with non-Gaussian observations. *IEEE Trans. Sig. Proc.*, 60(11):5987–6002, 2012.

[32] O. Devolder, F. Glineur, and Y. Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146:37–75, 2014.

[33] D. Jakovetic, J. Xavier, and J. M. F. Moura. Fast distributed gradient methods. *IEEE Trans. Autom. Contr.*, 59(5):1131–1146, May 2014.

[34] D. Jakovetic, J. Xavier, and J. M. F. Moura. Convergence rates of distributed Nesterov-like gradient methods on random networks. *IEEE Transactions on Signal Processing*, 62(4):868–882, February 2014.

[35] A. Nedic. Optimization I. lecture notes, August 2008.