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

Distributed Optimization Methods for Large Scale Unconstrained Optimization Problems

- PhD thesis -

Supervisor:
Prof. Dr. Nataša Krejić

Candidate:
Greta Malaspina

Novi Sad, 2022

УНИВЕРЗИТЕТ У НОВОМ САДУ **ОБРАЗАЦ – 5а**
ПРИРОДНО-МАТЕМАТИЧКИ ФАКУЛТЕТ

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА[1]

Врста рада:	Докторска дисертација
Име и презиме аутора:	Грета Маласпина
Ментор (титула, име, презиме, звање, институција)	Др Наташа Крејић, редовни професор, Универзитет у Новом Саду Природно-математички факултет
Наслов рада:	Методе дистрибуиране оптимизације за проблеме великих димензија без ограничења, Distributed Optimization Methods for Large Scale Unconstrained Optimization Problems
Језик публикације (писмо):	енглески
Физички опис рада:	Унети број: Страница: 213 Поглавља: 6 Референци: 75 Табела: 0 Слика: 15 Графикона: 0 Прилога: 1
Научна област:	Математика
Ужа научна област (научна дисциплина):	Нумеричка математика
Кључне речи / предметна одредница:	нумеричка оптимизација, дистрибуирана оптимизација, оптимизација без ограничења
Резиме на српском језику:	Многи савремени математички модели захтевају решавање проблема оптимизације на мрежи рачунарских чворова у кооперативном режиму, док растући интерес за велике скупове података и машинско учење захтева методе којима је могуће решити проблеме све већих димензија. Већина класичних оптимизационих метода није погодна за примену у дистрибуираном окружењу, те је потребно развити нове методе који могу да одговоре на изазове који потичу из дистрибуираног окружења. Ова теза је фокусирана на дистрибуиране методе за оптимизационе проблеме великих димензија. Резултати предављени у овој тези су допринос области дистрибуиране оптимизације у следећим аспектима. Прво је проширена анализа конвергенције за класу постојећих дистрибуираних метода првог реда, за случај комуникационих мрежа које се мењају у времену и за дужине корака које се мењају по времену без координације између комјутерских чворова. Ово проширење је од посебног значаја у практичним применама где су промене у комуникационим мрежама узроковане техничким проблемима у

[1] Аутор докторске дисертације потписао је и приложио следеће Обрасце:
5б – Изјава о ауторству;
5в – Изјава о истоветности штампане и електронске верзије и о личним подацима;
5г – Изјава о коришћењу.
Ове Изјаве се чувају на факултету у штампаном и електронском облику и не кориче се са тезом.

	комуникацији или у случају све значајнијих покретних сензорских мрежа. Једно од главних питања у методама дистрибуиране оптимизације је одређивање дужине корака. Наиме, класичне технике глобализације попут линијског претраживања нису применљиве у овом оквиру, а примена фиксне дужине корака често доводи до спорог метода. Сем тога, примена фиксног корака захтева познавање глобалних константи које се не могу једноставно оценити у дистрибуираном окружењу. Стога је у тези предложен дистрибуирани приближан Њутнов метод са адаптивном дужином корака која се може срачунати у диструбираном окружењу, а помоћу тако одређене дужине корака, уз уобичајене претпоставке о регуларности проблема, метод је глобално конвергентан и задржава локални ред конвергенције карактеристичан за централизовану оптимизацију. Системи линеарних једначина великих димензија су изазов и као независни проблеми и као део оптимизационих поступака другог реда. У општем случају, у свакој итерацији метода другог реда, попут приближног Њутновог метода који је већ поменут, потребно је решити, приближно или тачно, систем линеарних једначина да би се одредио правац претраживања. Дистрибуирано окружење и овде представља изазов. Методе непокретне тачке су познате као ефикасан начин решавања система линеарних једначина у централизованом окружењу. Овде је представљена класа дистрибуираних метода типа непокретне тачке која је прилагођена дистрибуираном окружењу. Показано је да поступци предложеног типа конвергирају за статичне и променљиве комуникационе мреже, а резултати о конвергенцији су аналогни резултатима у централизованом случају. У последњем делу тезе је размтран проблем најмањих квадрата веома велике димензије мотивисан проблемом дигитализације катастарских мапа. Веома велика димензија проблема представља главни проблем за примену класичних метода, док је ретка структура проблема природна могућност за примену паралелних метода. У тези је представљен приближни Левенберг-Маркардов метод за решавање ретких проблема најмањих квадрата. Ретка структура је искоришћена као основ за дефинисање стратегије типа непокретне тачке којом се одређује правац претраживања на начин који је погодан за паралелизацију. Представљена је теоријска анализа и показана глобална и локална конвергенција под класичним претпоставкама за овај тип проблема. Сви предложени методи су имплементирани и тестирани на релевантним тест примерима. Нумерички резултати су емпиријски потврдили теоријска тврђења. Поред тога, предложени методи су упоређени са најзначајнијим постојећим методама за одговарајуће класе проблема и показана је њихова компетитивност.
Датум прихватања теме од стране надлежног већа:	30.06.2022
Датум одбране: (Попуњава одговарајућа служба)	
Чланови комисије: (титула, име, презиме, звање, институција)	Председник: др Наташа Крклец Јеринкић Члан: др Душан Јаковетић Члан: др Наташа Крејић Члан: др Стефаниа Белавиа
Напомена:	

UNIVERSITY OF NOVI SAD
FACULTY OR CENTER

KEY WORD DOCUMENTATION[2]

Document type:	Doctoral dissertation
Author:	Greta Malaspina
Supervisor (title, first name, last name, position, institution)	Dr. Nataša Krejić, professor, University of Novi Sad Faculty of Sciences, Novi Sad
Thesis title:	Distributed Optimization Methods for Large Scale Unconstrained Optimization Problems
Language of text (script):	English
Physical description:	Number of: Pages: 213 Chapters: 6 References: 75 Tables: 0 Illustrations: 15 Graphs: 0 Appendices: 1
Scientific field:	Mathematics
Scientific subfield (scientific discipline):	Numerical mathematics
Subject, Key words:	numerical optimization, distributed optimization, unconstrained problems
Abstract in English language:	Many modern applications require networks of computational agents to solve optimization problems in a cooperative manner, while the growing interest in Big Data and machine learning calls for method that are able to solve problems of increasingly large dimension. This poses many challenges in the field of optimization as most classical methods are not suitable for the distributed framework: new algorithms need to be developed, that are able to deal with the practical limitations deriving from the distributed setting. This thesis focuses on distributed methods for large scale optimization. The contributions of this thesis to the area of distributed optimization are the following. First of all, we extend the convergence analysis of a class of existing first-order distributed methods to the case of time-varying networks and uncoordinated time-varying stepsizes. This extension is particularly relevant as changes in the communication network are common in practical applications due to possible technical failures in the communication and the increasing relevance of mobile sensor networks. One of the main issues for distributed optimization is the selection of the stepsize: classical globalization such as line search are not feasible in the

[2] The author of doctoral dissertation has signed the following Statements:
5б – Statement on the authority,
5в – Statement that the printed and e-version of doctoral dissertation are identical and about personal data,
5г – Statement on copyright licenses.
The paper and e-versions of Statements are held at he faculty and are not included into the printed thesis.

	multi-agent framework, while employing a fixed stepsize is known to often cause the method to be slow and usually requires the knowledge of the regularity constants of the problem, which can be hard to estimate distributedly. To overcome these issues we propose a distributed Inexact Newton method that relies on an adaptive choice of the stepsize, which can be computed distributedly and, under suitable regularity assumptions, ensures both global convergence and local fast convergence as in the centralized case. Systems of linear equations and large dimensions are a problem of interest by itself and as a part of second-order optimization methods. In general, in each iteration of the second-order method, like Inexact Newton method mentioned above, one has to solve a system of linear equations, either approximately or exactly, to get the search direction. Again, distributed environment represents a challenge. Fixed point methods are a known to be very effective in the centralized framework for linear system of large dimensions. We propose here a class of distributed fixed point methods that works in the distributed setting. We show that such methods converge for both static and time-varying network, achieving convergence results analogous to those that can be proved for the centralized case. Finally, in the last part of the thesis we consider the least square problems of very large dimension motivated by digitalization of cadastral maps. The dimension of the problem represent the main difficulty for classical method while the sparse structure presents an opportunity to be exploited in parallel computational framework. We present an Inexact Levenberg-Marquardt method for sparse least squares problem. The method exploits the underlying structure of the problem to define a fixed-point strategy for the computation of the search direction that is suitable for parallelization. Theoretical analysis is presented and we show both global and fast local convergence under the classical assumptions for least squares problems. All presented methods are implemented, and tested on relevant examples. The numerical results reveal that the theoretical results are confirmed by empirical evidence. Furthermore, the proposed methods are compared with the corresponding state-of-the-art methods and their competitiveness is demonstrated.
Accepted on Scientific Board on:	30.06.2022
Defended: (Filled by the faculty service)	
Thesis Defend Board: (title, first name, last name, position, institution)	President: Dr. Nataša Krklec Jerinkić Member: Dr. Dušan Jakovetić Member: Dr. Nataša Krejić Member: Dr. Stefania Bellavia
Note:	

Abstract

Many modern applications require networks of computational agents to solve optimization problems in a cooperative manner, while the growing interest in Big Data and machine learning calls for method that are able to solve problems of increasingly large dimension. This poses many challenges in the field of optimization as most classical methods are not suitable for the distributed framework: new algorithms need to be developed, that are able to deal with the practical limitations deriving from the distributed setting. This thesis focuses on distributed methods for large scale optimization.
The contributions of this thesis to the area of distributed optimization are the following. First of all, we extend the convergence analysis of a class of existing first-order distributed methods to the case of time-varying networks and uncoordinated time-varying stepsizes. This extension is particularly relevant as changes in the communication network are common in practical applications due to possible technical failures in the communication and the increasing relevance of mobile sensor networks.
One of the main issues for distributed optimization is the selection of the stepsize: classical globalization such as line search are not feasible in the multi-agent framework, while employing a fixed stepsize is known to often cause the method to be slow and usually requires the knowledge of the regularity constants of the problem, which can be hard to estimate distributedly. To overcome these issues we propose a distributed Inexact Newton method that relies on an adaptive choice of the stepsize, which can be computed distributedly and, under suitable regularity assumptions, ensures both global convergence and local fast convergence as in the centralized case.
Systems of linear equations and large dimensions are a problem of in-

terest by itself and as a part of second-order optimization methods. In general, in each iteration of the second-order method, like Inexact Newton method mentioned above, one has to solve a system of linear equations, either approximately or exactly, to get the search direction. Again, distributed environment represents a challenge. Fixed point methods are a known to be very effective in the centralized framework for linear system of large dimensions. We propose here a class of distributed fixed point methods that works in the distributed setting. We show that such methods converge for both static and time-varying network, achieving convergence results analogous to those that can be proved for the centralized case.

Finally, in the last part of the thesis we consider the least square problems of very large dimension motivated by digitalization of cadastral maps. The dimension of the problem represent the main difficulty for classical method while the sparse structure presents an opportunity to be exploited in parallel computational framework. We present an Inexact Levenberg-Marquardt method for sparse least squares problem. The method exploits the underlying structure of the problem to define a fixed-point strategy for the computation of the search direction that is suitable for parallelization. Theoretical analysis is presented and we show both global and fast local convergence under the classical assumptions for least squares problems.

All presented methods are implemented, and tested on relevant examples. The numerical results reveal that the theoretical results are confirmed by empirical evidence. Furthermore, the proposed methods are compared with the corresponding state-of-the-art methods and their competitiveness is demonstrated.

Apstrakt

Mnogi savremeni matematički modeli zahtevaju rešavanje problema optimizacije na mreži računarskih čvorova u kooperativnom režimu dok rastući interes za velike skupove podataka i mašinsko učenje zahteva metode kojima je moguće rešiti probleme sve većih dimenzija. Većina klasičnih optimizacionih metoda nije pogodna za primenu u distribuiranom okruženju, te je potrebno razviti nove metode koji mogu da odgovore na izazove koji potiču iz distribuiranog okruženja. Ova teza je fokusirana na distribuirane metode za optimizacione probleme velikih dimenzija.

Rezultati predstavljeni u ovoj tezi su doprinos oblasti distribuirane optimizacije u sledećim aspektima. Prvo je proširena analiza konvergencije za klasu postojećih distribuiranih metoda prvog reda, za slučaj komunikacionih mreža koje se menjaju u vremenu i za dužine koraka koje se menjaju po vremenu bez koordinacije između kompjuterskih čvorova. Ovo proširenje je od posebnog značaja u praktičnim primenama gde su promene u komunikacionim mrežama uzrokovane tehničkim problemima u komunikaciji ili u slučaju sve značajnijih pokretnih senzorskih mreža.

Jedno od glavnih pitanja u metodama distribuirane optimizacije je određivanje dužine koraka. Naime, klasične tehnike globalizacije poput linijskog pretraživanja nisu primenljive u ovom okviru, a primena fiksne dužine koraka često dovodi do sporog metoda. Sem toga primena fiksnog koraka zahteva poznavanje globalnih konstanti koje se ne mogu jednostavno oceniti u distribuiranom okruženju. Stoga je u tezi predložen distribuirani približan Njutnov metod sa adaptivnom dužinom koraka koja se može sračunati u distribuiranom okruženju, a pomoću tako određene dužine koraka, uz uobičajene pretpostavke o regularnosti problema, metod je globalno konvergentan i zadržava

lokalni red konvergencije karakterističan za centralizovanu optimizaciju. Sistemi linearnih jednačina velikih dimenzija su izazov i kao nezavisni problemi i kao deo optimizacionih postupaka drugog reda. U opštem slučju, u svakoj iteraciji metoda drugog reda, poput približnog Njutnovog metoda koji je već pomenut, potrebno je rešiti, približno ili tačno, sistem linearnih jednačina da bi se odredio pravac pretraživanja. Distribuirano okruženje i ovde predstavlja izazov. Metode nepokretne tačke su poznate kao efikasan način rešavanja sistema linearnih jednačina velikih dimenzija u centralizovanom okruženju. Ovde je predstavljana klasa distribuiranih metoda tipa nepokretne tačke koja je prilagođena distribuiranom okruženju. Pokazano je da postupci predloženog tipa konvergiraju za statične i promenljive komunikacione mreže, a rezultati o konvergenciji su analogni rezultatima u centralizovanom slučaju. U poslednjem delu teze je razmatran problem najmanjih kvadrata veoma velike dimenzije motivisan problemom digitalizacije katastarskih mapa. Veoma velika dimenzija problema predstavlja glavni problem za primenu klasičnih metoda dok je retka struktura problema prirodna mogućnost za primenu paralelnih metoda. U tezi je predstavljen približni Levenbrg-Markardov metod za rešavanje retkih problema najmanjih kvadrata. Retka strukutura problema je iskorišćena kao osnov za definisanje strategije tipa nepokretne tačke kojom se određuje pravca pretraživanja na način koji je pogodan za paralelizaciju. Predstavljena je teorijska analiza i pokazna globalna i lokalna konvergencija pod klasičnim pretpostavkama za ovaj tip problema.
Svi predloženi metodi su implementirani, i testirani na relevantnim test primerima. Numerički rezultati su emprijski potvrdili teorijska tvrđenja. Pored toga, predloženi metodi su upoređeni sa najznačajnim postojećim metodama za odgovarajuće klase problema i pokazna je njihova kompetitivnost.

Acknowledgements

First of all I wish to thank my supervisor Professor Nataša Krejić, for all the time she spent working with me, her precious guidance and patience, and for giving me the opportunity to learn so much from her. I couldn't have hoped for a better advisor.
I also would like to thank Professors Dušan Jakovetić and Nataša Krklec Jerinkić, for reading this thesis in detail and all our interesting collaborations during these years, and Professor Stefania Bellavia, for her valuable comments regarding the thesis and most of all for introducing me first to the topic of numerical optimization and guiding me in the very first steps of my career.

I am grateful to all the students, the professors, and everyone involved in the BIGMATH project. It has been an invaluable opportunity. In particular I want to thank Lense Swaenen and Sioux Mathware, for taking the time to work with me and for introducing me to one of the main topics of my research.
I would also like to thank all the friends and colleagues that I met during these years, in particular my colleagues from the University of Novi Sad, for making this PhD an even greater experience, Stevo Racković and Filipa Valdeira for our scientific collaborations, our trips around Europe, and for sharing with me joys and pains of the PhD.

I thank Sebastiano and all my life-long friends for their encouragements and support, despite our geographical distance in these past years. Finally, my deepest gratitude goes to my family, mamma, papà and Edoardo, for the love and the unconditional support they have always shown me.

Introduction

Recent advantages in technology such as the advent of Big Data and the continuous development of wireless communication systems able to support computational networks of increasing size has given rise to many practical applications that require the solution of optimization problems involving data sets of extremely large size and that are often stored over a large number of computational units. Applications of this kind require optimization methods that are able to deal with the challenges offered by the new framework. Optimization problems need to be solved in a cooperative manner by several computational units, appropriately handling size, privacy requirements, time constraints, as well as some technical limitations such as limited communication capabilities and computational power of the machines.

The need for methods that are able to solve optimization problems in this framework created the field of distributed optimization, which focuses on developing new methods designed specifically for the framework at hand, as well as adapting well-established classical optimization methods, such as gradient descent, Newton's method and ADMM, by designing strategies that work around the limitations posed by the distributed setting. Such methods should ideally be able to provide a solution to the given problem without explicitly sharing data among the machines and in general without excessive communication along the network. Moreover, given the practical nature of the applications that originate the framework, in order to be considered effective, methods should be somehow robust to possible technical issues in the network such as machines temporarily not working as well as occasional disruptions in the communications. While the distributed nature of the problem is the main obstacle to the application of clas-

sical methods, and the main focus of a large part of the algorithms proposed in this context, it is important from the practical point of view to develop methods that work in this framework but that at the same time are also suitable for large scale problems by applying typical strategies such as iterative procedures for the solution of linear systems, careful selection of the stepsizes, and so on.

The main topic of this thesis are distributed methods for large scale optimization problems. In Chapter 1 we set the notation and recall some useful classical results from linear algebra, real analysis and graph theory, as well as classical optimization methods that are the basis of the methods the we propose and discuss in Chapters 3-6. In Chapter 2 we introduce the distributed optimization framework by discussing typical problems and assumptions and presenting a review of the current literature. In Chapters 3-6 we present the original contributions of the thesis.
In Chapter 3 we consider a class of distributed gradient based methods and we extend its convergence analysis to the case of time-varying directed networks, and time-varying uncoordinated (that is, node- dependant) stepsizes, [39]. We show that given assumptions over the sequence of networks, which in particular do not include strong connectivity at all times, a suitable interval for the stepsizes can be found, such that the considered method retain convergence properties analogous to those proved for constant networks and stepsizes. Numerical results show that asynchronous choices of the stepsizes can significantly improve the performance of the methods in the considered class, compared to the fixed step-size.
In Chapter 4, we propose a distributed Inexact Newton method that combines penalty formulation of the problem, Jacobi Overrelaxation method to compute the direction distributedly and uses an adaptive stepsize to generate a new iterate, [29]. Under suitable assumptions on the regularity of the objective function and on the connectivity

of the underlying network, we can prove both global and fast local convergence, with convergence order depending on the choice of the forcing parameters. The adaptive stepsize strategy yields larger steps and thus generate a faster method, with the additional advantage of working without a priori knowledge of the global constants. The results are a generalization of Polyak stepsize (both deterministic an adaptive) for the Newton method in the centralized case. The method is compared numerically with other second and first order distributed methods showing the good performance of the proposed method when considering problems of large dimension.
In Chapter 5 we propose a class of fully distributed fixed point methods for the solution of linear systems of equations, [28]. The proposed method works for both static and time-varying networks and we are able to prove convergence results that are analogous to those of fixed-point methods in the centralized framework. We also present a set of numerical results that show the effectiveness of the proposed strategy compared to other distributed methods for the solution of linear systems and to distributed optimization methods applied to the quadratic reformulation of the linear system.
In Chapter 6 we consider a parallel method for large sparse Least-Squares problem that arise from localization problems, [20]. In particular we present an inexact Levenberg-Marquardt method that exploits the particular structure of the considered problems to compute a search direction at each iteration in a parallel fashion. We show that, under suitable assumptions over the objective function and the parameters of the algorithm, one can prove for the proposed method convergence results that are completely analogous to those of the centralized Inexact Levenberg-Marquardt method. We present a set of numerical results that demonstrate the effectiveness of the considered method compared to its centralized counterpart. This part of the research is motivated by the specific industrial application - digitalization of cadastral maps in the Netherlands, and is carried out in

collaboration between the University of Novi Sad and Sioux Technologies (Eindhoven, The Netherlands) within the BIGMATH project, an EU funded PhD program. Mr. Lense Swaenen from Sioux Technologies acted as the industrial advisor for this part of the research.

The BIGMATH project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 812912.

Contents

List of Figures

Chapter 1

Preliminaries

In this chapter, we set the notation used in the thesis and we present the basic definitions and many classical results that will be used in the following chapters.

1.1 Linear Algebra, Real Analysis and Graph Theory

We denote with $\mathbb{N}$ and $\mathbb{R}$ the set of natural and real numbers respectively, we define $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ and we use $\mathbb{R}_{>0}$ to indicate the set of positive real numbers. Given $n \in \mathbb{N}$, we denote with $\mathbb{R}^n$ the n-dimensional real vector space. If $\mathbf{x}$ is in $\mathbb{R}^n$ we assume that it is a column vector and we denote with $x_1, \dots, x_n \in \mathbb{R}$ its components. We use $\mathbb{R}^{n \times m}$ to represent the vector space of matrices with n rows, m columns and entries in $\mathbb{R}$. Given a matrix $A \in \mathbb{R}^{n \times m}$ we denote its components with a_{ij} for $i = 1, \dots, n$ and $j = 1, \dots, m$ while $A_i \in \mathbb{R}^{1 \times m}$ will denote the i-th row of A. For any $A \in \mathbb{R}^{n \times m}$ we denote with $A^\top \in \mathbb{R}^{m \times n}$ its transpose and we say that A is *symmetric* if $A = A^\top$. Given $n \in \mathbb{N}$, we denote with I_n the identity matrix in $\mathbb{R}^{n \times n}$. When

the order n is clear from context, we drop the index n. Moreover, we denote with $\mathbf{e}^i \in \mathbb{R}^n$ the the vector that has i-th component equal to 1 and zero everywhere else, and with $\mathbf{e} \in \mathbb{R}^n$ the vector with all components equal to 1.

Given a vector $\mathbf{x} \in \mathbb{R}^{nN}$ partitioned into N blocks, we denote with $\mathbf{x}_i \in \mathbb{R}^n$ the i-th block of $\mathbf{x}$. Analogously, given a matrix $A \in \mathbb{R}^{nN \times nN}$ partitioned into $N \times N$ block we denote with A_{ij} the block in position (i, j). That is,

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix}, \quad A = \begin{pmatrix} A_{11} & \dots & A_{1N} \\ \vdots & & \vdots \\ A_{N1} & \dots & A_{NN} \end{pmatrix}.$$

Given a symmetric matrix $A \in \mathbb{R}^{n \times n}$ we denote with $\text{eig}(A) = \{\lambda_i\}_{i=1:n}$ the set of eigenvalues of A and we always assume that $\lambda_1 \geq \cdots \geq \lambda_n$. We will also use the notation $\lambda_{\min}$ and $\lambda_{\max}$ to denote the eigenvalue with the smallest and the largest absolute value, respectively. Moreover, we denote with $\text{Span}(A)$ the subspace generated by the columns of A and with $\text{Ker}(A)$ its null space.
If $\lambda \neq 0$ for every $\lambda \in \text{eig}(A)$ we say that A is *nonsingular* and we denote with A^{-1} its inverse.

Definition 1.1. *Given a matrix $A \in \mathbb{R}^{n \times n}$ we say that A is* positive semi-definite *if for every $\mathbf{x} \in \mathbb{R}^n$ we have $\mathbf{x}^\top A\mathbf{x} \geq 0$. We say that the matrix is* positive definite *if the above inequality is strict.*

Given $a, b \in \mathbb{R}$ we use the notation $aI_n \preceq A \preceq bI_n$ to indicate the fact that for every $\lambda \in \text{eig}(A)$ we have $a \leq \lambda \leq b$.

Definition 1.2. *The function $\|\cdot\| : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a* norm *on $\mathbb{R}^n$ if*

i) $\|\mathbf{x}\| \geq 0$ *for every* $\mathbf{x} \in \mathbb{R}^n$*, and* $\|\mathbf{x}\| = 0$ *if and only if* $\mathbf{x} = 0$

ii) $\|a\mathbf{x}\| = |a|\|\mathbf{x}\|$ *for every* $a \in \mathbb{R}, \mathbf{x} \in \mathbb{R}^n$

iii) $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ *for every* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

Inequality *iii*) is referred to as *triangular inequality*. The following, called *reversed triangular inequality*, holds for every norm and every $\mathbf{x}, \mathbf{y}$:

$$\|\mathbf{x} - \mathbf{y}\| \geq |\|\mathbf{x}\| - \|\mathbf{y}\||$$

We will use the following vector and matrix norms:

$$\begin{aligned}
\|\mathbf{x}\|_2 &= \left(\sum_{i=1}^{n} x_i^2\right)^{1/2} \\
\|\mathbf{x}\|_\infty &= \max_{i=1:n} |x_i| \\
\|A\|_2 &= \left(\lambda_{\max}(A^\top A)\right)^{1/2} \\
\|A\|_\infty &= \max_{i=1:n} \sum_{j=1}^{m} |a_{ij}|
\end{aligned}$$

For every $\mathbf{x} \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times m}$ symmetric, the following inequalities hold

$$\begin{aligned}
&\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2 \leq n^{1/2}\|\mathbf{x}\|_2 \\
&\lambda_{\min}(A)\|\mathbf{x}\|_2 \leq \|A\mathbf{x}\|_2 \leq \lambda_{\max}(A)\|\mathbf{x}\|_2
\end{aligned}$$

Moreover the norms above are sub-multiplicative. That is, for every $A, B \in \mathbb{R}^{n \times n}$ we have

$$\|AB\|_2 \leq \|A\|_2\|B\|_2 \quad \|AB\|_\infty \leq \|A\|_\infty\|B\|_\infty.$$

Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ we have the following inequality, referred to as *Cauchy-Schwartz inequality*:

$$\mathbf{x}^\top \mathbf{y} \leq \|\mathbf{x}\|\|\mathbf{y}\|$$

Definition 1.3. *Given a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \subset \mathbb{R}^n$ *we say that* $\{\mathbf{x}^k\}$ *is*

i) bounded *if there exists* $M \in \mathbb{R}$ *such that* $\|\mathbf{x}^k\| \leq M$ *for every* $k \in \mathbb{N}_0$

ii) Cauchy *if for every* $\varepsilon > 0$ *there exists* $\bar{k} \in \mathbb{N}_0$ *such that* $\|\mathbf{x}^s - \mathbf{x}^l\| \leq \varepsilon$ *for every* $s, l \geq \bar{k}$.

iii) convergent *if there exists* $\mathbf{x}^*$ *such that* $\lim_{k \to +\infty} \mathbf{x}^k = \mathbf{x}^*$. *That is, if for every* $\varepsilon > 0$ *there exists* $\bar{k} \in \mathbb{N}_0$ *such that* $\|\mathbf{x}^k - \mathbf{x}^*\| \leq \varepsilon$ *for every* $k \geq \bar{k}$

Lemma 1.1. *Given a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \in \mathbb{R}^n$ *we have that*

1. $\{\mathbf{x}^k\}$ *is convergent if and only if it is a Cauchy sequence*
2. *if* $\{\mathbf{x}^k\}$ *converges to* $\mathbf{x}^* \in \mathbb{R}^n$ *then all its subsequences also converge to* $\mathbf{x}^*$.

Definition 1.4. *Given a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \subset \mathbb{R}^n$ *and a point* $\bar{\mathbf{x}} \in \mathbb{R}^n$, *we say that* $\bar{\mathbf{x}}$ *is an* accumulation point *of* $\{\mathbf{x}^k\}$ *if for every open subset* $C \subseteq \mathbb{R}^n$ *such that* $\bar{\mathbf{x}} \in C$, *we have that* $\mathbf{x}^k \in C$ *for infinitely many values of* $k \in \mathbb{N}_0$.

Lemma 1.2. *Given a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \in \mathbb{R}^n$ *we have that if* $\{\mathbf{x}^k\}$ *is bounded then it has at least one accumulation point. Moreover, if* $\bar{\mathbf{x}}$ *is an accumulation point of* $\{\mathbf{x}^k\}$, *then there exists a subsequence of* $\{\mathbf{x}^k\}$ *that converges to* $\bar{\mathbf{x}}$. *That is, there exists* $\mathcal{K} \subseteq \mathbb{N}_0$ *infinite subset such that* $\lim_{k \in \mathcal{K}} \mathbf{x}^k = \bar{\mathbf{x}}$.

Definition 1.5. *Consider a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \subset \mathbb{R}^n$ *such that* $\mathbf{x}^k$ *converges to* $\mathbf{x}^* \in \mathbb{R}^n$ *as* k *tends to* $+\infty$ *and assume that there exist* $q \geq 1$ *and* $C \geq 0$ *such that*

$$\lim_{k \to \infty} \frac{\|\mathbf{x}^{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}^k - \mathbf{x}^*\|^q} = C.$$

We say that the convergence is linear *if* $q = 1$ *and* $C \geq 1$, superlinear *if* $q \in (1, 2)$, *and* quadratic *if* $q = 2$.

Definition 1.6. *Given a sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty} \subset \mathbb{R}^n$ *such that* $\mathbf{x}^k$ *converges to* $\mathbf{x}^* \in \mathbb{R}^n$ *as* k *tends to* $+\infty$, *we say that the convergence is* R-linear *if there exist* $\{\varepsilon_k\}_{k=0}^{\infty} \subset \mathbb{R}$ *such that such that for every index* k *large enough*

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \varepsilon_k$$

and ε_k *converges to 0 linearly.*

Given $A \in \mathbb{R}^{n \times n}$ nonsingular and $\mathbf{b} \in \mathbb{R}^n$, consider the linear system

$$A\mathbf{x} = \mathbf{b} \tag{1.1}$$

and the sequence $\{\mathbf{x}\}_{k=0}^{\infty}$ generated by the following

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^n \\ \mathbf{x}^{k+1} = M\mathbf{x}^k + \mathbf{d} \end{cases} \tag{1.2}$$

where $M \in \mathbb{R}^{n \times n}$ and $\mathbf{d} \in \mathbb{R}^n$. We say that $\mathbf{x}^* \in \mathbb{R}^n$ is a *fixed point* of (1.2) if

$$\mathbf{x}^* = M\mathbf{x}^* + \mathbf{d}$$

and that (1.2) is a fixed point method for the solution of the linear system (1.1) if $A\mathbf{x}^* = \mathbf{b}$. The following theorem states sufficient conditions for the convergence of (1.2) and provides a bound to the distance of $\mathbf{x}^k$ from $\mathbf{x}^*$.

Theorem 1.1. *Given* $M \in \mathbb{R}^{n \times n}$ *and* $d \in \mathbb{R}^n$, *if* $\lambda_{\max}(M) < 1$ *then for every choice of* $\mathbf{x}^0 \in \mathbb{R}^n$ *the sequence* $\{\mathbf{x}^k\}_{k=0}^{\infty}$ *generated by* (1.2) *converges to the fixed point* $\mathbf{x}^*$. *Moreover, for every iteration index* k *we have*

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \leq \lambda_{\max}(M)\|\mathbf{x}^k - \mathbf{x}^*\|.$$

Given a linear system (1.1) several fixed-point methods have been developed in literature to find the solution, depending on the properties of the coefficient matrix A. We report here two examples that will be relevant in the following chapters.

Assume that $A \in \mathbb{R}^n$ is a nonsingular matrix with nonzero diagonal entries and let us consider the splitting $A = D - B$, where D is the diagonal matrix $D = \text{diag}(a_{11}, \ldots, a_{nn})$. The Jacobi method is defined by (1.2) with

$$\begin{aligned} M &= D^{-1}B =: M_J \\ \mathbf{d} &= D^{-1}\mathbf{b}. \end{aligned} \tag{1.3}$$

Equivalently, at iteration k, we define x_i^{k+1} for $i = 1, \ldots, n$ as follows:

$$x_i^{k+1} = -\frac{1}{a_{ii}} \sum_{j=1, j\neq i}^{n} a_{ij} x_j^k + d_i.$$

From Theorem 1.1 we know that the Jacobi method converges linearly to the solution of (1.1) if $\lambda_{\max}(D^{-1}B) < 1$ which is a condition satisfied, for example, by diagonally dominant matrices. To speed up convergence and extend the class of matrices for which the method is convergent, Jacobi Overrelaxation method (JOR) introduces the relaxation parameter $\omega \in \mathbb{R}$ and defines the sequence $\{\mathbf{x}^k\}_{k=0}^{\infty}$ as in (1.2) with

$$\begin{aligned} M &= \omega D^{-1}B + (1-\omega)I \\ \mathbf{d} &= D^{-1}\mathbf{b}. \end{aligned}$$

Component by component, the k-th iteration of JOR is given by

$$x_i^{k+1} = (1-\omega)x_i^k - \frac{\omega}{a_{ii}} \left(\sum_{j=1, j\neq i}^{n} a_{ij} x_j^k + b_i \right), \; i = 1, \ldots, n. \tag{1.4}$$

If the matrix A is symmetric and positive definite, we can prove that the sequence $\{\mathbf{x}^k\}$ generated by JOR method converges to the solution

of (1.1) for every choice of ω such that

$$\omega \in \left(0, \frac{2}{\lambda_{\max}(M_J)}\right),$$

where the matrix M_J is defined as in (1.3).
Given two matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$ we denote with $A \otimes B$ the Kronecker product of A and B. That is

$$A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1m}B \\ \vdots & & \vdots \\ a_{n1}B & \dots & a_{nm}B \end{pmatrix} \in \mathbb{R}^{np\times mq}$$

Definition 1.7. *Given a matrix $A \in \mathbb{R}^{n\times n}$ with $a_{ij} \geq 0$ for every $i, j = 1, \dots, n$ we say that A is:*

i) row-stochastic *if $\sum_{j=1}^{n} a_{ij} = 1$ for every row index i;*

ii) column-stochastic *if $\sum_{i=1}^{n} a_{ij} = 1$ for every column index j;*

iii) doubly-stochastic *if it is both row and column-stochastic.*

Lemma 1.3. *If $A \in \mathbb{R}^{n\times n}$ is a row (resp. column) stochastic matrix then $|\lambda_i(A)| \leq 1$ for every $i = 1, \dots, n$, $\lambda_{\max} = \lambda_1 = 1$ and $\mathbf{e} \in \mathbb{R}^n$ is a right (resp. left) eigenvector of A for the eigenvalue 1.*

In this thesis we will generally consider smooth real valued functions. Given a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ we say that it is continuously differentiable if for every $i = 1, \dots, n$ the partial derivative $\partial_{x_i} f$ exists and is continuous everywhere in $\mathbb{R}^n$. Analogously, we say that f is twice continuously differentiable, or smooth, if $\partial_{x_i}\partial_{x_j} f$ exists and is continuous everywhere in $\mathbb{R}^n$ for $i, j = 1, \dots, n$. Given a twice continuously function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ we denote with $\nabla f(\mathbf{x}) \in \mathbb{R}^n$ and

$\nabla^2 f(\mathbf{x}) \in \mathbb{R}^{n\times n}$ the gradient and the Hessian matrix of f respectively. That is, for every $i, j = 1, \ldots, n$

$$\nabla f(\mathbf{x})_i = \frac{\partial}{\partial_{x_i}} f(\mathbf{x}), \;\; \nabla^2 f(\mathbf{x})_{ij} = \frac{\partial^2}{\partial_{x_i}\partial_{x_j}} f(\mathbf{x}).$$

Given a continuously differentiable $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, with $f(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top$ we denote with $J_f(\mathbf{x}) \in \mathbb{R}^n$ the Jacobian matrix of f, defined as follows

$$J_f(\mathbf{x}) = \begin{pmatrix} \nabla f_1(\mathbf{x})^\top \\ \vdots \\ \nabla f_m(\mathbf{x})^\top \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

That is, for every $i = 1, \ldots, m$ and every $j = 1, \ldots, n$ we have

$$J_f(\mathbf{x})_{ij} = \frac{\partial}{\partial_{x_j}} f_i(\mathbf{x}).$$

We will often use the notation $J(\mathbf{x})$ when it is clear that the Jacobian is referred to a given function.

Definition 1.8. *Given a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ and a constant $L \geq 0$ we say that f is L-Lipschitz continuous if for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ we have*

$$\|f(\mathbf{x}) - f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$$

Definition 1.9. *Given a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, f is* convex *if for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and every $\alpha \in [0, 1]$ we have*

$$f(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}).$$

Definition 1.10. *Given a differentiable function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ and a constant $\mu > 0$ we say that f is μ-strongly convex if for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ we have*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top(\mathbf{y} - \mathbf{x}) + \frac{\mu}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

Lemma 1.4. *Assume that the function f is given by*

$$f(\mathbf{x}) = \sum_{i=1}^{m} f_i(\mathbf{x}), \text{ with } f_i : \mathbb{R}^n \longrightarrow \mathbb{R}.$$

If for every $i = 1, \ldots, m$ the function f_i is L_i-Lipschitz continuous with $L_i \geq 0$ then f is Lipschitz continuous with constant $L = \sum_{i=1}^{m} L_i$. Analogously, if f_i is μ_i-strongly convex for every i, then f is μ-strongly convex with $\mu = \sum_{i=1}^{m} \mu_i$.

Lemma 1.5. *Given a twice continuously differentiable function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, the following properties hold*

1. *the Hessian matrix $\nabla^2 f(\mathbf{x})$ is a symmetric;*
2. *if ∇f is L-Lipschitz continuous, then for every $\mathbf{x} \in \mathbb{R}^n$ we have that*
$$\lambda_{\max}\left(\nabla^2 f(\mathbf{x})\right) \leq L;$$
3. *f is μ-strongly convex if and only if*
$$\lambda_{\min}\left(\nabla^2 f(\mathbf{x})\right) \geq \mu;$$

We have the following results, referred to as first and second order Taylor expansion, respectively.

Theorem 1.2. *If $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is continuously differentiable then for every $\mathbf{x}, \mathbf{d} \in \mathbb{R}^n$ we have*

$$f(\mathbf{x} + \mathbf{d}) = f(\mathbf{x}) + \int_0^1 \nabla f(\mathbf{x} + t\mathbf{d})^\top \mathbf{d} \, dt$$

and there exists $s \in [0, 1]$ such that

$$f(\mathbf{x} + \mathbf{d}) = f(\mathbf{x}) + \nabla f(\mathbf{x} + s\mathbf{d})^\top \mathbf{d}.$$

In particular, we have

$$f(\mathbf{x} + \mathbf{d}) = f(\mathbf{x}) + \nabla f(\mathbf{x} + \mathbf{d})^\top \mathbf{d} + O(\|\mathbf{d}\|^2). \tag{1.5}$$

Theorem 1.3. *If $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is twice continuously differentiable then for every $\mathbf{x}, \mathbf{d} \in \mathbb{R}^n$ we have*

$$\nabla f(\mathbf{x}+\mathbf{d}) = \nabla f(\mathbf{x}) + \int_0^1 \nabla^2 f(\mathbf{x}+t\mathbf{d})\mathbf{d}\ dt$$

and there exists $s \in [0,1]$ such that

$$f(\mathbf{x}+\mathbf{d}) = f(\mathbf{x}) + \nabla f(\mathbf{x}+\mathbf{d})^\top \mathbf{d} + \frac{1}{2}\mathbf{d}^\top \nabla^2 f(\mathbf{x}+s\mathbf{d})\mathbf{d}. \tag{1.6}$$

The following theorem, which is an adaptation of a classical result in control theory [13] and is usually referred to as Small Gain Theorem, is widely used in the convergence analysis of optimization method in the distributed setting [47, 48, 24] and will be the basis of the results that we present in Chapter 3. Given a sequence of vectors $\mathbf{x} := \{\mathbf{x}^k\}_{k=0}^{\infty} \subset \mathbb{R}^n$, a norm $\|\cdot\|$ on $\mathbb{R}^n$, and two constant $\delta \in (0,1)$ and $K \in \mathbb{N}_0$ we define the following quantities

$$\|\mathbf{x}\|^{\delta,K} = \max_{k=0,1,\dots,K} \left\{ \frac{1}{\delta^k}\|\mathbf{x}^k\| \right\}$$
$$\|\mathbf{x}\|^{\delta} = \sup_{k\geq 0} \left\{ \frac{1}{\delta^k}\|\mathbf{x}^k\| \right\}.$$

One can notice that, since $0 < \delta < 1$, proving that $\|\mathbf{x}\|^\delta$ is bounded implies that $\|\mathbf{x}^k\|$ goes to zero at least as fast δ^k.

The version of the theorem that we state here assumes that two sequences are given, but the same result can be extended to the case of an arbitrary large number of sequences.

Theorem 1.4. *[13] Let $\mathbf{x} = \{\mathbf{x}^k\}_{k=0}^{\infty}$, $\mathbf{y} = \{\mathbf{y}^k\}_{k=0}^{\infty}$ be two sequences in $\mathbb{R}^n$ and assume that there exists $\delta \in (0,1)$, $\gamma_1, \gamma_2 \geq 0$ such that*

$\gamma_1 \cdot \gamma_2 \leq 1$ and for all $K \in \mathbb{N}_0$ the following inequalities hold

$$\|\mathbf{x}\|^{\delta,K} \leq \gamma_1 \|\mathbf{y}\|^{\delta,K} + w_1,$$
$$\|\mathbf{y}\|^{\delta,K} \leq \gamma_2 \|\mathbf{x}\|^{\delta,K} + w_2,$$

then

$$\|\mathbf{x}\|^{\delta} \leq \frac{w_1\gamma_2 + w_2}{1 - \gamma_1\gamma_2}$$

and $\lim_{k\to\infty} \mathbf{x}^k = 0$ R-linearly.

We now introduce a few definition and results in graph theory, which will be important when considering the underlying communication network in the distributed framework.

Definition 1.11. *We define an undirected graph $\mathcal{G}$ as the couple $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} \subset \mathbb{N}$ and $\mathcal{E} \subseteq \big\{\{u, v\} \mid u, v \in \mathcal{V}\big\}$.*

Definition 1.12. *We define a directed graph $\mathcal{G}$ as the couple $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} \subset \mathbb{N}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$.*

Given a graph (directed or undirected) $\mathcal{G}$, we refer to $\mathcal{V}$ and $\mathcal{E}$ as the set of *vertices* and *edges* of $\mathcal{G}$, respectively.

Definition 1.13. *We say that a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is simple if each edge appears at most once in $\mathcal{E}$.*

Equivalently, an undirected graph can also be defined as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that for every $(i, j) \in \mathcal{E}$ we also have $(j, i) \in \mathcal{E}$. For this reason, from now on we will use the notation (i, j) to indicate the edge between node i and node j, also in case the network $\mathcal{G}$ is undirected.

Definition 1.14. *Given an undirected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for every $i \in \mathcal{V}$, we denote with $\mathcal{N}_i$ the set of neighbors of node i. That is $\mathcal{N}_i = \{j \in \mathcal{V}$ such that $(i, j) \in \mathcal{E}\}$. We define the* degree *of i, denoted with $\deg(i)$, as the cardinality of $\mathcal{N}_i$.*

If the graph is directed we can define two neighbors and two degrees for each node, taking into account the direction of the edges.

Definition 1.15. *Given an undirected (resp. directed) graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *and two nodes* $i, j \in \mathcal{V}$ *we define a* (directed) path of length L *from* i *to* j *as an ordered sequence* $(v_1, \ldots, v_L)$ *such that* $v_l \in \mathcal{V}$ *for every* l, $v_1 = 1$, $v_L = j$ *and for every* $l = 1 : L - 1$ *we have* $(v_l, v_{l+1}) \in \mathcal{E}$.

Definition 1.16. *If* $\mathcal{G}$ *is a directed network, we say that it is* strongly connected *if for every* $i, j \in \mathcal{V}$ *there exists a directed path from* i *to* j. *We say that* $\mathcal{G}$ *is* fully strongly connected *if for every* $i, j \in \mathcal{V}$ *we have* $(i, j) \in \mathcal{E}$. *That is, if there is an edge in both direction between any to nodes in the graph.*

Definition 1.17. *If* $\mathcal{G}$ *is an undirected network, we say that it is* connected *if for every* $i, j \in \mathcal{V}$ *there exists a path between* i *and* j. *We say that* $\mathcal{G}$ *is* fully connected *if for every* $i, j \in \mathcal{V}$ *we have* $(i, j) \in \mathcal{E}$.

Definition 1.18. *Given a graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *we define the distance between two nodes* $i, j \in \mathcal{V}$ *as the length of the shortest (directed) path between* i *and* j. *We also define the diameter of the graph as the largest distance between two nodes in* $\mathcal{G}$.

Definition 1.19. *Given two graphs with the same set of nodes* $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$, $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$ *we define the composition between* $\mathcal{G}_1$ *and* $\mathcal{G}_2$ *as* $\mathcal{G} = \mathcal{G}_2 \circ \mathcal{G}_1 = (\mathcal{V}, \mathcal{E})$ *with*

$$\mathcal{E} := \{(j, i) \in \mathcal{V}^2 \mid \exists\ s \in \mathcal{V} \text{ such that } (j, s) \in \mathcal{E}_1, (s, i) \in \mathcal{E}_2\}.$$

That is, there is an edge from j *to* i *in* $\mathcal{G}_2 \circ \mathcal{G}_1$ *if we can find a path from* j *to* i *such that the first edge of the path is in* $\mathcal{G}_1$ *and the second edge is in* $\mathcal{G}_2$.

This definition can be extended to finite sequences of graphs of arbitrary length.

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$, if $\mathcal{G}$ is undirected and simple, then it is possible to find a symmetric matrix $W \in \mathbb{R}^{n \times n}$ such that W is doubly stochastic, and for every $i, j = 1, \ldots, n$, $w_{ij} \neq 0$ if and only if $(i, j) \in \mathcal{E}$. If $\mathcal{G}$ is connected, then we also have that $\lambda_1(W) = 1$ and $|\lambda_i(W)| < 1$ for every $i = 2, \ldots, n$.

1.2 Centralized Optimization

In this section we review the basic concepts and methods in nonlinear optimization. When not stated otherwise, for the results in this Section we refer to [50].

Given a set $C \subseteq \mathbb{R}^n$ and a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ we consider the problem of minimizing f over the set C. That is, we want to compute

$$f^* = \min_{\mathbf{x} \in C} f(\mathbf{x}). \tag{1.7}$$

The goal of an optimization method is to find a point $\mathbf{x}^*$ that realizes the minimum of the function f. That is, $\mathbf{x}^* \in X^*$ with X^* defined as

$$X^* = \arg\min_{\mathbf{x} \in C} f(\mathbf{x}) := \{\mathbf{x} \in C \mid f(\mathbf{x}) = f^*\}.$$

In this thesis we focus on unconstrained optimization. That is, we assume that $C = \mathbb{R}^n$ and therefore we want to solve

$$f^* = \min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}). \tag{1.8}$$

Definition 1.20. *Consider $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ and a point $\mathbf{x} \in \mathbb{R}^n$. We say that $\mathbf{x}^*$ is a* global minimizer *of f if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for every $\mathbf{x} \in \mathbb{R}^n$*

We say that $\mathbf{x}^*$ *is a* local minimizer *of* f *if there exists a neighbourhood* $\mathcal{N}$ *of* $\mathbf{x}^*$ *such that* $f(\mathbf{x}^*) \leq f(\mathbf{x})$ *for every* $\mathbf{x} \in \mathcal{N}$.
We say that $\mathbf{x}^*$ *is a* strict *global (resp. local) minimizer if it is a global (resp. local) minimizer and the inequalities above are strict.*

Definition 1.21. *If* f *is a continuously differentiable function* $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $\mathbf{x}^* \in \mathbb{R}^n$ *is a* stationary point *of* f *if* $\nabla f(\mathbf{x}^*) = 0$.

The following theorems give necessary and sufficient conditions for a point $\mathbf{x}^*$ to be a local minimizer of the function f.

Theorem 1.5 (Necessary Conditions)**.** *Given* $f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ *and* $\mathbf{x}^* \in \mathbb{R}$. *If* f *is continuously differentiable in a neighborhood of* $\mathbf{x}^*$, *and* $\mathbf{x}^*$ *is a local minimizer of* f *then* $\nabla f(\mathbf{x}^*) = 0$. *Moreover, if* f *is twice continuously differentiable in a neighborhood of* $\mathbf{x}^*$, *then we also have that the Hessian* $\nabla^2 f(\mathbf{x}^*)$ *is positive semi-definite.*

Theorem 1.6 (Sufficient Conditions)**.** *Given* f *twice continuously differentiable, if a point* $\mathbf{x}^* \in \mathbb{R}^n$ *is such that* $\nabla(\mathbf{x}^*) = 0$ *and* $\nabla^2 f(\mathbf{x}^*)$ *is positive definite, then* $\mathbf{x}^*$ *is a strict local minimizer of* f.

In general we will focus on problems where the objective function f is twice continuously differentiable and (strongly) convex. In this case, the following theorem guarantees that problem (1.8) is equivalent to finding a point where the gradient of f vanishes.

Theorem 1.7. *If* f *is a convex function, then all local minimizers of* f *are also global minimizers. Moreover, is* f *is continuously differentiable, then all stationary points are global minimizers of* f.

In this work we consider iterative for methods for the solution of (1.8). That is, algorithms that, starting from a given initial guess $\mathbf{x}^0 \in \mathbb{R}^n$ generate a sequence of iterates $\{\mathbf{x}^k\}_{k=0}^{\infty}$ that converges to a

solution $\mathbf{x}^*$ of (1.8). For all the methods that we consider, the sequence $\{\mathbf{x}^k\}_{k=0}^{\infty}$ is generated as follows

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^n \\ \mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k \end{cases} \tag{1.9}$$

where $\mathbf{d}^k \in \mathbb{R}^n$ and $\alpha_k \geq 0$. We refer to $\mathbf{d}^k$ and α_k and as the direction and the step size (or step length) at iteration k, respectively. Any iterative method of this form is characterized by the specific choice of the step size and the direction. That is, we define a method by specifying how α_k and $\mathbf{d}^k$ are computed at each iteration. Regarding the choice of $\mathbf{d}^k$ we will see here two main classes of methods: Gradient and Newton's method. We will also discuss a few strategies for the choice of the step size α_k.

1.2.1 Choice of the direction $\mathbf{d}^k$

Let us assume that f is continuously differentiable and, given a point $\mathbf{x}^k$, let us consider the first order Taylor expansion (1.5) of f around $\mathbf{x}^k$:

$$f(\mathbf{x}^k + \alpha \mathbf{d}^k) = f(\mathbf{x}^k) + \alpha \nabla f(\mathbf{x}^k)^\top \mathbf{d}^k + \alpha^2 O\left(\|\mathbf{d}^k\|^2\right).$$

It is easy to see that if $\nabla f(\mathbf{x}^k)^\top \mathbf{d}^k < 0$ then for α small enough, we will have

$$f(\mathbf{x}^k + \alpha \mathbf{d}^k) < f(\mathbf{x}^k).$$

That is, if at iteration k the direction $\mathbf{d}^k$ is such that $\nabla f(\mathbf{x}^k)^\top \mathbf{d}^k < 0$, then the value of the function f at the current iterate $\mathbf{x}^k$ can be reduced along the direction $\mathbf{d}^k$. A vector $\mathbf{d}^k$ that satisfies this property is called a *descent direction* for f at $\mathbf{x}^k$, and most methods of the form (1.9) assume the direction $\mathbf{d}^k$ to be descent.

The most simple choice for the descent direction is $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$, which defines Gradient Descent method. With this choice for $\mathbf{d}^k$ and assuming that the step size is fixed at each iteration, we can prove the following result.

Theorem 1.8. *Assume that f is a continuously differentiable convex function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ and that ∇f is L-Lipschitz continuous, and let $\{\mathbf{x}^k\}$ be the sequence generated by* (1.9) *with $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ and $\alpha_k = \alpha$ for every k. If $\alpha \leq 1/L$ then $\mathbf{x}^k$ converges linearly to a solution of* (1.8).

Gradient Descent method with fixed step size is widely used in many practical application because of its simplicity and because, since it only requires first order derivatives and no additional computation for the choice of the step size α_k, every iteration is very cheap. However, it is in general slow, in the sense that it may require many iteration to find a solution with good accuracy. First of all, depending on the considered function f, the first order Taylor expansion may not be a good local approximation of the function f, meaning that the direction $\mathbf{d}^k$ may not be very good. Moreover, the choice of taking a fixed step size also poses some issues: on the one hand the bound $1/L$ that ensures convergence in Theorem 1.8 may be extremely small, causing the method to preform a very large number if the initial guess $\mathbf{x}^0$ is far from the solution, on the other hand choosing a larger α may cause the sequence $\{\mathbf{x}^k\}$ to diverge. Later in this section we will discuss a strategy for the computation of the step size that, by considering the value of the function and its derivatives along the direction $\mathbf{d}^k$, ensures global convergence of the method, while allowing longer steps at some iterations.

When the objective function f is twice continuously differentiable, more sophisticated choices of the descent direction $\mathbf{d}^k$ can be obtained

by looking at the second order Taylor expansion (1.6). Given $\mathbf{x}^k \in \mathbb{R}^n$ we consider the following approximation of f around $\mathbf{x}^k$:

$$f(\mathbf{x}^k + \mathbf{d}) \approx m_k(\mathbf{d}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^\top \mathbf{d} + \frac{1}{2}\mathbf{d}^\top \nabla^2 f(\mathbf{x}^k)\mathbf{d} \quad (1.10)$$

and we compute the direction $\mathbf{d}^k$ at iteration k by minimizing the quadratic function m_k. That is, we take

$$\mathbf{d}_N^k = \arg\min_{\mathbf{d}\in\mathbb{R}^n} m_k(\mathbf{d}). \quad (1.11)$$

It is easy to see that, if the Hessian $\nabla^2 f(\mathbf{x}^k)$ is positive definite, then (1.11) has a unique solution, given by

$$\mathbf{d}_N^k = -\nabla^2 f(\mathbf{x}^k)^{-1} \nabla f(\mathbf{x}^k). \quad (1.12)$$

The method defined by (1.9) with direction $\mathbf{d}^k = \mathbf{d}_N^k$ is called Newton's method.

Compared to Gradient Descent, Newton's method is known for being generally fast, in the sense that it usually converges with a small number of iterations. In particular, with suitable regularity assumptions on the function f, it achieves local quadratic convergence. However, depending on the considered function f and the dimension n of the problem, there may be some issues. First of all, in order to compute the Newton direction $\mathbf{d}_N^K$ at each iteration one has to compute all second order derivatives and to solve a linear system of size n, which may cause each iteration of the method to become computationally expensive, especially for problems of large dimension. Secondly, in order for $\mathbf{d}_N^k$ to be a descent direction, the matrix $\nabla^2 f(\mathbf{x}^k)$ has to be positive definite at each iteration. If the Hessian is singular or not positive definite, the direction $\mathbf{d}^k$ is not well defined, and even in case it is positive definite, if at some iteration we have $\lambda_{\min}(\nabla^2 f(\mathbf{x}^k))$ is close to zero, the linear system could become difficult to solve in practice

and the method could become unstable.
To go around this second issue, the Hessian matrix in the local model (1.10) is often replaced by a matrix $B_k \in \mathbb{R}^{n\times n}$ which is purposefully chosen in such a way that it is both a good approximation of the true Hessian $\nabla^2 f(\mathbf{x}^k)$, but also has good spectral properties, so that the system $\mathbf{d}^k = -B_k^{-1}\nabla f(\mathbf{x}^k)$ is well-conditioned. Methods of this kind are referred to as Quasi-Newton methods. Several strategies for the computation of the matrix B_k have been proposed in literature that retain the good convergence properties of the classical Newton's method.
While replacing the true Hessian $\nabla^2 f(\mathbf{x}^k)$ with the approximation B_k makes Quasi-Newton methods suitable for problems where the Hessian of the objective function may be singular or nearly singular, they still require a linear system of size n to be solved at each iteration. A widely used strategy to reduce the per-iteration cost of Newton and Quasi-Newton methods and make them suitable for problems of large dimension, is to compute the direction $\mathbf{d}^k$ by solving the linear systems $\nabla^2 f(\mathbf{x}^k)\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ or $B_k\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ only approximately, up to a controlled accuracy. Also in this case, conditions can be posed on the accuracy for the computation of the direction $\mathbf{d}^k$, that ensure local fast convergence of the methods.

1.2.2 Selection of the Step Size

We now discuss a few choices for the step size α_k that will be relevant in the following chapters.
We saw that for gradient method, taking the step size constant at each iteration is a viable option, provided such a constant is small enough (namely, smaller than the inverse of the Lipschitz constant of the gradient of the objective function). However, we also noticed possible issue with this kind of choice: on the one hand the step size that ensures convergence could be small, causing the method to need

a very large number of iterations to arrive at the solution; on the other hand, the constant L is usually unknown, and working with a larger step size may cause the method to diverge.
In general we saw that, whenever $\mathbf{d}^k$ is a descent direction, for α_k small enough we have

$$f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) < f(\mathbf{x}^k). \tag{1.13}$$

That is, we can always find a value of the step size that decreases the value of the objective function. However, it can be proved that, in general, a simple decrease condition like (1.13) is not enough to ensure the convergence of the sequence to a solution of the problem. On the other hand, given any direction $\mathbf{d}^k$, the optimal choice for the step size α_k would be the value that minimizes f along $\mathbf{d}^k$, i.e.

$$\alpha_k = \underset{\alpha \in \mathbb{R}_{>0}}{\arg\min}\, \varphi(\alpha), \text{ with } \varphi(\alpha) = f(\mathbf{x}^k + \alpha \mathbf{d}^k). \tag{1.14}$$

This choice ensures that at each iteration we decrease the value of the objective function as much as possible along the current direction. However, computing such a value of the step size requires the solution of the additional optimization method (1.14) at each iteration, and it is usually too expensive to be applied in practice. In order for an iterative method to work and be practicable we need a way to compute the step size that does not require excessive computation, ensures sufficient decrease in the objective function and at the same time also ensures that the step size is not too small.

A main strategy for the selection of the step size is line search with Armijo and Wolfe condition, which we will now briefly describe. Given two constants $0 < c_1 < c_2 < 1$ we choose α_k such that the following two conditions hold

$$f(\mathbf{x} + \alpha_k \mathbf{d}^k) \leq f(\mathbf{x}^k) + c_1 \alpha_k \nabla f(\mathbf{x}^k)^\top \mathbf{d}^k \tag{1.15}$$

$$\nabla f(\mathbf{x} + \alpha_k \mathbf{d}^k)^\top \mathbf{d}^k \geq c_2 \alpha_k \nabla f(\mathbf{x}^k)^\top \mathbf{d}^k \tag{1.16}$$

The first condition ensures sufficient decrease of the objective function, while the second one, usually referred to as curvature condition, prevents the step size from becoming too small. The following lemma shows that, with suitable assumptions over the function f, we can always find an interval of values of the step size that satisfies both the conditions above.

Lemma 1.6. *Assume that f is a continuously differentiable function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $\mathbf{x}^k \in \mathbb{R}^n$, f is bounded from below on the semi-line $\{\mathbf{x}^k + \alpha \mathbf{d}^k | \alpha > 0\}$ and $0 < c_1 < c_2 < 1$. Then there exists and interval $I \subset \mathbb{R}_{>0}$ such that conditions* (1.15) *and* (1.16) *hold for every $\alpha \in I$.*

To avoid multiple evaluations of the derivative of f at each iteration, the second condition is usually not checked directly, and instead a backtracking strategy is used, combined with the sufficient decrease condition (1.15). The strategy consists in considering a decreasing sequence of possible values of the step sizes, until one is found that satisfies the Armijo condition or a given lower bound for the step size is reached. The following algorithm describes the backtracking strategy in more detail.

Algorithm 1.1 (Backtracking)**.**
Input: $0 < t_{\min} < t_0$, $q \in (0, 1)$, $c_1 > 0$, $\mathbf{x}^k$, $\mathbf{d}^k$.

1: *set m=0*
2: ***while*** $t_m \geq t_{\min}$ *and* $f(\mathbf{x} + t_m \mathbf{d}^k) > f(\mathbf{x}^k) + c_1 t_m \nabla f(\mathbf{x}^k)^\top \mathbf{d}^k$ ***do***
3: $\quad$ *set* $t_{m+1} = q t_m$
4: ***end while***
5: ***return*** t_m

That is, starting from a trial step size $t_0 > 0$, we check if Armijo condition holds. If it does, we set $\alpha_k = t_0$ and we proceed with the method, otherwise, we compute a new trial step $t_1 = q t_0$ for some

$q < 1$ and we check the condition again. The strategy proceeds in this way until a step t_m that satisfies Armijo condition is found, or until the step size becomes too small. While this backtracking strategy doesn't ensure, theoretically, that the curvature condition will be satisfied, since a lower bound on the step size is imposed directly by choosing $t_{\min}$, it is usually assumed in practice.

The following theorem shows how Armijo and Wolfe conditions can be used to analyze the behaviour of line-search methods.

Theorem 1.9. *Consider $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ and let $\{\mathbf{x}^k\}_{k=0}^{\infty}$ be the sequence generated by a method of the form* (1.9) *where, for every iteration index k, $\mathbf{d}^k$ is a descent direction for f at $\mathbf{x}^k$ and α_k satisfies conditions* (1.15) *and* (1.16)*. Assume that f is bounded from below and that there exists an open set $\Omega \subseteq \mathbb{R}^n$ such that*

$$\{\mathbf{x} \in \mathbb{R}^n | f(\mathbf{x}) \leq f(\mathbf{x}^0)\} \subset \Omega,$$

f is continuously differentiable on Ω and ∇f is L-Lipschitz continuous on Ω. Then, denoting with θ_k the angle between $\mathbf{d}^k$ and $\nabla f(\mathbf{x}^k)$ we have

$$\sum_{k=0}^{+\infty} \cos(\theta_k)^2 \|\nabla f(\mathbf{x}^k)\|^2 < +\infty. \tag{1.17}$$

Inequality (1.17) implies that

$$\lim_{k \to +\infty} \cos(\theta_k) \|\nabla f(\mathbf{x}^k)\| = 0$$

and therefore, if the descent direction $\mathbf{d}^k$ is defined in such a way that there exists $\sigma > 0$ such that $\cos(\theta_k) \geq \sigma$ for every k, then $\nabla f(\mathbf{x}^k)$ tends to zero as k tends to infinity. That is, if the step size satisfies conditions (1.15), (1.16) and the direction $\mathbf{d}^k$ does not tend to be orthogonal to the gradient, the sequence of gradients $\{\nabla f(\mathbf{x}^k)\}$ tends to

zero. In particular, every accumulation point of the sequence $\{\mathbf{x}^k\}_{k=0}^{\infty}$ is a stationary point of f.

When the Hessian matrix $\nabla^2 f(\mathbf{x}^k)$ is positive definite, we already noticed that the Newton direction defined in (1.11) is a descent direction and therefore the results of the previous theorem hold for Newton's method. The following theorems show that, under suitable regularity assumptions on the objective function, after a finite number of iteration of Newton's method, the full step size $\alpha_k = 1$ is always accepted. Moreover, we have that if the sequence starts close enough to a stationary point, then the method converges quadratically.

Theorem 1.10. *Assume that the same conditions of Theorem 1.9 hold, and additionally that the function f is strongly convex. Let $\{\mathbf{x}^k\}_{k=0}^{\infty}$ be the sequence generated by (1.9) with $\mathbf{d}^k = \mathbf{d}_N^k$ Newton direction and α_k computed like in Algorithm 1.1 with $t_0 = 1$ and $c_1 \in (0, 1)$. There exists c_1 small enough and $\bar{k} \in \mathbb{N}_0$ such that $\alpha_k = 1$ for every $k \geq \bar{k}$.*

Theorem 1.11. *Assume that f is a twice continuously differentiable function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, let $\mathbf{x}^* \in \mathbb{R}^n$ be a stationary point of f and let $\{\mathbf{x}^k\}_{k=0}^{\infty}$ be the sequence generated by (1.9) with $\mathbf{d}^k = \mathbf{d}_N^k$ and $\alpha_k = 1$ for every k. Assume also that there exists $r > 0$ such that the Hessian matrix $\nabla^2 f$ is positive definite and L-Lipschitz continuous on $B(\mathbf{x}^*, r)$. Then there exists $\varepsilon > 0$ such that if $\mathbf{x}^0 \in B(\mathbf{x}^*, \varepsilon)$ the sequence $\{\mathbf{x}^k\}$ converges to $\mathbf{x}^*$ quadratically. Moreover, $\{\nabla f(\mathbf{x}^k)\}$ converges to 0 quadratically.*

Theorems 1.9 - 1.11 together show that, when combined with the line search strategy that we discussed, for convex functions Newton's method exhibits both global convergence and local quadratic convergence.

We already remarked that, when the dimension n of the problem is large, computing the Newton direction at each iteration could be prohibitively expensive. Moreover, when the current point $\mathbf{x}^k$ is far away from the solution, it may not be necessary to use the exact Newton direction, as we could achieve similar results using a cheaper approximation. This is the idea behind Inexact Newton method: at each iteration of the algorithm we take $\mathbf{d}^k$ as a vector that solves the linear system inexactly, up to a given accuracy, and then perform the line search procedure described above. More formally, we take $\mathbf{d}^k \in \mathbb{R}^n$ such that

$$\|\nabla^2 f(\mathbf{x}^k)\mathbf{d}^k + \nabla f(\mathbf{x}^k)\| \leq \eta_k \|\nabla f(\mathbf{x}^k)\|, \text{ with } \eta_k \in [0, 1). \tag{1.18}$$

We can prove the following result, which is the analogous of Theorem 1.11 for Newton's method.

Theorem 1.12. *Let f be a twice continuously differentiable function and let $\mathbf{x}^*$ be a stationary point of f such that $\nabla^2 f(\mathbf{x}^*)$ is positive definite. Assume that the Hessian matrix $\nabla^2 f(\mathbf{x})$ is positive definite in a neighborhood of $\mathbf{x}^*$, and let $\{\mathbf{x}^k\}$ be the sequence generated by* (1.9) *with $\alpha_k = 1$ and $\mathbf{d}_k$ satisfying* (1.18) *for a sequence $\{\eta_k\}$ such that $0 \leq \eta_k \leq \bar{\eta} < 1$. Then there exists $\varepsilon > 0$ such that if $\mathbf{x}^0 \in B(\mathbf{x}^*, \varepsilon)$, we have*

i) for $\bar{\eta}$ small enough the sequence $\mathbf{x}^k$ converges to $\mathbf{x}^*$ linearly

ii) if $\lim_{k \to +\infty} \eta_k = 0$ then the convergence is superlinear

iii) if $\eta_k = \eta \|\nabla f(\mathbf{x}^k)\|^\delta$ for some $\eta > 0$, $\delta \in (0, 1]$ then the convergence is of order $1 + \delta$

In particular we notice that by choosing η_k proportional to $\|\nabla f(\mathbf{x}^k)\|$ we recover the local quadratic convergence result that we stated for

exact Newton's method.

In [52] the authors propose a strategy for the choice of the step size in Newton's method that tries to reduce the number of function evaluation required by the line-search strategy that we discussed, while still ensuring both global convergence and local fast convergence. The details regarding this choice of the step length will be presented in Chapter 3, together with its extension to Inexact Newton method and to the distributed framework.

Regarding Gradient method, so far we considered the fixed step size and the line-search strategy. In [2, 53, 54] the authors propose the Spectral Gradient method. That is, a gradient based that incorporate second order information into the choice of the step size. Given the current point $\mathbf{x}^k$ and the direction $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ the step size is given by $\alpha_k = \sigma_k^{-1}$, where σ_k is defined as follows

$$\sigma_k = \frac{(\mathbf{s}^{k-1})^\top \mathbf{y}^{k-1}}{(\mathbf{s}^{k-1})^\top \mathbf{s}^{k-1}}$$

with $\mathbf{s}^{k-1} = \mathbf{x}^k - \mathbf{x}^{k-1}$ and $\mathbf{y}^{k-1} = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$. The spectral step size corresponds to applying a Quasi Newton method with approximate Hessian matrix $B_k = \sigma_k^{-1} I$, where at each iteration σ_k is chosen as the scalar that best approximates the secant equation $B_k \mathbf{s}^k = \mathbf{y}^k$. That is, $\sigma_k = \arg\min_{\sigma \in \mathbb{R}} \|\sigma^{-1}\mathbf{s}^{k-1} - \mathbf{y}^{k-1}\|$.

1.2.3 Least-Squares Problem

We now briefly focus on the following unconstrained optimization problem, that will be relevant in Chapter 6:

$$\min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) \text{ with } F(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{m} r_j(\mathbf{x}) = \frac{1}{2}\|\mathbf{R}(\mathbf{x})\|_2^2 \tag{1.19}$$

where for every $j = 1, \dots, m$ $r_j : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{R} = (r_1, \dots, r_m) \in \mathbb{R}^m$. The functions r_j and $\mathbf{R}$ are usually referred to as *residuals*, and residuals vector respectively.
The derivatives of F are given by

$$\begin{aligned}\nabla F(\mathbf{x}) &= \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla r_j(\mathbf{x}) = J(\mathbf{x})^\top \mathbf{R}(\mathbf{x}) \\ \nabla^2 F(\mathbf{x}) &= \sum_{j=1}^{m} \nabla r_j(\mathbf{x}) \nabla r_j(\mathbf{x})^\top + \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla^2 r_j(\mathbf{x}) \\ &= J(\mathbf{x})^\top J(\mathbf{x}) + \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla^2 r_j(\mathbf{x})\end{aligned}$$

In particular, it is easy to see that if the residual at the solution $\mathbf{x}^*$ of (1.19) is zero, then the second term in the expression for the Hessian vanishes, and $\nabla^2 F(\mathbf{x}^*) = J(\mathbf{x}^*)^\top J(\mathbf{x}^*)$. More in general, $J(\mathbf{x})^\top J(\mathbf{x})$ provides a good approximation of the Hessian whenever $r_j(\mathbf{x})$ is small. This is particularly relevant in practice because it implies that the Hessian matrix of the objective function can be approximated using only first order derivatives. Classical methods for least-square problems (1.19) rely on this particular property of $\nabla^2 F$.

The first method we consider, called Gauss-Newton method, is an iterative method of the form 1.9, with $\mathbf{d}^k = \mathbf{d}^k_{GN}$ given by

$$J(\mathbf{x}^k)^\top J(\mathbf{x}^k) \mathbf{d}^k_{GN} = -J(\mathbf{x})^\top \mathbf{R}(\mathbf{x}^k). \tag{1.20}$$

Notice that the Gauss-Newton direction $\mathbf{d}^k_{GN}$ is the minimizer of the following local model for F at $\mathbf{x}^k$

$$m_k(\mathbf{d}) = F(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^\top \mathbf{d} + \frac{1}{2} \mathbf{d}^\top J(\mathbf{x}^k)^\top J(\mathbf{x}^k) \mathbf{d}. \tag{1.21}$$

$\mathbf{d}^k_{GN}$ an approximation of the Newton direction $\mathbf{d}^k_N$ (1.12), obtained replacing the Hessian matrix $\nabla^2 F(\mathbf{x})$ with $J(\mathbf{x})^\top J(\mathbf{x})$ and therefore

it is generally cheaper to compute than $\mathbf{d}_N^k$. Moreover, the matrix $J(\mathbf{x})^\top J(\mathbf{x})$ is always positive semi-definite and in particular $\mathbf{d}_{GN}^k$ is a descent direction whenever $J(\mathbf{x}^k)$ is nonsingular.

The following theorems show the global and local convergence properties of Gauss Newton method.

Theorem 1.13. *Let $\{\mathbf{x}^k\}_{k=0}^\infty$ be the sequence generated by Gauss-Newton method with α_k satisfying the line search conditions* (1.15), (1.16). *Let us assume that r_j is continuously differentiable and L-Lipschitz continuous for every $j = 1, \dots, m$, that the level set $\Omega = \{\mathbf{x} \in \mathbb{R}^n \mid F(\mathbf{x}) \leq f(\mathbf{x}^0)\}$ is bounded and that that there exists $\gamma > 0$ such that $\|J(\mathbf{x})\mathbf{z}\| \geq \gamma\|\mathbf{z}\|$ for every $\mathbf{z} \in \mathbb{R}^n$ and every $\mathbf{x} \in \mathcal{N}$ where $\mathcal{N}$ is a neighborhood of Ω. Then*

$$\lim_{k \to +\infty} J(\mathbf{x}^k)^\top \mathbf{R}(\mathbf{x}^k) = 0.$$

Theorem 1.14. *Let us denote with $\mathbf{x}^*$ the solution of* (1.19) *and, given $l > 0$ let us denote with $B_l = \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x} - \mathbf{x}^*\| \leq l\}$. Let us assume that r_j is continuously differentiable and L-Lipschitz continuous in B_r for every $j = 1, \dots, m$, that $F(\mathbf{x}^*) = 0$, and that $J(\mathbf{x})$ is nonsingular on B_l. There exists $\varepsilon > 0$ such that if $\mathbf{x}^0 \in B_\varepsilon$, the sequence $\{\mathbf{x}^k\}_{k=0}^\infty$ generated by Gauss-Newton method with $\alpha_k = 1$ converges to $\mathbf{x}^*$ quadratically.*

Theorem 1.14 shows that, despite using an approximation of the Hessian matrix, under suitable assumptions Gauss-Newton method achieves the same convergence rate as Newton method.

In order to be well defined, the Gauss-Newton method needs the Jacobian to be nonsingular at all considered points. Moreover, even when $J(\mathbf{x}^k)$ is nonsingular, if $\lambda_{\min}(J(\mathbf{x}^k)^\top J(\mathbf{x}^k))$ is small the system (1.20) may be ill-conditioned and therefore it may be hard to compute the direction $\mathbf{d}^k$. The Levenberg-Marquardt (LM) method solves

this issue by adding a regularization term to the coefficient matrix of (1.20). That is, the LM direction at a generic iteration k is defined as

$$(J(\mathbf{x}^k)^\top J(\mathbf{x}^k) + \mu_k I_n)\mathbf{d}_{LM}^k = -J(\mathbf{x})^\top \mathbf{R}(\mathbf{x}^k)$$

for some value of $\mu_k > 0$. We notice that for $\mu_k = 0$ the method becomes equivalent to the Gauss-Newton method discussed above, while if μ_k is large compared to $\|J(\mathbf{x})\|$ the direction $\mathbf{d}_{LM}^k$ tends to be close to the gradient direction. In general, the definition of the damping parameter μ_k is fundamental from both the theoretical and the practical point of view, and several choice have been proposed in literature depending on the assumptions on the considered problem.

In the original version of LM method, the damping parameter μ_k actually plays a double role: the one mentioned above of regularizing the linear system that arises at each iteration of Gauss-Newton method, and that of ensuring global convergence. The idea behind this choice of μ_k is the following. Let us assume that $\mathbf{x}^k$ is the current point, $\mathbf{d}_{LM}^k$ is the LM direction computed with the current value of the damping parameter μ_k, and $\hat{\mathbf{x}}$ is the candidate new step, defined as $\hat{\mathbf{x}} = \mathbf{x}^k + \mathbf{d}_{LM}^k$. We compare the decrease in the objective function with the decrease in the local model (1.21), then we accept/reject the candidate step and update μ_k according to the comparison. That is, given m_k as in (1.21), we consider the ratio

$$\rho = \frac{F(\mathbf{x}^k) - F(\hat{\mathbf{x}})}{m_k(0) - m_k(\mathbf{d}_{LM}^k)}.$$

If ρ is small, the new point is rejected, μ_k is increased and a new direction $\mathbf{d}_{LM}^k$ is computed. If ρ is large enough, the candidate step is accepted and we take $\mu_{k+1} \leq \mu_k$ with the inequality being strict or not depending again on the value of ρ. For this choice of the parameter μ_k and $\alpha_k = 1$ for every k one can prove convergence results

completely analogous to those stated above for Gauss-Newton method.

In particular, for the described method to achieve the same convergence properties of Gauss-Newton, one has to assume that the Jacobian is nonsingular and that $\|\mathbf{R}(\mathbf{x}^*)\| = 0$. These assumptions can be restrictive: many practical applications require the solution of a non-zero residual least squares problem, and the Jacobian matrix is often singular or nearly singular in practice. Different modifications of LM method have been proposed in literature that employs different updating scheme for the sequence $\{\mu_k\}$ and ensure good convergence properties with weaker assumptions [31, 3, 18, 19, 71].

1.2.4 Constrained Optimization

We now briefly consider problem (1.7), in the case where C, referred to as *feasible set* is the subset of points of $\mathbb{R}^n$ that satisfy a given system of equations. That is, we assume

$$C = \{\mathbf{x} \in \mathbb{R}^n \mid h_j(\mathbf{x}) = 0, j = 1, \ldots, m\}$$

where for every $i = j, \ldots, m$ h_j is a real-valued function on $\mathbb{R}^n$.

Analogously to the unconstrained case, we give the following definition

Definition 1.22. *A point* $\mathbf{x}^* \in \mathbb{R}^n$ *is a* local solution *of* (1.7) *if there exists a neighbourhood* $\mathcal{N}$ *of* $\mathbf{x}^*$ *such that* $f(\mathbf{x}^*) \leq f(\mathbf{x})$ *for every* $\mathbf{x} \in \mathcal{N} \cap C$.
A point $\mathbf{x}^* \in \mathbb{R}^n$ *is a* strict local solution *if it is a local solution and the inequality is strict.*

Given a problem of this form, we define the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m :\longrightarrow \mathbb{R}$ as

$$\mathcal{L}(\mathbf{x}, \mathbf{s}) = f(\mathbf{x}) + \sum_{j=1}^{m} s_j h_j(\mathbf{x}).$$

The Langrangian function can be used to describe necessary and sufficient optimality condition for problem (1.7).

Theorem 1.15 (Necessary Conditions)**.** *Assume that for every* $j = 1, \dots, m$ $f, h_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ *are continuously differentiable functions, that* $\mathbf{x}^*$ *is a local solution of* (1.7) *and that the vectors* $\nabla h_1(\mathbf{x}^*), \dots, \nabla h_m(\mathbf{x}^*)$ *are linearly independent. Then* $\mathbf{x}^*$ *is a KKT point for* (1.7)*. That is,* $h_j(\mathbf{x}^*) = 0$ *for* $j = 1, \dots, m$ *and there exists* $\mathbf{s}^* \in \mathbb{R}^m$ *such that*

$$\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \mathbf{s}^*) = 0.$$

If f *and* h_j *are twice continuously differentiable for every* $j = 1, \dots, m$ *the we also have*

$$\mathbf{v}^\top \nabla^2_{\mathbf{xx}} \mathcal{L}(\mathbf{x}^*, \mathbf{s}^*) \mathbf{v} \geq 0 \ \ \forall \mathbf{v} \in \mathcal{C}(\mathbf{x}^*, \mathbf{s}^*)$$

where $\mathcal{C}(\mathbf{x}^*, \mathbf{s}^*) = \{\mathbf{v} \in \mathbb{R}^n \mid \nabla h_j(\mathbf{x}^*)^\top \mathbf{v} = 0, j = 1, \dots, m\}$.

Theorem 1.16 (Sufficient Conditions)**.** *Assume that for every* $j = 1, \dots, m$ $f, h_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ *are twice continuously differentiable functions and that* $\mathbf{x}^* \in C$*. If there exists* $\mathbf{s}^* \in \mathbb{R}^m$ *such that*

$$\begin{cases} \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \mathbf{s}^*) = 0 \\ \mathbf{v}^\top \nabla^2_{\mathbf{xx}} \mathcal{L}(\mathbf{x}^*, \mathbf{s}^*) \mathbf{v} > 0 \ \ \forall \mathbf{v} \in \mathcal{C}(\mathbf{x}^*, \mathbf{s}^*), \mathbf{v} \neq 0 \end{cases}$$

then $\mathbf{x}^*$ *is a strict local solution for* (1.7)*.*

Several methods have been developed in literature for the solution of constrained problems, depending on the properties of the objective function f and the feasible set C.We describe here the quadratic penalty method, that will be relevant in the next chapters.

Assume that we want to solve

$$\min f(\mathbf{x}) \ \text{ s.t. } h_j(\mathbf{x}) = 0 \ j = 1, \dots, m \tag{1.22}$$

with $f, h_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ for $j = 1, \ldots, m$. Given $\beta > 0$ we define the quadratic penalty function Φ_β as follows

$$\Phi_\beta(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2\beta} \sum_{j=1}^{m} h_j(\mathbf{x})^2.$$

The second term in Φ_β, referred to as the penalty term, is a measure of the constraint violation, weighted by the penalty parameter β. The penalty term is 0 at $\mathbf{x} \in \mathbb{R}^n$ if all constraints are satisfied (that is if $h_j(\mathbf{x}) = 0$ for every j) and it becomes larger the further $\mathbf{x}$ is from satisfying the constraint. Intuitively, solving the unconstrained optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi_\beta(\mathbf{x}) \tag{1.23}$$

gives and approximation of the solution of (1.22), with the quality of the approximation depending on the penalty parameter β: if β is very large, minimizing Φ_β is similar to minimizing f over the whole $\mathbb{R}^n$, while the constraint violation becomes more relevant as β decreases. On the one hand, taking a smaller β ensure a better approximation of the solution of (1.22) as in ensures that the solution of (1.23) will be close to satisfy the constraints. On the other hand, typically the condition number of problem (1.23) increases as β approaches 0, thus solving the problem for β small may be difficult.

The idea behind the Quadratic Penalty method is that of solving a sequence of unconstrained problems with objective function Φ_{β_s} for $\{\beta_s\}$ decreasing, taking as initial guess for the $(s+1)$-th problem, the approximate minimizer of Φ_{β_s}.

Algorithm 1.2 (Quadratic Penalty Method)**.**
Input: $\varepsilon_0, \beta_0 > 0,\ \hat{\mathbf{x}}^0 \in \mathbb{R}^n$
1: ***for*** $s = 1, 2, \ldots$ ***do***

2: *use an iterative method starting at* $\hat{\mathbf{x}}^{s-1}$ *to find* $\hat{\mathbf{x}}^s$ *such that*

$$\|\Phi_{\beta_s}(\hat{\mathbf{x}}^s)\| \leq \varepsilon_s$$

3: *choose* $\beta_{s+1} < \beta_s$
4: *choose* $\varepsilon_{s+1} \leq \varepsilon_s$
5: ***end for***

The sequence of penalty parameters β_s can be predefined or updated in an adaptive fashion, depending on the progress made by the algorithm and on the difficulty observed in solving the problem at line 2.

Theorem 1.17. *Let us assume that* f, h_j *are continuously differentiable functions and that* $\{\hat{\mathbf{x}}^s\}$ *is the sequence generated by Algorithm 1.2 with* $\lim_{s\to+\infty} \varepsilon_s = \lim_{s\to+\infty} \beta_s = 0$. *Let* $\bar{\mathbf{x}}$ *be a limit point of* $\{\hat{\mathbf{x}}^s\}$. *If* $\bar{\mathbf{x}}$ *is not feasible, then it is a stationary point of the constraint violation* $\sum_{j=1}^m h_j(\mathbf{x})^2$. *If* $\bar{\mathbf{x}}$ *is feasible and and that the vectors* $\nabla h_1(\bar{\mathbf{x}}), \ldots, \nabla h_m(\bar{\mathbf{x}})$ *are linearly independent, then* $\bar{\mathbf{x}}$ *satisfies the first order necessary optimality conditions for* (1.22).

We notice that the penalty function can also be defined as

$$\Phi_\beta(\mathbf{x}) = \beta f(\mathbf{x}) + \frac{1}{2}\sum_{j=1}^m h_j(\mathbf{x})^2$$

where $\beta > 0$ as the same meaning discussed above. This formulation will be relevant in the following chapters.

Chapter 2

Distributed Optimization

In this chapter we introduce the concept of distributed optimization, as well as some important methods in the field. In Section 2.1 we define the computational framework that we are considering, we discuss the main challenges that it presents, and the main requirements that method have to satisfied in order to be effective in this context. In Section 2.2 we provide a literature review of distributed methods, with particular focus on those algorithms that are most relevant for the results presented in Chapters 3-5.

2.1 Distributed Framework

In distributed optimization, a set of computational agents connected by a given communication network is required to solve an optimization problem, while having only partial information regarding the objective function and communicating with the other nodes according to the architecture of the network. A typical problem in this framework is distributed learning [6], where the goal is to estimate a set of decision variables based on the observations in a given data set, that

happens to be distributed among different computational agents. When designing and studying optimization algorithms in the distributed framework, one is typically concerned, as in the classical centralized case, with convergence to the solution, order of convergence and computational cost, but also with the amount of information that the agents share with each other, which is referred to as *communication traffic*. In general, communication traffic is treated analogously to computational cost: since sharing variables along the network is a time-consuming and energy-consuming operation, one has to try to keep the traffic as low as possible, and we are interested in studying how the communication grows with the number of variables and the number of agents in the network. Moreover, we can expect that there would be some trade-off involving computational and communication cost: if one could assume that communication traffic is inessential (that is, that the nodes can instantly communicate any amount of information), and algorithm could share all relevant quantities (e.g. function values and derivatives) in such a way that all nodes work with complete information, and then typical method from classical optimization could be applied in this context directly. The fact that communication traffic needs to stay low is one of the main reasons why methods need to be developed, that are designed specifically for the distributed framework.

We can distinguish two main types of network architectures, that lead to the development of two main classes of methods:

- **server/worker**: one of the agents (the server) is able to communicate with all other nodes in the network, while the remaining agents typically cannot communicate with each other. This is the standard setting for parallel methods and federated learning. In this framework, the server typically handles the the combination and transmission of the results of the computation carried

out by the workers, and it may manage the synchronization of the network. In some applications (e.g. federated learning) each worker holds privately a local dataset and performs local optimization steps by using its partial knowledge of the problem and the information received by the server. In other cases the partitioning of the dataset and the distribution of the subsets of data to the nodes is handled by the server, which also assigns to the workers computational tasks to be carried out in parallel.

- **decentralized**: none of the computational agents plays a particular role, and the communication network does not usually have any particular topology. In general, one cannot assume the network to be fully connected, nor that there exists a node that is able to communicate with all others. In this setting, all nodes perform both the local optimization and the aggregation step, by combining the results received from the agents in their neighborhood.

We begin by focusing on the decentralized framework. Consider the following unconstrained optimization problem

$$\min_{\mathbf{y}\in\mathbb{R}^n} f(\mathbf{y}), \text{ with } f(\mathbf{y}) = \sum_{i=1}^{N} f_i(\mathbf{y}) \tag{2.1}$$

where for every $i = 1, \ldots, N$ f_i is a real-valued function of $\mathbb{R}^n$. We assume that a set of N computational agents is given, such that for every $i = 1, \ldots, N$ agent i holds privately the function f_i, and that the agents can exchange information with each other according to a given communication network $\mathcal{G}$. That is, agents i and j can communicate if and only if there is an edge in $\mathcal{G}$ between node i and node j.
Given the network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and assuming for the moment that it is simple, undirected, connected and with self-loops at every node, we can define a matrix $W \in \mathbb{R}^{N\times N}$ such that W is symmetric, doubly

stochastic, and respects the sparsity of the the graph $\mathcal{G}$. That is, $w_{ij} = 0$ for every $i, j = 1, \ldots, N$ with $i \neq j$, such that $(i, j) \notin \mathcal{E}$. We refer to W as *consensus matrix* for the network $\mathcal{G}$. A typical choice for W is the Metropolis matrix [65], whose components are defined as follows:

$$\begin{aligned} w_{ij} &= 0 && \text{if} j \notin \mathcal{N}_i \\ w_{ij} &= \frac{1}{1 + \max\{\deg(i), \deg(j)\}} && \text{if} j \in \mathcal{N}_i \\ w_{ii} &= 1 - \sum_{j \neq i} w_{ij}. \end{aligned} \tag{2.2}$$

We assume that each agent i holds a local version $\mathbf{x}_i \in \mathbb{R}^n$ of the vector of variables and we denote with $\mathbf{x} \in \mathbb{R}^{nN}$ the aggregated vector and with F the function $F : \mathbb{R}^{nN} \longrightarrow \mathbb{R}$ associated with the given f. That is

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} \in \mathbb{R}^{nN}, \;\; F(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}_i). \tag{2.3}$$

It is immediate to see that problem (2.1) is equivalent to the following constrained problem

$$\min_{\mathbf{x} \in \mathbb{R}^{nN}} F(\mathbf{x}), \;\; \text{s.t. } \mathbf{x}_i = \mathbf{x}_j, \;\; \forall \; i, j = 1, \ldots, N.$$

Assuming that W is a consensus matrix associated with the network $\mathcal{G}$, we define $\mathcal{W} = W \otimes I_n$. Since W is a doubly stochastic matrix, we have that $\mathcal{W}\mathbf{x} = \mathbf{x}$ if and only if $\mathbf{x}_i = \mathbf{x}_j$ for every $i, j = 1, \ldots, N$ and therefore the problem can be further reformulated as

$$\min_{\mathbf{x} \in \mathbb{R}^{nN}} F(\mathbf{x}), \;\; \text{s.t. } (I - \mathcal{W})^{1/2}\mathbf{x} = 0. \tag{2.4}$$

In Section 1.2.4 we discussed the penalty formulation of constrained problems. Given $\beta > 0$, we define the penalty function for (2.4) as

follows

$$\min_{\mathbf{x}\in\mathbb{R}^{nN}} \Phi_\beta(\mathbf{x}) \text{ with } \Phi_\beta(\mathbf{x}) = \beta F(\mathbf{x}) + \frac{1}{2}\mathbf{x}^\top (I - \mathcal{W})\mathbf{x}. \tag{2.5}$$

An important concept when discussing distributed optimization methods that work within this setting is that of *consensus*. That is, the constraint $(I-\mathcal{W})^{1/2}\mathbf{x} = 0$ being satisfied. In the methods that we consider, each node defines a local sequence of iterates $\{\mathbf{x}_i^k\}_{k=0}^\infty \subset \mathbb{R}^n$ and, ideally, all the local sequences should converge to the same point. Regarding consensus we can distinguish two main classes of methods:

- **exact**: methods in this class enforce the consensus constraint directly, either by applying a primal-dual scheme for the solution of (2.4) or by employing a two-step iteration that combines an optimization phase and a consensus phase and ensures that the sequence of local iterates converges to the average of the iterates at all nodes

- **inexact**: methods in this class apply to the penalty reformulation of problem (2.4). The accuracy of the solution computed by these methods, both in terms of optimality and consensus error is regulated by the penalty parameter.

In the following section, we will discuss methods from both of this classes.

At the beginning of this section we mentioned two main types of network architecture. While the distinction between the server/worker and the decentralized case is neat and gives origin to different computational frameworks, problems and strategies, when considering either of these cases it is important to distinguish a few additional cases, depending on the nature of the network and how it may change over time. In particular, we distinguish the three following situations:

- **constant network**: all nodes are working at all times and the communication network remains the same during the execution of the algorithm

- **time-varying network**: all nodes are working at all times but the communication network may change from one iteration to another. In the decentralized framework this means that an agent may be able to communicate with a different subset of nodes at each iteration, while in the server/worker case it implies that at a given iteration some of the workers may not be able to communicate with the server. This framework typically models situations where some failure in communication between the nodes occurs at some point during the execution of the method, and the case where the agents move in space, so that devices that may have been in each other communication range at the beginning may move further away from each other during the execution of the algorithm.

- **asynchronous**: there is no coordinating clock among the agents so they do not execute the iterations of the considered algorithm in a coordinated fashion. This framework is also used to model the case where only a subset of nodes is active at each iteration. In particular, it is not assumed that all nodes update their local variables at the same time.

This thesis mainly focuses on the decentralized framework, and it discusses method for both the constant and the time-varying network case. The server/worker framework is considered in Chapter 6, where we present an optimization method that exploits a parallelization strategy to solve least squares problems of large dimension. Given the specifical application that we consider, for the server/worker framework we are only interested in the constant network case.

2.2 Literature Review

We consider the same computational framework defined in the previous section. In particular we assume that we have the following optimization problem

$$\min_{\mathbf{y}\in\mathbb{R}^n} f(\mathbf{y}), \text{ with } f(\mathbf{y}) = \sum_{i=1}^{N} f_i(\mathbf{y}) \tag{2.6}$$

and a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of computational agents such that agent i holds the function f_i and can communicate directly with all its neighbors in $\mathcal{G}$. Moreover, we assume that each agent holds a local vector of variables $\mathbf{x}_i \in \mathbb{R}^n$ and we define the function $F : \mathbb{R}^{nN} \to \mathbb{R}$ as in (2.3).

Typical applications that require an optimization problem to be solved in this context are sensor network localization [30], distributed control [45] and distributed learning [6], and many methods have been developed in literature for the solution of problems of this form.

2.2.1 First Order Method

A basic method for the solution of problem (2.6) in the decentralized framework is Distributed Gradient (DG) [49], which also provides the structure for many first order methods. Every iteration of the method consists of two phases: optimization and consensus. In the optimization phase, each agents performs a step in the direction of the local gradient (thus reducing the value of the local objective function), while in the consensus phase each agent computes a weighted average of the local iterate and the local iterates received from its neighbors. The generic k-th iteration of Distributed Gradient method, at node i is defined by the following two equations, corresponding to the two phases

described above.

$$\begin{cases} \hat{\mathbf{x}}_i^{k+1} = \mathbf{x}_i^k - \alpha_k \nabla f_i(\mathbf{x}_i^k) \\ \mathbf{x}_i^{k+1} = \sum_{j=1}^N w_{ij} \hat{\mathbf{x}}_j^{k+1} \end{cases} \tag{2.7}$$

While the global vector $\mathbf{x}^k$ holds all the local variables $\mathbf{x}_i^k$ and therefore it is not available at any node during the execution of the algorithm, to ease the notation we can rewrite (2.7) as follows

$$\mathbf{x}^{k+1} = \mathcal{W}\mathbf{x}^k - \alpha_k \nabla F(\mathbf{x}^k),$$

where $F : \mathbb{R}^{nN} \to \mathbb{R}$ is defined in (2.3). We notice that sequence generated by DG method with fixed step size $\alpha_k = \beta$ for a given $\beta > 0$ is the same as the sequence generated by centralized gradient method with step size 1 applied to the penalty problem (2.5).

It is proved in [49] that, with suitable assumptions over the function f and the underlying communication network $\mathcal{G}$, the sequence generated by (2.7) achieves sublinear convergence to a solution of (2.6) in the case of diminishing step sizes and linear inexact convergence if the step size $\alpha_k = \alpha$ is fixed, where the accuracy reached by the method depends on the size of the step size α.

In [35] the authors show that a gradient-based method with the above structure and fixed step size, cannot in general achieve convergence to the exact solution of (2.6), unless additional information is shared among the agents.
In the same paper the authors propose a method that follows the same structure of DG. However, instead of using as direction at each iteration the local gradient, they propose a gradient tracking strategy: each node holds an additional vector of variables $\mathbf{s}_i^k$, initialized as $\mathbf{s}_i^0 = \nabla f_i(\mathbf{x}_i^0)$ that is updated and shared at each iteration in order to capture information about the gradient of the aggregated objective

function. The resulting method, presented also in [47], is referred to here as DIGing and is defined as follows:

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^{nN} \\ \mathbf{s}^0 = \nabla F(\mathbf{x}^0) \in \mathbb{R}^{nN} \\ \mathbf{x}^{k+1} = \mathcal{W}\mathbf{x}^k - \alpha_k \mathbf{s}^k \\ \mathbf{s}^{k+1} = \mathcal{W}\mathbf{s}^k + \nabla F(\mathbf{x}^{k+1}) - \nabla F(\mathbf{x}^k) \end{cases}$$

Before we state the convergence results for this method, we state the following assumptions over the local objective function f_i and the network $\mathcal{G}$, which will be similar for all the methods that we consider here.

Assumption A1. *(Regularity Assumptions) For every $i = 1, \ldots, N$, f_i is a continuously differentiable function $f_i : \mathbb{R}^n \longrightarrow \mathbb{R}$ and there exist $0 < \mu_i \leq L_i$ such that*

- *∇f_i is L_i-Lipschitz continuous*
- *f_i is μ_i-strongly convex*

Assumption A2. *(Communication Network) The network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is undirected, simple and connected, and it has self-loops at every node (i.e. $(i, i) \in \mathcal{E}$ for every $i = 1, \ldots, N$).*

Assumption A3. *(Consensus Matrix) The matrix $W \in \mathbb{R}^{N \times N}$ is such that*

- *if $(i, j) \in \mathcal{E}$ then $w_{ij} > 0$*
- *if $i \neq j$ and $(i, j) \notin \mathcal{E}$ then $w_{ij} = 0$*
- *W is symmetric and doubly stochastic*

We notice that Assumption A1 implies that the aggregated objective function f is strongly convex and that the gradient ∇f is Lipschitz continuous, with constants given by $\mu = \sum_i \mu_i$ and $L = \sum_i L_i$, respectively.

We also define the following quantities, relevant for the convergence analysis of the methods that we consider.

$$\begin{aligned} \bar{\mathbf{x}}^k &= \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^k \in \mathbb{R}^n \\ \mathbf{g}^k &= \frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{x}_i^k) \in \mathbb{R}^n \\ \mathbf{z}^k &= \begin{pmatrix} \|\mathbf{s}^k - \mathbf{e}\otimes \mathbf{g}^k\| \\ \|\mathbf{x}^k - \mathbf{e}\otimes \bar{\mathbf{x}}^k\| \\ \sqrt{N}\|\bar{\mathbf{x}}^k - \mathbf{y}^*\| \end{pmatrix} \in \mathbb{R}^3. \end{aligned} \tag{2.8}$$

In particular, the three components of vector $\mathbf{z}^k$ provide a measure of the different kinds of errors: $\mathbf{z}_3^k$ is the distance of the average iterate $\bar{\mathbf{x}}$ from the true solution of (2.6), $\mathbf{z}_2^k$ is the consensus error and it is equal to zero if and only if the local iterate is the same at each node, and $\mathbf{z}_1^k$ measures how well the local vector $\mathbf{s}_i^k$ approximates the aggregated gradient $\mathbf{g}^k$.

Theorem 2.1. *[35] Let Assumptions A1-A3 hold and consider the sequence generated by* (2.8) *with* $\mathbf{x}_i^0 \in \mathbb{R}^n$ *and* $\mathbf{z}_i^0 = \nabla f_i(\mathbf{x}_i^0)$ *for every* $i = 1, \ldots, N$ *and fixed step-size* $\alpha_k = \alpha$ *for every* k*, for some* $\alpha > 0$*. Let us define the following matrix, whose entries depend on the choice of* α*:*

$$M(\alpha) = \begin{pmatrix} \sigma + \alpha L & L(\alpha L + 2) & \alpha L^2 \\ \alpha & \lambda_2 & 0 \\ 0 & \alpha L & \tau \end{pmatrix}$$

with $\tau = \max\{|1 - \alpha\mu|, |1 - \alpha L|\}$ *and* $\sigma = \max\{\lambda_2(W), -\lambda_N(W)\}$ *Then we have*

i) for every $k \in \mathbb{N}_0$ the vector $\mathbf{z}^k$ satisfies the following inequality

$$\mathbf{z}^{k+1} \leq M(\alpha)\mathbf{z}^k \leq \lambda_{\max}(M(\alpha))^k \mathbf{z}^0.$$

ii) if α is such that $\lambda_{\max}(M(\alpha)) < 1$ then $\|\mathbf{s}^k - \mathbf{e}\otimes\mathbf{g}^k\|$, $\|\mathbf{x}^k - \mathbf{e}\otimes\bar{\mathbf{x}}^k\|$, and $\|\bar{\mathbf{x}}^k - \mathbf{y}^*\|$ tend to 0 $R-$linearly.

Lemma 2.1. *[35] With the same assumptions of Theorem 2.1, we have the following bounds for $\lambda_{\max}(M(\alpha))$*

i) if $\alpha = \frac{\mu}{L^2}\left(\frac{1-\sigma}{6}\right)^2$ then

$$\lambda_{\max}(M(\alpha)) \leq 1 - \frac{1}{2}\left(\frac{\mu(1-\sigma)}{6L}\right)^2 < 1$$

ii) if $\alpha < 1/L$ then

$$\lambda_{\max}(M(\alpha)) \leq \max\left\{1 - \frac{\alpha\mu}{2},\ \sigma + 5\sqrt{\frac{\alpha L^2}{\mu}}\right\}$$

In particular, we notice that point i) of Lemma 2.1 ensures the existence of a step size such that the sequence generated by the method converges. Moreover, we see that, depending on the value of the regularity constants and σ, the step size $1/L$ that ensures convergence of gradient method in the centralized framework does not necessarily ensure convergence of the method in this case.

In [47] the authors propose a modification of DG, called EXTRA method, that ensures convergence by adding a correction term that depends on the previous iterates and that balances the fact that the

local gradients ∇f_i do not vanish at a solution of (2.1). Extra method is defined by the following equations

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^{nN} \\ \mathbf{x}^1 = \mathcal{W}\mathbf{x}^0 - \alpha_k \nabla F(\mathbf{x}^0) \\ \mathbf{x}^{k+2} = (I + \mathcal{W})\mathbf{x}^{k+1} - \widetilde{\mathcal{W}}\mathbf{x}^k - \alpha \left(\nabla F(\mathbf{x}^{k+1}) - \nabla F(\mathbf{x}^k)\right) \end{cases} \tag{2.9}$$

with $\mathcal{W} = W \otimes I$ and $\widetilde{\mathcal{W}} = \tilde{W} \otimes I$ where $W, \tilde{W}$ are consensus matrices for the network $\mathcal{G}$ that satisfy the following assumptions

Assumption A3'. *The matrices $W, \tilde{W} \in \mathbb{R}^{N\times N}$ are such that*

- *if $(i,j) \in \mathcal{E}$ then $w_{ij}, \tilde{w}_{ij} > 0$*
- *if $i \neq j$ and $(i,j) \notin \mathcal{E}$ then $w_{ij} = \tilde{w}_{ij} = 0$*
- *$W = W^\top$, $\tilde{W} = \tilde{W}^\top$*
- *$\mathrm{Ker}(W - \tilde{W}) = \mathrm{Span}(\mathbf{e})$, $\mathrm{Span}(\mathbf{e}) \subseteq \mathrm{Ker}(I - W)$*
- *$\tilde{W}$ is positive definite and $W \preceq \tilde{W} \preceq \frac{1}{2}(I + W)$*

We notice that if $W \in \mathbb{R}^{n\times n}$ satisfies Assumption A3 and $\tilde{W} = \frac{1}{2}(I + W)$, then Assumption A3' holds.

For this method, the authors prove a similar convergence results to that of (2.8). That is, they prove that if assumptions A1, A2 and A3' hold, then for a fixed step size α small enough, the sequence $\mathbf{x}_i^k$ converges R-linearly to the solution $\mathbf{y}^*$ of (2.6).

In [24] a unified analysis of a class of first-order distributed methods is presented, which in particular includes and extends (2.8) and (2.9) presented above. We assume that at each iteration node i holds two

vectors $\mathbf{x}_i^k$ and $\mathbf{u}_i^k$ in R^n and that the global vectors $\mathbf{x}^k,\ \mathbf{u}^k \in \mathbb{R}^{nN}$, are updated according to the following rules:

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^{nN} \\ \mathbf{u}^0 = 0 \in \mathbb{R}^{nN} \\ \mathbf{x}^{k+1} = \mathcal{W}\mathbf{x}^k - \alpha(\mathbf{u}^k + \nabla F(\mathbf{x}^k)) \\ \mathbf{u}^{k+1} = \mathbf{u}^k + (\mathcal{W} - I)(\nabla F(\mathbf{x}^k) + \mathbf{u}^k - \mathcal{B}^k\mathbf{x}^k) \end{cases} \tag{2.10}$$

where $\mathcal{B}^k \in \mathbb{R}^{nN\times nN}$ is given by $\mathcal{B}^k = bI$ or $\mathcal{B}^k b = \mathcal{W}$ for some constant $b \geq 0$. We notice that for $\mathcal{B}^k = 0$ and $\mathcal{B}^k = \frac{1}{\alpha}\mathcal{W}$ we get the method described in (2.8) and (2.9), respectively.

Theorem 2.2. *[24] Assume that A1-A3 hold and let* $\{\mathbf{x}^k\}_{k=0}^{\infty}$ *be the sequence generated by* (2.10) *starting from a given initial guess* $\mathbf{x}^0 \in \mathbb{R}^{nN}$. *If the step size* α *satisfies the following inequality with* $C = (L + \|\mathcal{B}\|)$ *and* $\sigma = \max\{\lambda_2(W), -\lambda_N(W)\}$

$$\alpha \leq \min\left\{\frac{(1-\sigma)\mu}{19L^2}, \frac{(1-\sigma)^2\mu}{192LC}\right\}$$

then $\mathbf{x}_i^k$ *converges to* $\mathbf{y}^*$ *R-linearly for every* $i = 1, \ldots, N$.

As we already noticed, the assumption that the communication network stays the same at each iteration is very restrictive in practice and therefore it is interesting to generalize the convergence results of distributed methods to the the case of time-varying networks. In [59] the convergence of several first-order methods was generalized to the case of a time-varying network, provided that the network is connected at each iteration. In the same paper, the authors also show that the EXTRA method (2.9) may diverge if the network is not the same at each iteration.

In [47] the same method as in [35] is presented and analyzed in the time-varying case. We denote with $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ the underlying network at iteration k and we assume that for every k we define a consensus matrix W_k. In (2.8) we replace $\mathcal{W}$ with $\mathcal{W}^k = W^k \otimes I$. We make the following assumptions over $\{\mathcal{G}_k\}$ and $\{W^k\}$.

Assumption A2". *For every $k \in \mathbb{N}_0$ the network $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ is undirected and simple, with self-loops at every node*

Assumption A3". *For every $k \in \mathbb{N}_0$ the matrix W^k satisfies assumption A3 for the network $\mathcal{G}_k$. Moreover, there exists $m \in \mathbb{N}$ such that for every $k \in \mathbb{N}_0$*

$$\delta := \sup_{k \geq m-1} \lambda_{\max}\left(W_m^k - \frac{1}{N}\mathbf{e}\mathbf{e}^\top\right) < 1$$

where $W_m^k = W^k W^{k-1} \dots W^{k-m+1}$.

When the networks $\mathcal{G}_k$ are undirected, Assumption A3" is satisfied if for every k the matrix W^k is defined as in (2.2) and there exists $\bar{m} \in \mathbb{N}$ such that the graph $\mathcal{G}_{\bar{m}}^k = (\mathcal{V}, \mathcal{E}_{\bar{m}}^k)$ is connected for every k, where

$$\mathcal{E}_m^k = \bigcup_{j=0}^{\bar{m}-1} \mathcal{E}_{k-j}.$$

In particular, we remark that the communication network $\mathcal{G}_k$ is not required to be connected at all iterations.

Theorem 2.3. *[47] Let Assumptions A1 and A2" hold and assume that $\{\mathbf{x}^k\}_{k=0}^\infty$ is the sequence generated by (2.8) with fixed step size $\alpha_k = \alpha$, $\mathcal{W}_k = W_k \otimes I$ and $\{W^k\}_{k=0}^\infty$ satisfying A3". If the step size α is such that*

$$\alpha \leq \frac{2(1-\delta)^2}{\mu C} \text{ with } C = 3m^2 \frac{L}{\mu}\left(1 + 4\sqrt{NL/\mu}\right)$$

and m, δ given by assumption A3", then $\mathbf{x}_i^k$ converges R-linearly to $\mathbf{y}^*$ for every $i = 1, \ldots, N$.

In [55] an accelerated gradient-based method for the time-varying directed case is proposed, with weaker assumptions over the underlying networks. In [14] the authors considered the problem of minimizing $f(y)+G(y)$ over a closed and convex set $\mathcal{K}$, where f is a possibly non-convex function as in (2.6) and G is a convex nonseparable term, and they propose a gradient-tracking method that achieves convergence in the case of time-varying directed jointly-connected networks for diminishing synchronized step sizes. In [56] the method proposed in [14] is extended with constant step-sizes to a more general framework while in [58] R-linear convergence is proved for [56] with strongly convex $f(y)$. A unifying framework of these methods is presented in [69] and, for the case of constant and undirected networks, in [1].

In all the decentralized methods we considered so far the sequence of the step-sizes is assumed to be fixed and coordinated among all the agents. In [48], [70], [67], [68], and [73] the case of uncoordinated time-constant step sizes is considered, that is, each node has a different step-size but these step sizes are constant in all iterations.

In [25] the authors propose a modification of (2.8), with step-sizes varying both across nodes and iterations. That is, they consider the method given by

$$\begin{cases} \mathbf{x}^0 \in \mathbb{R}^{nN} \\ \mathbf{s}^0 = \nabla F(\mathbf{x}^0) \\ \mathbf{x}^{k+1} = \mathcal{W}\mathbf{x}^k - A_k\mathbf{s}^k \\ \mathbf{s}^{k+1} = \mathcal{W}\mathbf{s}^k + \nabla F(\mathbf{x}^{k+1}) - \nabla F(\mathbf{x}^k), \end{cases} \tag{2.11}$$

where $A_k = \operatorname{diag}(\alpha_1^k I, \ldots, \alpha_N^k I)$ is the matrix of local step sizes at iteration k.

Theorem 2.4. *[25] If Assumptions A1-D2 hold and* $\{\mathbf{x}^k\}_{k=0}^{\infty}$ *is the sequence generated by* (2.11)*, then there exists* $\alpha_{\min} < \alpha_{\max}$ *such that if* $\alpha_{\min} \leq \alpha_i^k \leq \alpha_{\max}$ *for every* $k \in \mathbb{N}_0$ *and every* $i = 1, \ldots, N$ *the sequence converges R-linearly to* $\mathbf{y}^*$*.*

The theorem states that there always exists an interval of step sizes $[\alpha_{\min}, \alpha_{\max}]$ such that the considered method converges, provided that all step sizes falls inside the interval, independently of how the step sizes are chosen by each node at each iteration. In the same paper, the authors propose the following choice for the step size α_i^k, which is an adaptation to the distributed case of the spectral gradient method described in Chapter 1. Such a choice is given by $\alpha_i^k = (\sigma_i^k)^{-1}$ with σ_i^k given by

$$\sigma_i^k = \mathcal{P}_{[\sigma_{min}, \sigma_{max}]} \left(\frac{(\mathbf{d}_i^{k-1})^T \mathbf{y}_i^{k-1}}{(\mathbf{d}_i^{k-1})^T \mathbf{d}_i^{k-1}} + \sigma_i^{k-1} \sum_{j=1}^{n} w_{ij}^k \left(1 - \frac{(\mathbf{d}_i^{k-1})^T \mathbf{d}_j^{k-1}}{(\mathbf{d}_i^{k-1})^T \mathbf{d}_i^{k-1}} \right) \right)$$

where $\mathbf{d}_i^{k-1} = \mathbf{x}_i^k - \mathbf{x}_i^{k-1}$, $\mathbf{y}_i^{k-1} = \nabla f_i(\mathbf{x}_i^k) - \nabla f_i(\mathbf{x}_i^{k-1})$, $\sigma_{min} = 1/\alpha_{\max}$, $\sigma_{max} = 1/\alpha_{min}$ and, given a closed set U, $\mathcal{P}_U$ denotes the projection onto U.
Theorem 2.4 ensures that there exist $\sigma_{\min}, \sigma_{max}$ such that the method (2.11) with α_i^k as above converges. In [25] the authors also provide numerical results that show the effectiveness of the method when compared to order first order method with fixed step size.

2.2.2 Second Order Methods

In [74] the idea behind [35] of sharing the local gradients at each iteration with the goal of approximating the global gradient, is extended to Newton method: at each iteration each node computes the local direction by solving a linear system involving the local Hessian matrix and a right hand side that combines the local gradient and information received from the neighbors at the previous iteration.

The method introduced in [74], referred to as Newton Tracking is defined as follows.

$$\begin{cases} \mathbf{x}^0 = 0 \in \mathbb{R}^{nN} \\ \mathbf{u}^0 = \left(\nabla^2 F(\mathbf{x}^0) + \varepsilon I\right)^{-1} \nabla F(\mathbf{x}^0) \in \mathbb{R}^{nN} \\ \mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{u}^k \\ \mathbf{u}^{k+1} = \left(\nabla^2 F(\mathbf{x}^{k+1}) + \varepsilon I\right)^{-1} \Big(\left(\nabla^2 F(\mathbf{x}^k) + \varepsilon I\right) \mathbf{u}^k \\ \qquad + \nabla F(\mathbf{x}^{k+1}) - \nabla F(\mathbf{x}^k) + \alpha(I - \mathcal{W})(2\mathbf{x}^{k+1} - \mathbf{x}^k)\Big) \end{cases}$$

where $\alpha, \varepsilon > 0$. Equivalently, the update at node i is given by

$$\begin{cases} \mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \mathbf{u}_i^k \\ \mathbf{u}_i^{k+1} = \left(\nabla^2 f_i(\mathbf{x}_i^{k+1}) + \varepsilon I_n\right)^{-1} \Big(\left(\nabla^2 f_i(\mathbf{x}_i^k) + \varepsilon I_n\right) \mathbf{u}_i^k \\ \qquad + \nabla f_i(\mathbf{x}_i^{k+1}) - \nabla f_i(\mathbf{x}_i^k) + \alpha(2\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \\ \qquad - \alpha \sum_{j=1}^N w_{ij}(2\mathbf{x}_j^{k+1} - \mathbf{x}_j^k)\Big) \end{cases}$$

with $\mathbf{x}_i^0 = 0 \in \mathbb{R}^n$ and $\mathbf{u}_i^0 = \left(\nabla^2 f_i(\mathbf{x}_i^{k+1}) + \varepsilon I_n\right)^{-1} \nabla f_i(\mathbf{x}_i^0) \in \mathbb{R}^n$.

Theorem 2.5. *Let Assumptions A1-A3 hold and $\{\mathbf{x}^k\}_{k=0}^{\infty}$ be the sequence generated by the method above for given values of $\alpha, \varepsilon > 0$. If α and ε are such that*

$$\alpha < \varepsilon - \frac{4}{\lambda_{\min}(I - W)} \frac{L^2}{\mu}$$

then $\mathbf{x}_i^k$ converges R-linearly to $\mathbf{y}^*$.

In [43] the authors propose a distributed version of classical Newton method that relies on a truncated Taylor expansion of the inverse Hessian matrix to compute distributedly an approximation of the Newton direction, and employs a penalty formulation of the original problem to ensure consensus among the nodes.

Given $\beta > 0$, we consider the quadratic penalty reformulation of problem (2.4). That is,

$$\min_{\mathbf{x}\in\mathbb{R}^{nN}} \Phi_\beta(\mathbf{x}) \text{ with } \Phi_\beta(\mathbf{x}) = \beta F(\mathbf{x}) + \frac{1}{2}\mathbf{x}^\top (I - \mathcal{W})\mathbf{x}.$$

For $i = 1, \dots, N$ and $k \in \mathbb{N}_0$ we define the following matrices and vectors:

$$\mathbf{g}^k = \nabla\Phi_\beta(\mathbf{x}^k) = \begin{pmatrix} \mathbf{g}_1^k \\ \vdots \\ \mathbf{g}_N^k \end{pmatrix}$$

$$H^k = \nabla^2\Phi_\beta(\mathbf{x}^k) = \begin{pmatrix} H_{11}^k & \dots & H_{1N}^k \\ \vdots & & \vdots \\ H_{N1}^k & \dots & H_{NN}^k \end{pmatrix}.$$

Notice that for $i, j = 1, \dots, N$ we have

$$\begin{aligned} \mathbf{g}_i^k &= \beta\nabla f_i(\mathbf{x}_i^k) + (1 - w_{ii})\mathbf{x}_i^k - \textstyle\sum_j w_{ij}\mathbf{x}_j^k \\ H_{ij}^k &= -w_{ij}I \text{ for } i \neq j \\ H_{ii}^k &= \beta\nabla^2 f_i(\mathbf{x}_i^k) + (1 - w_{ii})I. \end{aligned}$$

Moreover, we denote with D^k the block-diagonal matrix that has diagonal blocks equal to H_{ii}^k for $i = 1, \dots, N$, and we define $B = H^k - D^k$ (notice that B only depends on $\mathcal{W}$ so it is the same at each iteration).

Algorithm 2.1. *[Network Newton]*
Input: $\alpha > 0,\ M \in \mathbb{N}_0$
Iteration k***, node*** i***:***

1: *compute* $\mathbf{g}_i^k$ *and* D_i^k
2: *compute* $\mathbf{d}_i^{(0)} = -(D_{ii}^k)^{-1}\mathbf{g}_i^k$
3: ***for*** $m = 0, \dots, M-1$ ***do***
4: *share* $\mathbf{d}_i^{(m)}$ *with the neighbors*

5: *compute* $\mathbf{d}_i^{(m+1)} = (D_{ii}^k)^{-1}\left(\sum_j B_{ij}\mathbf{d}_j^{(m)} - \mathbf{g}_i^k\right)$
6: ***end for***
7: *set* $\mathbf{d}_i^k = \mathbf{d}_i^{(M)}$
8: *compute* $\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \alpha\mathbf{d}_i^k$
9: *share* $\mathbf{x}_i^{k+1}$ *with the neighbors*

Linear convergence of the method is proved if assumptions A1-A3 hold, for a suitable choice of the step size α.

In [44] the same approximation strategy for the computation of the direction is considered, but the authors propose the application of a primal-dual strategy rather than solving the penalty problem. In [27] the authors propose an inexact version of the method introduced in [44].

All the second order methods we have seen so far achieve linear convergence, provided that the underlying network is strongly connected and constant in time. As remarked in [75], when considering the distributed framework, there are two main issues that need to be addressed in order to design a second order method that achieves superlinear or quadratic convergence: exact consensus and choice of the step size. Both the weighted averaging strategy [11] used in [74] and the primal-dual scheme used by [44] only yield linear convergence of the local vectors to the global consensus vector and therefore methods involving this strategies cannot achieve overall superlinear convergence. On the other hand, it is know from the theory of second order methods in the classical centralized framework that the choice of the step size is of fundamental importance in order for a method to achieve both global convergence and fast local convergence. However, classical line search strategies, which effectively solve this issue in the centralized framework, are not applicable in the distributed setting as they require knowledge of the value of the global function at all nodes and may require several evaluation at each iteration, which would be

prohibitively expensive from the point of view of communication traffic.

To overcome these obstacles, in [75] the authors propose a second order method that achieves quadratic converges by employing a finite-time communication strategy to share the necessary quantities among the whole network, and the step size proposed in [52] for Newton method in the centralized case, which ensures both global convergence and fast local convergence without requiring multiple function evaluations at each iteration. Additional details about this choice of the step size, including its derivation, are provided in Chapter 4.

Algorithm 2.2 (DSF)**.**
Input: $\{S_i\}_{i=1:N}$ *messages at each node*
Node i***:***

set $I_i^0 = \{S_i\}$
for $k = 0, \dots, N-1$ ***do***
  for $j \in \mathcal{N}_i$ ***do***
    take $S \in I_0^{k-1}$ *such that* S *was not received from node* j *and not sent to node* j *at the previous iterations*
    send S *to node* j
  end for
  update I_i^K *adding the messages received by the neighbors*
end for

Assuming that every node starts with a local message S_i, the previous algorithm ensures that after $N-1$ rounds of communication per nodes, all node have the complete set of messages $S_1, \dots, S_N$.

Algorithm 2.3 (DAN)**.**
Input: $\mathbf{x}^0 i = \mathbf{y} \in \mathbb{R}^n \ \forall i = 1, \dots, N, \ \alpha > 0, \ M \in \mathbb{N}_0$
Iteration k***, node*** i***:***

compute $\mathbf{g}_i^k = \nabla f_i(\mathbf{x}^k)$ *and* $H_{ii}^k = \nabla^2 f_i(\mathbf{x}^k)$

2: run DSF with $S_i = (\mathbf{g}_i^k, H_{ii}^k)$
3: compute $\bar{\mathbf{g}}^k = \sum_{j=1}^N \mathbf{g}_i^k$
4: compute $\bar{H}^k = \sum_{j=1}^N H_{ii}^k$
5: compute $\alpha_k = \min\{1, \frac{\mu^2}{L}\frac{1}{\|\bar{\mathbf{g}}^k\|}\}$
6: compute $\mathbf{d}^k = (\bar{H}^k)^{-1}\bar{\mathbf{g}}^k$
7: compute $\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \alpha\mathbf{d}^k$

We notice that the step size and the direction are the same in all nodes at each iteration and therefore, since all agents start with the same initial guess, we have $\mathbf{x}_i^k = \mathbf{x}_j^k$ for every $i, j = 1, \ldots, N$ and for every $k \in \mathbb{N}_0$. Regarding the communication traffic, we notice that DSF (Algorithm 2.2) ensures that the information held by each node is transmitted to the whole network in a finite number of communication rounds. Despite this fact, running DSF at each iteration for all the local Hessian matrices may cause the communication traffic of DAN algorithm to become quite large, making it not suitable for problems of large dimensions. In the same paper the authors also propose a version of DAN method that avoids sharing the full Hessian at each iteration and works with a rank 1 approximation that significantly reduces the communication cost of the method.

2.2.3 Linear Systems

Given a network of computational agents as described in the previous part of the chapter, a matrix $A \in \mathbb{R}^{n\times n}$ and a vector $\mathbf{b} \in \mathbb{R}^n$, we consider the linear system

$$A\mathbf{y} = \mathbf{b} \tag{2.12}$$

where we assume that each node i holds a subset $R_i \subset \{1, \ldots, n\}$ of the rows of the matrix A and the corresponding components of the vector b.

The problem of finding the solution of the linear system above can clearly be reformulated as the problem of minimizing the sum of the residuals in each equation. That is

$$\mathbf{y} = \arg\min_{\mathbf{y}\in\mathbb{R}^n} \sum_{i=1}^{N} f_i(\mathbf{y}) \text{ with } f_i = \sum_{j\in\mathbb{R}_i} \|A_j\mathbf{y} - \mathbf{b}_j\|^2.$$

Since node i holds equation j of the system for $j \in R_i$, it holds the function f_i and therefore the methods from the previous subsections can be used to solve (2.12). Moreover, several method have been developed specifically for the solution of linear systems in the distributed setting [37, 46, 38, 64, 66, 36], while a survey of the methods is presented in [63]. In general, we can distinguish two classes of method. In [46, 66] each node holds a local copy of a subset of the variables and the sequence generated by each node converges to the corresponding components of the solution. That is, when the algorithm terminates, each node does not have the full solution. In [37, 38, 64, 36] each node holds a local copy of the whole vector of variables and therefore, when convergence is reached, each node has the full solution. Here we are interested in the methods that achieve convergence to the full solution at each node.

For every $i = 1, \dots, N$, let us denote with n_i the number of equations held by node i and let us define the following quantities.

$$A_{[i]} = (A_j)_{j\in R_i} \in \mathbb{R}^{n_i\times n}, \quad \mathbf{b}_{[i]} = (b_j)_{j\in R_i} \in \mathbb{R}^{n_i}$$

Each node generates a sequence $\{\mathbf{x}_i^k\}_{k=0}^{\infty}$ as follows.

$$\begin{cases} \mathbf{x}_i^0 \in \mathbb{R}^n \text{ such that } A_{[i]}\mathbf{x}_i^0 = \mathbf{b}_{[i]} \\ \mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \frac{1}{m_i}\mathcal{P}_i\left(m_i\mathbf{x}_i^k - \sum_{j\in\mathcal{N}_i}\mathbf{x}_j^k\right) \end{cases}$$

where for every $i = 1, \dots, N$ $\mathcal{N}_i$ is the neighborhood on i in the network $\mathcal{G}$, $m_i = |\mathcal{N}_i|$ and $\mathcal{P}_i$ is the orthogonal projection over $\mathrm{Ker}(A_{[i]})$.

In [36] and [37] the authors prove that the method converges to the solution of (1.1) for time-varying networks, provided that such solution exists unique and with suitable connectivity assumptions over the sequence of networks.

Chapter 3

Time-Varying First Order Methods

In this chapter, we consider the class of distributed first order methods (2.10), presented in [24] in the case where the underlying communication network changes from iteration to iteration and each node employs a different step size, that also changes through time.
Of special interest is the spectral gradient method (or Barzilai and Borwein method). This method is very popular in centralized optimization due to its efficiency, as reported in numerous studies, for example [8, 15]. In general, the method avoids the famous zig-zag behaviour of the steepest descent and converges much faster. The method was first proposed by Barzilai and Borwein [2]. This reference proves the method's convergence for two-dimensional problems and convex quadratic functions. The analysis is then extended to arbitrary dimensions and convex quadratic functions by Raydan [53]. Minimization of generic functions is considered in [54] in combination with a nonmonomote line search. R-linear convergence for convex quadratic functions has been proved in [9]. In summary, despite its excellent numerical performance, spectral gradient methods are proved to con-

verge without any safeguarding lower and upper bounds on the step size only for strongly convex quadratic costs. Convergence for generic functions beyond convex quadratic is proved only under step size safeguarding, coupled with a line search strategy. Distributed variants of spectral gradient methods and *fixed network topologies* are studied in [25].
The main contributions, given in [39], are the following

- We prove that the methods proposed in [24], referred to here (and also in [59]) as the unified EXTRA and the unified DIGing are robust to time-varying directed networks and time-varying uncoordinated step sizes, i.e., they converge R-linearly in this setting. Up to now, it is only known that these methods converge under static undirected networks [24] or time-varying networks where the network is connected at each iteration [59]. These methods have been previously considered only for time-invariant coordinated step sizes.

- We prove that the method proposed in [25] is robust to time-varying directed networks, which until now is only known to converge for static, undirected networks.

- It is shown in [59] that the Extra method [57] may diverge over time-varying networks, even when the network is connected at every iteration. On the other hand, as we show here, the unified Extra, a variant of Extra proposed in [24], is robust to time-varying networks. Hence, our results reveal that the unified Extra can be considered as a mean to modify Extra and make it robust.

- We provide a thorough numerical study and an analytical study for a special problem structure that demonstrates that the unification strategy in [24] and the spectral gradient-like step-size

selection strategy in [25] exhibit a high degree of robustness to time-varying networks and uncoordinated time-varying step-sizes. More precisely, we show that these strategies converge, when working on time-varying networks, for wider step-size ranges than commonly used strategies such as constant coordinated step-sizes and DIGing algorithmic forms. In addition, we show by simulation that actually a combination of the unification and the spectral step-size strategies further improves robustness.

This chapter is organized as follows. In Section 3.1 we describe the computational framework that we consider and we present the methods that we analyse. In Section 3.2 we prove a convergence theorem for the considered class of methods. In Section 3.3 we show analytically and by simulation that the unification and spectral step-size selection strategies increase robustness of the methods to time-varying networks and uncoordinated step-sizes. We conclude the chapter with some insights over the obtained results and comments on possible interesting research directions.

3.1 The Model and the Class of Considered Methods

We consider the same computational setting of Chapter 2. That is, we assume a set of computational agents is given, and we consider the following optimization problem

$$\min_{\mathbf{y}\in\mathbb{R}^n} f(\mathbf{y}), \text{ with } f(\mathbf{y}) = \sum_{i=1}^{N} f_i(\mathbf{y}) \tag{3.1}$$

where for every $i = 1, \ldots, N$ node i holds the local function f_i, and a local vector of variables $\mathbf{x}_i \in \mathbb{R}^n$.

Given a sequence of matrices $\{M^k\}_k$ and $m \in \mathbb{N}$, we define M_m^k as follows

$$\begin{cases} M_0^k = I \\ M_m^k := M^k M^{k-1} \dots M^{k-m+1} \end{cases}$$

We assumed that at iteration k the N agents are the nodes of a given network $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$, and for every k we define a consensus matrix for the network $\mathcal{G}_k$. Our analysis is based on the following assumptions over the objective function and the sequence of networks.

Assumption B1. *(Regularity Assumptions) For every $i = 1, \dots, N$, f_i is a continuously differentiable function from $\mathbb{R}^n$ to $\mathbb{R}$ and there exist $0 < \mu_i \leq L_i$ such that*

- *∇f_i is L_i-Lipschitz continuous*
- *f_i is μ_i-strongly convex*

In particular this implies that f is μ-strongly convex with $\mu = \sum_i \mu_i$ and the gradient is L-Lipschitz continuous with $L = \sum_i L$, which also imply that for every $\mathbf{y} \in \mathbb{R}^n$

$$\begin{aligned} &\mu_i I \preceq \nabla^2 f_i(\mathbf{y}) \preceq L_i I \text{ for } i = 1, \dots, N \\ &\mu I \preceq \nabla^2 f(\mathbf{y}) \preceq L I \end{aligned}$$

Assumption B2. *(Sequence of Networks) For every $k \in \mathbb{N}_0$, $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ is a directed graph with self-loops at every node, and $\{W^k\}_{k=0}^{\infty} \in \mathbb{R}^{N \times N}$ is such that*

- *W^k is doubly stochastic*
- *if $(i,j) \in \mathcal{E}_k$ then $w_{ij}^k > 0$*
- *$w_{ij}^k = 0$ if $i \neq j$ and $(i,j) \notin \mathcal{E}_k$*

- *there exists* $m \in \mathbb{N}$ *such that* $\sup_{k=0:m-1} \nu_k < 1$, *where*

$$\nu_k = \lambda_{\max}\left(W_m^k - \frac{1}{N}\mathbf{e}\mathbf{e}^\top\right)$$

The assumptions above are the same as Assumption A1, A2",A3" that we saw for DIGing method in the previous chapter.

Remark 3.1. *Assumption B2 does not require the underlying network* $\mathcal{G}_k$ *to be connected at each iteration and can be satisfied by a sequence of jointly connected networks, if the consensus matrix* W^k *is chosen as the Metropolis matrix, defined in* (2.2). *In more detail, the following can be shown. Assume that the positive entries of the weight matrices* W_k*'s are always bounded from below by a positive constant* $\underline{w}$*- (including also the diagonal entries, i.e., assume that the diagonal entries of* W_k *are always greater than or equal to* $\underline{w}$*). Furthermore, assume network connectedness over bounded intercommunication intervals. That is, for any fixed iteration* k*, consider the graph* $G_k^m = (\{1,\dots,n\}, E_k^m)$, $E_k^m = \cup_{\ell=k-m+1}^{k} E_k$, *whose set of links is the union of the sets of links of graphs at time instances* $\ell = k-m+1, ..., k$. *Assume that* G_k^m *is strongly connected, for every* k. *Now, it is easy to show that the above assumptions imply that* $\nu_{\max}\left(W_k^m - \frac{1}{n}\mathbf{e}\mathbf{e}^\top\right) < 1$.

We assume that each node holds two local vectors of variables $\mathbf{x}_i, \mathbf{u}_i \in \mathbb{R}^n$ and we define the aggregated vectors $\mathbf{x}, \mathbf{u}$ and the aggregated function F as

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} \in \mathbb{R}^{nN}, \quad \mathbf{u} = \begin{pmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_N \end{pmatrix} \in \mathbb{R}^{nN}, \quad F(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}_i). \tag{3.2}$$

We consider the class of methods defined in (2.10), extended to the case of time-varying network and time-varying uncoordinated step

sizes. That is, we assume that each node generates two sequences $\{\mathbf{x}_i^k\}_{k=0}^{\infty}, \{\mathbf{u}_i^k\}_{k=0}^{\infty}$, with $\mathbf{x}_i^0 \in \mathbb{R}^n$, $\mathbf{u}_i^0 = 0$ and we assume that at each iteration the following update is performed:

$$\begin{cases} \mathbf{x}^{k+1} = \mathcal{W}^k \mathbf{x}^k - A^k(\mathbf{u}^k + \nabla F(\mathbf{x}^k)) \\ \mathbf{u}^{k+1} = \mathbf{u}^k + (\mathcal{W}^k - I)(\nabla F(\mathbf{x}^k) + \mathbf{u}^k - B^k \mathbf{x}^k) \end{cases} \tag{3.3}$$

where $\mathcal{W}^k := (W^k \otimes I) \in \mathbb{R}^{nN \times nN}$, $A^k = \operatorname{diag}(\alpha_1^k I, \ldots, \alpha_n^k I)$ with α_i^k being the step-size for node i at iteration k and B^k is a symmetric $n \times n$ matrix that respects the sparsity structure of the communication network $\mathcal{G}_k$ and such that for every $\mathbf{y} \in \mathbb{R}^n$ we have $B^k(1 \otimes \mathbf{y}) = c(1 \otimes \mathbf{y})$ for some $c \in \mathbb{R}$. Moreover, we assume that $x^0 \in \mathbb{R}^{nN}$ is an arbitrary vector and $\mathbf{u}^0 = 0 \in \mathbb{R}^{nN}$.

For $B^k = 0$ and appropriate choice of the step-sizes α_i^k we get the method introduced in [25]. For $A^k = \alpha I$, if $B^k = bI$ or $B^k = b\mathcal{W}$ we retrieve the class of methods analyzed in [24] while if $B_k = 0$ we retrieve the DIGing method proposed in [47, 35]. For $A^k = \alpha I$ and $B^k = b\mathcal{W}$ with $b = \frac{1}{\alpha}$ we have the EXTRA method [57], but the analysis the we carry out in the following section requires the matrix B^k to be independent on the step sizes and therefore it does not apply to EXTRA method. In fact, it is known [59] that EXTRA method does not converge in general for time-varying networks, even with stronger assumptions on the sequence of networks than those we consider here. In our analysis, we consider the case $B^k = bI$ and $B^k = b\mathcal{W}^k$ with b nonnegative constant and $\alpha_{\min} \leq \alpha_j^k \leq \alpha_{\max}$ for every k and every $j = 1, \ldots, n$ for appropriately chosen safeguards $0 < \alpha_{\min} < \alpha_{\max}$.

3.2 Convergence Analysis

Let us denote with $\mathbf{y}^*$ the solution of (3.1) and with $\mathbf{x}^*$ the following vector

$$\mathbf{x}^* := \begin{pmatrix} \mathbf{y}^* \\ \vdots \\ \mathbf{y}^* \end{pmatrix} \in \mathbb{R}^{nN}.$$

In this section, we prove that, under Assumptions B1 and B2, there exist an interval $[\alpha_{\min}, \alpha_{\max}] \subset \mathbb{R}$ with $0 < \alpha_{\min} < \alpha_{\max}$ such that, the sequence $\{\mathbf{x}^k\}_{k=0}^{\infty}$ generated by (3.3) converges to $\mathbf{x}^*$, provided that the step size α_i^k belongs to the interval for every $i = 1, \ldots, N$ and for every $k \in \mathbb{N}_0$.

Given a vector $\mathbf{v} \in \mathbb{R}^{nN}$, denote with $\bar{\mathbf{v}}$ the average $\bar{\mathbf{v}} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{v}_j \in \mathbb{R}^n$ and with J the $n \times n$ matrix $\left(I - \frac{1}{N}\mathbf{e}\mathbf{e}^\top\right)$. Moreover, we define the following quantities.

$$\begin{aligned} \tilde{\mathbf{x}}^k &= \mathbf{x}^k - \mathbf{e}\bar{\mathbf{x}}^k \in \mathbb{R}^{nN}, \\ \tilde{\mathbf{u}}^k &= \mathbf{u}^k + \nabla F(\mathbf{x}^*) \in \mathbb{R}^{nN}, \\ \mathbf{q}^k &= \mathbf{x}^k - \mathbf{x}^* = \tilde{\mathbf{x}}^k + \mathbf{e}\bar{\mathbf{q}}^k \in \mathbb{R}^{nN}. \end{aligned}$$

To ease the notation, in the rest of this section we assume that $n = 1$. In the general case the analysis proceeds analogously.
Since W^k is doubly stochastic, we have that $\frac{1}{N}\mathbf{e}\mathbf{e}^\top(W^k - I) = 0$. Using this equality and the definition of $\mathbf{u}^{k+1}$ we get

$$\begin{aligned} \bar{\mathbf{u}}^{k+1} &= \frac{1}{N}\mathbf{e}\mathbf{e}^\top \mathbf{u}^{k+1} = \\ &= \frac{1}{N}\mathbf{e}\mathbf{e}^\top \left(\mathbf{u}^k + (W^k - I)(\mathbf{u}^k + \nabla F(\mathbf{x}^k) - B^k\mathbf{x}^k)\right) = \\ &= \frac{1}{N}\mathbf{e}\mathbf{e}^\top \mathbf{u}^k = \bar{\mathbf{u}}^k \end{aligned}$$

and given that $\mathbf{u}^0 = 0$, we have

$$\bar{\mathbf{u}}^k = 0 \text{ for every } k \in \mathbb{N}_0. \tag{3.4}$$

Since $\mathbf{x}_i^* = \mathbf{y}^*$ for every index i, we have that

$$\frac{1}{N}\mathbf{e}\mathbf{e}^\top \nabla F(\mathbf{x}^*) = \frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{x}^*) = \frac{1}{N}\nabla f(\mathbf{y}^*)$$

and therefore, from the definition of $\tilde{\mathbf{u}}^k$, (3.4) and the fact that $\mathbf{y}^*$ is stationary point of the function f we get

$$\frac{1}{N}\mathbf{e}\mathbf{e}^\top \tilde{\mathbf{u}}^k = \frac{1}{N}\mathbf{e}\mathbf{e}^\top(\mathbf{u}^k + \nabla F(\mathbf{x}^*)) = \bar{\mathbf{u}}^k + \frac{1}{N}\nabla f(\mathbf{y}^*) = 0. \tag{3.5}$$

From Assumption B1, for every iteration index k there exists a matrix $H_k \preceq LI$ such that

$$\nabla F(\mathbf{x}^k) - \nabla F(\mathbf{x}^*) = H_k(\mathbf{x}^k - \mathbf{x}^*). \tag{3.6}$$

Lemma 3.1. *[47] If the matrix sequence $\{W^k\}_k$ satisfies assumption A2, then for every $k \geq m$ we have*

$$\left\|JW_m^k y\right\| \leq \nu_k \left\|Jy\right\|$$

Lemma 3.2. *[51] If the function f satisfies assumption A1 and $0 < \alpha < \frac{1}{L}$, then*

$$\|y - \alpha\nabla f(y) - y^*\| \leq \tau\|y - y^*\|$$

where $\tau = \max\{|1 - \alpha\mu|, |1 - \alpha L|\}$

The convergence result that we prove is based on the Small Gain Theorem [13], stated in Section 1.1, which relies on the following definitions. Given an infinite sequence of vectors $\mathbf{v} := \{\mathbf{v}^k\}_{k=0}^{\infty}$ with

$\mathbf{v}^k \in \mathbb{R}^n$ and two constant $\delta \in (0,1)$ and $K \in \mathbb{N}_0$ we define the following quantities

$$\|\mathbf{v}\|^{\delta,K} = \max_{k=0,1,\dots,K} \left\{ \frac{1}{\delta^k} \|\mathbf{v}^k\| \right\}$$
$$\|\mathbf{v}\|^{\delta} = \sup_{k\geq 0} \left\{ \frac{1}{\delta^k} \|\mathbf{v}^k\| \right\}.$$

We will use the following technical Lemma to show that the sequences $\|\bar{\mathbf{q}}^k\|$ and $\|\tilde{\mathbf{x}}^k\|$ satisfy the hypotheses of Theorem 1.4.

Lemma 3.3. *Given $b, \mu, L \geq 0$, $\nu \in (0,1)$ and $n, m \in \mathbb{N}$, there exists $\delta \in (0,1)$ and $0 \leq \alpha_{\min} < \alpha_{\max}$ such that the following conditions hold:*

i) $\nu < \delta^m$;

ii) $\frac{\alpha_{\min}}{N} < \frac{2}{L}$;

iii) $1 - \mu\alpha_{\min} + \Delta L < \delta$;

iv) $\gamma\beta_2 < 1$;

v) $\beta_3 < 1$;

vi) $\frac{\beta_5\gamma}{1-\beta_3} < 1$;

vii) $\frac{\beta_1+\gamma\beta_2}{1-\gamma\beta_2} \cdot \frac{\beta_4+\gamma\beta_5}{1-\beta_3-\gamma\beta_5} < 1$,

where

$$\gamma = \frac{(b+L)C}{\delta^m - \nu}, \qquad \beta_1 = \frac{L\alpha_{\max}}{\delta - 1 + \mu\alpha_{\min} - \Delta L},$$
$$\beta_2 = \frac{\Delta}{L\alpha_{\max}}\beta_1, \qquad \beta_3 = \frac{\nu}{\delta^m} + \beta_4,$$
$$\beta_4 = L\beta_5, \qquad \beta_5 = \frac{C\alpha_{\max}}{\delta^m},$$
$$\Delta = \alpha_{\max} - \alpha_{\min}, \qquad C = \frac{\delta(1-\delta^m)}{1-\delta}.$$

Proof. Take $\delta^m > \nu$ and $\alpha_{\min} < \frac{2n}{L}$ so that i) and ii) hold. For $\alpha_{\max} > \alpha_{\min}$ and close enough to $\alpha_{\min}$ one can ensure that

$$\frac{\alpha_{\max}}{\alpha_{\min}} < 1 + \frac{\mu}{L} \tag{3.7}$$

holds. By the previous inequality, we have $1 - \mu\alpha_{\min} + \Delta L < 1$ and therefore, for fixed $\alpha_{\max}$ and $\alpha_{\min}$ we can always take $\delta \in (0, 1)$ such that iii) is satisfied and i). still holds. Moreover, we can take $\alpha_{\min}$ arbitrarily small and $\alpha_{\max}$ arbitrarily close to $\alpha_{\min}$ without violating conditions i)-iii). Notice that $C = \frac{\delta(1-\delta^m)}{1-\delta}$ is an increasing function of δ.
Let us now consider condition iv) given by

$$\frac{(b + L)C\Delta}{(\delta^m - \nu)(\delta - 1 + \mu\alpha_{\min} - \Delta L)} < 1.$$

The left hand side expression is an increasing function of Δ and it is equal to 0 for $\Delta = 0$. Therefore, taking $\alpha_{\max}$ close enough to $\alpha_{\min}$, condition iv) holds.
Condition v) holds for $\alpha_{\max} < \frac{\delta^m - \nu}{\delta^m LC}$.
Consider now condition vi),

$$\frac{(b + L)C^2\alpha_{\max}}{(\delta^m - \nu)(\delta^m - \nu)(\delta^m - \nu - L\alpha_{\max}C)} < 1$$

The left hand side expression is an increasing function of $\alpha_{\max}$ and taking $\alpha_{\max}$ small enough we conclude that the previous inequality holds. Since we need $\alpha_{\max} > \alpha_{\min}$, in order to be able to take $\alpha_{\max}$ small, we need to take $\alpha_{\min}$ small enough, but this can be done without violating the previous conditions.

By definition, $\frac{\beta_2 + \gamma\beta_3}{1-\gamma\beta_3}$ and $\frac{\beta_5 + \gamma\beta_6}{1-\beta_4-\gamma\beta_6}$ are also increasing functions of $\alpha_{\max}$ and Δ. Thus, we can apply the same reasoning that we applied to iv) and vi) to get $\gamma_2 < 1$ and $\gamma_3 < 1$. In particular, we can take $\alpha_{\min}$ and $\alpha_{\max}$ such that condition vii) holds. □

We can now prove the following convergence result.

Theorem 3.1. *Assume that B1 and B2 hold and let* $\{\mathbf{x}^k\}$ *be the sequence generate by* (3.3) *with* $B^k = bW^k$ *or* $B^k = bI$ *for any* $b > 0$ *or* $B^k = 0$*, and* $\alpha_{\min} \leq \alpha_i^k \leq \alpha_{\max}$ *for every* $i = 1, \ldots, N$ *and every* $k \in \mathbb{N}_0$*. Then there exists* $\alpha_{\min} < \alpha_{\max}$ *such that the sequence* $\{\mathbf{x}^k\}$ *converges R-linearly to* $\mathbf{x}^*$*.*

Proof. Let us define $\nu = \sup_{k=0:m-1} \nu_k < 1$ where the sequence $\{\nu_k\}$ is given in assumption B2, and take $\delta \in (0,1), 0 \leq \alpha_{\min} < \alpha_{\max}$ given by Lemma 3.3. We prove that $N^{1/2}\bar{\mathbf{q}}^k$ and $\tilde{\mathbf{x}}^k$ satisfy the assumptions of Theorem 1.4 thus ensuring R-linear convergence. That is, we will prove that there exist $\gamma_2, \gamma_3 \geq 0$ and $w_2, w_3 \in \mathbb{R}$ such that the product $\gamma_2\gamma_3$ is smaller than 1 and the following inequalities hold

$$\begin{aligned} \|N^{1/2}\bar{\mathbf{q}}\|^{\delta,K} &\leq \gamma_2 \|\tilde{\mathbf{x}}\|^{\delta,K} + w_2, \\ \|\tilde{\mathbf{x}}\|^{\delta,K} &\leq \gamma_3 \|N^{1/2}\bar{\mathbf{q}}\|^{\delta,K} + w_3. \end{aligned} \tag{3.8}$$

Since $B^k = bI$ or $B^k = bW^k$, we have $B^k\mathbf{x}^* = b\mathbf{x}^*$, therefore $(W^k - I)B^k\mathbf{x}^* = 0$ and

$$(W^k - I)B^k\mathbf{x}^k = (W^k - I)B^k(\mathbf{x}^k - \mathbf{x}^*) = (W^k - I)B^k\mathbf{q}^k$$

For $k \geq m-1$, using (3.3), the previous equality and (3.6), we get

$$\begin{aligned} \tilde{\mathbf{u}}^{k+1} &= \mathbf{u}^{k+1} + \nabla F(x^*) = \\ &= \mathbf{u}^k + (W^k - I)(\mathbf{u}^k + \nabla F(x^k) - B^k\mathbf{x}^k) + \nabla F(x^*) = \\ &= W^k(\mathbf{u}^k + \nabla F(x^*)) + (W^k - I)(\nabla F(x^k) - \nabla F(x^*)) \\ &\quad - (W^k - I)B^k\mathbf{x}^k = \\ &= W^k\tilde{\mathbf{u}}^k + (W^k - I)H_k\mathbf{q}^k - (W^k - I)B^k\mathbf{q}^k. \end{aligned}$$

Applying this equality recursively, we have

$$\tilde{\mathbf{u}}^{k+1} = W_m^k\tilde{\mathbf{u}}^{k-m+1} + \sum_{t=0}^{m-1} W_t^k(W^{k-t} - I)\left(H_{k-t} - B^{k-t}\right)\mathbf{q}^{k-t}. \tag{3.9}$$

By (3.5) and Lemma 3.1 we can bound the first term on the right hand side,

$$\begin{aligned}\|W_m^k \tilde{\mathbf{u}}^{k-m+1}\| = \|W_m^k J\tilde{\mathbf{u}}^{k-m+1}\| \leq \nu\|J\tilde{\mathbf{u}}^{k-m+1}\| = \\ = \nu\|\tilde{\mathbf{u}}^{k-m+1}\|\end{aligned}$$

and by (3.6), the definition of B^k and the fact that W^k is doubly stochastic, we get the following inequality for each term of the sum

$$\|W_t^k(W^{k-t} - I)(H_{k-t} - B^k)\mathbf{q}^{k-t}\| \leq (L+b)\|\mathbf{q}^{k-t}\|.$$

Taking the norm in (3.9) and using the two bounds that we just found, we have that for $k \geq m-1$

$$\|\tilde{\mathbf{u}}^{k+1}\| \leq \nu\|\tilde{\mathbf{u}}^{k-m+1}\| + (b+L)\sum_{t=0}^{m-1}\|\mathbf{q}^{k-t}\|. \tag{3.10}$$

Notice that the above inequality also holds for the case $B^k = 0$, taking $b = 0$. Multiplying both side of (3.10) by $\frac{1}{\delta^{k+1}}$, taking the maximum for $k = -1 : \bar{k} - 1$, and defining

$$\tilde{\omega}_1 = \max_{k=-1:m-1}\left\{\frac{1}{\delta^{k+1}}\|\tilde{\mathbf{u}}^{k+1}\|\right\}$$

we get

$$\begin{aligned}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} &= \max_{k=-1:m-1}\left\{\frac{1}{\delta^{k+1}}\|\tilde{\mathbf{u}}^{k+1}\|\right\} + \max_{k=m:\bar{k}}\left\{\frac{1}{\delta^{k+1}}\|\tilde{\mathbf{u}}^{k+1}\|\right\} \\ &\leq \frac{\nu}{\delta^m}\max_{k=m:\bar{k}}\left\{\frac{1}{\delta^{k-m+1}}\|\tilde{\mathbf{u}}^{k-m+1}\|\right\} \\ &\quad + (b+L)\sum_{t=0}^{m-1}\frac{1}{\delta^t}\max_{k=m:\bar{k}}\left\{\frac{1}{\delta^{k-t}}\|\mathbf{q}^{k-t}\|\right\} + \tilde{\omega}_1 \\ &\leq \frac{\nu}{\delta^m}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} + \frac{(b+L)}{\delta^m}\frac{\delta(1-\delta^m)}{(1-\delta)}\|q\|^{\delta\bar{k}} + \tilde{\omega}_1.\end{aligned}$$

Since by condition 1. in Lemma 3.3 we have $\nu < \delta^m$, reordering the terms in the previous inequality and using the fact that $\mathbf{q}^k = \tilde{\mathbf{x}}^k + \mathbf{e}\bar{\mathbf{q}}^k$, we get

$$\begin{aligned} \|\tilde{\mathbf{u}}\|^{\delta\bar{k}} &\leq \gamma_1 \|q\|^{\delta\bar{k}} + \omega_1 \\ &\leq \gamma_1 \|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \gamma_1 N^{1/2} \|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \omega_1 \end{aligned} \tag{3.11}$$

with

$$\gamma_1 = \frac{(b+L)\delta(1-\delta^m)}{(1-\delta)(\delta^m - \nu)}, \qquad \omega_1 = \frac{\delta^m}{\delta^m - \nu}\tilde{\omega}_1.$$

Let us now consider $\bar{\mathbf{q}}^k$.

$$\begin{aligned} \bar{\mathbf{q}}^{k+1} &= \bar{\mathbf{x}}^{k+1} - y^* = \frac{1}{N}\mathbf{e}\mathbf{e}^\top \mathbf{x}^{k+1} - y^* = \\ &= \frac{1}{N}\mathbf{e}\mathbf{e}^\top \left(W^k \mathbf{x}^k - A^k(\mathbf{u}^k + \nabla F(\mathbf{x}^k)\right) - y^* = \\ &= \bar{\mathbf{x}}^k - y^* - \frac{\alpha_{\min}}{N}\nabla F(\bar{\mathbf{x}}^k) \\ &+ \frac{\alpha_{\min}}{N}\sum_{j=1}^n (\nabla f_j(\bar{\mathbf{x}}^k) - \nabla f_j(y^*)) \\ &- \frac{1}{N}\sum_{j=1}^n (\alpha_j^k - \alpha_{\min})(\nabla f_j(\mathbf{x}_j^k) - \nabla f_j(y^*)) \\ &+ \frac{1}{N}\sum_{j=1}^n (\alpha_{\min} - \alpha_j^k)\tilde{\mathbf{u}}_j^k. \end{aligned}$$

Taking the norm, by Lipschitz continuity of the gradient and denoting with $\Delta = \alpha_{\max} - \alpha_{\min}$, we have

$$\begin{aligned} \|\bar{\mathbf{q}}^{k+1}\| = \left\|\bar{\mathbf{x}}^k - y^* - \frac{\alpha_{\min}}{N}\nabla F(\bar{\mathbf{x}}^k)\right\| + \frac{L\alpha_{\min}}{N}\|\bar{\mathbf{x}}^k - y^*\|_1 \\ + \frac{L\Delta}{N}\|\mathbf{x}^k - \mathbf{x}^*\|_1 + \frac{\Delta}{N}\|\tilde{\mathbf{u}}_j^k\|_1. \end{aligned}$$

Since $\frac{\alpha_{\min}}{N} < \frac{2}{L}$, Lemma 3.2 ensures that

$$\left\| \bar{\mathbf{x}}^k - y^* - \frac{\alpha_{\min}}{N} \nabla F(\bar{\mathbf{x}}^k) \right\| \leq \tau \|\bar{\mathbf{x}}^k - \mathbf{y}^*\|$$

and we get

$$\begin{aligned} N^{1/2}\|\bar{\mathbf{q}}^{k+1}\| &\leq N^{1/2}\tau\|\bar{\mathbf{x}}^k - y^*\| + L\alpha_{\min}\|\bar{\mathbf{x}}^k - y^*\| \\ &\quad + L\Delta\|\mathbf{x}^k - \mathbf{x}^*\| + \Delta\|\tilde{\mathbf{u}}_j^k\| \\ &\leq N^{1/2}(\tau + \Delta L)\|\bar{\mathbf{q}}^k\| + L\alpha_{\max}\|\tilde{\mathbf{x}}^k\| + \Delta\|\tilde{\mathbf{u}}^k\|. \end{aligned}$$

Multiplying both sides by $\frac{1}{\delta^{k+1}}$ and taking the maximum for $k = -1 : \bar{k} - 1$ we have

$$N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} \leq \frac{\tau + \Delta L}{\delta} N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \frac{L\alpha_{\max}}{\delta}\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \frac{\Delta}{\delta}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}}.$$

By Lemma 3.3 we have $\tau = 1 - \mu\alpha_{\min}$ and $\tau + \Delta L < \delta$, thus reordering and using (3.11), we get

$$\begin{aligned} N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} &\leq + \frac{L\alpha_{\max}}{\delta - \tau - \Delta L}\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \frac{\Delta}{\delta - \tau - \Delta L}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} \\ &\leq (\beta_1 + \gamma_1\beta_2)\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \gamma_1\beta_2 N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \beta_2\omega_1 \end{aligned}$$

where β_1 and β_2 are defined in Lemma 3.3. Take

$$\gamma_2 = \frac{\beta_1 + \gamma_1\beta_2}{1 - \gamma_1\beta_2}, \quad \omega_2 = \frac{\beta_2\omega_1}{1 - \gamma_1\beta_2}.$$

From 4. in Lemma 3.3 we get

$$N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} \leq \gamma_2\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \omega_2. \tag{3.12}$$

Finally, let us consider $\tilde{\mathbf{x}}^k$. For $k \geq m-1$, we have

$$\begin{aligned}
\tilde{\mathbf{x}}^{k+1} &= J(W^k\mathbf{x}^k - A^k(\mathbf{u}^k + \nabla F(\mathbf{x}^k)) = \\
&= JW^kW^{k-1}\mathbf{x}^{k-1} - JW^kD^{k-1}(\mathbf{u}^{k-1} + \nabla F(x^{k-1}))- \\
&- JD^k(\mathbf{u}^k + \nabla F(x^k)) = \\
&= JW_m^k\mathbf{x}^{k-m+1} - J\sum_{t=0}^{m-1} W_t^kD^{k-t}(\tilde{\mathbf{u}}^{k-t} + H_{k-t}\mathbf{q}^{k-t}).
\end{aligned}$$

Taking the norm, applying Lemma 3.1 and (3.6), we get

$$\|\tilde{\mathbf{x}}^{k+1}\| \leq \nu\|\tilde{\mathbf{x}}^{k-m+1}\| + \alpha_{\max}\sum_{t=0}^{m-1}(\|\tilde{\mathbf{u}}^{k-t}\| + L\|\mathbf{q}^{k-t}\|).$$

Multiplying with $\frac{1}{\delta^{k+1}}$ and taking the maximum for $k = -1 : \bar{k}-1$ we get

$$\begin{aligned}
\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} &\leq \frac{\nu}{\delta^m}\|\tilde{\mathbf{x}}^{k-m+1}\| + \alpha_{\max}\frac{\delta(1-\delta^m)}{\delta^m(1-\delta)}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} \\
&\quad + L\alpha_{\max}\frac{\delta(1-\delta^m)}{\delta^m(1-\delta)}\|q\|^{\delta\bar{k}} + \tilde{\omega}_3 \\
&\leq \beta_3\|\tilde{\mathbf{x}}^{k-m+1}\| + \beta_4N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \beta_5\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} + \tilde{\omega}_3.
\end{aligned}$$

where

$$\tilde{\omega}_3 = \max_{k=-1:m-1}\left\{\frac{1}{\delta^{k+1}}\|\tilde{\mathbf{x}}^{k+1}\|\right\}$$

and $\beta_3, \beta_4, \beta_5$ are defined in Lemma 3.3. In particular, we have $\beta_3 < 1$, and we can rearrange the terms of the previous inequality to get

$$\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} \leq \frac{\beta_4}{1-\beta_3}N^{1/2}\|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \frac{\beta_5}{1-\beta_3}\|\tilde{\mathbf{u}}\|^{\delta\bar{k}} + \frac{\tilde{\omega}_3}{1-\beta_3}.$$

Now, applying (3.11) and 6. from Lemma 3.3, we obtain

$$\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} \leq \gamma_3 N^{1/2} \|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \omega_3$$

with

$$\gamma_3 = \frac{\beta_4 + \beta_5\gamma_1}{1 - \beta_3 - \gamma_1\beta_5}, \quad \omega_3 = \frac{\tilde{\omega}_3 + \beta_5\omega_1}{1 - \beta_3 - \gamma_1\beta_5}.$$

We thus proved

$$N^{1/2} \|\bar{\mathbf{q}}\|^{\delta\bar{k}} \leq \gamma_2 \|\tilde{\mathbf{x}}\|^{\delta\bar{k}} + \omega_2$$
$$\|\tilde{\mathbf{x}}\|^{\delta\bar{k}} \leq \gamma_3 N^{1/2} \|\bar{\mathbf{q}}\|^{\delta\bar{k}} + \omega_3$$

with $\gamma_2\gamma_3 < 1$ by condition 7. in Lemma 3.3. By the Small Gain Theorem, we have that $\|\bar{\mathbf{q}}^k\|$ and $\|\tilde{\mathbf{x}}^k\|$ converge to 0, and thus $\|\mathbf{q}^k\|$ converges to zeros, which gives the thesis. □

3.3 Robustness - Analytical and Numerical Study

Theorem 3.1 and Lemma 3.3 ensure convergence of the considered class of methods. That is, they ensure existence of $\alpha_{\min} < \alpha_{\max}$ such that, under the given assumptions over the objective function and the sequence of networks, the sequence generated at each node converges R-linearly if all the step sizes α_i^k are between $\alpha_{\min}$ and $\alpha_{\max}$. However the results we prove do not provide any information about the difference $\Delta = \alpha_{\max} - \alpha_{\min}$ and thus about how much the step sizes employed by different nodes and at different iterations can differ. In this section we try to address this issue by investigating in practice the length of the interval of admissible step-sizes.
First we show a particular example where the upper bound $\alpha_{\max}$ is

not necessary to ensure convergence of the method, then we present a set of numerical results that investigate how the step bounds $\alpha_{\min}$ and $\alpha_{\max}$ and their difference Δ influence the behaviour of the method.

We consider the same framework considered in [25] (Section 4.2) extended to the case we are considering of time-varying network. Consider the following objective function

$$f(y) = \sum_{i=1}^{N} f_i(y) \text{ with } f_i(y) = \frac{1}{2}(y - a_i)^2 \tag{3.13}$$

where $y \in \mathbb{R}$ and $a_i \in \mathbb{R}$ for every $i = 1, \ldots, N$, and assume that at iteration k the consensus matrix is of the following form, for a given $\theta_k \in (0, 1)$

$$W_k = (1 - \theta_k)I + \theta_k J.$$

Lemma 3.4. *Assume that $\theta_k \in (\frac{1}{3}, \frac{3}{4})$ for every k, and that $\{\mathbf{x}^k\}_{k=0}^{\infty}$ is the sequence generated by (3.3) with $b = 0$, $\mathbf{e}^\top(\mathbf{u}^0 + \nabla F(\mathbf{x}^0)) = 0$ and $\mathbf{e}^\top \mathbf{x}^0 = \mathbf{e}^\top \mathbf{a}$ where $\mathbf{a} = (a_1, \ldots, a_N) \in \mathbb{R}^N$.*
If $\alpha_i^k = \alpha$ for every $i = 1, \ldots, n$ and for every k, then the method converges R-linearly to the solution of (3.13) if $\alpha_{\min} \leq \alpha \leq \frac{2}{3}$ and $\alpha_{\min} > 0$ small enough. On the other hand, for any $\alpha > 2$, there exists a sequence $\{\theta_k\}$, $k = 0, 1, 2, \ldots$ that satisfies the assumptions of the Lemma such that the method diverges, i.e., $\|\mathbf{x}^k\| \to \infty$.
If $\alpha_i^k = (\sigma_i^k)^{-1}$ with

$$\sigma_i^{k+1} = \mathcal{P}_{[\sigma_{\min}, \sigma_{\max}]} \left(1 + \sigma_i^k \sum_{j=1}^{n} w_{ij}^k \left(1 - \frac{s_j^k}{s_i^k} \right) \right), \tag{3.14}$$

with $s_i^k = x_i^{k+1} - x_i^k$, $\sigma_{\min} = 0$, $\sigma_{\max} = 3/2$ and $\sigma_i^0 = \sigma \in (\sigma_{\min}, \sigma_{\max})$ for every $1, \ldots, n$, then $\{\mathbf{x}^k\}$ converges R-linearly to the solution of (3.13).

Proof. In the case we are considering (3.3) is equivalent to

$$\begin{cases} \mathbf{x}^{k+1} = W^k \mathbf{x}^k - A^k \mathbf{z}^k \\ \mathbf{z}^{k+1} = W^k \mathbf{z}^k + \mathbf{x}^{k+1} - \mathbf{x}^k \end{cases}$$

where $A^k = \text{diag}(\alpha_1^k I, \dots, \alpha_n^k I)$.
Let us consider the case with fixed step-size $\alpha_i^k = \alpha$ and let us denote with $\boldsymbol{\xi}^k$ the vector $(\mathbf{q}^k, \mathbf{z}^k) \in \mathbb{R}^{2N}$. We can see that for every k we have $\boldsymbol{\xi}^{k+1} = P^k \boldsymbol{\xi}^k$ where the matrix P^k is given by

$$P^k = \begin{pmatrix} W_k - J & -\alpha I \\ W_k - I & W_k - \alpha I \end{pmatrix} \in \mathbb{R}^{2N \times 2N}.$$

In order to prove the first part of the Lemma, it is enough to show that there exists $\mu < 1$ such that $\|P^k\|_2^2 < \mu$ for every iteration index k. That is, we have to prove that there exists $\mu < 1$ such that the spectral radius of $(P^k)^\top P^k$ is smaller than μ for every k. Denoting with $1, \delta_2^k, \dots, \delta_n^k$ the eigenvalues of W^k, it can be proved that the eigenvalues of $(P^k)^\top P^k$ are given by the eigenvalues of the 2×2 matrices M_i^k defined as

$$M_1^k = \begin{pmatrix} \alpha^2 & \alpha(\alpha - 1) \\ \alpha(\alpha - 1) & (\alpha - 1)^2 \end{pmatrix}$$

$$M_i^k = \begin{pmatrix} (\delta_i^k)^2 + \alpha^2 & (\delta_i^k)^2 - (1 + \alpha)\delta_i^k + \alpha^2 \\ (\delta_i^k)^2 - (1 + \alpha)\delta_i^k + \alpha^2 & 2(\delta_i^k)^2 - 2(1 + \alpha)\delta_i^k + 1 + \alpha^2 \end{pmatrix}$$

for $i = 2, \dots, n$. By direct computation we can see that the eigenvalues of M_1^k are given by 0 and $2\alpha^2 - 2\alpha + 1 < 1 - \frac{2}{3}\alpha_{min}$ and therefore it is enough to take $\mu > 1 - \frac{2}{3}\alpha_{min}$. Denoting with $p_i^k(t)$ the characteristic polynomial of D_i^k we can see that with the values of $\theta_{min}, \theta_{max}$ and α_{max} given by the assumptions, we can always find $1 - \frac{2}{3}\alpha_{min} < \mu < 1$ such that $p_i^k(\mu) > 0$ and $p_i^k(-\mu) > 0$ and thus such that the eigenvalues of M_i^k belong to $(-\mu$ and $\mu)$ for every k and for every $i = 1, \dots, n$. To prove that if $\alpha > 2$ the method is in general not convergent it is

enough to consider the case when $\theta_k = \theta_0$ for every iteration index k. In this case we have that $P^k = P^0$ for every k and thus $\xi^k = (P^0)^k \xi^0$. In this case we can see [25] that $1 - \alpha$ is an eigenvalue of P^0 an therefore if $\alpha > 2$ we have that $\rho(P^0) > 1$ and thus the sequence $\{\xi^k\}$ does not converge. This concludes the first part of the proof.

Assume now that the step-sizes are computed as in (3.14). Proceeding as in the proof of Proposition 4.3 in [25] we can prove that $\sigma_i^{k+1} = \sigma^{k+1}$ for every i with σ_{k+1} given by

$$\sigma_{k+1} = \begin{cases} \min\{\sigma_{\max}, 1 + \sigma_{\max}\theta_k\} & \text{if } \sigma_k = \sigma_{\max} \\ \min\{\sigma_{\max}, \hat{\sigma}^{k+1}\} & \text{otherwise} \end{cases}$$

where $\hat{\sigma}^{k+1} = 1 + \theta_k + \theta_k\theta_{k-1} + \cdots + \prod_{j=1}^{k} \theta^j + \sigma^0 \prod_{j=0}^{k} \theta^j$. By using the fact that $\theta_k > 1/3$ and $\sigma_{\max} = 3/2$ we can prove that there exists $\bar{k}$ such that $\sigma^k = \sigma_{\max}$ for every $k > \bar{k}$. Therefore, for $k > \bar{k}$ the step-size becomes the same for all nodes and equal to $\alpha_i^k = \sigma_{\max}^{-1} = 2/3$ and thus the method converges by the first part of the Lemma. □

The above Lemma certifies convergence of the spectral-like method [25] for time-varying networks and a very specific problem structure with all-to-all communication network and consensus quadratic costs. It is worth noting that, for generic quadratic cost functions and sparse time-varying networks, an upper bound on the step-size is necessary (see Figures 1 and 2 below). We now make an analogy on the achieved results for the distributed spectral-like method [25] and the spectral (Barzilei-Borwein) gradient method from the centralized optimization. Namely, in centralized settings, the spectral gradient method's convergence without step size safeguarding has been proved only for a *strictly convex quadratic cost function*. In the case of generic functions beyond strictly convex quadratic, some safeguards Δ_{min} and Δ_{max} are necessary, even in the centralized case. Though, in the centralized case,

these safeguards can be arbitrarily small (Δ_{min}) and arbitrarily large (Δ_{max}). Therefore, the need for safeguards is to be expected in the distributed optimization scenario as well. This matches with the results that we present here. It turns our that the price to be payed in the distributed time-varying networks scenario is two-fold: 1) the no-safeguards case happens in a more restricted cost functions setting, namely the consensus quadratic costs (see Lemma 4); and 2) the safeguard step size bounds in the general case are no longer arbitrary and take a network-dependent form.

We also have the following Lemma where we continue to assume the consensus problem but relax the requirement that the network is fully connected at all times. When the network is not fully connected, in general we need safeguarding for global convergence. However, as explained below, the following Lemma sheds some light on the behavior of the spectral-like distributed method. While it is not to be considered as a global convergence result, it highlights that the next step size has a controlled length provided that the current solution estimate is close to consensus.

Lemma 3.5. *Let us assume that the objective function is given by* (3.13)*, and that* x^0, z^0 *are such that* $e^T x^0 = e^T a$*,* $z_i^0 = \nabla f(x_i^0) = x_i^0$*. Moreover, for every* $i = 1, \ldots, n$ *let the local stepsize* α_i^k *be defined as* $\alpha_i^0 = \alpha^0 > 0$ *and, for every* $k \geq 1$*,* $\alpha_i^k = 1/\sigma_i^k$*, with*

$$\sigma_i^{k+1} = 1 + \sigma_i^k \sum_{j=1}^{n} w_{ij}^k \left(1 - \frac{s_j^k}{s_i^k}\right) \tag{3.15}$$

where $s_j^k = x_j^{k+1} - x_j^k$*. Moreover, let us assume that at each iteration assumption A2 holds with* $m = 1$*.*
Given any $\hat{\alpha} > 1$*, if* $\|\mathbf{x}^0 - \mathbf{e}\bar{x}^0\| \leq \hat{\varepsilon}$ *with* $\hat{\varepsilon}$ *satisfying*

$$\hat{\varepsilon} \leq \frac{1}{\nu_0 + d^0} \frac{(\alpha^0)^2(\hat{\alpha} - 1)|\bar{x}^0|}{2\hat{\alpha} + \alpha^0(\hat{\alpha} - 1)}$$

then $\alpha_i^1 \leq \hat{\alpha}$ *for every* $i = 1, \ldots, N$.

Proof. Let us denote with $J \in \mathbb{R}^{n \times n}$ the matrix $\frac{1}{n}\mathbf{e}\mathbf{e}^t$ and with $\mathbf{v}^k = \mathbf{s}^k - e\bar{s}^k$. From the assumptions and the double stochasticity of the matrix W^k, we have

$$\begin{aligned}
&\|\mathbf{v}^0\| = \|(I-J)(\mathbf{x}^1 - \mathbf{x}^0)\| = \|(I-W)(W^0\mathbf{x}^0 - \mathbf{x}^0 - \alpha^0\mathbf{z}^0)\| \\
&= \|(I-J)(W^0 - I - \alpha^0 I)\mathbf{x}^0\| \leq \|(W^0 - I)\mathbf{x}^0\| + \alpha^0\|(I-J)\mathbf{x}^0\| \\
&\leq (\nu_0 + \alpha^0)\|\mathbf{x}^0 - \mathbf{e}\bar{x}^0\| \leq (\nu_0 + \alpha^0)\hat{\varepsilon} = \varepsilon
\end{aligned}$$

Where we defined $\varepsilon = (\nu_0 + \alpha^0)\hat{\varepsilon}$. Moreover,

$$|\bar{s}^0| = \frac{1}{n}|e^t(W^0\mathbf{x}^0 - \alpha^0\mathbf{x}^0 - \mathbf{x}^0)| = \alpha^0|\bar{x}^0|.$$

These imply that, for every $j = 1, \ldots, n$

$$1 - \frac{2\varepsilon}{\alpha^0|\bar{x}^0| - \varepsilon} \leq \frac{s_j^0}{s_i^0} \leq 1 + \frac{2\varepsilon}{\alpha^0|\bar{x}^0| - \varepsilon}.$$

Replacing these bounds in (3.15), and defining $\sigma^0 = 1/\alpha^0$, we get

$$1 - \frac{2\varepsilon\sigma^0}{\alpha^0|\bar{x}^0| - \varepsilon} \leq \sigma_i^1 \leq 1 + \frac{2\varepsilon\sigma^0}{\alpha^0|\bar{x}^0| - \varepsilon}. \tag{3.16}$$

It's easy to see that the first inequality, together with the assumption over $\hat{\varepsilon}$, imply $\sigma_i^1 \geq 1/\hat{\alpha}$, which in turn implies the thesis.

□

Intuitively, the Lemma above says that, for the considered problem, if algorithm (3.3) with stepsize (3.15) starts from a point close to consensus (i.e., a point where solution estimates across different nodes are mutually close), then the next step size at each node will not be too large. More precisely, the size of the next step size is controlled

by the consensus neighborhood $\hat{\epsilon}$ that we start from. In other words, if the next step size is to be upper bounded by an arbitrary constant $\hat{\alpha} > 1$, we can find a problem-dependent constant $\hat{\epsilon}$ such that, starting at most $\hat{\epsilon}$ away from consensus, the next step size at each node is at most $\hat{\alpha}$. To further explain this, suppose that all the quantities s_j^k/s_i^k's are ϵ'-close to one, $|s_j^k/s_i^k| \in (1-\epsilon', 1+\epsilon')$, for all nodes i, j. Then, in view of (3.16), quantity σ_i^{k+1}, for all nodes i, is approximated as:

$$1 \pm \sigma_i^k \, n \, \epsilon'.$$

In other words, for the special case of the consensus problem, provided that all the quantities s_j^k/s_i^k's are ϵ'-close to one, the next step-size $1/\sigma_i^{k+1}$ is in a neighborhood of one, and is hence bounded.

Let us now present some numerical results. We consider the problem of minimizing a logistic loss function with l_2 regularization, that is, we assume the local objective function f_i at node i is given by

$$f_i(\mathbf{y}) = \ln\left(1 + \exp(-b_i \mathbf{a}_i^\top \mathbf{y})\right) + \frac{1}{2} R \|\mathbf{y}\|_2^2 \tag{3.17}$$

where $\mathbf{a}_i \in \mathbb{R}^n$, $b_i \in \{-1, 1\}$ and $R > 0$. Nine methods are considered, obtained combining three possible choices of the matrix B and three different ways of computing the step sizes α_i^k. We consider an increasing sequence of values of the upper bound $\alpha_{\max}$, and for each method and each value of the step bound the number of iterations necessary to arrive at convergence is plotted in Figure 3.1.

The dimension n is set to 10 and we generate $\mathbf{a}_i \in \mathbb{R}^n$, $b_i \in \{-1, 1\}$ in (3.17) as follows. For $i = 1, \ldots, n$ we take $a_i = (a_{i1}, \ldots, a_{i,n-1}, 1)^\top$ where the components a_{ij} are independent and come from the standard normal distribution, and $b_i = sign(\mathbf{a}_i^\top \mathbf{y}^* + \epsilon_i)$ where $\mathbf{y}^* \in \mathbb{R}^n$ with independent components drawn from the standard normal distribution, and ϵ_i are generated according to the normal distribution

with mean 0 and standard deviation 0.4. Finally, we take the regularization parameter $R = 0.25$. The initial vectors $\mathbf{x}_i^0$ are generated independently, with components drawn from the uniform distribution on $[0, 1]$, and at each iteration we define the consensus matrix W^k as the Metropolis matrix [65]. The convergence analysis we carried out in Section 3 does not rely on any particular definition of the step-sizes α_i^k, therefore we need to specify how each node chooses the step-size at each iteration. Two cases are considered. The first one, referred to as *spectral* in Figure 3.1, is the case where $\alpha_i^k = (\sigma_i^k)^{-1}$ with σ_i^k as in (3.14). The second case we consider is the one where each node performs local line search by employing a backtracking strategy starting at $\alpha_{\max}$ to satisfy classical Armijo condition on the local objective function. That is, to satisfy

$$f_i\left(\sum_{j=1}^{N} w_{ij}^k \mathbf{x}_j^k - \alpha_i^k \mathbf{z}_i^k\right) \leq f_i(\mathbf{x}_i^k) - c\alpha_i^k \nabla f_i(\mathbf{x}_i^k)^\top \mathbf{z}_i^k$$

with $c = 10^{-3}$ and $\mathbf{z}_i^k = \mathbf{u}_i^k + \nabla f_i(\mathbf{x}_i^k)$. We refer to this method as *line search*. It is worth noting that there are no convergence guarantees for the line search method. The rationale for including a comparison with it is to show that the method [25] exhibits a significantly higher degree of robustness with respect to a meaningful, time-varying and node-varying, local step size strategy that can be employed.
For comparison, we also consider the method with fixed step-size $\alpha_i^k = \alpha_{\max}$ for every k and every $i = 1, \ldots, N$. The choices of the matrix B^k are given by $B^k = 0$ (plot (a) in Figure 3.1), $B^k = \alpha_{\max}^{-1} I$ (plot (b)) and $B^k = \alpha_{\max}^{-1} \mathcal{W}^k$ (plot(c)), where for the case $B \neq 0$ the choice is made following [24]. Notice that the case $\alpha_i^k = \alpha_{\max}$ and $B^k = 0$ corresponds to [47, 35] with constant, coordinated step-sizes. We consider increasing values of $\alpha_{\max}$ in $[\frac{1}{50L}, \frac{10}{L}]$, while we fix $\alpha_{\min} = 10^{-8}$ as we saw that, in the considered framework, its choice does not influence the performance of the methods significantly.

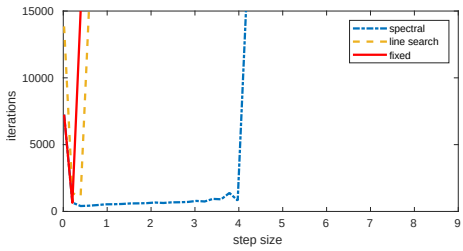
The sequence of communication networks $\{\mathcal{G}_k\}$ is defined as follows. We consider a network $\mathcal{G}$ with $N = 25$ nodes, undirected and connected, generated as a random geometric graph with communication radius $\sqrt{N^{-1}\ln(N)}$, and we define the sequence of networks $\{\mathcal{G}_k\}$ by deleting each edge with probability $\frac{1}{4}$. We carried out analogous tests in the cases where $\mathcal{G}$ is symmetric and constant and in the case where it is given by a directed ring. The obtained results were comparable to the ones that we present. We also observed in practice that double stochasticity of the underlying network appears to be essential for the convergence of the considered methods.

We are interested in the number of iterations required by each method to reach a prescribed accuracy. More precisely, we terminate the execution of the method when

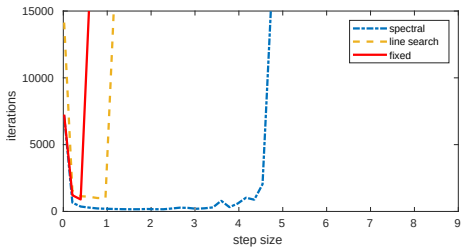
$$\max_{i=1,\dots,n} \|x_i^{\bar{k}} - y^*\| < \varepsilon,$$

where $\varepsilon = 10^{-5}$. That is, we stop when the local error at each node is smaller than ε. In Figure 3.1, on the x-axis we show the upper bound $\alpha_{\max}$ while on the y-axis we show $\bar{k}$ for each method. To facilitate the comparison among the methods, in Figure 3.2 we plot the same results, with y-axis cut at 2000.

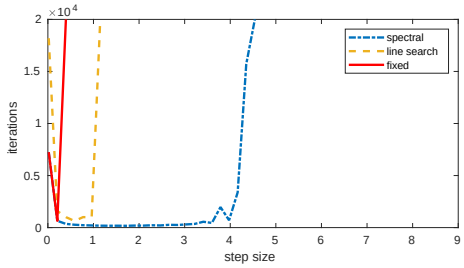
We can see in Figure 3.1 that, for all the choices of the matrix B_k that we consider, the method that employs time-varying asynchronous step sizes given by (3.14) converges for maximum step bound $\alpha_{\max}$ that is at least 10 times larger than the method that uses the same step size at every node and each iteration, while the method that selects the step size with local line search converges for $\alpha_{\max}$ equal to 2 to 3 times larger than the method with fixed step size. Moreover, we can see that choosing $B = bI$ seems to increase the maximum value of $\alpha_{\max}$ that yields convergence for all the considered methods. Finally, in



(a) $B = 0$



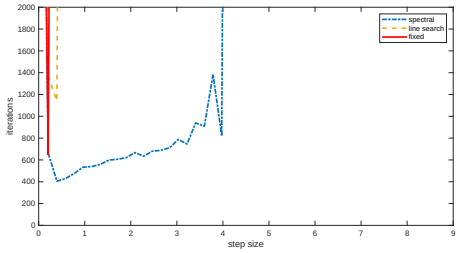
(b) $B = bI$



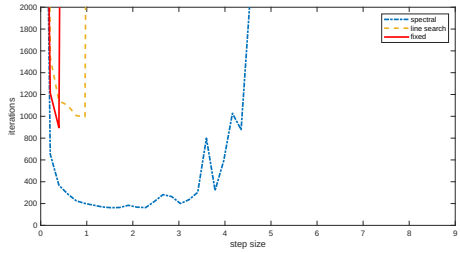
(c) $B = bW$

Figure 3.1: Number of iteration for increasing upper bound on the step size

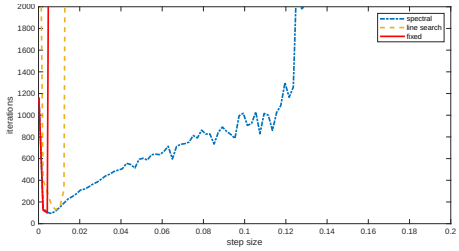
Figure 3.2, we can notice that for most of the tested values of $\alpha_{\max}$ we can see that the spectral methods requires a smaller number of iterations than the method with fixed step-size, therefore in the considered framework, using uncoordinated time-varying step-sizes given by [25] helps to significantly improve the robustness of the method and also the performance. Notice also that the spectral step-size strategy exhibits a "stable", practically unchanged, performance for a wide range of $\alpha_{\max}$; hence, it is not sensitive to tuning of $\alpha_{\max}$. This is in contrast with the constant step-size strategy that is very sensitive to the step-size choice $\alpha_{\max}$. This is important also from the practical point of view: the bounds to $\alpha_{\max}$ and Δ given by Theorem 3.1, as well



(a) $B = 0$



(b) $B = bI$



(c) $B = bW$

Figure 3.2: Number of iteration for increasing upper bound on the step size

as the value of the step size that ensures convergence of the method with fixed step size, are usually unknown in practice. The numerical results show that, if the step size is computed as in (3.14), then the method is robust to the choice of $\alpha_{\max}$ and that a bound of Δ is not necessary for convergence. That is, $\alpha_{\min}$ can be set to a small value independent of system parameters, e.g., $\alpha_{\min} = 10^{-8}$, and setting $\alpha_{\max}$ requires only a coarse upper bound on quantity $1/L$.

Chapter 4

Distributed Inexact Newton Method with Adaptive Step Size

4.1 Introduction

In this chapter we propose a distributed Inexact Newton method with adaptive step size, for the finite sum minimization problem

$$\min_{\mathbf{y}\in\mathbb{R}^n} f(\mathbf{y}), \quad f(\mathbf{y}) = \sum_{i=1}^{N} f_i(\mathbf{y}) \tag{4.1}$$

that we already discussed in Section 2.1 and Chapter 3.

When considering second order methods in the distributed and large-scale framework, there a few issues that need to be addressed. First of all, as in the classical framework, Hessian-based methods are typically quite expensive: while in the distributed framework the work of computing the Hessian is shared by the nodes and therefore does

not usually represent a problem, solving the linear system that is typically required at each iteration to compute the direction may be prohibitively expensive for problems of very large sizes. Secondly, while we normally assume that nodes can share the local vector of variables and the local gradients (or some other vector), sharing the Hessian would cause the communication traffic to became too large. In particular this implies that, while we can usually define the algorithms in such a way that each node works with an approximation of the global gradient that becomes more and more accurate along the iterations, the same thing cannot be done with the Hessian matrix, as sharing the Hessian with the neighbours would be impractical. Finally, the order of convergence. Ideally, one would like for a second order method to show superlinear or quadratic local convergence. However, when considering the distributed framework, there are two main obstacles: exact consensus and choice of the step size. Both the weighted averaging strategy [11] used in [74] and the primal-dual scheme used in [44, 27] only yield linear convergence of the local vectors to the global consensus vector and therefore methods involving this strategies cannot achieve overall superlinear convergence. On the other hand, it is know from the theory of second order methods in the classical centralized framework that the choice of the step size is of fundamental importance in order for a method to achieve both global convergence and fast local convergence. However, classical line search strategies, which effectively solve this issue in the centralized framework, are not applicable in the distributed setting as they require knowledge of the value of the global function at all nodes and may require several evaluations at each iteration, which would be prohibitively expensive from the point of view of communication traffic. The distributed method proposed in [75] achieves quadratic converges by employing a finite-time communication strategy to share the local Hessians (or their rank one approximation) through the whole network, and the step size proposed in [52] for Newton's method in the centralized case, which ensures both

global convergence and fast local convergence without requiring multiple function evaluations at each iteration.

We present here a Distributed Inexact Newton method that employs an adaptive step size that can be computed at each iteration without knowing the global constants of the objective function, [29]. The method computes the search direction using a suitable iterative solver in order to distributedly compute an approximate solution of the Newton linear system at each iteration. After the direction is computed, the step size is defined based on the value of the gradient and the forcing term in the solution of the linear system. The considered step size is an adaptation of the one presented in [52], to the distributed framework, and to the case where the Newton system is solved inexactly. Under suitable assumptions over the objective function f and the underlying communication network, we prove that the proposed method achieves global convergence and local linear, superlinear or quadratic convergence depending on the choice of the forcing term, analogously to the centralized Inexact Newton method [12].

This chapter is organized as follows. In Section 4.2 we describe the framework we consider, state the basic assumption on the considered problem, and describe the penalty reformulation. DINAS algorithm is stated and discussed in Section 4.3 while the theoretical analysis is presented in Section 4.4. Section 4.5 contains a set of numerical results that investigate the performance of the DINAS and compare it with relevant methods from the literature.

4.2 Preliminaries

The computational framework is analogous to that considered in Chapter 3: we assume that a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of computational agents

is given, such that agent i holds the function f_i and can communicate directly with all its neighbors in $\mathcal{G}$. Moreover, we assume that each agent holds a local vector of variables $\mathbf{x}_i \in \mathbb{R}^n$. The properties of the communication network are stated in the assumption below.

Assumption D1. *The network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is undirected, simple and connected, and it has self-loops at every node, i.e., $(i, i) \in \mathcal{E}$ for every $i = 1, \ldots, N$.*

We associate the consensus matrix W to the graph $\mathcal{G}$ such that the following assumption hold.

Assumption D2. *The matrix $W \in \mathbb{R}^{N \times N}$ is such that*

- *if $(i, j) \notin \mathcal{E}$ then $w_{ij} = 0$*
- *W is symmetric and doubly stochastic*

Moreover, we define $\mathcal{W} = W \otimes I_n \in \mathbb{R}^{nN \times nN}$.

Regarding f, we make the following assumption, which is standard for Newton-type methods.

Assumption D3. *For every $i = 1, \ldots, N$ $f_i : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a twice continuously differentiable function and there exist $\mu_i, L_i > 0$ such that f_i is μ_i-strongly convex and $\nabla^2 f$ is L_i-Lipschitz continuous for $\|\cdot\|_\infty$. That is, for every $\mathbf{y}, \mathbf{z} \in \mathbb{R}^n$*

- $\mu_i I \prec \nabla^2 f_i(\mathbf{y})$,
- $\|\nabla^2 f_i(\mathbf{y}) - \nabla^2 f_i(\mathbf{z})\|_\infty \leq L_i \|\mathbf{y} - \mathbf{z}\|_\infty$

We define $L = \sum_{i=1}^N L_i$ and, analogously, $\mu = \sum_{i=1}^N \mu_i$.
We notice that Assumption D3 implies in particular that for every $i = 1, \ldots, N$

$$\|\nabla^2 f_i(\mathbf{y})^{-1}\|_\infty \leq \|\nabla^2 f_i(\mathbf{y})^{-1}\|_2 \leq \frac{1}{\mu_i}.$$

As described in Section 2.1, problem (4.1) can be equivalently written as

$$\min_{\mathbf{x}\in\mathbb{R}^{nN}} F(\mathbf{x}),\ \text{s.t. } (I-\mathcal{W})^{1/2}\mathbf{x}=0 \tag{4.2}$$

with

$$F(\mathbf{x})=\sum_{i=1}^{N} f_i(\mathbf{x}_i).$$

In the following, we consider the quadratic penalty reformulation of (4.2). That is,

$$\min_{\mathbf{x}\in\mathbb{R}^{nN}} \Phi_\beta(\mathbf{x}) \text{ with } \Phi_\beta(\mathbf{x})=F(\mathbf{x})+\frac{1}{2\beta}\mathbf{x}^\top(I-\mathcal{W})\mathbf{x}. \tag{4.3}$$

The relationship between the penalty problem above and the original problem (4.1) is analysed in [72]. In particular, it is know that the penalty problem yields an approximate solution of (4.1) such that $\|\mathbf{x}_i^*-\mathbf{y}^*\|=\mathcal{O}(\beta/(1-\lambda_2))$ where $\mathbf{x}^*=(\mathbf{x}_1^*,\ldots,\mathbf{x}_N^*)$ is the solution of (4.3), $\mathbf{y}$ is the solution of (4.1) and λ_2 is the second largest eigenvalue of W. The method we present in the following section solves the penalty problem (4.3) for a given value of $\beta>0$. Later in the same section, we discuss how the method can be used to solve a sequence of penalty problems with decreasing values of the parameter β to find the solution of (4.1).

4.3 Algorithm DINAS

The classical Inexact Newton iteration for (4.3), given the current point $\mathbf{x}^k$ and the forcing parameter η_k is defined as

$$\mathbf{x}^{k+1}=\mathbf{x}^k-\alpha_k\mathbf{d}^k,$$

where $\mathbf{d}^k$ is such that

$$H^k\mathbf{d}^k=\mathbf{g}^k+\mathbf{r}^k \tag{4.4}$$

where $H^k = \nabla^2\Phi_\beta(\mathbf{x}^k)$, $\mathbf{g}^k = \nabla\Phi_\beta(\mathbf{x}^k)$ and the residual vector $\mathbf{r}^k \in \mathbb{R}^{nN}$ satisfies

$$\|\mathbf{r}^k\|_\infty \leq \eta_k \|\mathbf{g}^k\|_\infty. \tag{4.5}$$

In general, it is assumed that $\eta_k \leq \eta < 1$ for every k, while the step size α_k is determined by line search or some other globalization strategy [16]. If $\eta_k = 0$ the method is in fact Newton's method. Notice that we are using the norm $\|\cdot\|_\infty$ in the above expressions, as it is suitable for the distributed framework we are interested in.

To apply the above Inexact Newton iteration in the distributed framework we need to solve two problems: to distributedly compute the direction $\mathbf{d}^k$ such that (4.5) holds and to choose the step size α_k. A number of methods is available for solving linear systems in the distributed case [28, 36, 37], however it is easy to see that, because of the particular structure of the penalty function $\Phi_\beta(\mathbf{x})$, the system that we are considering (4.4) follows the sparsity pattern of the communication matrix and therefore it can be solved with a suitable fixed point method without any changes [26, 27]. In the following, we assume that the system is solved with Jacobi Overrelaxation (JOR) method but the theoretical analysis is valid for any solver, provided that they can be applied distributedly to (4.4) and achieve (4.5) for any given forcing term $\eta_k > 0$.

For the sake of clarity let us briefly state the JOR method for solving (4.4), while further details can be seen in [26, 27]. First of all notice that the coefficient matrix of the linear system

$$H^k = \nabla^2\Phi_\beta(\mathbf{x}^k) = \nabla^2 F(\mathbf{x}^k) + \frac{1}{\beta}(I - \mathcal{W})$$

is symmetric and positive definite due to strong convexity of F, the fact that $(I - \mathcal{W})$ is positive semidefinite and $\beta > 0$. Moreover, $\mathbf{g}^k$

and H^k have the following block structure:

$$\mathbf{g}^k = \begin{pmatrix} \mathbf{g}_1^k \\ \vdots \\ \mathbf{g}_N^k \end{pmatrix} \quad H^k = \begin{pmatrix} H_{11}^k & \dots & H_{1N}^k \\ \vdots & & \vdots \\ H_{N1}^k & \dots & H_{NN}^k \end{pmatrix}$$

with blocks given by, for $i, j = 1, \dots, N$

$$\mathbf{g}_i^k = \nabla f_i(\mathbf{x}_i^k) + \frac{1}{\beta}\left(\mathbf{x}_i^k - \sum_{j=1}^{N} w_{ij}\mathbf{x}_j^k\right)$$

$$H_{ij}^k = \begin{cases} \nabla^2 f_i(\mathbf{x}_i^k) + \frac{1}{\beta}(1 - w_{ii})I_n & \text{if } j = i \\ -\frac{1}{\beta}w_{ij}I_n & \text{if } j \neq i \end{cases}$$

In the following, we use H_i^k to denote the i-th block row of matrix H^k. That is $H_i^k = (H_{i1}^k, \dots, H_{iN}^k) \in \mathbb{R}^{n \times nN}$. We remark that by definition of the consensus matrix W, we have that $w_{ij} = 0$ if nodes i and j are not neighbors in the communication network. In particular, if $(i, j) \notin \mathcal{E}$ we have $H_{ij}^k = 0$. Given the linear system (4.4), a relaxation parameter $\omega > 0$, and an initial guess $\mathbf{d}^{k,0} \in \mathbb{R}^{nN}$, JOR method computes a sequence $\mathbf{d}^{k,m}$ that, for a suitable choice of the parameter ω converges to the solution of (4.4). Because of the particular block structure of the coefficient matrix, in the case that we are considering JOR method can be implemented distributedly, in such a way that node i computes the i-th block $\mathbf{d}_i^{km}$ of the approximate solution, and it only needs to communicate with its neighbors in $\mathcal{G}$. Iteration m of JOR at node i is given by

$$\mathbf{d}_i^{k,m+1} = \mathbf{d}_i^{k,m} + \omega(D_{ii}^k)^{-1}\left(\mathbf{g}_i^k - \sum_{j=1}^{N} H_{ij}^k \mathbf{d}_j^{k,m}\right), \tag{4.6}$$

where $D_{ii}^k = \mathrm{diag}(H_{ii}^k)$. Since each node holds $\mathbf{g}_i^k$ and H_{ij}^k for every $j = 1, \dots, N$ and $H_{ij}^k = 0$ if $(i, j) \notin \mathcal{E}$, we have that the nodes need to

share their approximate solutions $\mathbf{d}_i^{k,m}$ only among neighbours. It is known [26] that the method converges for

$$0 < \omega < 2\beta \frac{1 - \bar{w}}{L + 2\beta},$$

with $\bar{w} = \max_{1 \leq i \leq N} w_{ii}$. From now on we will assume that ω is chosen in such a way that JOR method converges, with further implementation details postponed to Section 4.5.

Having a distributed algorithm for solving the system of linear equations, let us explain the adaptive step size strategy we will employ here. The basic assumption we have is that the global constants μ and L are not available. Thus we employ the procedure that is governed by a sequence of parameters $\{\gamma_k\}_{k=0}^{\infty}$ that are used to generate the step sizes, adopting the reasoning from [52] to the case of distributed and inexact method. At each iteration we compute a trial step size based on the current value of γ_k and check if such step size induces enough decrease in the gradient. If the decrease is large enough the step size is accepted and we proceed to a new iteration. If not, the step size is rejected, the value of γ_k is reduced and a new trial step size is computed. We will prove that this procedure eventually leads to an acceptable step. Moreover, since the direction $\mathbf{d}^k$ stays the same until a suitable step size is found, the check is inexpensive. The step size computation includes the infinity norm of the gradient at the previous iteration and thus all nodes needs this value. For this reason, we use the algorithm from [75] in order to exchange of information around the network. Notice that in [75] the nodes exchange either local Hessians or their rank one approximations, while in DINAS they will exchange only scalars. For the sake of completeness we include the algorithm here.

Algorithm 4.1 (DSF)**.** *[75]*
Input: $\{S_i\}_{i=1:N}$ *messages at each node*
Node i***:***

set $I_i^0 = \{S_i\}$
for $k = 0, \dots, N-1$ ***do***
  for $j \in \mathcal{N}_i$ ***do***
    take $S \in I_0^{k-1}$ *such that* S *was not received from node* j *and not sent to node* j *at the previous iterations*
    send S *to node* j
  end for
  update I_i^k *adding the messages received by the neighbors*
end for

We can now state the main algorithm. In Algorithm 4.2 the proposed method is described in a node-wise fashion, to facilitate the understanding of the distributed flow. A condensed formulation will be used later on, for theoretical considerations.

Algorithm 4.2 (DINAS)**.**
Iteration k***,***
each node i ***holds:*** $\eta_k,\ \mathbf{x}_i^k,\ \mathbf{g}_i^k,\ H_i^k,\ \gamma_k > 0,\ \|\mathbf{g}^k\|_\infty,\ q \in (0,1)$

All nodes run DJOR iterations (4.6) to compute $\{\mathbf{d}_i^k\}_{i=1}^N$ *such that*

$$\|H_i^k \mathbf{d}_i^k - \mathbf{g}_i^k\|_\infty \leq \eta_k \|\mathbf{g}^k\|_\infty,\ \ \forall i = 1, \dots, N \tag{4.7}$$

All nodes compute the step size

$$\alpha_k = \min\left\{1, \frac{1-\eta_k}{(1+\eta_k)^2}\gamma_k \frac{1}{\|\mathbf{g}^k\|_\infty}\right\} \tag{4.8}$$

Each node i *computes* $\hat{\mathbf{x}}_i = \mathbf{x}_i^k - \alpha_k \mathbf{d}_i^k$
All nodes share $\hat{\mathbf{x}}_i$ *with the neighbors in* $\mathcal{N}_i$
Each node i *computes* $\hat{\mathbf{g}}_i = \nabla f_i(\hat{\mathbf{x}}_i) + \frac{1}{\beta}\left(\hat{\mathbf{x}}_i - \sum_j w_{ij}\hat{\mathbf{x}}_j\right)$
All nodes run DSF (Algorithm 4.1) with message at node i *given by* $S_i = \|\hat{\mathbf{g}}_i\|_\infty$

7: Each node i computes $\|\hat{\mathbf{g}}\|_\infty = \max_{j=1:N} S_j$
8: Each node i performs the following check and conditional updates
9: ***if***

$$\begin{cases} \alpha_k < 1 \\ \|\hat{\mathbf{g}}\|_\infty \leq \|\mathbf{g}^k\|_\infty - \frac{1}{2}\frac{(1-\eta_k)^2}{(1+\eta_k)^2}\gamma_k \end{cases} \tag{4.9}$$

or

$$\begin{cases} \alpha_k = 1 \\ \|\hat{\mathbf{g}}\|_\infty \leq \eta_k\|\mathbf{g}^k\|_\infty + \frac{1}{2\gamma_k}(1+\eta_k)^2\|\mathbf{g}^k\|_\infty^2 \end{cases} \tag{4.10}$$

then
10: set $\gamma_{k+1} = \gamma_k$
11: set $\mathbf{x}_i^{k+1} = \hat{\mathbf{x}}_i$, $\mathbf{g}_i^{k+1} = \hat{\mathbf{g}}_i$, $\|\mathbf{g}^{k+1}\|_\infty = \|\hat{\mathbf{g}}\|_\infty$
12:
13: define $H_i^{k+1} = \left(H_{i1}^{k+1}, \ldots, H_{iN}^{k+1}\right) \in \mathbb{R}^{n\times nN}$ with

$$H_{ij}^{k+1} = \begin{cases} \nabla^2 f_i(\mathbf{x}_i^{k+1}) + \frac{1}{\beta}(1-w_{ii})I_n & \text{if } j = i \\ -\frac{1}{\beta}w_{ij}I_n & \text{if } j \neq i \end{cases}$$

14: ***else***
15: set $\gamma_k = q\gamma_k$ and return to line 2
16: ***end if***

Notice that the local conditions (4.7), which is the stopping criterion for DJOR, implies the global condition

$$\|H^k\mathbf{d}^k - \mathbf{g}^k\|_\infty \leq \eta_k\|\mathbf{g}^k\|_\infty.$$

The application of JOR method for solving (4.4) implies that in each inner (JOR) iteration every node shares its current approximations $\mathbf{d}_i^{k,m}$ with its neighbours. Each iteration of JOR method includes only the inversion of the local diagonal matrix and is hence rather cheap. Depending on the value of η_k the number of inner iterations

can vary but the right hand side of (4.5) ensures that initially (that is, when the gradient is large) we solve (4.4) rather loosely, with relative large residual and small computational effort while the accuracy requirements increase as the gradients gets smaller and we approach the solution. Since we avoid the computation of the exact solution of the linear system, DINAS should be particularly effective if the dimension of (4.1) is relatively large. From the point of view of communication traffic, in addition to what we already noticed about JOR, computing $\|\hat{\mathbf{g}}\|_\infty$ requires all nodes to know the infinity norm of all the local gradients an it is achieved through Algorithm 4.1. While this implies that several rounds of communication have to be performed at each iteration, this procedure is relatively cheap, as only scalar values are shared at each round. The step size computation at line 2 is performed locally by each agent but the resulting step size is the same for all of them, as all the involved quantities are available to all nodes. The check and update defined at line 9 is again performed by all nodes using the same values and hence the results will be the same. Therefore, all nodes go back to line 2 or all nodes update the approximate solution $\mathbf{x}_i^k$. Therefore the if loop at line 9 is well defined. In the next section we will prove that the check cannot fail infinitely many times, and therefore the algorithm is well defined.

4.4 Convergence of DINAS algorithm

Lemma 4.1. *If Assumptions D1 - D3 are satisfied, then the following holds*

i) if $\gamma_k \leq \frac{\mu^2}{L}$ *then the condition at line 9 is satisfied*

ii) the number of times γ_k *is reduced is bounded from above by*

$$\log_{1/q}\left(\gamma_0 \frac{L}{\mu^2}\right)$$

iii) for every $k \in \mathbb{N}_0$ *we have* $\bar{\gamma} < \gamma_k \leq \gamma_0$, *with* $\bar{\gamma} = q\frac{\mu^2}{L}$

iv) there exists $\bar{m}_1 \in \mathbb{N}_0$ *such that* $\gamma_k = \gamma_{\bar{m}_1}$ *for every* $k \geq \bar{m}_1$

In particular, this implies that the algorithm is well defined, as there are only a finite number of iterations where the candidate step is rejected.

Proof. Since f is continuously differentiable we have that for any $\alpha \in \mathbb{R}$

$$\mathbf{g}(\mathbf{x}^k - \alpha\mathbf{d}^k) = \mathbf{g}(\mathbf{x}^k) - \alpha H^k\mathbf{d}^k - \int_0^1 \alpha \left(H(\mathbf{x}^k - s\alpha\mathbf{d}^k) - H^k\right)\mathbf{d}^k ds.$$

From Assumption D3, (4.4) and (4.5) we then have

$$\begin{aligned}
\|\mathbf{g}(\mathbf{x}^k - \alpha\mathbf{d}^k)\|_\infty &\leq \|\mathbf{g}^k - \alpha(\mathbf{g}^k + \mathbf{r}^k)\|_\infty \\
&\quad + \alpha\int_0^1 \|H(\mathbf{x}^k - s\alpha\mathbf{d}^k) - H^k\|_\infty \|\mathbf{d}^k\|_\infty ds \\
&\leq |1-\alpha|\|\mathbf{g}^k\|_\infty + \alpha\|\mathbf{r}^k\|_\infty + \alpha^2 L\int_0^1 s\|\mathbf{d}^k\|_\infty^2 ds \\
&\leq |1-\alpha|\|\mathbf{g}^k\|_\infty + \alpha\eta_k\|\mathbf{g}^k\|_\infty + \frac{\alpha^2 L}{2}\|(H^k)^{-1}(\mathbf{g}^k + \mathbf{r}^k)\|_\infty^2 \\
&\leq (|1-\alpha| + \eta_k\alpha)\|\mathbf{g}^k\|_\infty + \frac{1}{2}\frac{L}{\mu^2}(1+\eta_k)^2\alpha^2\|\mathbf{g}^k\|_\infty^2.
\end{aligned} \tag{4.11}$$

To prove the first statement we have to prove that if $\gamma_k \leq \mu^2/L$ then either (4.9) or (4.10) hold. From (4.11), if $\alpha_k = 1$ we have

$$\|\hat{\mathbf{g}}\|_\infty \leq \eta_k\|\mathbf{g}^k\|_\infty + \frac{1}{2}\frac{L}{\mu^2}(1+\eta_k)^2\|\mathbf{g}^k\|_\infty^2,$$

otherwise, replacing the value of $\alpha_k < 1$, we get

$$\begin{aligned}\|\hat{\mathbf{g}}\|_\infty &\leq \left(1 - \gamma_k \frac{(1-\eta_k)^2}{(1+\eta_k)^2} \frac{1}{\|\mathbf{g}^k\|_\infty}\right) \|\mathbf{g}^k\|_\infty \\ &+ \frac{1}{2}\frac{L}{\mu^2}(1+\eta_k)^2 \left(\gamma_k \frac{1-\eta_k}{(1+\eta_k)^2} \frac{1}{\|\mathbf{g}^k\|_\infty}\right)^2 \|\mathbf{g}^k\|_\infty^2 \\ &\leq \|\mathbf{g}^k\|_\infty + \frac{(1-\eta_k)^2}{(1+\eta_k)^2}\gamma_k \left(\frac{1}{2}\frac{L}{\mu^2}\gamma_k - 1\right).\end{aligned}$$

Since we are assuming $\gamma_k \leq \frac{\mu^2}{L}$ the desired inequalities follow immediately in both cases and we get i).
By definition of γ_{k+1} (lines 10 and 13 in Algorithm DINAS), the sequence $\{\gamma_k\}$ is non increasing, the value of γ_k is reduced only when neither (4.9) nor (4.10) are satisfied and in that case $\gamma_{k+1} = q\gamma_k$ for a fixed $q \in (0,1)$. This, together with i) implies ii) and iii). Since we proved that $\{\gamma_k\}$ is bounded from below, iv) follows immediately. □

Lemma 4.1 implies in particular that for k large enough γ_k becomes constant. While by iii) we know that $\gamma_{\bar{m}_1} \geq q\frac{\mu^2}{L}$, the Lemma does not state that γ_k will eventually reach $q\frac{\mu^2}{L}$.

Notice that the iteration of DINAS can be written in the following compact form. Given $\mathbf{x}^k$ and $\mathbf{d}^k$ such that (4.5) holds, we have $\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \mathbf{d}^k$ where

$$\alpha_k = \min\left\{1, \frac{1-\eta_k}{(1+\eta_k)^2}\gamma_k \frac{1}{\|\mathbf{g}^k\|_\infty}\right\}. \tag{4.12}$$

In the next theorem we prove that the sequence converges to the unique solution of the penalty problem.

Theorem 4.1. *Assume that D1 - D3 hold, $\{\eta_k\}$ be a nonincreasing sequence of forcing parameters such that $0 \leq \eta_k \leq \bar{\eta} < 1$ and $\gamma_0 > 0$. Let $\{\mathbf{x}^k\}$ be the sequence generated by DINAS for any $\mathbf{x}^0 \in \mathbb{R}^n$. Then*

i) there exists $\bar{m}_2 \in \mathbb{N}_0$ *such that* $\alpha_k = 1$ *for every* $k \geq \bar{m}_1 + \bar{m}_2$, *and*

$$\bar{m}_2 \leq \left\lceil \frac{1}{C} \left(\|\mathbf{g}^{\bar{m}_1}\|_\infty - \bar{\gamma} \frac{1-\bar{\eta}}{(1+\bar{\eta})^2} \right) + 1 \right\rceil, \tag{4.13}$$

$$C = q \frac{\mu^2}{L} \frac{(1-\bar{\eta})^2}{(1+\bar{\eta})^2};$$

ii) $\lim_{k\to\infty} \|\mathbf{g}^k\|_\infty = 0$;

iii) $\lim_{k\to\infty} \mathbf{x}^k = \mathbf{x}^*$ *where* $\mathbf{x}^*$ *is the unique solution of the penalty problem (4.3).*

Proof. Let us first assume that at iteration k we have step size

$$\alpha_k = \frac{1-\eta_k}{(1+\eta_k)^2} \gamma_k \frac{1}{\|\mathbf{g}^k\|_\infty} < 1. \tag{4.14}$$

From (4.9) we have

$$\|\mathbf{g}^{k+1}\|_\infty \leq \|\mathbf{g}^k\|_\infty - \frac{1}{2} \gamma_k \frac{(1-\eta_k)^2}{(1+\eta_k)^2}. \tag{4.15}$$

By Lemma 4.1 we have $\gamma_k \geq q\frac{\mu^2}{L}$. Moreover, since $\frac{(1-\eta)^2}{(1+\eta)^2}$ is a decreasing function of η for $\eta \in (0,1)$ and $\eta_k \leq \bar{\eta} < 1$ we have that, for every k

$$\frac{(1-\eta_k)^2}{(1+\eta_k)^2} \geq \frac{(1-\bar{\eta})^2}{(1+\bar{\eta})^2} > 0$$

which implies that

$$\gamma_k \frac{(1-\eta_k)^2}{(1+\eta_k)^2} \geq q \frac{\mu^2}{L} \frac{(1-\bar{\eta})^2}{(1+\bar{\eta})^2} = C > 0$$

Replacing this inequality in (4.15) we get

$$\|\mathbf{g}^{k+1}\|_\infty \leq \|\mathbf{g}^k\|_\infty - C \tag{4.16}$$

for every iteration index k such that $\alpha_k < 1$.

Let us now consider the case where $\alpha_k = 1$ and let us notice that, by definition of α_k, this implies

$$\|\mathbf{g}^k\|_\infty \leq \gamma_k \frac{1-\eta_k}{(1+\eta_k)^2}. \tag{4.17}$$

From this inequality and (4.10) we have

$$\begin{aligned} \|\mathbf{g}^{k+1}\|_\infty &\leq \eta_k \|\mathbf{g}^k\|_\infty + \frac{1}{2\gamma_k}(1+\eta_k)^2 \|\mathbf{g}^k\|_\infty^2 \\ &\leq \eta_k \|\mathbf{g}^k\|_\infty + \frac{1}{2}(1-\eta_k)\|\mathbf{g}^k\|_\infty = \frac{1}{2}(1+\eta_k)\|\mathbf{g}^k\|_\infty. \end{aligned} \tag{4.18}$$

Since $\eta_k \leq \bar{\eta} < 1$, this implies that for every k such that $\alpha_k = 1$ we have

$$\|\mathbf{g}^{k+1}\|_\infty \leq \rho \|\mathbf{g}^k\|_\infty \tag{4.19}$$

with $\rho = \frac{1}{2}(1+\bar{\eta})$.

Let us now assume that $k > \bar{m}_1$. If $\alpha_k = 1$, by (4.17), (4.18), $\eta_{k+1} \leq \eta_k$, and $\gamma_k = \gamma_{\bar{m}_1} = \gamma_{k+1}$ we have

$$\|\mathbf{g}^{k+1}\|_\infty \leq \rho \|\mathbf{g}^k\|_\infty \leq \rho \gamma_k \frac{(1-\eta_k)}{(1-\eta_k)^2} \leq \rho \gamma_{k+1} \frac{(1-\eta_{k+1})}{(1-\eta_{k+1})^2}$$

which implies $\alpha_{k+1} = 1$. Denote with $\bar{m}_2$ the smallest positive integer such that $\alpha_{\bar{m}_1+\bar{m}_2} = 1$. This, together with (4.15) implies

$$\begin{aligned} \gamma_{\bar{m}_1+\bar{m}_2-1} \frac{1-\eta_{\bar{m}_1+\bar{m}_2-1}}{(1+\eta_{\bar{m}_1+\bar{m}_2-1})^2} &< \|\mathbf{g}^{\bar{m}_1+\bar{m}_2-1}\|_\infty \leq \|\mathbf{g}^{\bar{m}_1+\bar{m}_2-2}\|_\infty - C \\ &\leq \|\mathbf{g}^{\bar{m}_1}\|_\infty - (\bar{m}_2 - 1)C, \end{aligned}$$

and thus

$$\begin{aligned} \bar{m}_2 &< \frac{1}{C}\left(\|\mathbf{g}^{\bar{m}_1}\|_\infty - \gamma_{\bar{m}_1+\bar{m}_2-1} \frac{1-\eta_{\bar{m}_1+\bar{m}_2-1}}{(1+\eta_{\bar{m}_1+\bar{m}_2-1})^2} \right) + 1 \\ &\leq \frac{1}{C}\left(\|\mathbf{g}^{\bar{m}_1}\|_\infty - \bar{\gamma} \frac{1-\bar{\eta}}{(1+\bar{\eta})^2} \right) + 1. \end{aligned}$$

Since we already proved that $\alpha_k = 1$ implies $\alpha_{k+1} = 1$ for every $k \geq \bar{m}_2$, this proves i).
Inequalities (4.16) and (4.19), together with i), imply part ii) of the Lemma.
We now prove that the sequence of iterates $\{\mathbf{x}^k\}$ converges to the solution of $\mathbf{x}^*$. For every $j \in \mathbb{N}$ we have, by definition (4.4), (4.5), (4.19) and the bound on η_k

$$\begin{aligned}
&\|\mathbf{d}^{\bar{m}_1+\bar{m}_2+j}\|_\infty \leq \| \left(H^{\bar{m}_1+\bar{m}_2+j}\right)^{-1} (\mathbf{g}^{\bar{m}_1+\bar{m}_2+j} + \mathbf{r}^{\bar{m}_1+\bar{m}_2+j})\|_\infty \\
&\leq \frac{1}{\mu}(1+\bar{\eta})\|\mathbf{g}^{\bar{m}_1+\bar{m}_2+j}\|_\infty \leq \frac{1}{\mu}(1+\bar{\eta})\rho\|\mathbf{g}^{\bar{m}_1+\bar{m}_2+j-1}\|_\infty \\
&\leq \frac{1}{\mu}(1+\bar{\eta})\rho^j\|\mathbf{g}^{\bar{m}_1+\bar{m}_2}\|_\infty.
\end{aligned}$$

For every $s \geq l \geq 0$ we then have

$$\begin{aligned}
&\|\mathbf{x}^{\bar{m}_1+\bar{m}_2+s} - \mathbf{x}^{\bar{m}_1+\bar{m}_2+l}\|_\infty \leq \sum_{j=l}^{s-1} \|\mathbf{d}^{\bar{m}_1+\bar{m}_2+j}\|_\infty \leq \\
&\leq \frac{1}{\mu}(1+\bar{\eta})\|\mathbf{g}^{\bar{m}_1+\bar{m}_2}\|_\infty \sum_{j=l}^{s-1} \rho^j = \frac{1}{\mu}(1+\bar{\eta})\|\mathbf{g}^{\bar{m}_1+\bar{m}_2}\|_\infty \frac{\rho^s - \rho^l}{1-\rho}.
\end{aligned}$$

This implies that $\{\mathbf{x}^k\}$ is a Cauchy sequence and therefore there exists $\bar{\mathbf{x}} = \lim_{k\to+\infty} \mathbf{x}^k$. Since we already proved that $\lim_{k\to+\infty} \|\mathbf{g}^k\|_\infty = 0$, we have that $\bar{\mathbf{x}} = \mathbf{x}^*$ and therefore iii). □

Remark 4.1. *Theorem 4.1 shows that, like line search in the classical framework, the adaptive strategy employed by Algorithm 4.2 ensures convergence to the solution for any initial guess (i.e. global convergence), and also that the full step $\alpha_k = 1$ is accepted whenever the norm of the gradient $\|\mathbf{g}^k\|_\infty$ is small enough.*

Regarding the forcing terms, Theorem 4.1 ensures convergence of the method to the solution whenever $\eta_k \leq \bar{\eta} < 1$. On the other

hand, for suitable choices of the relaxation parameter, one can ensure that the spectral radius of the iterative matrix of the JOR method is bounded away from 1. This implies that the assumptions of Theorem 4.1 still hold if at each iteration of Algorithm 4.2 the nodes perform only one iteration of DJOR method.

The following theorem shows that, as in the centralized case [12], the forcing sequence $\{\eta_k\}$ determines the rate of convergence.

Theorem 4.2. *Assume that D1 - D3 hold, $\gamma_0 > 0$ and let $\{\mathbf{x}^k\}$ be a sequence generated by DINAS algorithm. If $\{\eta_k\}$ is a nonincreasing sequence with $\eta_k \leq \bar{\eta}$ and $\bar{\eta}$ small enough the method converges linearly in norm $\|\cdot\|_\infty$. If for k large enough we have $\eta_k \leq \eta \|\mathbf{g}^k\|_\infty^\delta$ for some $\eta \geq 0$, then the convergence of $\mathbf{x}^k$ is superlinear for $\delta \in (0,1)$ and quadratic for $\delta = 1$.*

Proof. We already proved in Theorem 4.1 that $\|\mathbf{g}^k\|_\infty$ and $\mathbf{x}^k$ converge to 0 and $\mathbf{x}^*$ respectively. In the following, we always assume that $k \geq \bar{m} = \bar{m}_1 + \bar{m}_2$, and hence $\alpha_k = 1$. For $\delta = 0$, linear convergence of $\|\mathbf{g}^k\|_\infty$ follows directly from (4.19). Let us now consider the case $\delta > 0$. For k large enough, from (4.18) we have

$$\begin{aligned}\|\mathbf{g}^{k+1}\|_\infty &\leq \eta_k \|\mathbf{g}^k\|_\infty + \frac{1}{2\gamma_k}(1+\eta_k)^2 \|\mathbf{g}^k\|_\infty^2 \\ &\leq \eta \|\mathbf{g}^k\|_\infty^{1+\delta} + \frac{1}{\gamma_k}\|\mathbf{g}^k\|_\infty^2 \leq \eta \|\mathbf{g}^k\|_\infty^{1+\delta} + \frac{1}{\gamma_{\bar{m}}}\|\mathbf{g}^k\|_\infty^2.\end{aligned}$$

If $\delta = 1$ this implies

$$\lim_{k\to+\infty} \frac{\|\mathbf{g}^{k+1}\|_\infty}{\|\mathbf{g}^k\|_\infty^2} \leq \lim_{k\to+\infty} \frac{\eta\|\mathbf{g}^k\|_\infty^2 + \frac{1}{\gamma_{\bar{m}}}\|\mathbf{g}^k\|_\infty^2}{\|\mathbf{g}^k\|_\infty^2} = \eta + \frac{1}{\gamma_{\bar{m}}}$$

which ensures quadratic convergence. If $\delta \in (0,1)$, using the fact that

$\|\mathbf{g}^k\| \longrightarrow 0$ as $k \longrightarrow +\infty$, we get

$$\lim_{k\to+\infty} \frac{\|\mathbf{g}^{k+1}\|_\infty}{\|\mathbf{g}^k\|_\infty} \leq \lim_{k\to+\infty} \frac{\eta\|\mathbf{g}^k\|_\infty^{1+\delta} + \frac{1}{\gamma_{\bar{m}}}\|\mathbf{g}^k\|_\infty^2}{\|\mathbf{g}^k\|_\infty^2}$$
$$= \lim_{k\to+\infty} \eta\|\mathbf{g}^k\|_\infty^{\delta} + \frac{1}{\gamma_{\bar{m}}}\|\mathbf{g}^k\|_\infty = 0$$

and therefore superlinear convergence.
We now consider the sequence $\mathbf{x}^k$. For every $j \in \mathbb{N}$ we have

$$\begin{aligned}
&\|\mathbf{x}^{\bar{m}+j+1} - \mathbf{x}^*\|_\infty = \|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^* - \mathbf{d}^{\bar{m}+j}\|_\infty = \\
&= \|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^* - (H^{\bar{m}+j})^{-1}(\mathbf{g}^{\bar{m}+j} + \mathbf{r}^{\bar{m}+j})\|_\infty = \\
&= \|(H^{\bar{m}+j})^{-1}\|_\infty \|H^{\bar{m}+j}(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*) - (\mathbf{g}^{\bar{m}+j} + \mathbf{r}^{\bar{m}+j})\|_\infty \\
&\leq \frac{1}{\mu}\|H^{\bar{m}+j}(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*) - (\mathbf{g}^{\bar{m}+j} - \mathbf{g}^*)\|_\infty + \frac{1}{\mu}\|\mathbf{r}^{\bar{m}+j}\|_\infty.
\end{aligned} \tag{4.20}$$

Since the objective function is twice continuously differentiable, we have

$$\mathbf{g}^{\bar{m}+j} - \mathbf{g}^* = \int_0^1 H(\mathbf{x}^{\bar{m}+j} + s(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*))(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*)ds$$

and therefore

$$\begin{aligned}
&\|H^{\bar{m}+j}(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*) - (\mathbf{g}^{\bar{m}+j} - \mathbf{g}^*)\|_\infty \\
&= \left\| \int_0^1 (H^{\bar{m}+j} - H(\mathbf{x}^{\bar{m}+j} + s(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*))(\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*)ds \right\|_\infty \\
&\leq \int_0^1 sL\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*)\|_\infty^2 ds \leq \frac{L}{2}\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*)\|_\infty^2.
\end{aligned} \tag{4.21}$$

Replacing this term in (4.20) and using the bound on $\|\mathbf{r}^k\|_\infty$

$$\|\mathbf{x}^{\bar{m}+j+1} - \mathbf{x}^*\|_\infty \leq \frac{1}{\mu}\frac{L}{2}\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty^2 + \frac{1}{\mu}\eta_{\bar{m}+j}\|\mathbf{g}^{\bar{m}+j}\|_\infty. \tag{4.22}$$

Since $\mathbf{g}(\mathbf{x})$ is continuous and $\{\mathbf{x}^k\}$ is bounded, there exists $L_2 \geq 0$ such that

$$\|\mathbf{g}^k\|_\infty = \|\mathbf{g}^k - \mathbf{g}^*\|_\infty \leq L_2 \|\mathbf{x}^k - \mathbf{x}^*\|_\infty.$$

Let us consider the case $\delta > 0$ and let us notice that, since $\mathbf{x}^k$ converges to $\mathbf{x}^*$, we can assume j is large enough so that $\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty < 1$. By definition of η_k we then have

$$\begin{aligned} \|\mathbf{x}^{\bar{m}+j+1} - \mathbf{x}^*\|_\infty &\leq \frac{1}{\mu}\frac{L}{2}\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty^2 + \frac{1}{\mu}\eta\|\mathbf{g}^{\bar{m}+j}\|_\infty^{1+\delta} \\ &\leq \frac{1}{\mu}\left(\frac{L}{2} + L_2\eta\right)\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty^{1+\delta} \end{aligned} \tag{4.23}$$

which proves superlinear and quadratic convergence for $\delta \in (0, 1)$ and $\delta = 1$, respectively. For $\delta = 0$, let us assume that $\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty \leq \varepsilon$. From (4.22), using the fact that $\eta_k \leq \bar{\eta}$, we have

$$\|\mathbf{x}^{\bar{m}+j+1} - \mathbf{x}^*\|_\infty \leq \frac{1}{\mu}\left(\frac{L}{2}\varepsilon + L_2\bar{\eta}\right)\|\mathbf{x}^{\bar{m}+j} - \mathbf{x}^*\|_\infty$$

For $\varepsilon, \bar{\eta}$ small enough we have that

$$\frac{1}{\mu}\left(\frac{L}{2}\varepsilon + L_2\bar{\eta}\right) < 1,$$

which ensures linear convergence and concludes the proof. □

The adaptive step size we use in DINAS can be traced back to the adaptive step sizes proposed in [52] for the Newton method. Generalizing the reasoning presented there, to take into account both the distributed computational framework and approximate Newton search direction. Assuming that $N = 1$, i.e., going back to the classical problem of solving (4.1) on a single computational node we can state the adaptive step size method for Inexact method with the same analysis

as already presented. Going one step further and assuming that the constants μ and L are available, we get the following statement for Inexact Newton method with arbitrary linear solver.

Corollary 4.1. *Assume that D1 - D3 hold and that the iterative sequence is generated as*

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \hat{\alpha}_k \mathbf{d}^k$$

where $\mathbf{d}^k$ *satisfies (4.5), and*

$$\hat{\alpha}_k = \min\left\{1, \frac{1-\eta_k}{(1+\eta_k)^2}\frac{\mu^2}{L}\frac{1}{\|\mathbf{g}^k\|_\infty}\right\}. \tag{4.24}$$

Then $\lim_k \mathbf{x}^k = \mathbf{x}^*$ *and the rate of convergence is governed by the forcing sequence* $\{\eta_k\}$ *as in Theorem 4.2.*

Proof. The step size employed in DINAS algorithm reduces to $\hat{\alpha}_k$ whenever $\gamma_k = \frac{\mu^2}{L}$, while from part *i*) of Lemma 4.1 we have that for this choice of γ_k either condition (4.9) or (4.10) is always satisfied. That is, for the considered sequence, we have

$$\|\mathbf{g}^{k+1}\|_\infty \leq \|\mathbf{g}^k\|_\infty - \frac{1}{2}\frac{\mu^2}{L}\frac{(1-\eta_k)^2}{(1+\eta_k)^2}$$

for every k such that $\hat{\alpha}_k < 1$, and

$$\|\mathbf{g}^{k+1}\|_\infty \leq \eta_k\|\mathbf{g}^k\|_\infty + \frac{1}{2}\frac{L}{\mu^2}(1+\eta_k)^2\|\mathbf{g}^k\|_\infty^2$$

for every k such that $\hat{\alpha}_k = 1$. The thesis then follows directly from the convergence analysis of DINAS. □

Remark 4.2. *The previous corollary provides a choice of the step size that is accepted at all iterations. However, compared to* $\hat{\alpha}_k$*, the*

adaptive step size (4.8) *presents several advantages. First of all, the definition of* $\hat{\alpha}_k$ *involves the regularity constants* L *and* μ*, which are generally not known. Moreover, depending on the considered objective function,* $\hat{\alpha}_k$ *could be very small, especially if in the initial iterations the value of the gradient is large. In fact a number of methods we mentioned throughout this thesis assume the step sizes (at least theoretically) to be depending on the global constants and are rather small which potentially makes the methods slow. The numerical experience so far implies that for a reasonable value of* γ_0 *we have a rather small number of rejections in Step 9 and the step size is mostly accepted although* $\gamma_k > \frac{\mu^2}{L}$*. So, the adaptive step size approach allows us to take larger steps and proceed faster than the method that employes the exact value* $\frac{\mu^2}{L}$*, at least when we are far away from the solution. Finally, we notice that when* $\gamma_k > \frac{\mu^2}{L}$*, the right hand sides of inequalities* (4.9) *and* (4.10) *are smaller than their equivalent for* $\gamma_k = \frac{\mu^2}{L}$*. That is, when the adaptive step size is accepted, the decrease in the gradient is larger than the decrease induced by* $\hat{\alpha}_k$*.*

Theorem 4.3. *Assume that D1 - D3 hold,* $\{\eta_k\}$ *is a nonincreasing sequence of forcing parameters such that* $0 \leq \eta_k \leq \bar{\eta} < 1$ *and* $\gamma_0 > 0$*. Let* $\{\mathbf{x}^k\}$ *be the sequence generated by DINAS for any* $\mathbf{x}^0 \in \mathbb{R}^n$*. The number of iterations necessary to reach* $\|\mathbf{g}^k\|_\infty \leq \varepsilon$ *for a given* $\varepsilon > 0$ *is bounded from above by*

$$k_\varepsilon = \left\lceil \frac{\log\left(\varepsilon^{-1}\|\mathbf{g}^0\|_\infty\right)}{\log(\hat{\rho}^{-1})} + 1 \right\rceil$$

with

$$\hat{\rho} = \max\left\{ \frac{(1+\bar{\eta})}{2},\ 1 - q\frac{\mu^2}{L}\frac{(1-\bar{\eta})^2}{(1+\bar{\eta})^2}\frac{1}{\|\mathbf{g}^0\|_\infty} \right\} < 1.$$

Proof. Let us consider inequalities (4.15) and(4.19) derived in the proof of Theorem 4.1. For every index k we have that if $\alpha_k = 1$

$$\|\mathbf{g}^{k+1}\|_\infty \leq \rho\|\mathbf{g}^k\|_\infty$$

with $\rho = (1+\bar{\eta})/2$. If $\alpha_k < 1$ and $C = q\frac{\mu^2}{L}\frac{(1-\bar{\eta})^2}{(1+\bar{\eta})^2}$,

$$\|\mathbf{g}^{k+1}\|_\infty \leq \|\mathbf{g}^k\|_\infty - C \leq \|\mathbf{g}^k\|_\infty \left(1 - \frac{C}{\|\mathbf{g}^0\|_\infty}\right) =: \rho_2 \|\mathbf{g}^k\|_\infty.$$

Notice that since $\alpha_k < 1$, from the definition of α_k and the fact that $\gamma_k \geq q\mu^2/L$ and $\eta_k \leq \bar{\eta} < 1$, we have

$$\|\mathbf{g}^k\|_\infty > \gamma_k \frac{1-\bar{\eta}_k}{(1+\bar{\eta}_k)^2} \geq q\frac{\mu^2}{L}\frac{1-\bar{\eta}}{(1+\bar{\eta})^2} > C$$

and thus $\rho_2 \in (0,1)$. That is, denoting with $\hat{\rho} = \max\{\rho, \rho_2\} < 1$, we have that for every iteration index k

$$\|\mathbf{g}^{k+1}\|_\infty \leq \hat{\rho}\|\mathbf{g}^k\|_\infty.$$

Let us denote with k_ε the first iteration such that $\|\mathbf{g}^{k_\varepsilon}\|_\infty \leq \varepsilon$. From the inequality above, we have

$$\varepsilon < \|\mathbf{g}^{k_\varepsilon - 1}\|_\infty \leq \hat{\rho}\|\mathbf{g}^{k_\varepsilon - 2}\|_\infty \leq \hat{\rho}^{k_\varepsilon - 1}\|\mathbf{g}^0\|_\infty,$$

which implies

$$\hat{\rho}^{k_\varepsilon - 1} > \frac{\varepsilon}{\|\mathbf{g}^0\|_\infty}$$

and thus

$$k_\varepsilon - 1 < \log_{1/\hat{\rho}}\left(\varepsilon^{-1}\|\mathbf{g}^0\|_\infty\right) = \frac{\log\left(\varepsilon^{-1}\|\mathbf{g}^0\|_\infty\right)}{\log(\hat{\rho}^{-1})}.$$

□

The proposed method minimizes the penalty function Φ_β. That is, given a value of the penalty parameter β, Algorithm 4.2 can be applied to find the solution of (4.3) with arbitrary precision. As showed in Section 1.2.4, under suitable assumptions problem (4.2) can be solved by solving a sequence of penalty problems with objective function Φ_{β_s} for decreasing β_s. We consider the following procedure, analogous to Algorithm 1.2.

Algorithm 4.3.
Input: $\varepsilon_0, \beta_0 > 0$, $\hat{\mathbf{x}}^0 \in \mathbb{R}^n, q \in (0,1)$.
1: ***for*** $s = 1, 2, \ldots$ ***do***
2: *use DINAS starting at* $\hat{\mathbf{x}}^{s-1}$ *to find* $\hat{\mathbf{x}}^s$ *such that*

$$\|\Phi_{\beta_s}(\hat{\mathbf{x}}^s)\|_\infty \leq \varepsilon_s$$

3: *set* $\beta_{s+1} = q\beta_s$
4: *set* $\varepsilon_{s+1} = q\varepsilon_s$
5: ***end for***

Remark 4.3. *Similarly to what we commented for Algorithm 1.2, different choices could in theory be made at lines 3 and 4 for the update of the penalty parameter* β_s *and the tolerance* ε_s*. The fixed decrease proposed here is suitable for the distributed case as it does not require any additional communication among the nodes. The convergence theorem below works with more general conditions over* $\{\beta_s\}$ *and* $\{\varepsilon_s\}$*.*

The following theorem shows that every accumulation point of the sequence generated by Algorithm 4.3 is the solution of (4.2). We notice that Theorem 1.17 cannot be applied directly as the matrix $(I - \mathcal{W})$ is singular and thus the assumption of linear independence of the gradients $\{\nabla h_i(\mathbf{x}^*)\}$ does not hold.

Theorem 4.4. *Let Assumptions D1 - D3 hold and let us assume that two sequences* $\{\beta_s\}, \{\varepsilon_s\} \subset \mathbb{R}_{>0}$ *are given. For every* $s \in \mathbb{N}_0$ *let us denote with* $\hat{\mathbf{x}}^s$ *the sequence of* $\mathbb{R}^n$ *such that*

$$\|\nabla\Phi_{\beta_s}(\hat{\mathbf{x}}^s)\|_\infty \leq \varepsilon_s.$$

If $\lim_{s\to+\infty} \beta_s = \lim_{s\to+\infty} \varepsilon_s = 0$*, then every accumulation point of the sequence* $\hat{\mathbf{x}}^s$ *satisfies the sufficient optimality conditions (Theorem 1.16) for problem* (4.2)*.*

Proof. We proceed as in the proof of Theorem 3.1 in [26]. Let $\bar{\mathbf{x}}$ be an accumulation point of $\{\hat{\mathbf{x}}^s\}$ and let $\mathcal{K}_1 \subseteq \mathbb{N}_0$ be an infinite subset such that $\lim_{k\in\mathcal{K}_1} \hat{\mathbf{x}}^s = \bar{\mathbf{x}}$. By definition of $\hat{\mathbf{x}}^s$ and Φ_{β_s}, we have

$$\begin{aligned}\varepsilon_{\beta_s} \geq \|\nabla\Phi_{\beta_s}(\hat{\mathbf{x}}^s)\|_\infty &= \left\|\nabla F(\hat{\mathbf{x}}^s) + \frac{1}{\beta_{\beta_s}}(I-\mathcal{W})\hat{\mathbf{x}}^s\right\|_\infty \\ &\geq \frac{1}{\beta_{\beta_s}}\|(I-\mathcal{W})\hat{\mathbf{x}}^s\|_\infty - \|\nabla F(\hat{\mathbf{x}}^s)\|_\infty,\end{aligned}$$

which implies

$$\|(I-\mathcal{W})\hat{\mathbf{x}}^s\|_\infty \leq \beta_s(\varepsilon_s + \|\nabla F(\hat{\mathbf{x}}^s)\|_\infty). \tag{4.25}$$

Since ∇F is bounded over $\{\hat{\mathbf{x}}^s\}_{\mathcal{K}_1}$, and β_s tends to zero, taking the limit of the previous inequality for $k \to +\infty$ in $\mathcal{K}_1$, we get

$$\|(I-\mathcal{W})\bar{\mathbf{x}}\|_\infty \leq \lim_{k\in\mathcal{K}_1} \beta_s(\varepsilon_s + \|\nabla F(\hat{\mathbf{x}}^s)\|_\infty) = 0,$$

which also implies that $(I-\mathcal{W})^{1/2}\bar{\mathbf{x}} = 0$ and therefore $\bar{\mathbf{x}}$ is a feasible point for (4.2). Let us now define for every $s \in \mathbb{N}_0$ the vectors

$$\mathbf{v}_s = \frac{1}{\beta_s}(I-\mathcal{W})^{1/2}\hat{\mathbf{x}}^s,$$

$$\mathbf{z}_s = \frac{1}{\beta_s}(I-\mathcal{W})\hat{\mathbf{x}}^s.$$

We want to prove that $\{\mathbf{v}^s\}_{s\in\mathcal{K}_1}$ is bounded. Let $(I-\mathcal{W}) = U\Lambda U^\top$ be the eigendecomposition of $(I-\mathcal{W})$, with $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_{nN})$. From (4.25) we have that $\{\mathbf{z}^s\}_{s\in\mathcal{K}_1}$ is bounded. That is, there exists $Z \in \mathbb{R}$ such that $\|\mathbf{z}^s\| \leq Z$ for every $s \in \mathcal{K}_1$. Since U is an orthogonal matrix, by definition of $\mathbf{z}$ we have

$$\begin{aligned}Z \geq \|\mathbf{z}^s\| \geq \|U^\top\mathbf{z}^s\| &= \frac{1}{\beta_s}\|U^\top U\Lambda U^\top\hat{\mathbf{x}}^s\| = \frac{1}{\beta_s}\|\Lambda U^\top\hat{\mathbf{x}}^s\| \\ &= \frac{1}{\beta_s}\left(\sum_{i=1}^{nN}\lambda_i^2(U^\top\hat{\mathbf{x}}^s)_i^2\right)^{1/2}\end{aligned}$$

Since $\lambda_i \geq 0$ for every i, this implies that $\{\frac{1}{\beta_s}\lambda_i^2(U^\top \mathbf{x}^s)_i^2\}_{s\in\mathcal{K}_1}$ is bounded for every i and therefore $\{\frac{1}{\beta_s}\lambda_i(U^\top \mathbf{x}^s)_i^2\}_{s\in\mathcal{K}_1}$ is also bounded. By definition of $\mathbf{v}$ we get

$$\frac{1}{\beta_s}\left(\sum_{i=1}^{nN}\lambda_i(U^\top \mathbf{x}^s)_i\right)^{1/2} = \frac{1}{\beta_s}\|\Lambda^{1/2}U^\top \mathbf{x}\| = \|U^\top \mathbf{v}^s\|.$$

This implies that $\{\mathbf{v}^s\}_{s\in\mathcal{K}_1}$ is bounded and therefore there exists $\bar{\mathbf{v}} \in \mathbb{R}^n$ and $\mathcal{K}_2 \subseteq \mathcal{K}_1$ infinite subset such that $\lim_{k\in\mathcal{K}_2}\mathbf{v}^s = \bar{\mathbf{v}}$. By definition of $\hat{\mathbf{x}}^s$, Φ_s, and $\mathbf{v}^s$ we have

$$\begin{aligned}\varepsilon_s \geq \|\nabla\Phi_s(\hat{\mathbf{x}}^s)\|_\infty &= \left\|\nabla F(\hat{\mathbf{x}}^s) + \frac{1}{\beta_s}(I-\mathcal{W})\hat{\mathbf{x}}^s\right\|_\infty = \\ &= \left\|\nabla F(\hat{\mathbf{x}}^s) + (I-\mathcal{W})^{1/2}\mathbf{v}^s\right\|_\infty.\end{aligned}$$

Taking the limit for $s \in \mathcal{K}_2$ we get

$$\left\|\nabla F(\bar{\mathbf{x}}) + (I-\mathcal{W})^{1/2}\bar{\mathbf{v}}\right\|_\infty = 0,$$

and thus $\bar{\mathbf{x}}$ is a satisfies the KKT conditions for (4.2). Denoting with $\mathcal{L}$ the Lagrangian function of problem (4.2), by Assumption D3 we have that $\nabla^2_{\mathbf{xx}}\mathcal{L}(\bar{\mathbf{x}}, \bar{\mathbf{v}})$ is positive definite, and therefore we get the thesis. □

4.5 Numerical Results

We now present a set of numerical results to investigate the behavior of DINAS and how it compares with relevant methods from the literature, for different kinds of problems.
We begin studying how the choice of the forcing terms η_k influences the performance of the method. As in the previous chapter, we consider

the problem of minimizing a logistic loss function with l_2 regularization. That is, given $\{\mathbf{a}_j\}_{j=1}^m \subset \mathbb{R}^n$, $\{b_j\}_{j=1}^m \subset \{-1, 1\}, \rho > 0$, the objective function f is defined as

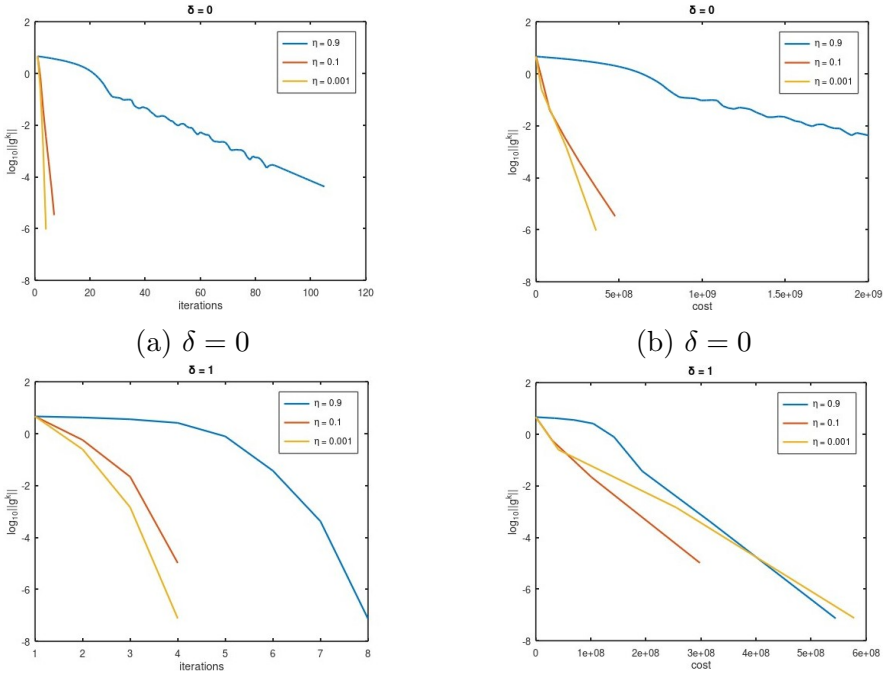
$$f(\mathbf{y}) = \sum_{j=1}^{m} \ln\left(1 + \exp(-b_j \mathbf{a}_j^\top \mathbf{y})\right) + \frac{1}{2}\rho\|\mathbf{y}\|_2^2 \tag{4.26}$$

We set $n = 100$, $m = 1000$ and we assume that node i holds $\{\mathbf{a}_j\}_{j\in\mathcal{R}_i}$, $\{b_j\}_{j\in\mathcal{R}_i}$ for $\mathcal{R}_i = \{(i-1)100+1, \ldots, 100i\}$. For every $j = 1, \ldots, m$ the components of $\mathbf{a}_j$ are independent and uniformly drawn from $(0, 1)$, while b_j takes value 1 or -1 with equal probability. We take the regularization parameter $\rho = 0.01m$. The underlying communication network is defined as a random geometric graph with communication radius $\sqrt{N^{-1}\ln(N)}$, and the consensus matrix W as the Metropolis matrix [65] (see (2.2)). To evaluate the methods, we define the per-iteration total cost of each method as the sum of the computational cost plus the communication traffic multiplied by a scaling constant r, [5]. That is,

$$\text{total cost} = \text{computation} + r \cdot \text{communication} \tag{4.27}$$

The computational cost is expressed in terms of scalar operations, while the communication traffic is the total number of scalar quantities shared by all nodes. The reason why the scaling factor r is introduced is that the time necessary to share a variable between two nodes compared with the time necessary to execute scalar computations depends on many factors of technical nature, such as the kind of computational agents that form the network and the technology they use to communicate, that are beyond the purpose of these experiments.

We consider a logistic regression function f generated as above and, given $\beta = 0.1$, we solve the associated penalty problem (4.3) with DINAS algorithm for different choices of the sequence of forcing



(a) $\delta = 0$ (b) $\delta = 0$

(c) $\delta = 1$ (d) $\delta = 1$

Figure 4.1: Choice of the forcing terms, Logistic Regression

terms $\{\eta_k\}$. All nodes start with initial guess $\mathbf{x}_i^0 = 0 \in \mathbb{R}^n$ and the execution terminates when $\|\nabla\Phi_\beta(\mathbf{x}^k)\| \leq 10^{-5}$. We assume that η_k is given by

$$\eta_k = \min\{\eta, \eta\|\mathbf{g}^k\|_\infty^\delta\}$$

for different choices of the parameters η and δ. In particular we consider here $\delta = 0, 1$ and $\eta = 0.9, 0.1, 0.001$. In Figure 4.1 we plot the results for the six methods given by the different combinations of the two parameters. In Figure 4.1a, 4.1b we consider the case $\delta = 0$ (that is, $\eta_k = \eta$ for every iteration index k), while in Figure 4.1c,4.1d we have $\delta = 1$. in each subfigure we plot the value of $\log_{10}(\|\mathbf{g}^k\|)$ versus

iterations (Figure 4.1a, 4.1c) and cost (4.1b, 4.1d), with scaling factor $r = 1$. For all the methods we define $\gamma_0 = 1$
Figures 4.1a, 4.1c confirm the results in Theorem 4.2: the sequence $\|\mathbf{g}^k\|$ is linearly decreasing for all the considered choices of δ and η, while for $\delta = 1$ the convergence is locally quadratic.
For both values of δ we see that the number of iterations required by the methods to arrive at termination depends directly on the choice of the forcing term: smaller values of η ensure the stopping criterion to be satisfied in a smaller number of iterations. However, for $\delta = 1$ we notice that, when compared in terms of overall cost, the method with the smallest value of η performs worse than the other two. For $\delta = 0$ the comparison among the methods for the cost gives the same result as that in terms of iterations. The results for different values of the scaling factor r are completely analogous and are therefore omitted here.

4.5.1 Comparison with Exact Methods

We compare DINAS with Network Newton [43], DAN and DAN-LA [75], Newton Tracking [74], DIGing [47] and EXTRA [57], which were all described in Section 2.2. The proposed method DINAS is designed to solve the penalty formulation of the problem therefore, in order to minimize (4.1), we apply Algorithm 4.3 with $\beta_0 = 0.1$, $q = 10$ and $\varepsilon_s = 0.01\beta_s$. For Network Newton we proceed analogously, replacing DINAS in line 2 with Network Newton. All other methods are exact, and therefore can be applied directly to minimize f. We take $\gamma_0 = 1$, $\delta = 0$ and $\eta = 0.9$ in Algorithm 4.2, while for all the methods the step sizes are computed as in the respective papers. For DAN-LA the constants are set as in [75]. In particular, we consider four different values of the parameter $c = M, 10M, 100M, 1000M$

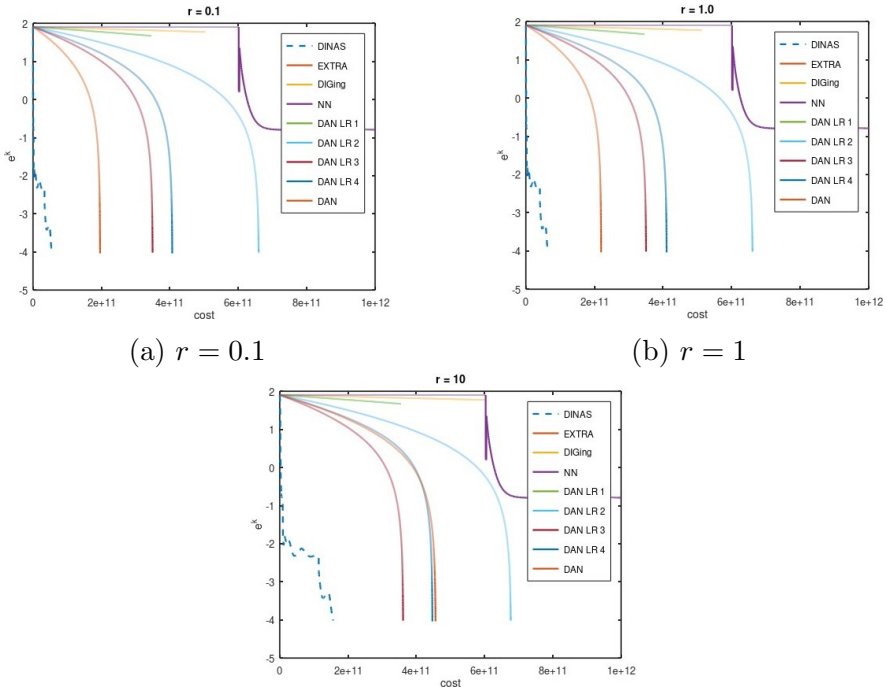
First, we consider a logistic regression problem with the same parameters as in the previous test. The exact solution $\mathbf{y}^*$ of (4.26) is

computed by running the classical gradient method with tolerance 10^{-8} on the norm of the gradient. As in [43], the methods are evaluated considering the average squared relative error, defined as

$$e_k = \frac{1}{N} \sum_{j=1}^{N} \frac{\|\mathbf{x}_i^k - \mathbf{x}^*\|^2}{\|\mathbf{x}^*\|^2}$$

where $\mathbf{x}^* = (\mathbf{y}^*, \dots, \mathbf{y}^*)^\top \in \mathbb{R}^{nN}$. For all methods we take initial guess $\mathbf{x}_i^0 = 0$ at every node, which yields $e_0 = 1$, and we terminate the execution when $e^k \leq 10^{-4}$. We consider the same combined measure of computational cost and communication defined in (4.27), with scaling factor $r = 0.1, 1, 10$ and we plot the results in Figure 4.2.

We can see that for all values of r DINAS outperforms all the other methods. Network Newton, DIGing and EXTRA all work with fixed step sizes that, in order to ensure global convergence of the methods, need to be very small. Despite the fact that each iteration of DIGing and EXTRA is very cheap compared to an iteration of DINAS, this is not enough to compensate the fact that both these methods require a large number of iterations to arrive at termination. DAN and DAN-LA all use an adaptive step size that depends on the constants of the problem μ and L and on the inverse of $\|\nabla F^k\|$ in such a way that the full step size is accepted when the solution is approached. In fact, we can clearly see from the plots that all these methods reach a quadratic phase where e_t decreases very quickly. However, the per-iteration cost of these methods is, in general, significantly higher than the cost of DINAS. DAN method requires all the local Hessians $\nabla^2 f_i(\mathbf{x}^k)$ to be shared among all the nodes at each iteration. While using Algorithm 4.1 this can be done in a finite number of rounds of communications, the overall communication traffic is large as it scales quadratically with both the dimension n of the problem and the number of nodes N. DAN-LA avoids the communication of matrices by computing and sharing the rank-1 approximations of the local Hessians. While this



(a) $r = 0.1$ (b) $r = 1$

(c) $r = 10$

Figure 4.2: Total cost, Logistic Regression

reduces significantly the communication traffic of the method, it increases the computational cost, as two eigenvalues and one eigenvector need to be computed by every node at all iterations, and the number of iterations, since the direction is computed using an approximation of the true Hessian. Overall, this leads to a larger per-iteration cost than DINAS. Since $\gamma_0 = 1$ and it only decreases when the conditions (4.9),(4.10) do not hold, we have that α_k in DINAS is relatively large compared to the fixed step sizes employed by the other methods that we considered. The per-iteration cost of DINAS is largely dominated

by the cost of JOR that we use to compute the direction $\mathbf{d}^k$. Since the method is run with η_k large, and $\mathbf{d}^{k-1}$ is used as initial guess at the next iteration, a small number of JOR iteration is needed on average to satisfy (4.5), which makes the overall computational and communication traffic of DINAS small compared to DAN and DAN-LA.

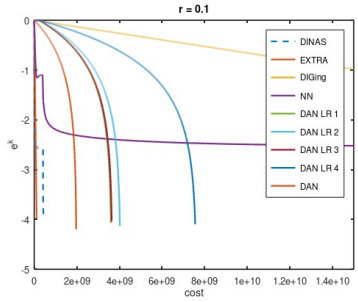
We now consider a quadratic problem. That is, we assume

$$f(\mathbf{y}) = \sum_{i=1}^{N} f_i(\mathbf{y}), \quad f_i(\mathbf{y}) = \mathbf{y}^\top A_i \mathbf{y} + \mathbf{y}^\top \mathbf{b}_i \tag{4.28}$$

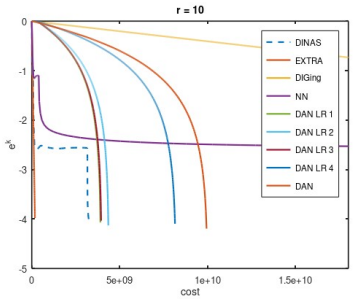
with $A_i \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$ for every $i = 1, \dots, N$. We take $n = 100$ and $N = 10$ and we generate A_i, b_i as follows. Given $0 < \lambda_{\min} < \lambda_{\max}$, we define the diagonal matrix $D_i = \mathrm{diag}(\lambda_1^i, \dots, \lambda_n^i)$ where the scalars λ_j^i are independent and uniformly sampled in $[\lambda_{\min}, \lambda_{\max}]$. Given a randomly generated orthogonal matrix $P_i \in \mathbb{R}^{n\times n}$ we define $A_i = PDP^\top$. For every $i = 1, \dots, n$ the components of $\mathbf{b}_i$ are independent and from the uniform distribution in $[0, 1]$. We set $\lambda_{\min} = 0.1$ and we consider different problems of the form (4.28) with increasing values of $\lambda_{\max}$. For each problem the exact solution $\mathbf{y}^*$, the initial guess and the termination condition are all set as in the previous test. We also use the same combined measure of the cost, with scaling factor r. All methods are run with step sizes from the respective papers, while for Network Newton we use step size equal to 1, as suggested in [43] for quadratic problems. In Figure 4.3 we plot the obtained results for $\lambda_{\max} = 0.5, 1, 10, 100$ and $r = 0.1, 10$.

For this set of problems the advantages of DINAS, compared to the other considered methods, become more evident as $\lambda_{\max}$ increases. When $\lambda_{\max}$ is larger, the Lipschitz constant of the problem also increases and therefore the step sizes that ensure convergence of DIGing and EXTRA become progressively smaller. In fact we can see that EX-

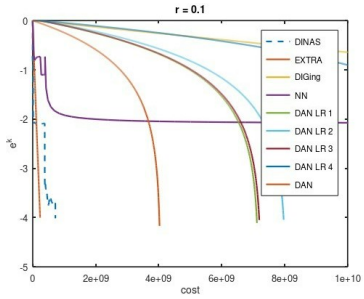
TRA outperforms the proposed method for $\lambda_{\max} \leq 1$ when the cost is computed with $r = 0.1$ and for $\lambda_{\max} \leq 10$ when $r = 10$, but DINAS becomes more efficient for larger values of $\lambda_{\max}$. Regarding DAN and DAN-LA, what we noticed for the previous test also holds here. Moreover, their step size depends on the ratio μ^2/L which, for large values of $\lambda_{\max}$ causes the step size to be small for many iterations. While Network Newton uses the full step size in this test, we can see that its performance is in general more influenced by the condition number of the problem than that of DINAS. Moreover, while the per-iteration communication traffic of Network Newton is fixed and generally lower than that of DINAS, the computational cost is typically larger, as at each iteration every node has to solve multiple linear systems of size n, exactly. Finally, we notice that for all the considered values of $\lambda_{\max}$ the comparison between DINAS and the other method is better for $r = 0.1$ which is a direct consequence of assigning different weight to the communication traffic when computing the overall cost.



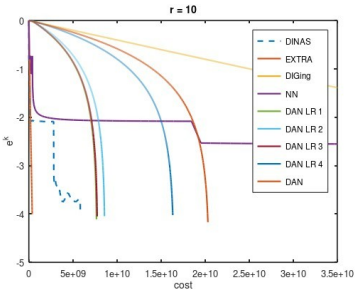
(a) $\lambda_{\max} = 0.5,\ r = 0.1$



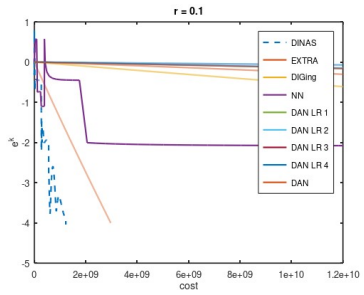
(b) $\lambda_{\max} = 0.5,\ r = 10$



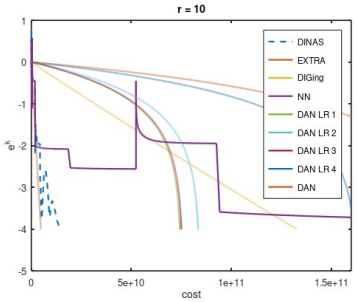
(c) $\lambda_{\max} = 1,\ r = 0.1$



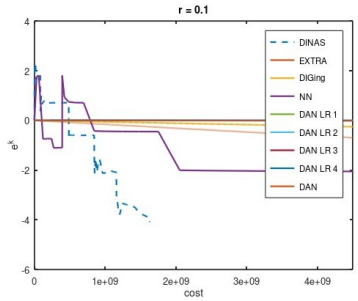
(d) $\lambda_{\max} = 1,\ r = 10$



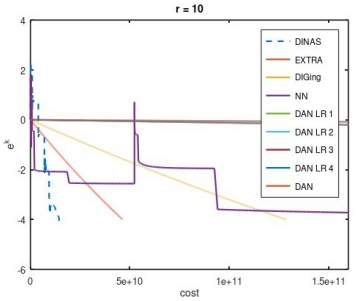
(e) $\lambda_{\max} = 10,\ r = 0.1$



(f) $\lambda_{\max} = 10,\ r = 10$



(g) $\lambda_{\max} = 100,\ r = 0.1$



(h) $\lambda_{\max} = 100,\ r = 10$

Figure 4.3: Total cost, quadratic function

Chapter 5

Distributed Fixed Point Methods for Linear Systems

In this chapter we consider the same problem as in Section 2.2.3. That is, we assume we have a linear system

$$A\mathbf{y} = \mathbf{b} \tag{5.1}$$

where $A \in \mathbb{R}^{n\times n}$ and $\mathbf{b} \in \mathbb{R}^n$ are given, and $\mathbf{y} \in \mathbb{R}^n$ is the vector of the unknowns. We assume that the matrix A is nonsingular, so that there exists a unique $\mathbf{y}^* \in \mathbb{R}^n$ solution of (5.1). Moreover, we assume that a set of computational agents is given, such that each node hold a subset of the equations of the system, and that can communicate according to a given network $\mathcal{G}$.

In this Chapter, we propose a class of distributed methods to solve (5.1), which we refer to as DFIX (Distributed Fixed Point), based on a given centralized fixed point method associated with the linear system, [28]. That is, we extend the convergence theory of fixed point methods to the decentralized framework. In particular we prove that under suitable assumptions on the communication network $\mathcal{G}$, we can prove in the distributed case a convergence result that is completely

analogous to the convergence theorem of general fixed point methods in the classical framework: if the spectral radius of the iterative matrix is smaller than one, the method converges for any choice of the initial guess.

This Chapter is organized as follows. In Section 5.1 we present DFIX method and its theoretical analysis. We prove linear convergence of DFIX under directed strongly connected fixed network and explicitly quantify the corresponding convergence factor in terms of the parameters of the linear system and the underlying network. We also propose a modification of DFIX that works in the case where each nodes hold a subset of the equations, and we shouw that the same theoretical results still apply. In Section 5.2 we extend the convergence analysis of the proposed method to the case of time-varying networks. We provide assumptions over the sequence of networks that ensure convergence of DFIX and we compare such assumptions with the assumptions made for similar papers in the literature. In particular we prove that they are equivalent to those considered in [38, 36, 64]. We also prove that the case with fixed network is a particular case of the time-varying case. In Section 5.3 we provide extensive numerical results. We study how the computational cost and communication traffic of DFIX depend on the density of the communication network and on the number of nodes in the network. Moreover, we compare DFIX with several methods for the literature, including the method for linear system in the decentralized framework presented in [36] and different first order optimization methods.

5.1 DFIX method

Let us now define precisely the computational environment we consider in this chapter. Assume that the network of nodes is a directed

network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of all edges, i.e., all pairs (i, j) of nodes where node i can send information to node j through a communication link.

Definition 5.1. *The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is* strongly connected *if for every couple of nodes i, j there exists an oriented path from i to j in $\mathcal{G}$. That is, if there exist $s_1, \ldots, s_l$ such that $(i, s_1), (s_1, s_2), \ldots, (s_l, j) \in \mathcal{E}$.*

Assumption C1. *The network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is directed, strongly connected, with self-loops at every node.*

Remark 5.1. *The case of undirected network $\mathcal{G}$ can be seen as the particular case of directed graph where $\mathcal{G}$ is symmetric. That is, $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$. In this case, the hypothesis that $\mathcal{G}$ is strongly connected is equivalent to $\mathcal{G}$ connected.*

Let us denote by O_i the in-neighborhood of node i, that is, the set of nodes that can send information to node i directly. Since the graph has self loops at each node, then $i \in O_i$ for every i. We associate with $\mathcal{G}$ an $n \times n$ matrix W, such that the elements of W are all nonnegative and each row sums up to one. More precisely, we assume the following.

Assumption C2. *The matrix $W \in \mathbb{R}^{n \times n}$ is row stochastic with elements w_{ij} such that*

$$w_{ij} > 0 \text{ if } j \in O_i, \; w_{ij} = 0 \text{ if } j \notin O_i$$

Let us denote by w_{min} a constant such that all nonzero elements of W satisfy $w_{ij} \geq w_{\min} > 0$. Under the previously stated assumptions we know that such constant exists. Moreover, we have $w_{\min} \in (0, 1)$. Therefore, for all elements of W we have

$$w_{ij} \neq 0 \Rightarrow w_{ij} \geq w_{\min}. \tag{5.2}$$

The diameter of a network is defined as the largest distance between two nodes in the graph. Let us denote with δ the diameter of $\mathcal{G}$.

We consider a generic fixed point method for solving (5.1). That is,

$$\mathbf{y}^{k+1} = M\mathbf{y}^k + \mathbf{d}, \tag{5.3}$$

with $M = [m_{ij}] \in \mathbb{R}^{n\times n}$, $d = [d_i] \in \mathbb{R}^n$ defined in such a way that node i contains the i-th row $M_i \in \mathbb{R}^{1\times n}$ and $d_i \in \mathbb{R}$. Moreover, we assume that the fixed point y^* of (5.3) is a solution of (5.1). We assume that at each iteration each node holds a local copy of the vector of variables, that is, each node hold a vector of size n equal to the dimension of the considered problem. We denote with $\mathbf{x}_i^k$ the local vector of variable for node i at iteration k.

The DFIX method is presented in the algorithm below.

Algorithm 5.1. *[DFIX]*
Input: $\{\mathbf{x}_i^0\}_{i=1,\dots,N} \subset \mathbb{R}^n$
Iteration k***, node*** i***:***

1: compute $\hat{\mathbf{x}}_i^{k+1}$ *as follows*

$$\begin{aligned} \hat{x}_{ii}^{k+1} &= \sum_{j=1}^{n} m_{ij}x_{ij}^k + d_i, \\ \hat{x}_{ij}^{k+1} &= \hat{x}_{ij}^k, \ i \neq j. \end{aligned} \tag{5.4}$$

2: share $\hat{\mathbf{x}}_i^k$ *with its neighbors*
3: update the local estimate $\mathbf{x}_i^k$

$$\mathbf{x}_i^{k+1} = \sum_{j=1}^{n} w_{ij}\hat{\mathbf{x}}_j^{k+1} \tag{5.5}$$

Notice that in line 1 each node i updates only the i-th component of its solution estimate and leaves all other components unchanged, while in line 3 all nodes perform a consensus step [11, 23, 61] using the set of vector estimates $\hat{\mathbf{x}}_j^{k+1}$. Defining the global variable at iteration k as

$$X^k = \left(\mathbf{x}_1^k; \ldots; \mathbf{x}_n^k\right) \in \mathbb{R}^{n^2},$$

Algorithm DFIX can be stated in a condensed form using X^k and the following notation

$$\widehat{M}_i = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ m_{i1} & \ldots & m_{ii} & \ldots & m_{in} & \\ & & & \ddots & & \\ & & & & 1 & \end{pmatrix} \in \mathbb{R}^{n\times n}, \qquad \widehat{\mathbf{d}}_i = \begin{pmatrix} 0 \\ \vdots \\ d_i \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^n.$$

More precisely, matrix $\widehat{M}_i$ has the i-th row equal to M, the rest of diagonal elements are equal to 1 and the remaining elements are equal to 0. Vector $\widehat{\mathbf{d}}_i$ has only one nonzero element in the i-th row which is equal to d_i. Now, Step 1 can be rewritten as

$$\hat{\mathbf{x}}_i^{k+1} = \widehat{M}_i \mathbf{x}_i^k + \widehat{\mathbf{d}}_i,$$

and we can rewrite the Steps 1-2 in matrix form as

$$X^{k+1} = (W \otimes I)(\mathcal{M} X^k + \hat{d}) \tag{5.6}$$

where $\mathcal{M} = \operatorname{diag}\left(\widehat{M}_1, \ldots, \widehat{M}_n\right) \in \mathbb{R}^{n^2\times n^2}$, $\widehat{\mathbf{d}} = \left(\widehat{\mathbf{d}}_1; \ldots; \widehat{\mathbf{d}}_n\right) \in \mathbb{R}^{n^2}$ and $\otimes$ denotes the Kronecker product of matrices. We remark here that equation (5.6) is only theoretical, in the sense that since each agent has access only to partial information, the global vector X^k, the matrix $\mathcal{M}$ and the vector $\widehat{d}$ are not computed at any node. We derived

equation (5.6) to get a compact representation of Algorithm 5.1 and to use it in the convergence analysis.

The following theorem shows that for every $i \in \{1, \ldots, n\}$ the local sequence $\{\mathbf{x}_i^k\}$ converges to the fixed point $\mathbf{y}^*$ of (5.3). Denote

$$X^* = (\mathbf{y}^*; \ldots; \mathbf{y}^*) \in \mathbb{R}^{n^2}.$$

Theorem 5.1. *Let Assumptions A1 and A2 hold,* $\|M\|_\infty = \mu < 1$ *and let* $\{X^k\}$ *be a sequence generated by* (5.6). *There exists a constant* $\tau < 1$ *such that for every* k *the global error* $E^k = X^k - X^*$ *satisfies*

$$\|E^{k+1}\|_\infty \leq \tau \|E^{k-\delta+1}\|_\infty, \tag{5.7}$$

where δ *denotes the diameter of the underlying computational graph* $\mathcal{G}$.

Proof. Since W is assumed to be row stochastic there holds $(W \otimes I)X^* = X^*$. Moreover, using the fact that $\hat{d} = (I \otimes I - \mathcal{M})X^*$, we obtain the following recursion

$$E^{k+1} = (W \otimes I)\mathcal{M}E^k. \tag{5.8}$$

Notice that $\|(W \otimes I)\mathcal{M}\|_\infty \leq 1$, so we have

$$\|E^{k+1}\|_\infty \leq \|E^k\|_\infty. \tag{5.9}$$

Now, denoting by $\mathbf{e}_i^k$ the i-th block of E^k (the local error corresponding to node i) and by e_{ij}^k its j-th component, from (5.8) we obtain the following

$$e_{ij}^{k+1} = w_{ij} M_j \mathbf{e}_j^k + \sum_{s \neq j} w_{is} e_{sj}^k. \tag{5.10}$$

We prove the thesis by proving that if the distance between j and i in the graph is equal to l, then for every k

$$|e_{ij}^{k+1}| \leq \tau' \|E^{k-l+1}\|_\infty, \text{ for a constant } \tau' < 1. \tag{5.11}$$

We proceed by induction over the distance l. If $l = 1$, that is, if there is an edge from j to i, then $w_{ij} \geq w_{min} > 0$. By (5.10) we get

$$\begin{aligned} |e_{ij}^{k+1}| &\leq w_{ij}|M_j \mathbf{e}_j^k| + \sum_{s \neq j} w_{is}|e_{sj}^k| \leq w_{ij}\mu\|E^k\|_\infty + \|E^k\|_\infty \sum_{s \neq j} w_{is} \\ &\leq \big(1 - w_{ij}(1-\mu)\big)\|E^k\|_\infty \leq \big(1 - w_{\min}(1-\mu)\big)\|E^k\|_\infty, \end{aligned} \tag{5.12}$$

and defining $\tau' = \big(1 - w_{\min}(1-\mu)\big) < 1$, we get

$$|e_{ij}^{k+1}| \leq \tau' \|E^k\|_\infty. \tag{5.13}$$

Assume now that (5.11) holds for distance equal to $l-1$, and let us prove it for l. Let $(j, s_{l-1}, s_{l-2}, \ldots, s_1, i)$ be a path of length l from j to i. In particular we have that $w_{is_1} > 0$ and thus

$$|e_{ij}^{k+1}| \leq w_{is_1}|e_{s_1 j}^k| + \sum_{s \neq s_1} w_{is}|e_{sj}^k|. \tag{5.14}$$

For each of the terms $|e_{sj}^k|$ in the sum, by (5.9), we have

$$|e_{sj}^k| \leq \|E^k\|_\infty \leq \|E^{k-l+1}\|_\infty. \tag{5.15}$$

Let us now consider the term $|e_{s_1 j}^k|$. Since $(j, s_{l-1}, s_{l-2}, \ldots, s_1, i)$ is a path of length l from j to i and the distance between j and i is equal to l, we have that the distance between j and s_1 is equal to $l-1$ and therefore, by inductive hypothesis

$$|e_{s_1 j}^k| \leq \tau' \|E^{k-(l-1)}\|_\infty = \tau' \|E^{k-l+1}\|_\infty, \text{ for } \tau' < 1. \tag{5.16}$$

Replacing (5.15) and (5.16) in (5.14), we get

$$\begin{aligned} |e_{ij}^{k+1}| &\leq w_{is_1}\tau' \|E^{k-l+1}\|_\infty + \sum_{s \neq s_1} w_{is}\|E^{k-l+1}\|_\infty = \\ &= \left(1 - w_{s_1 j}(1-\tau')\right)\|E^{k-l+1}\|_\infty \\ &\leq \left(1 - w_{min}(1-\tau')\right)\|E^{k-l+1}\|_\infty \end{aligned} \tag{5.17}$$

and defining $\tau := (1 - w_{min}(1 - \tau')) < 1$ we get (5.11). Now the thesis follows directly from the fact that the distance between any two nodes is smaller or equal than the diameter δ of the graph.□

The previous analysis shows that the global error in nonexpanding and that we have a decrease after at most δ iterations, where δ is the diameter of the underlying graph. Next we quantify the R-linear convergence factor.

Corollary 5.1. *Suppose that the assumptions of Theorem 5.1 are satisfied. Then each node's solution estimate* $\mathbf{x}_i^k$ *converges to the solution* $\mathbf{y}^*$ *of the problem* (5.3) *R-linearly with the factor* $\gamma = \tau^{1/\delta}$, *i.e., for each* $i \in \{1, 2, ..., N\}$ *there holds* $\|\mathbf{x}_i^k - \mathbf{y}^*\|_\infty = \mathcal{O}(\gamma^k)$, *where* $\gamma = \left(1 - w_{min}^{\delta}(1 - \mu)\right)^{1/\delta}$.

Proof. Denote $\xi_k := \|X^k - X^*\|_\infty = \|E^k\|_\infty$. Notice that (5.9) implies that $\xi_{k+1} \leq \xi_k$ for every k. Moreover, every iteration k can be represented as $k = s\delta + c$, where $s, c \in \mathbb{N}_0$ and $c < \delta$. Then,

$$\xi_k \leq \xi_{k-c} \leq \tau^s \xi_0 = \tau^{(k-c)/\delta} \xi_0 \leq \tau^{k/\delta} \tau^{-1} \xi_0 := \gamma^k C,$$

where

$$\gamma = \tau^{1/\delta} = \left(1 - w_{min}^{\delta}(1 - \mu)\right)^{1/\delta}$$

and

$$C = \frac{\xi_0}{\tau} = \frac{\|X^0 - X^*\|_\infty}{1 - w_{min}^{\delta}(1 - \mu)}.$$

By definition of X^k and X^* we have $\|\mathbf{x}_i^k - \mathbf{y}^*\|_\infty \leq \xi_k,\ i = 1, \ldots, N$ and the result follows. □

5.1.1 DFIX Method - multirow case

We consider now the case where each of the nodes holds a subset of rows of the fixed point method, opposed to the previous section, where

each node had exactly one row. We consider again the fixed point iterative method (5.3) and we assume that N computational agents are given. We denote with R_i the set if indecess of the rows available to agent i and we assume $R_i \bigcap R_j = \varnothing$ for $i \neq j$ and $\bigcup_{i=1}^{N} R_i = \{1, 2, ..., n\}$. More precisely, each node i holds $M_j \in \mathbb{R}^{1\times n}$ and $d_i \in \mathbb{R}$, for all $j \in R_i$. As in the previous section. the algorithm is designed in such a way that each node computes a local estimate of the solution $\mathbf{y}^*$.

Algorithm 5.2. *[DFIXM]*
Input: $\{\mathbf{x}_i^0\}_{i=1,...,N} \subset \mathbb{R}^n$
Iteration k***, node*** i***:***

1: compute $\hat{\mathbf{x}}_i^{k+1}$ *as follows*

$$\begin{aligned} \hat{x}_{ij}^{k+1} &= \sum_{l=1}^{n} m_{jl} x_{il}^k + d_j, \quad j \in R_i, \\ \hat{x}_{ij}^{k+1} &= \hat{x}_{ij}^k, \quad j \notin R_i. \end{aligned} \tag{5.18}$$

2: share $\hat{\mathbf{x}}_i^k$ *with its neighbors*
3: update the local estimate $\mathbf{x}_i^k$

$$\mathbf{x}_i^{k+1} = \sum_{j=1}^{N} w_{ij} \hat{\mathbf{x}}_j^{k+1} \tag{5.19}$$

Notice that, analogously to the single-row case, in line 1 each node i updates only the components $j \in R_i$ of its solution estimate, while in line 3 a consensus step is performed using the set of iterates $\hat{\mathbf{x}}_j^{k+1}$ obtained from the immediate neighbours.

Algorithm DFIXM can be stated in the condensed form with

$$X^k = \begin{pmatrix} \mathbf{x}_1^k \\ \vdots \\ \mathbf{x}_n^k \end{pmatrix} \in \mathbb{R}^{Nn}, \quad \widehat{d}_i = \begin{pmatrix} 0 \\ \vdots \\ d_j \\ d_{j+1} \\ \vdots \\ d_{j+q_i} \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^n$$

and $\widehat{M}_i \in \mathbb{R}^{n\times n}$ such that the j-th row of $\widehat{M}_i$ is equal to the j-th row of M for all $j \in R_i$, the rest of diagonal elements are equal to 1 and the remaining elements are equal to 0.

Now, line 1 can be rewritten as

$$\hat{\mathbf{x}}_i^{k+1} = \widehat{M}_i \mathbf{x}_i^k + \hat{\mathbf{d}}_i,$$

and each iteration of Algorithm DFIXM can be written as

$$X^{k+1} = (W \otimes I)(\mathcal{M}X^k + \widehat{\mathbf{d}}) \tag{5.20}$$

where $\mathcal{M} = \operatorname{diag}\left(\widehat{M}_1, \ldots, \widehat{M}_n\right) \in \mathbb{R}^{Nn\times Nn}$, and $\widehat{\mathbf{d}} = \left(\hat{d}_1; \ldots; \hat{d}_n\right) \in \mathbb{R}^{Nn}$. As we already notice, the global expression (5.6) is never computed at any node and it is derived only for theoretical analysis.

The following theorem shows that for every $i \in \{1, \ldots, N\}$ the local sequence $\{\mathbf{x}_i^k\}$ converges to the fixed point $\mathbf{y}^*$ of (5.3) as in the case of DFIX.

Theorem 5.2. *Let Assumptions C1 and C2 hold, $\|M\|_\infty = \mu < 1$ and let $\{X^k\}_{k=0}^\infty$ be the sequence generated by* (5.20)*. Then, for every k, the global error $E^k = X^k - X^*$ satisfies*

$$\|E^{k+1}\|_\infty \leq \tau \|E^{k-\delta+1}\|_\infty, \tag{5.21}$$

where δ denotes the diameter of the underlying computational graph $\mathcal{G}$ and

$$\tau = 1 - w_{min}^{\delta}(1-\mu) \in (0,1). \tag{5.22}$$

Proof. The proof is essentially the same as the proof of Theorem 5.1 with some technical changes. The error expression is now

$$e_{ij}^{k+1} = w_{ij} M_h \mathbf{e}_j^k + \sum_{s \neq j} w_{is} e_{sj}^k, \tag{5.23}$$

where h depends on i and j. As in the previous case, we prove the thesis by proving that if the distance between j and i in the graph is equal to l, then

$$|e_{ij}^{k+1}| \leq \tau \|E^{k-l+1}\|_\infty, \tag{5.24}$$

for every k, with $\tau = 1 - w_{min}^{l}(1-\mu) \in (0,1)$. Let us proceed by induction over the distance l. If $l = 1$, that is, if there is an edge from j to i, then $w_{ij} \geq w_{min} > 0$. By (5.23) we get

$$\begin{aligned}
|e_{ij}^{k+1}| &\leq w_{ij}|M_h \mathbf{e}_j^k| + \sum_{s \neq j} w_{is}|e_{sj}^k| \\
&\leq w_{ij}\|E^k\|_\infty \sum_{l=1}^{n} |m_{hl}| + \|E^k\|_\infty \sum_{s \neq j} w_{is} \\
&\leq w_{ij}\mu\|E^k\|_\infty + \|E^k\|_\infty(1 - w_{ij}) \\
&\leq \big(1 - w_{ij}(1-\mu)\big)\|E^k\|_\infty \\
&\leq \big(1 - w_{\min}(1-\mu)\big)\|E^k\|_\infty,
\end{aligned}$$

and defining $\tau' = 1 - w_{\min}(1-\mu) < 1$, we get

$$|e_{ij}^{k+1}| \leq \tau' \|E^k\|_\infty. \tag{5.25}$$

The rest of the proof is completely analogous to the proof of Theorem 5.1 and hence omitted here.□

Analogously, we can quantify the convergence factor in the same way as before and the corollary below holds.

Corollary 5.2. *Suppose that the assumptions of Theorem 5.2 are satisfied. Then each node's solution estimate x_i^k converges to the solution y^* of the problem (5.3) R-linearly with the factor $\gamma = \tau^{1/\delta}$, i.e., for each $i \in \{1, 2, ..., N\}$ there holds $\|\mathbf{x}_i^k - \mathbf{y}^*\|_\infty = \mathcal{O}(\gamma^k)$, where $\gamma = \left(1 - w_{min}^\delta (1-\mu)\right)^{1/\delta}$.*

DFIXM is a generalization of DFIX that might be of practical importance as it allows us to solve an n dimensional linear system with an arbitrary number of nodes $N \leq n$ which might be the case in many applications. However we will continue with DFIX method for time-varying networks in the next Section to avoid notation cluttering and to facilitate reading. The changes in the proofs are of the same type as above.

5.2 Time-varying Network

The theoretical analysis presented in the previous section relies on the fact that the communication network is the same at all iterations. As we noticed in Chapter 1 this assumption could be impractical since it does not take into account possible failures of the communication link between two agents. In this section, we extend DFIX to the time-varying framework and we give assumptions on the sequence of graphs that yield a convergence result analogous to Theorem 5.1. In particular we show that, in order to achieve convergence, the underlying network does not need to be strongly connected at any time.

Assume that a sequence of directed graphs $\{\mathcal{G}_k\}_k$ is given, such that $\mathcal{G}_k$ represents the network of nodes at iteration k. That is, at iteration k, each node can communicate with its neighbours in $\mathcal{G}_k$.

The DFIX algorithm described by equations (5.4) and (5.5) can be applied in this case if we replace (5.5) with

$$\mathbf{x}_i^{k+1} = \sum_{j=1}^{n} w_{ij}^k \hat{\mathbf{x}}_j^{k+1} \tag{5.26}$$

where W^k is the consensus matrix associated with the graph $\mathcal{G}_k$, that is, W^k satisfies Assumption C2 with $\mathcal{G} = \mathcal{G}_k$. With this modification, the equation describing the global iteration becomes

$$X^{k+1} = (W^k \otimes I)(\mathcal{M}X^k + \hat{d}). \tag{5.27}$$

We will prove a convergence result for a class of sequences of graphs. We first present and analyze the assumptions on such sequence.

Remark 5.2. *Let us consider a generic set of graphs $\mathcal{G}_1, \ldots, \mathcal{G}_m$. It is easy to see that if for every index j the graph $\mathcal{G}_j$ has self-loops at every node then the set of edges of the composition $\mathcal{G}_1 \circ \cdots \circ \mathcal{G}_m$ contains the set of edges of $\mathcal{G}_j$ for every j. In particular, if there exists an index $\hat{\jmath} \in \{1, \ldots, m\}$ such that $\mathcal{G}_{\hat{\jmath}}$ is fully connected, then $\mathcal{G}_1 \circ \cdots \circ \mathcal{G}_m$ is also fully connected.*

Definition 5.2. *Given an infinite sequence of networks $\{\mathcal{G}_k\}_k$ and a positive integer $\bar{m}$, we say that the sequence is* jointly fully *(respectively,* strongly*)* connected for sequences of length $\bar{m}$ *if for every index k, the composition $\mathcal{G}_k \circ \mathcal{G}_{k+1} \circ \cdots \circ \mathcal{G}_{k+\bar{m}-1}$ is fully (respectively, strongly) connected.*

Definition 5.3. *Given an infinite sequence of networks $\{\mathcal{G}_k\}_k$ and two integers τ_0, l, we say that the sequence is* repeatedly jointly strongly connected with constants τ_0, l, *if for every index k, the composition $\mathcal{G}_{\tau_0+kl} \circ \mathcal{G}_{\tau_0+kl+1} \circ \cdots \circ \mathcal{G}_{\tau_0+(k+1)l}$ is strongly connected.*

Definition 5.4. *Given two vertices i, j we say that there is a* joint path *of length l from i to j in $\mathcal{G}_k, \ldots, \mathcal{G}_{k+\bar{m}-1}$ if there exist $s_1, \ldots, s_{l-1}$ such that $(i, s_1) \in \mathcal{E}_{k+\bar{m}-1}$, $(s_1, s_2) \in \mathcal{E}_{k+\bar{m}-2}, \ldots, (s_{l-1}, j) \in \mathcal{E}_{k+\bar{m}-l}$, and we say that i, j have* joint distance *l in $\mathcal{G}_k, \ldots, \mathcal{G}_{k+\bar{m}-1}$ if the shortest joint path from i to j is of length l.*

Our analysis is based on the following assumption.

Assumption C3. *$\{\mathcal{G}_k\}$ is a sequence of directed graphs, with self-loops at every node, jointly fully connected for sequences of length $\bar{m}$, for some positive integer $\bar{m}$.*

The algorithm presented in [38] works for time-varying network in a similar framework. Formally, the hypothesis on $\{\mathcal{G}_k\}$ in [38] is the following.

Assumption C3'. *$\{\mathcal{G}_k\}$ is a sequence of directed graphs, with self-loops at every node, jointly strongly connected for sequences of length $\bar{p}$, for some positive integer $\bar{p}$.*

We show now that Assumptions C3 and C3' are equivalent, in the sense specified by Proposition 5.1. In the following, given an integer m, we denote with $\mathcal{G}^m$ the composition of m copies of $\mathcal{G}$.

Lemma 5.1. *If $\mathcal{G}$ is a directed strongly connected graph with self-loops at every node and diameter δ, then $\mathcal{G}^\delta$ is fully connected.*

Proof. By definition of composition we have that (i, j) is an edge in $\mathcal{G}^\delta$ if and only if

$$\exists s_1, \ldots, s_{\delta-1} \in \mathcal{V} \text{ such that } (i, s_1), (s_1, s_2), \ldots, (s_{\delta-1}, j) \in \mathcal{G}. \quad (5.28)$$

We want to prove that for every $i, j \in \mathcal{V}$ a sequence of nodes s_h as in (5.28) exists.

Since $\mathcal{G}$ is fully connected with diameter δ, there exists a path in $\mathcal{G}$ from i to j of length $l \leq \delta$. That is, there exist a set of nodes $v_1, \dots, v_{l-1}$ such that $(i, v_1), (v_1, v_2), \dots, (v_{l-1}, j)$ are edges in $\mathcal{G}$ and therefore a sequence satisfying (5.28) is given by

$$s_h = \begin{cases} v_h & h = 1 : l-1 \\ j & h = l : \delta. \end{cases}$$

□

Proposition 5.1. *Let $\{\mathcal{G}_k\}$ be a sequence of graphs where, for each k, $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ is a directed graph with self-loops at every node. The following are equivalent:*

(1) there exist $\tau_0, l \in \mathbb{N}$ such that $\{\mathcal{G}_k\}$ is repeatedly jointly strongly connected with constants τ_0, l

(2) there exists $\bar{p} \in \mathbb{N}$ such that $\{\mathcal{G}_k\}$ is strongly connected for sequences of length $\bar{p}$

(3) there exists $\bar{m} \in \mathbb{N}$ such that $\{\mathcal{G}_k\}$ is fully connected for sequences of length $\bar{m}$

Proof. It is easy to see that (2) ⇒ (1) with $\tau_0 = 0$ and $l = \bar{p}$ and since full connectivity clearly implies strong connectivity, we have that (3) ⇒ (2) with $\bar{p} = \bar{m}$.
We now prove that (1) ⇒ (2) with $\bar{p} = 2l$. That is, we prove that if (1) holds, then for every index s the composition $\mathcal{G}_s \circ \cdots \circ \mathcal{G}_{s+2l-1}$ is strongly connected. Given an index s, we denote with $\bar{r}$ the remainder of the division of $(s - \tau_0)$ by l, we define $\bar{h} := l^{-1}(s - \tau_0 + l - \bar{r})$. By definition of $\bar{r}$ and $\bar{h}$ and applying (1) with $k = \bar{h}$ we have that the graph

$$\begin{aligned} H :&= \mathcal{G}_{s+l-\bar{r}} \circ \cdots \circ \mathcal{G}_{s+2l-\bar{r}-1} = \\ &= \mathcal{G}_{\tau_0+\bar{h}l} \circ \cdots \circ \mathcal{G}_{\tau_0+(\bar{h}+1)l-1} \end{aligned}$$

is strongly connected and thus

$$\mathcal{G}_s \circ \cdots \circ \mathcal{G}_{s+2l-2} = \mathcal{G}_s \circ \cdots \circ \mathcal{G}_{s+l-\bar{r}-1} \circ H \circ \mathcal{G}_{s+2l-\bar{r}} \circ \cdots \circ \mathcal{G}_{s+2l-1}$$

is strongly connected. Since $2l - \bar{r} \in l+1, \ldots, 2l$ we have the thesis. Finally, we prove that (2) $\Rightarrow$ (3). Since the size of $\mathcal{V}$ is finite, there exists a finite number of graphs with vertices $\mathcal{V}$. In particular, there exists a finite integer L equal to the number of strongly connected graphs with vertices $\mathcal{V}$. We denote with $H_1, \ldots H_L$ such graphs, with δ_j the diameter of H^j and with $\bar{\delta} := \max \delta_j$. Given any index k, we consider $(\bar{\delta}-1)L+1$ sequences of length $\bar{p}$ as follows:

$$\begin{aligned}
&S_1 = \mathcal{G}_k \circ \mathcal{G}_{k+1} \cdots \circ \mathcal{G}_{k+\bar{p}-1} \\
&S_2 = \mathcal{G}_{k+\bar{p}} \circ \mathcal{G}_{k+\bar{p}+1} \cdots \circ \mathcal{G}_{k+2\bar{p}-1} \\
&\vdots \\
&S_{(\bar{\delta}-1)L+1} = \mathcal{G}_{k+(\bar{\delta}-1)L\bar{p}} \circ \mathcal{G}_{k+(\bar{\delta}-1)L\bar{p}+1} \cdots \circ \mathcal{G}_{k+(\bar{\delta}-1)L\bar{p}+\bar{p}-1}.
\end{aligned}$$

Statement (2) implies that, for every $j \in \{1, \ldots, (\bar{\delta}-1)L+1\}$, $S_j \in \{H_1, \ldots H_L\}$ and thus there exists an index $\hat{\imath} \in \{1, \ldots, L\}$ such that at least $\bar{\delta}$ elements of $\{S_1, \ldots, S_{(\bar{\delta}-1)L+1}\}$ are equal to $H_{\hat{\imath}}$. Using the fact that, by Lemma 1, $H_{\hat{\imath}}^{\delta_{\hat{\imath}}}$ is fully connected and Remark 5.2, we have

$$\mathcal{G}_k \circ \mathcal{G}_{k+1} \circ \cdots \circ \mathcal{G}_{k+(\bar{\delta}-1)L\bar{p}+\bar{p}-1} = S_1 \circ \cdots \circ S_{(\bar{\delta}-1)L+1}$$

fully connected, and thus (3) holds with $\bar{m} = (\bar{\delta}-1)L\bar{p} + \bar{p}$.□

To conclude the considerations on the sequence of networks we remark that, since we are assuming that the linear system (5.1) has unique solution and that each node contains exactly one row of the coefficient matrix, the D-connectivity hypothesis introduced in [36] is equivalent to Assumption C3' and thus, by Proposition 5.1, to Assumption C3'.

Theorem 5.3. *Assume that a sequence of networks $\{\mathcal{G}_k\}_k$ is given, satisfying Assumption C3, and that for every index k the corresponding consensus matrix W^k satisfies Assumption A2. Let $\{X^k\}$ be a sequence generated by (5.27) with $\|M\|_\infty = \mu < 1$. There exists a constant $\sigma < 1$ such that for every $k \in \mathbb{N}_0$ the global error $E^k = X^k - X^*$ satisfies*

$$\|E^{k+1}\|_\infty \leq \sigma \|E^{k-\bar{m}+1}\|_\infty, \tag{5.29}$$

where $\bar{m}$ is the constant given by Assumption A3.

Proof. We follow the proof of Theorem 5.1. For every index k, the matrix W^k is row stochastic and $\|(W^k \otimes I)\mathcal{M}\|_\infty \leq 1$, so we get

$$E^{k+1} = (W^k \otimes I)\mathcal{M}E^k. \tag{5.30}$$

and

$$\|E^{k+1}\|_\infty \leq \|E^k\|_\infty. \tag{5.31}$$

For every node i, j and for every iteration index k, we have

$$e_{ij}^{k+1} = w_{ij}^k M_j e_j^k + \sum_{s \neq j} w_{is}^k e_{sj}^k. \tag{5.32}$$

We now prove that if the joint distance between j and i in $\mathcal{G}_{k-\bar{m}+1}$, $\mathcal{G}_{k-\bar{m}+2}$, $\ldots$, $\mathcal{G}_k$ is equal to l, then for every k

$$|e_{ij}^{k+1}| \leq \sigma' \|E^{k-l+1}\|_\infty, \text{ for } \sigma' < 1. \tag{5.33}$$

We proceed by induction over the joint distance l. If $l = 1$, that is, if $w_{ij}^k > 0$, proceeding as in the derivation of (5.12) we get

$$|e_{ij}^{k+1}| \leq \big(1 - w_{ij}^k(1-\mu)\big)\|E^k\|_\infty \leq \big(1 - w_{\min}(1-\mu)\big)\|E^k\|_\infty =: \sigma\|E^k\|_\infty.$$

We assume now that (5.33) holds for distance equal to $l-1$ and we prove it for l. Let $(j, s_{l-1}, s_{l-2}, \ldots, s_1, i)$ be a joint path of length l from

j to i in $\mathcal{G}_{k-\bar{m}+1}, \mathcal{G}_{k-\bar{m}+2}, \ldots, \mathcal{G}_k$ In particular we have that $w_{is_1}^k > 0$ and thus

$$|e_{ij}^{k+1}| \leq w_{is_1}|e_{s_1 j}^k| + \sum_{s \neq s_1} w_{is}|e_{sj}^k|. \tag{5.34}$$

Using the fact that $(j, s_{l-1}, s_{l-2}, \ldots, s_1)$ is a joint path of length $l-1$ from j to s_1 in $\mathcal{G}_{k-\bar{m}+1}, \mathcal{G}_{k-\bar{m}+2}, \ldots, \mathcal{G}_{k-1}$, applying the inductive hypothesis and proceeding as in the proof of the previous theorem, we get

$$|e_{ij}^{k+1}| \leq (1 - w_{min}(1-\sigma'))\, \|E^{k-l+1}\|_\infty \tag{5.35}$$

with σ' given by (5.33) for distance $l-1$, and defining

$$\sigma := (1 - w_{min}(1-\sigma')) < 1$$

we get (5.33) for distance equal to l.
Since the sequence $\{\mathcal{G}_k\}$ is fully connected for sequences of length $\bar{m}$ we have that for every couple of nodes i, j the joint distance between j and i in $\mathcal{G}_{k-\bar{m}+1}, \mathcal{G}_{k-\bar{m}+2}, \ldots, \mathcal{G}_k$ is smaller or equal than $\bar{m}$ and we get the thesis.□

Lemma 1 shows that if we consider the time-independent case as the particular instance of the time-varying case where each of the graphs $\mathcal{G}_k$ is equal to $\mathcal{G}$ with diameter δ, then Assumption C3 holds with $\bar{m} = \delta$ and the two theorems give the same inequality for the error vectors.

5.3 Numerical results

We now present the results of several numerical tests on DFIX method and on the comparison with the state-of-the-art distributed optimization algorithms from [35, 47, 57, 59] referred to here as DIGing, EXTRA and SVL respectively, and the method for solving systems of linear equations presented in [36], abbreviated here as Projection. The

test set consists of two types of problems: Simple Kriging problems and linear systems with strictly diagonally dominant coefficient matrix. In Section 5.3.1 we consider Simple Kriging problems. We study the influence of the connectivity of the underlying network on the computational cost and the communication traffic of DFIX, and we compare DFIX with the mentioned methods. In Section 5.3.2 we repeat the comparison considering a randomly generated linear system. Moreover, we tests the multi-row method DFIXM method is considered and we analyze how the number of nodes influence the performance of the method. In Sections 5.3.3 and 5.3.4 we consider the cases of directed and time-varying networks, respectively.

The results demonstrate that DFIX, analogously to the classical results, outperforms the optimization method for solving the unconstrained quadratic problem both in terms of computation and communication. With respect to the method from[36] the comparison is again favorable for DFIX. Clearly, the method from [36] is designed for a wider class of problems, but in the case of unique solution and a suitable iterative matrix its efficiency is significantly lower than DFIX.

In the following, the DFIX method we consider is defined using Jacobi Overrelaxation as the underlying fixed point method. That is, iteration k of the distributed method at each node is given by

$$\begin{aligned} \hat{x}_{ii}^{k+1} &= (1-\omega)x_{ii}^k - \frac{\alpha}{a_{ii}}\left(\sum_{j\neq i} a_{ij}x_{ij}^k - b_i\right), \\ \hat{x}_{ij}^{k+1} &= x_{ij}^k \quad \text{for } j \neq i, \end{aligned} \tag{5.36}$$

and

$$\mathbf{x}_i^{k+1} = \sum_{j=1}^{n} w_{ij}\hat{\mathbf{x}}_j^{k+1}. \tag{5.37}$$

In the rest of the section we refer to the method defined by equations (5.36), (5.37) as DFIX-JOR, and we choose the relaxation parameter ω

in (5.36) as $2/\|D^{-1}A\|_\infty$ where $D = diag(a_{11}, \ldots, a_{nn})$. The methods for distributed optimization DIGing, EXTRA and SVL are applied to solve the unconstrained problem with quadratic objective function given by $\frac{1}{2}\mathbf{x}^\top A\mathbf{x} - \mathbf{b}^\top\mathbf{x}$, which is equivalent to finding a solution of (5.1). The step-size parameter for DIGing and EXTRA is chosen as $\eta = 1/(3L)$ where $L = \max_{i=1:n} 2\|A_i\|_2^2$, while the parameters for SVL method are computed through the procedure described in [59]. We remark that the relaxation parameter for DFIX and the step-size η for DIGing and EXTRA can be easily computed in the distributed framework, the computation of the optimal parameters for SVL requires knowledge of the extremal eigenvalues of the matrix A and the spectral gap of the consensus matrix W. Finally, Projection method deals with the linear system (5.1) directly and it does not require the computation of any additional parameter, but it requires a local initial vector $\mathbf{x}_i^0$ for each node i.

5.3.1 Simple Kriging problem

The first kind of problems that we consider is Simple Kriging [7, 34, 42]: an optimal linear prediction technique of the expected value of a spatial random field $\mathcal{Z}(\mathbf{s})$, $s \in \mathbb{R}^n$. Let us consider a physical process modeled as a spatial random field and assume that a network of sensors is given in the region of interest, taking measurements of the field. The goal is to estimate the field in any given point of the region. Assuming that the field is Gaussian and stationary, and that the expected value and covariance function are known at any point, this kind of problem can be solved by Simple Kriging method.

Denote with $\mathcal{Z}(\mathbf{s})$ the value of the random field at the point s, and with $\mu(\mathbf{s})$ its expected value, which is assumed to be known. Moreover, by the stationarity assumption, the covariance between the value of $\mathcal{Z}$

at two points is given by

$$Cov(\mathcal{Z}(\mathbf{s}_1), \mathcal{Z}(\mathbf{s}_2)) = \mathcal{K}(\|\mathbf{s}_1 - \mathbf{s}_2\|_2)$$

for some nonnegative function $\mathcal{K}$. Given the positions in space $\{\mathbf{s}_1, \ldots, \mathbf{s}_n\} \subset \mathbb{R}^2$ of the n sensors of the network, let $\{\mathcal{Z}(\mathbf{s}_1), \ldots, \mathcal{Z}(\mathbf{s}_n)\}$ be the sampled values at those points and define the covariance matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ as $a_{ij} = \mathcal{K}(\|\mathbf{s}_i - \mathbf{s}_j\|_2)$. Now, given a point $\bar{s}$ where we want to estimate the field, the vector $b \in \mathbb{R}^n$ is defined as $b_i = \mathcal{K}(\|\mathbf{s}_i - \bar{\mathbf{s}}\|_2)$. The predicted value of $\mathcal{Z}(\bar{s})$ is then given by

$$\hat{p}(s) := \mu(\bar{\mathbf{s}}) + \sum_{i=1}^{n} x_i(\mathcal{Z}(\mathbf{s}_i) - \mu(\mathbf{s}_i))$$

where $(x_1, \ldots, x_n)$ is the approximate solution of the linear system

$$A\mathbf{x} = \mathbf{b}. \tag{5.38}$$

First of all, we study the influence of connectivity within the network in terms of communication traffic and computational cost for the kriging problem, with covariance function given by

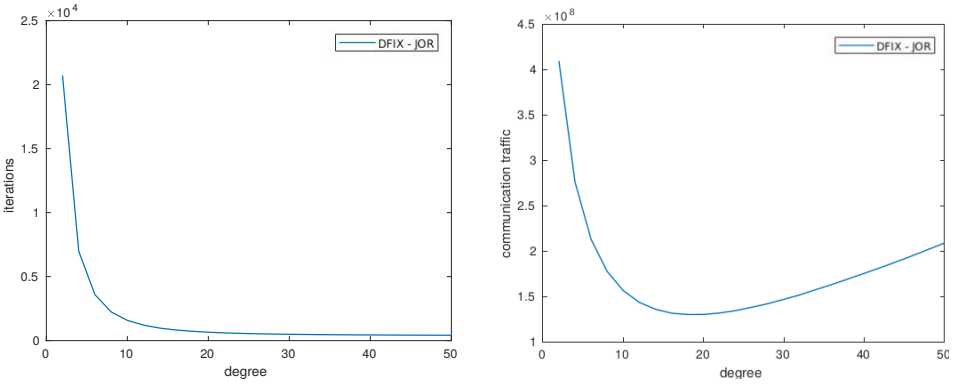
$$\mathcal{K}(t) := \exp(-5t^2). \tag{5.39}$$

We assume that a set $\{\mathbf{s}_1, \ldots, \mathbf{s}_{100}\} \subset [-30, 30]^2$ of agents is given and for any $m \in \{2, 4, \ldots, 48, 50\}$ we take the m-regular graph with vertices $\{\mathbf{s}_1, \ldots, \mathbf{s}_{100}\}$. The matrix W is defined using the Metropolis weights [65], defined in (2.2). For each value of the degree m the system $A\mathbf{x} = b$ is solved with DFIX-JOR. For every considered value of the degree m, in Figure 5.1a and 5.1b we plot the number of iterations and the total communication cost, respectively, until the stopping criterion

$$\max_{i=1,\ldots,n} \|A\mathbf{x}_i^k - b\| \leq 10^{-4} \tag{5.40}$$

is satisfied, for graphs of increasing degree. The communication cost is computed as follows. At each iteration, line 1 of DFIX does not require any communication, while in line 2 node i shares $\mathbf{x}_i^k$ with all the agents in its neighborhood. The per-iteration traffic is thus given by $n^2m = 2|\mathcal{E}|n$, where $\mathcal{E}$ is the set of edges of the underlying network and m is the degree. Note that here we implicitly assume that we consider the case of peer-to-peer communication. That is, there is a dedicated communication link between any pair of agents. In the other tests that we present the broadcasting scenario will also be considered: in that case, the per-iteration communication cost is independent to the number of edges in the network and it is given by the number of nodes times the size of the shared vectors, thus it is proportional to the number of performed iterations.
From Figure 5.1 one can see that, as the degree of the network increases, the number of iterations required to satisfy (5.40) decreases, while the total communication traffic first decreases then increases again. As the connectivity of the graph improves, the local information is distributed through the network more efficiently, and a smaller number of iterations is necessary. On the other hand, if the degree is larger, the consensus step (5.5) of the algorithm requires each node to share its local vector with a larger number of neighbours, yielding a higher communication traffic at each iteration. The fact that the overall communication traffic (Figure 5.1b) is nonmonotone suggests that for large values of the degree, the decrease in the number of iterations in not enough to balance the higher per-iteration traffic.

Let us now compare the DFIX-JOR with DIGing [35, 47], EXTRA [57], SVL [59] and Projection method [36]. We consider a 10×10 grid of nodes located at $\{s_1, \dots, s_{100}\} \subset [-3, 3]^2$ and, given a communication radius $R > 0$, we define the network so that nodes i and j are neighbours if and only if their distance is smaller than R. The linear system that we consider is derived by the kriging problem described at the beginning of this section. That is, we consider again $A\mathbf{x} = \mathbf{b}$



(a) Number of iterations (b) Communication cost

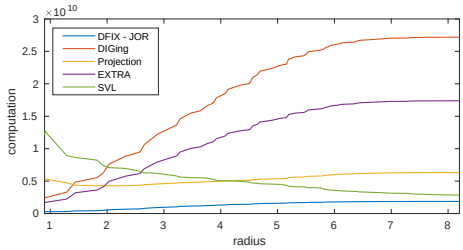
Figure 5.1: Dependence of number of iterations and communication traffic on the degree of the network

with

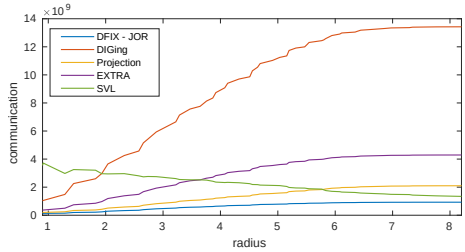
$$a_{ij} = \mathcal{K}(\|s_i - s_j\|_2), \quad b_i = \mathcal{K}(\|s_i - \bar{s}\|_2) \tag{5.41}$$

where $\mathcal{K}$ is given by (5.39) and $\bar{s}$ is a fixed random point in $[-3, 3]^2$. Proceeding as in the previous test, we compute the communication traffic and computational cost required by the three methods to achieve the tolerance specified at (5.40), for different values of the communication radius R. For each method, the overall computational cost is given by the number of iterations performed times the per-iteration cost, calculated as the number of scalar operations in one iteration. Similarly, the communication traffic is given by the number of iterations times the total number of vectors shared by the nodes during one iteration, times the length n of the vector. The matrix W is defined as in [65], with off-diagonal elements $w_{ij} = \frac{1}{1+\max\{m_i, m_j\}}$ if $j \in \mathcal{O}_i$, and $w_{ij} = 0$ otherwise, where m_i denotes the degree of node i. The diagonal elements are $w_{ii} = 1 - \sum_{j \neq i} w_{ij}$. The resulting matrix W is stochastic. The stopping criterion is the same as in the previous

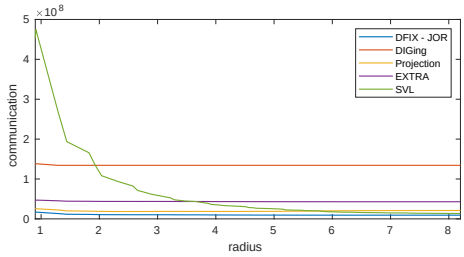
test. The initial point at each node is the same for all the methods, $x_{ii}^0 = b_i/a_{ii}$ and $x_{ij}^0 = 0$ for every $j \neq i$. In Figure 5.2 we plot the obtained results. As we can see, in this framework, DFIX method is more efficient than the methods we compare with, both in terms of computational and communication costs.



(a) computational cost



(b) peer-to-peer traffic



(c) broadcasting traffic

Figure 5.2: Simple kriging problem (5.41)

5.3.2 Strictly diagonally dominant systems

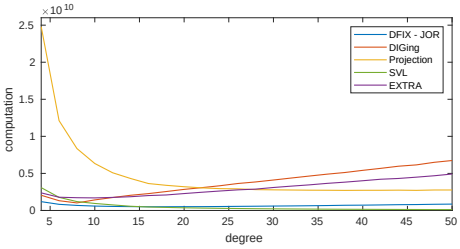
Let us now consider a linear system $A\mathbf{x} = \mathbf{b}$ of order $n = 100$, where A and b are generated as follows. For every index i we take b_i randomly generated with uniform distribution in $(0, 1)$ and A is a symmetric diagonally dominant random matrix obtained as follows: take

$\hat{a}_{ij} \in (0,1)$ with uniform distribution and then set $\tilde{A} = \frac{1}{2}(\hat{A} + \hat{A}^T)$ and finally $A = \hat{A} + (n-1)I$, where I is the identity matrix of order n. The underlying network is an m-regular graph with n nodes. For every fixed value of the degree m, 10 random linear systems are generated, solved with all methods and the average number of iterations needed to fulfill (5.40) is computed. The total amount of computation and communication for each method are then obtained multiplying the average number of iterations and the per-iteration computational cost and communication traffic, respectively. The matrix W is defined as in (2.2), the parameter of the methods are computed as described at the beginning of the section, while the initial guess at each node and the termination condition are as in the previous test. In Figures 5.3 we plot the results for $m \in \{2, 4, \ldots, 48, 50\}$. DFIX outperforms DIGing, EXTRA and Projection method in terms of computation and communication both in the peer-to-peer and in the broadcasting scenario, while SVL method performs better than DFIX for values of the degree larger than 15. We remark again that SVL method is run with the optimal choice of the parameters, exploiting information on the eigenvalues of A and W.

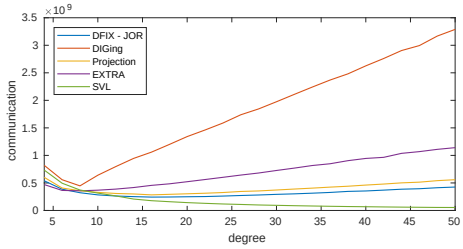
The same tests were performed on random Erdos-Renyi [17] graphs with given expected average degree for a sequence of increasing degrees. In these tests all methods are more expensive in term of both communication and computational effort but the the mutual comparison is the same as in the case of m-regular graphs.

To confirm the effectiveness of DFIXM, we repeat the previous test with a linear system of size $n = 500$ and $N = 100$ nodes, where each node is assigned 5 equations. As we can see in Figure 5.4 the results for all the methods are completely analogous to the case where each node holds one equation.

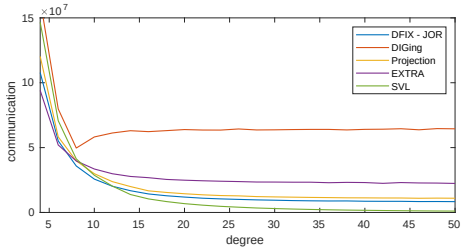
Let us now show the influence of the number of nodes in the network on performance of the five methods. We consider a lin-



(a) computational cost



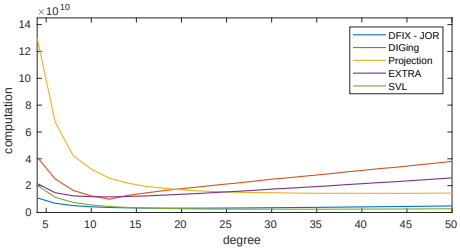
(b) peer-to-peer traffic



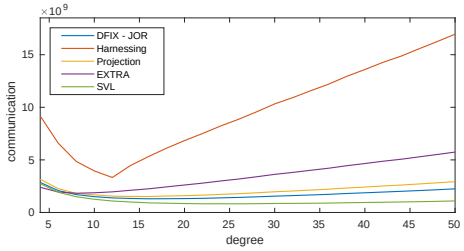
(c) broadcasting traffic

Figure 5.3: m-regular graph

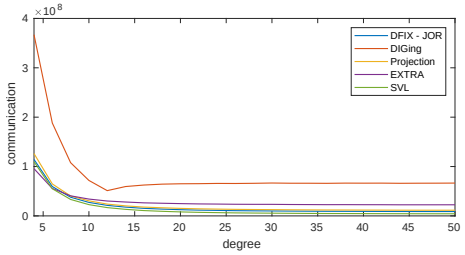
ear system of size $n = 100$ generated as described above, and for $N = 10, 20, \ldots, 100$ consider a regular network of size N. For each value of N the degree of the network is chosen so that the ratio between N and the degree is constant. The results are plotted in figures 5.5. The amount of both computation and communication of all the methods increases together with the number of nodes. Moreover, DFIX seems to outperform all the methods that we compare to in terms of computational costs, while in terms of communication it seems to be comparable with Projection and both methods seems to be cheaper than the optimization methods.



(a) computational cost



(b) peer-to-peer traffic

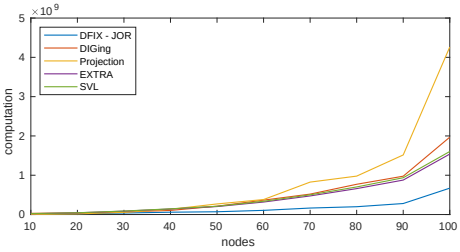


(c) broadcasting traffic

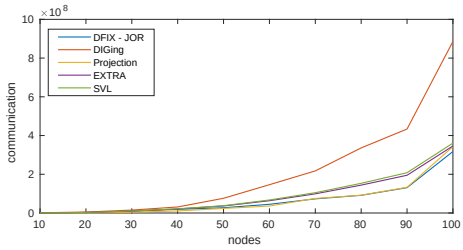
Figure 5.4: DFIXM

5.3.3 Directed Networks

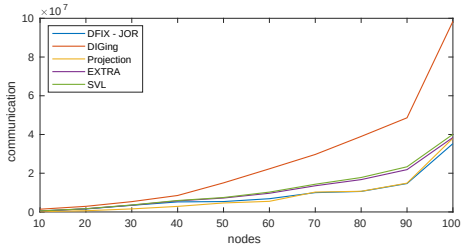
We now consider underlying directed networks. Let $n = 100$ be the size of linear system generated as in the previous tests. We consider a randomly generated directed network of size n such that the average out-degree of the nodes is equal to a fixed m. The consensus matrix W is defined with off-diagonal elements $w_{ij} = 1/(1 + \hat{m}_i)$ if $j \in \mathcal{O}_i$, and $w_{ij} = 0$ otherwise, where $\hat{m}_i$ denotes the out-degree of node i, and the diagonal elements are $w_{ii} = 1 - \sum_{j \neq i} w_{ij}$. The resulting matrix W is row-stochastic. In Figures 5.6 we plot the results for $m = 8, \ldots, 50$ for DFIX, DIGing, EXTRA and Projection. The SVL method fails to converge in this framework. The resulting comparison among the



(a) computational cost



(b) peer-to-peer traffic



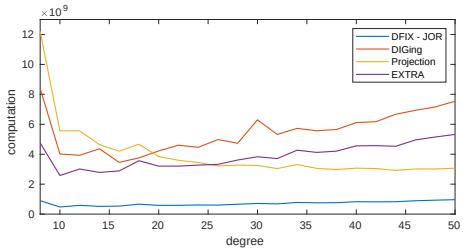
(c) broadcasting traffic

Figure 5.5: Varying number of nodes

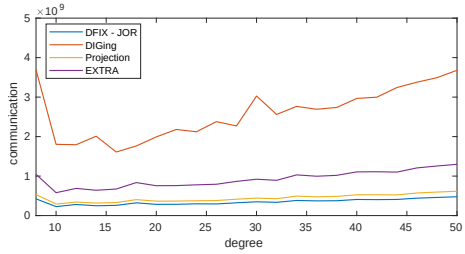
four methods is analogous to the case of undirected networks: DFIX seems to require the smallest computational effort among all methods and a similar communication traffic as Projection.

5.3.4 Time-varying Network

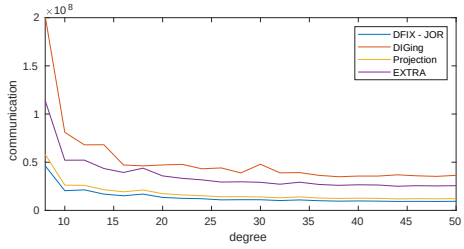
We now compare the performance of the five methods in the time-varying case described in Section 5.2. The sequence $\{\mathcal{G}_k\}$ is generated as follows. For a fixed strongly connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a scalar $\gamma \in (0, 1]$, at every iteration k we randomly generate $\mathcal{E}_k$ by uniformly sampling $\gamma|E|$ edges from $\mathcal{E}$ and we set $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$. This construction can be interpreted as having a fixed underlying graph $\mathcal{G}$



(a) computational cost



(b) peer-to-peer traffic

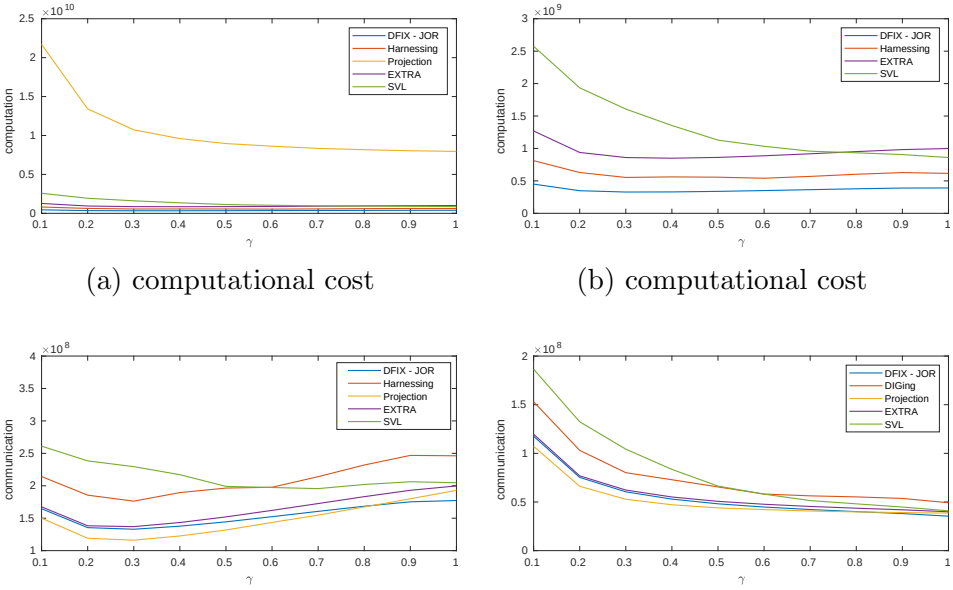


(c) broadcasting traffic

Figure 5.6: Comparison for directed networks

that represents the available communication links among the nodes, and employing at each iteration only a fraction γ of the links. In particular, $\gamma = 1$ corresponds to the case $\mathcal{G}_k = \mathcal{G}$ for every k. As we already remarked, this is equivalent to the time-independent case. The tests we present here compare the communication and computational costs required by the five methods to solve a given linear system using the same sequence of networks $\{\mathcal{G}_k\}$. We generated the linear system as in Section 5.3.2 and chose $\mathcal{G}$ as the undirected m-regular graph with $n = 100$ vertices and degree $m = 8$. The same test is repeated for γ in $\{0.1, 0.2, \ldots, 1\}$. For every k the consensus matrix W^k associated with $\mathcal{G}_k$ is defined as in (2.2), the termination condition and all the parameters of the methods are chosen as in the previous sections.

In Figure 5.7 we plot the results (note that Figure 5.7b repeats the results of Figure 5.7a, excluding Projection method). The computational cost and the communication traffic are calculated as described in Section 5.3.1. DFIX outperforms the three methods for distributed optimization both in terms of computation and communication in this framework. Comparing with Projection, for every value of the parameter γ, the computational cost of DFIX is significantly lower, but it requires a smaller amount of communication only for large values of γ (that is, when each graph $\mathcal{G}_k$ is equal or close to $\mathcal{G}$). Moreover, we can see that for all the methods except for SVL there is an optimal value of $\gamma < 1$, that minimizes the communication traffic, suggesting that using the whole graph $\mathcal{G}$ at every iteration (that is, setting $\gamma = 1$) is inefficient. A similar phenomena happens for DIGing, EXTRA and DFIX also for the computational cost (Figure 5.7b), while we can see in Figures 5.7a and 5.7b that Projection and SVL methods are most efficient when all the available communication links are used at each iterations. For $\gamma < 1$ the networks $\mathcal{G}_k$ are in general not connected, but the joint connectivity of the overall sequence is enough to ensure the convergence of the methods.



(a) computational cost (b) computational cost

(c) peer-to-peer traffic (d) broadcasting traffic

Figure 5.7: Comparison for time-varying networks

Chapter 6

Parallel Inexact Levenberg-Marquardt Method for Network Adjustment Problems

In this chapter we consider a nonlinear least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}), \quad F(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{m} r_j(\mathbf{x})^2 = \frac{1}{2}\|\mathbf{R}(\mathbf{x})\|_2^2. \tag{6.1}$$

where for every $j = 1, \ldots, m$, $r_j : \mathbb{R}^n \to \mathbb{R}$, $\mathbf{R}(\mathbf{x}) = (r_1(\mathbf{x}), \ldots, r_m(\mathbf{x}))^\top \in \mathbb{R}^m$ is the vector of residuals, and F is the aggregated residual function. We assume that the problem is large scale, in the sense that both the dimension n and the number of functions m are large, and nearly separable.

In particular here we are interested in localization problems such as Least Squares Network Adjustment [60], Bundle Adjustment [33] and

Wireless Sensor Network Localization [41], where the variables correspond to the coordinates of physical points in the 2 or 3 dimensional space, residuals correspond to observations of geometrical quantities involving these points and the goal is to find the set of coordinates that minimizes the residuals in the least squares sense. Typical observations are the distance between two points and the angle formed by a set of three points. In problems of these kinds each of the observations usually involves a small number of points. Moreover, when the problem is large and the points are deployed in a large region, each point is typically involved in a small number of observations, compared with the total amount of observations m. That is, for the problems we consider, we have that each residual function r_j only involves a small number of variables, and each variable is involved in a relatively small number of residual functions. A direct consequence of this kind of sparsity, is that these problems are, usually, *nearly-separable*, meaning that it is possible to partition the set of variables in such a way that the number of residual functions involving variables in different subsets is small compared to the number of residuals involving variables in the same subset.

The particular application we focus on is the refinement of cadastral maps and in this case n is prohibitively large for direct application of the LM method. For example, the Dutch Kadaster is pursuing making the cadastral map more accurate by making it consistent with accurate field surveyor measurements [21, 62]. This application yields a nonlinear least squares problem which is known as *least squares adjustment* in the field of geography. If the entire Netherlands were to be considered as one big adjustment problem, the number of variables would be twice the number of feature points in the Netherlands, which is on the order of 1 billion variables and even considering separate parts of the Netherlands still yields a very large-scale problem.

The Levenberg-Marquardt (LM) method is a classical method for the solution of nonlinear least squares problems. As noticed in Section 1.2.3, at each iteration of the LM method, the search direction is computed by solving a linear system of equations of size n equal to the number of variables in the problem. In its original version, LM methods achieves local quadratic convergence, assuming, in particular, that the residual at solution is zero and that the Jacobian matrix is nonsingular in a neighborhood of the solution. Moreover, solving a linear system at each iteration may be impractical for problems of large dimension.

Many modification of the classical Levenberg-Marquardt scheme have been proposed in literature to retain convergence while relaxing the assumptions on the objective function and to improve the performance of the method. In [18, 19, 71] the damping parameter is defined as a multiple of the objective function. With this choice of the parameter local superlinear or quadratic convergence is proved under a local error bound assumption for zero residual problems, while global convergence is achieved by employing a line search strategy. In [31] the authors propose an updating strategy for the parameter that, in combination with Armijo line search, ensures global convergence and q-quadratic local convergence under the same assumption of the previous papers. In [3] the non-zero residual case is considered and a Levenberg-Marquardt scheme is proposed that achieves local convergence with order depending on the rank of the Jacobian matrix and of a combined measure of nonlinearity and residual size around stationary points. In [10] an Inexact Levenberg-Marquardt is considered and local convergence is proved under a local error bound condition. In [4] the authors propose an approximated Levenberg-Marquardt method, suitable for large-scale problems, that relies on inaccurate function values and derivatives.

Given that we are interested in least squares problems of very large

dimension, we propose an Inexact Levenberg-Marquardt method suitable for the solution of these problems in the master/worker framework, which allows us to distribute the data and the computational effort among different nodes, [20]. At each iteration of the proposed method, a search direction is computed by approximately solving the linear system that arises at each iteration of Levenberg-Marquardt method, using a fixed point strategy that relies on the partition of the variables induced by the near-separability property and is suitable for parallel implementation. We prove that the proposed method, combined with a nonmonotone line search strategy, achieves global convergence to a stationary point of the considered problem, while for full step size we prove local convergence with order depending on the choice of the parameters of the method. We see that, aside from near-separability, the required assumptions are the same as for the convergence analysis of classical Levenberg-Marquardt method. A sequential method for the solution of the same kind of problems was proposed in [40].

This chapter is organized as follows. In Section 6.1 we formalize the near-separability assumption, describe the consequent block-partition of the Levenberg-Marquardt equation and present the Parallel Inexact Levenberg-Marquardt Method. In Section 6.2 we carry out the convergence analysis, while in Section 6.3 we present a set of numerical results to investigate the performance of the method.

6.1 Parallel Inexact LM method

The problem we consider is stated in (6.1). Let us define $\mathcal{I} = \{1, \ldots, n\}$ and $\mathcal{J} = \{1, \ldots, m\}$. Given a partition $I_1, \ldots, I_K$ of $\mathcal{I}$ we define the

corresponding partition of $\mathcal{J}$ into $E_1, \ldots, E_K$ as follows:

$$\begin{aligned} &E_s = \{j \in \mathcal{J} | r_j \text{ only depends on variables in } I_s\}, \ s = 1, \ldots, K \\ &\widehat{E} = \mathcal{J} \setminus \bigcup_{i=s}^{K} E_i. \end{aligned} \tag{6.2}$$

That is, given a partition of the set of variables, each of the subsets E_s contains the indices corresponding to residual functions that only involve variables in I_s, while $\widehat{E}$ contains the indices of residuals that involve variables belonging to different subsets I_s. We say that problem (6.1) is *separable* if there exist $K \geq 2$ and a partition $\{I_s\}_{s=1,\ldots,K}$ of $\mathcal{I}$ such that $\widehat{E} = \emptyset$, while we say that it is *nearly-separable* if there exist $K \geq 2$ and a partition $\{I_s\}_{s=1,\ldots,K}$ of $\mathcal{I}$ such that the cardinality of $\widehat{E}$ is small with respect to the cardinality of $\bigcup_{s=1}^{K} E_s$.

Given the partition $\{I_s\}_{s=1}^{K}$ of the variables and the corresponding partition $\{E_s\}_{s=1,\ldots,K}, \ \widehat{E}$ of the residuals, for $s = 1, \ldots, K$ we define $\mathbf{x}_s \in \mathbb{R}^{n_s}$ as the vector of the variables in I_s where n_s denotes the cardinality of I_s, and we introduce the following functions

$$\begin{aligned} \mathbf{R}_s(\mathbf{x}_s) &:= (r_j(\mathbf{x}))_{j \in E_s}, & \rho(\mathbf{x}) &:= (r_j(\mathbf{x}))_{j \in \widehat{E}} \\ F_s(\mathbf{x}_s) &:= \|\mathbf{R}_s(\mathbf{x}_s)\|_2^2 & \Phi(\mathbf{x}) &:= \|\rho(\mathbf{x})\|_2^2 \end{aligned} \tag{6.3}$$

so that, denoting with $|\cdot|$ the cardinality, for every $s = 1, \ldots, K$, $\mathbf{R}_s : \mathbb{R}^{n_s} \to \mathbb{R}^{|E_s|}$ is the vector of residuals involving only variables in I_s, while $\rho : \mathbb{R}^n \to \mathbb{R}^{|\widehat{E}|}$ is the vector of residuals in $\widehat{E}$ and F_s, Φ are the corresponding local aggregated residual functions. Since $\{I_s\}_{s=1}^{K}$ is a partition of $\mathcal{I}$, we have $\sum_{s=1}^{K} n_s = n$. With this notation problem (6.1) can be equivalently written as

$$\min_{\mathbf{x} \in \mathbb{R}^n} \left(\Phi(\mathbf{x}) + \sum_{s=1}^{K} F_s(\mathbf{x}_s) \right) = \min_{\mathbf{x} \in \mathbb{R}^n} \left(\|\rho(\mathbf{x})\|_2^2 + \sum_{s=1}^{K} \|\mathbf{R}_s(\mathbf{x}_s)\|_2^2 \right) \tag{6.4}$$

In particular, if the problem is separable (and therefore $\widehat{E}$ is empty) $\Phi(\mathbf{x}) = 0$ for every $\mathbf{x} \in \mathbb{R}^n$ and solving problem (6.4) is equivalent to solving K independent least squares problems given by

$$\min_{\mathbf{x}_s \in \mathbb{R}^{n_s}} F_s(\mathbf{x}_s) = \min_{\mathbf{x}_s \in \mathbb{R}^{n_s}} \|\mathbf{R}_s(\mathbf{x}_s)\|_2^2 \text{ for } s = 1, \ldots, K. \tag{6.5}$$

If the problem is not separable then in general Φ is not equal to zero and that is the case we are interested in.

Let $\{I_s\}_{s=1,\ldots,K}$ be a partition of $\mathcal{I}$ and $\{E_s\}_{s=1,\ldots,K}$ be the corresponding partition of $\mathcal{J}$ as defined in (6.2). To ease the notation we assume that the vectors $\mathbf{x}$ and $\mathbf{R}$ are reordered according to the given partition. That is, for $\mathbf{x} \in \mathbb{R}^n$

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_K \end{pmatrix}, \qquad \mathbf{R}(\mathbf{x}) = \begin{pmatrix} \mathbf{R}_1(\mathbf{x}_1) \\ \vdots \\ \mathbf{R}_K(\mathbf{x}_K) \\ \rho(\mathbf{x}) \end{pmatrix}$$

With this reordering, denoting with $J_{s\mathbf{R}_j}$ the Jacobian of the partial residual vector $\mathbf{R}_j$ defined in (6.3) with respect to the variables in I_s, and with $J_{s\rho}$ the Jacobian of the partial residual ρ with respect to $\mathbf{x}_s$, we have

$$J(\mathbf{x}) = \begin{pmatrix} J_{1\mathbf{R}_1}(\mathbf{x}_1) & & & 0 \\ & J_{2\mathbf{R}_2}(\mathbf{x}_2) & & \\ & & \ddots & \\ 0 & & & J_{K\mathbf{R}_K}(\mathbf{x}_K) \\ J_{1\rho}(\mathbf{x}) & J_{2\rho}(\mathbf{x}) & \ldots & J_{K\rho}(\mathbf{x}) \end{pmatrix}.$$

Notice that for every $i = 1, \ldots, K$ the block $J_{i\mathbf{R}_i}(\mathbf{x}_i)$ only involves variables in I_i and residual functions in E_i. Therefore, assuming that K nodes are given such that node i holds the portion of the dataset

relative to the functions in E_i, we have that each of the nodes can compute one of the diagonal blocks of the Jacobian. Communication is only needed for the last row of blocks. From this structure of $\mathbf{R}$ and J we get the corresponding block structure of the gradient $\mathbf{g}(\mathbf{x}) = J(\mathbf{x})^\top \mathbf{R}(\mathbf{x})$ and the matrix $J(\mathbf{x})^\top J(\mathbf{x})$:

$$\mathbf{g}(\mathbf{x})^\top = \left(\mathbf{g}_1^\top(\mathbf{x}), \mathbf{g}_2^\top(\mathbf{x}), \ldots, \mathbf{g}_K^\top(\mathbf{x})\right), \tag{6.6}$$

$$J(\mathbf{x})^\top J(\mathbf{x}) = \begin{pmatrix} P_1(\mathbf{x}) & B_{12}(\mathbf{x}) & \ldots & B_{1K}(\mathbf{x}) \\ B_{21}(\mathbf{x}) & P_2(\mathbf{x}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & B_{K-1K}(\mathbf{x}) \\ B_{K1}(\mathbf{x}) & \ldots & B_{KK-1}(\mathbf{x}) & P_K(\mathbf{x}) \end{pmatrix}, \tag{6.7}$$

where, for $s, i, j = 1, \ldots, K$

$$\begin{aligned} \mathbf{g}_s(\mathbf{x}) &= J_{s\mathbf{R}_s}(\mathbf{x}_s)^\top \mathbf{R}_s(\mathbf{x}_s) + J_{s\rho_s}(\mathbf{x})^\top \rho(\mathbf{x}) \in \mathbb{R}^{n_s} \\ P_s(\mathbf{x}) &= J_{s\mathbf{R}_s}(\mathbf{x}_s)^\top J_{s\mathbf{R}_s}(\mathbf{x}_s) + J_{s\rho}(\mathbf{x})^\top J_{s\rho}(\mathbf{x}) \in \mathbb{R}^{n_s \times n_s} \\ B_{ij}(\mathbf{x}) &= J_{i\rho}(\mathbf{x})^\top J_{j\rho}(\mathbf{x}) \in \mathbb{R}^{n_i \times n_j}. \end{aligned} \tag{6.8}$$

The algorithm we introduce here is motivated by the near-separability property. Regarding this, we state the following formal assumption,

Assumption E1. *There exists a constant $C_B > 0$ such that for all $\mathbf{x} \in \mathbb{R}^n$*

$$\|B(\mathbf{x})\| \leq C_B \|J(\mathbf{x})^\top J(\mathbf{x})\|. \tag{6.9}$$

We notice that $B(\mathbf{x})$ is a submatrix of $J(\mathbf{x})^\top J(\mathbf{x})$ and there is no upper bound over the magnitude of C_B, so the assumption above is not restrictive.

Consider a standard iteration of LM method for a given iteration $\mathbf{x}^k$

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{d}^k,$$

where $\mathbf{d}^k \in \mathbb{R}^n$ is the solution of

$$\left((J^k)^\top J^k + \mu_k I\right) \mathbf{d}^k = -(J^k)^\top \mathbf{R}^k, \tag{6.10}$$

where $J^k = J(\mathbf{x}^k) \in \mathbb{R}^{m\times n}$ denotes the Jacobian matrix of $R^k = R(\mathbf{x}^k)$ and $\mu_k > 0$ is a scalar. When n is very large solving (6.10) at each iteration of the method may be prohibitively expensive. In the following we propose an Inexact Levenberg-Marquardt method that relies on the near-separability of the problem to define a fixed-point iteration for the solution of the linear system (6.10). Such a method is suitable for the server/worker framework: if each worker holds the data relative to one of the subsets E_s, the fixed point iterations can be efficiently carried out in parallel, with moderate communication traffic.

The linear system (6.10) at iteration k can be rewritten as

$$(P^k + \mu_k I + B^k)\mathbf{d}^k = -\mathbf{g}^k \tag{6.11}$$

where $\mathbf{g}^k = (J^k)^\top \mathbf{R}^k$ is the vector with s-th block component equal to $g_s(\mathbf{x}^k)$, $P^k = P(\mathbf{x}^k)$ is the block diagonal matrix with diagonal blocks given by $P_s(\mathbf{x}^k)$ for $s = 1, \ldots, K$, $B^k = B(\mathbf{x}^k)$ is the block partitioned matrix with diagonal blocks equal to zero and off-diagonal blocks equal to $B_{ij}(\mathbf{x}^k)$, namely

$$\begin{aligned} P^k &= \begin{pmatrix} P_1(\mathbf{x}^k) & & & \\ & P_2(\mathbf{x}^k) & & \\ & & \ddots & \\ & & & P_K(\mathbf{x}^k) \end{pmatrix}, \\ B^k &= \begin{pmatrix} 0 & B_{12}(\mathbf{x}^k) & \ldots & B_{1K}(\mathbf{x}^k) \\ B_{21}(\mathbf{x}^k) & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & B_{K-1K}(\mathbf{x}^k) \\ B_{K1}(\mathbf{x}^k) & \ldots & B_{KK-1}(\mathbf{x}^k) & 0 \end{pmatrix}. \end{aligned} \tag{6.12}$$

Consider the sequence $\{\mathbf{y}^l\}$ generated as follows

$$\begin{cases} \mathbf{y}^1 = -(P^k + \mu_k I)^{-1}\mathbf{g}^k \\ \mathbf{y}^{l+1} = -(P^k + \mu_k I)^{-1}\left(\mathbf{g}^k + B^k\mathbf{y}^l\right) \quad l \geq 1. \end{cases} \tag{6.13}$$

The equations above define a fixed-point method for the solution of (6.11). From the theory of fixed point methods (Theorem 1.1), we know that if

$$\|(P^k + \mu_k I)^{-1}B^k\| < 1$$

then the sequence $\{\mathbf{y}^l\}$ converges to the solution $\mathbf{d}^k$ of (6.11). Moreover, denoting with $\mathbf{r}_k^l$ the residual in the linear system at the l-th inner iteration, namely

$$\mathbf{r}_k^l = ((J^k)^\top J^k + \mu_k I)\mathbf{y}^l + \mathbf{g}^k,$$

for every $l \in \mathbb{N}$ we have the following

$$\begin{aligned} \|\mathbf{r}_k^{l+1}\| &= \|((J^k)^\top J^k + \mu_k I)\mathbf{y}^{l+1} + \mathbf{g}^k\| \\ &= \| - (P^k + B^k + \mu_k I)(P^k + \mu_k I)^{-1}\left(\mathbf{g}^k + B^k\mathbf{y}^l\right) + \mathbf{g}^k\| \\ &= \|B^k\mathbf{y}^l + B^k(P^k + \mu_k I)^{-1}\left(\mathbf{g}^k + B^k\mathbf{y}^l\right)\| \\ &= \|B^k(P^k + \mu_k I)^{-1}\left((P^k + \mu_k I)\mathbf{y}^l + \mathbf{g}^k + B^k\mathbf{y}^l\right)\| \\ &= \|B^k(P^k + \mu_k I)^{-1}\left(((J^k)^\top J^k + \mu_k I)\mathbf{y}^l + \mathbf{g}^k\right)\| \\ &\leq \|B^k(P^k + \mu_k I)^{-1}\|\|\mathbf{r}_k^l\| = \rho_k\|\mathbf{r}_k^l\| \end{aligned} \tag{6.14}$$

where we defined $\rho_k = \|B^k(P^k + \mu_k I)^{-1}\|$.

In the following, we use the fixed point iteration in (6.13) to define an inexact LM method. More details regarding the implementation of the algorithm in the server/worker framework will be discussed in Section 6.3, together with the numerical results.

Algorithm 6.1 (PILM).
Parameters: $c > 0, \{\ell_k\}_{k=0}^{\infty} \in \mathbb{N},\ \{\varepsilon_k\}_{k=0}^{\infty} \in \mathbb{R}_{>0}$
Iteration k***:***

1: *compute* $\mathbf{R}^k$, J^k, P^k, B^k
2: *choose* μ_k
3: ***for*** $i = 1, \ldots, K$ ***do***
4: *compute* $\mathbf{y}_i^1$ *such that* $\left(P_i^k + \mu_k I_{n_i}\right) \mathbf{y}_i^1 = -\mathbf{g}_i^k$
5: ***end for***
6: ***for*** $l = 1, \ldots, \ell_k - 1$ ***do***
7: ***for*** $i = 1, \ldots, K$ ***do***
8: *compute* $\mathbf{y}_i^{l+1}$ *such that* $\left(P_i^k + \mu_k I_{n_i}\right) \mathbf{y}_i^{l+1} = -\left(\mathbf{g}_i^k + \sum_{j=1}^{K} B_{ij} \mathbf{y}_i^l\right)$
9: ***end for***
10: ***end for***
11: *set* $\mathbf{d}^k = (\mathbf{y}_1^{\ell_k}, \ldots, \mathbf{y}_K^{\ell_k})^\top$
12: *use backtracking to find the largest positive* $\alpha_k \leq 1$ *such that*

$$F(\mathbf{x}^k + \alpha_k \mathbf{d}^k) \leq F(\mathbf{x}^k) - c\alpha_k^2 \|\mathbf{g}^k\|^2 + \varepsilon_k \tag{6.15}$$

13: *set* $\mathbf{x}^k = \mathbf{x}^k + \alpha_k \mathbf{d}^k$

Remark 6.1. *Let us consider the line search condition* (6.15)*. For* α_k *that tends to zero, the term on the left-hand side tends to* $F(\mathbf{x}^k)$*, while the negative term in the right-hand side tends to zero. Since we assume that* $\varepsilon_k > 0$*, one can always find* $\alpha_k > 0$ *such that the line search condition* (6.15) *is satisfied. Note that this argument holds even in case the direction* $\mathbf{d}^k$ *is not a descent direction for* F *at* $\mathbf{x}^k$*. In particular, Algorithm PILM is well defined.*

Remark 6.2. *The* K *linear systems in line 4 are independent. In particular the rounds of the* for *loop in lines 3-5 can be executed in a parallel fashion. The same holds also for the* for *loop at lines 7-9.*

6.2 Convergence Analysis

The following assumptions are regularity assumptions commonly used in LM methods

Assumption E2. *The vector of residuals* $\mathbf{R} : \mathbb{R}^n \to \mathbb{R}^m$ *is continuously differentiable.*

Assumption E3. *The Jacobian matrix* $J \in \mathbb{R}^{m\times n}$ *of* $\mathbf{R}$ *is L-Lipschitz continuous. That is, for every* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

$$\|J(\mathbf{x}) - J(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|.$$

6.2.1 Global Convergence

Lemma 6.1. *Assume that* $\mathbf{d}^k$ *is computed as in algorithm PILM for a given* $\ell_k \in \mathbb{N}$*. The following inequalities hold*

i) for every $k \in \mathbb{N}_0$

$$\rho_k \leq \frac{\|B^k\|}{\mu_k}$$

ii) for every $k \in \mathbb{N}_0$

$$\|\mathbf{r}_k^{\ell_k}\| \leq \rho_k^{\ell_k}\|\mathbf{g}^k\|$$

iii) for every $k \in \mathbb{N}_0$

$$\|\mathbf{d}^k\| \leq \frac{(1+\rho_k^{\ell_k})}{\mu_k}\|\mathbf{g}^k\|$$

iv) for every $k \in \mathbb{N}_0$

$$(\mathbf{d}^k)^\top \mathbf{g}^k \leq \left(\frac{\rho_k^{\ell_k}}{\mu_k} - \frac{1}{\|J^k\|^2 + \mu_k}\right)\|\mathbf{g}^k\|^2$$

v) if μ_k in line 2 is chosen as $\mu_k = \max\{\mu_{\min}, C_\mu \|\mathbf{B}^k\|\}$ for some $\mu_{\min} > 0$ and $C_\mu > 1$, then

$$\|\mathbf{r}_k^{\ell_k}\| \leq \left(\frac{1}{C_\mu}\right)^{\ell_k} \|\mathbf{g}^k\|$$

Proof. By sub-multiplicativity of the norm, we have

$$\rho_k = \|B^k(P^k + \mu_k I)^{-1}\| \leq \|B^k\|\|(P^k + \mu_k I)^{-1}\| \leq \frac{\|B^k\|}{\mu_k} \tag{6.16}$$

which is *i*). Using the definition of $\mathbf{y}^1$ in line 3 of Algorithm ILM we have

$$\begin{aligned} \|\mathbf{r}_k^1\| &= \| - (P^k + B^k + \mu_k I)(P^k + \mu_k I)^{-1}\mathbf{g}^k + \mathbf{g}^k\| = \\ &= \|B^k(P^k + \mu_k I)^{-1}\|\|\mathbf{g}^k\| = \rho_k \|\mathbf{g}^k\| \end{aligned} \tag{6.17}$$

This proves part *ii*) of the thesis in case $\ell_k = 1$. If $\ell_k > 1$, recursively applying (6.14), and using the equality above, we get

$$\|\mathbf{r}_k^{\ell_k}\| \leq \rho_k \|\mathbf{r}_k^{\ell_k - 1}\| \leq \rho_k^{\ell_k - 1}\|\mathbf{r}_k^1\| = \rho_k^{\ell_k}\|\mathbf{g}^k\|,$$

and *ii*) is proved. By definition of $\mathbf{d}^k$ and $\mathbf{r}_k^{\ell_k}$ we have

$$\mathbf{d}^k = ((J^k)^\top J^k + \mu_k I)^{-1}(-\mathbf{g}^k + \mathbf{r}_k^{\ell_k}) \tag{6.18}$$

Taking the norm, we have

$$\begin{aligned} \|\mathbf{d}^k\| &= \left\|((J^k)^\top J^k + \mu_k I)^{-1}\left(\mathbf{g}^k + \mathbf{r}_k^{\ell_k}\right)\right\| \\ &\leq \|((J^k)^\top J^k + \mu_k I)^{-1}\|\left(\|\mathbf{g}^k\| + \|\mathbf{r}_k^{\ell_k}\|\right) \\ &\leq \frac{(1 + \rho_k^{\ell_k})}{\mu_k}\|\mathbf{g}^k\|, \end{aligned}$$

that is *ii*).
By (6.18), Cauchy-Schwartz inequality and *i*), we have

$$\begin{aligned}(\mathbf{d}^k)^\top \mathbf{g}^k &= (\mathbf{g}^k)^\top ((J^k)^\top J^k + \mu_k)^{-1} \left(-\mathbf{g}^k + \mathbf{r}_k^{\ell_k}\right) \\ &\leq \lambda_{\max}\left((J^k)^\top J^k + \mu_k)^{-1}\right) \|\mathbf{r}^k\| \|\mathbf{g}^k\| \\ &- \lambda_{\min}\left((J^k)^\top J^k + \mu_k)^{-1}\right) \|\mathbf{g}^k\|^2 \\ &\leq \frac{1}{\mu_k} \rho_k^{\ell_k} \|\mathbf{g}^k\|^2 - \frac{1}{\|J^k\|^2 + \mu_k} \|\mathbf{g}^k\|^2 \\ &\leq \left(\frac{\rho_k^{\ell_k}}{\mu_k} - \frac{1}{\|J^k\|^2 + \mu_k}\right) \|\mathbf{g}^k\|^2,\end{aligned}$$

and we have part *iii*) of the statement.
To prove *iv*) it is enough to notice that if $\mu_k \geq C_\mu \|B^k\|$ we have

$$\frac{\|B^k\|}{\mu_k} \leq \frac{1}{C_\mu},$$

and thus *iv*) follows directly from *i*). □

Theorem 6.1. *Assume that Assumptions E2 and E3 hold, $\ell_k \geq \ell$ for every k, $\{\varepsilon_k\}$ is such that $\sum_{k=0}^{\infty} \varepsilon_k < +\infty$, and that in line 2 μ_k is chosen as $\mu_k = \max\{\mu_{\min}, C_\mu \|\mathbf{B}^k\|\}$. Then for ℓ large enough we have that for every $\mathbf{x}_0 \in \mathbb{R}^n$, each accumulation point of the sequence $\{\mathbf{x}^k\}_{k=1}^{\infty}$ is a stationary point of $\mathbf{F}(\mathbf{x})$*

Proof. Applying recursively the line search condition (6.15) we have that for every $k \in \mathbb{N}_0$

$$\begin{aligned}F(\mathbf{x}^{k+1}) &\leq F(\mathbf{x}^k) - c\alpha_k^2 \|\mathbf{g}^k\|^2 + \varepsilon_k \\ &\leq F(\mathbf{x}^0) - c \sum_{j=0}^{k} \alpha_j^2 \|\mathbf{g}^j\|^2 + \sum_{j=0}^{k} \varepsilon_j.\end{aligned} \tag{6.19}$$

Reordering inequality (6.19) and taking the limit for $k \to +\infty$ we get

$$\sum_{k=0}^{+\infty} \alpha_k^2 \|\mathbf{g}^k\|^2 \leq F(\mathbf{x}^0) + \sum_{k=0}^{\infty} \varepsilon_k < +\infty$$

which implies that

$$\lim_{k \to +\infty} \alpha_k \|\mathbf{g}^k\| = 0.$$

Let us consider $\mathbf{x}^* \in \mathbb{R}^n$ any accumulation point of the sequence $\{\mathbf{x}^k\}$. By definition of accumulation point, there exists an infinite subset of indices $\mathcal{K}_0 \subseteq \mathbb{N}_0$ such that the subsequence $\{\mathbf{x}^k\}_{k \in \mathcal{K}_0}$ converges to $\mathbf{x}^*$. The limit above implies

$$\lim_{k \in \mathcal{K}_0} \alpha_k \|\mathbf{g}^k\| = 0.$$

If there exists $\alpha > 0$ such that $\alpha_k \geq \alpha$ for every index $k \in \mathcal{K}_0$, then, by continuity of the gradient $\mathbf{g}$,

$$0 = \lim_{k \in \mathcal{K}_0} \alpha_k \|\mathbf{g}^k\| \geq \alpha \lim_{k \in \mathcal{K}_0} \|\mathbf{g}^k\| = \alpha \|\mathbf{g}(\mathbf{x}^*)\|$$

and therefore $\mathbf{x}^*$ is a stationary point of F. If such $\alpha > 0$ does not exist, then one can find $\mathcal{K}_1 \subseteq \mathcal{K}_0$ infinite set of indices such that $\lim_{k \in \mathcal{K}_1} \alpha_k = 0$ and $\alpha_k < 1$ for every $k \in \mathcal{K}_1$. In particular this implies that for every $k \in \mathcal{K}_1$ there exists $1 \geq \hat{\alpha}_k > \alpha_k$ such that condition (6.15) does not hold. That is, for every $k \in \mathcal{K}_1$

$$F(\mathbf{x}^k + \hat{\alpha}_k \mathbf{d}^k) > -c\hat{\alpha}_k^2 \|\mathbf{g}^k\|^2 + \varepsilon_k + F(\mathbf{x}^k) \geq -c\hat{\alpha}_k^2 \|\mathbf{g}^k\|^2 + \varepsilon_k.$$

Since $\varepsilon_k \geq 0$, reordering the inequality above and applying the mean value theorem we have, for some $s_k \in [0, 1]$

$$-c\hat{\alpha}_k \|\mathbf{g}^k\|^2 < \frac{1}{\hat{\alpha}_k} \left(F(\mathbf{x}^k + \hat{\alpha}_k \mathbf{d}^k) - F(\mathbf{x}^k) \right) = \mathbf{g}(\mathbf{x}^k + s_k \hat{\alpha}^k \mathbf{d}^k)^\top \mathbf{d}^k. \tag{6.20}$$

Let us now consider $\mathbf{d}^k$. By part *ii*) of Lemma 6.1, the definition of μ_k, and the fact that $C_\mu > 1$, we have

$$\|\mathbf{d}^k\| \leq \frac{1+\rho_k^{\ell_k}}{\mu_k}\|\mathbf{g}^k\| \leq \frac{2}{\mu_k}\max_{k\in\mathcal{K}_1}\|\mathbf{g}^k\|$$

where the maximum in the last term of the inequality exists because $\{\mathbf{x}^k\}_{k\in\mathcal{K}_1}$is a compact subset of $\mathbb{R}^n$ and the gradient is continuous. Since the maximum is finite and $\mu_k \geq \mu_{\min} > 0$ we have that $\{\mathbf{d}^k\}_{k\in\mathcal{K}_1}$ is a bounded subsequence of $\mathbb{R}^n$ and therefore it has an accumulation point $\mathbf{d}^*$. That is

$$\lim_{k\in\mathcal{K}_2}\mathbf{d}^k = \mathbf{d}^*$$

for some $\mathcal{K}_2 \subseteq \mathcal{K}_1$ infinite subset.
Since $\lim_{k\in\mathcal{K}_2}\alpha_k = 0$, by definition of $\hat{\alpha}_k$ we have $\lim_{k\in\mathcal{K}_2}\hat{\alpha}_k = 0$ and thus $\lim_{k\in\mathcal{K}_2}\mathbf{x}^k + s_k\hat{\alpha}_k\mathbf{d}^k = \mathbf{x}^*$, which in turn implies

$$\lim_{k\in\mathcal{K}_2}\mathbf{g}(\mathbf{x}^k + s_k\hat{\alpha}_k\mathbf{d}^k)^\top\mathbf{d}^k = (\mathbf{g}^*)^\top\mathbf{d}^*.$$

Adding and subtracting $(\mathbf{d}^k)^\top\mathbf{g}^k$ in the right-hand side of (6.20) and taking the limit for $k \in \mathcal{K}_2$, we then get

$$0 \leq \lim_{k\in\mathcal{K}_2}(\mathbf{d}^k)^\top\mathbf{g}^k = (\mathbf{d}^*)^\top\mathbf{g}^*. \tag{6.21}$$

On the other hand, by Lemma 6.1 *iii*) and the definition of μ_k we have that

$$\begin{aligned}(\mathbf{d}^*)^\top\mathbf{g}^* = \lim_{k\in\mathcal{K}_2}(\mathbf{d}^k)^\top\mathbf{g}^k &\leq \lim_{k\in\mathcal{K}_2}\left(\frac{\rho_k^{\ell_k}}{\mu_k} - \frac{1}{\|J^k\|^2+\mu_k}\right)\|\mathbf{g}^k\|^2 \\ &\leq \lim_{k\in\mathcal{K}_2}\left(\frac{\rho_k^{\ell_k}}{\mu_{\min}} - \frac{1}{\max\limits_{k\in\mathcal{K}_2}\|J^k\|^2+\mu_{\max}}\right)\|\mathbf{g}^k\|^2.\end{aligned} \tag{6.22}$$

where $\mu_{\max} = C_\mu \max_{k\in\mathcal{K}_2} \|B^k\|$ and the maxima exist by compactness of $\{\mathbf{x}^k\}_{k\in\mathcal{K}_2}$ and continuity of $J(\mathbf{x})$ and $B(\mathbf{x})$.

If $\lim_{k\in\mathcal{K}_2} \|\mathbf{g}^k\| = 0$ then, by uniqueness of the limit, we have that

$$\|\mathbf{g}^*\| = \lim_{k\in\mathcal{K}_1} \|\mathbf{g}^k\| = \lim_{k\in\mathcal{K}_2} \|\mathbf{g}^k\| = 0$$

and therefore $\mathbf{x}^*$ is a stationary point of F. Otherwise, we proceed by contradiction. If $\|\mathbf{g}^k\|$ does not vanish for $k \in \mathcal{K}_2$, there exist $\gamma > 0$ and $\mathcal{K}_3 \subseteq \mathcal{K}_2$ infinite sequence such that $\|\mathbf{g}^k\| \geq \gamma$ for every $k \in \mathcal{K}_3$. Fix $\nu > 0$. From Lemma 6.1 and we have that $\rho_k^{\ell_k} \leq C_\mu^{-\ell_k}$. Since $C_\mu > 1$ and $\ell_k \geq \ell$ for every k, one can find ℓ large enough such that

$$\left(\frac{\rho_k^{\ell_k}}{\mu_{\min}} - \frac{1}{\max_{k\in\mathcal{K}_2} \|J^k\|^2 + \mu_{\max}} \right) \leq -\nu.$$

For this choice of ℓ, from (6.22) we have, for every $k \in \mathcal{K}_3$

$$(\mathbf{d}^*)^\top \mathbf{g}^* \leq -\nu\gamma^2.$$

This contradicts (6.21) and therefore concludes the proof. □

6.2.2 Local Convergence

Let S denote the set of all stationary points of $\|R(\mathbf{x})\|^2$, namely $S = \{\mathbf{x} \in \mathbb{R}^N | J(\mathbf{x})^\top R(\mathbf{x}) = 0\}$. Consider a stationary point $\mathbf{x}^* \in S$ and a ball $B_r := B(\mathbf{x}^*, r)$ with radius $r \in (0, 1)$ around it. From now on, given a point $\mathbf{x} \in \mathbb{R}^n$ we denote with $\bar{\mathbf{x}}$ a point in S that minimizes the distance from $\mathbf{x}$. That is,

$$\|\mathbf{x} - \bar{\mathbf{x}}\| = \min_{z\in S} \|\mathbf{x} - \mathbf{z}\| = dist(\mathbf{x}, S).$$

Since B_r is a bounded subset of $\mathbb{R}^n$, and $\mathbf{R}$ and J are continuous functions on $\mathbb{R}^n$, we have that there exist $R_{\max}, L_2 \geq 0$ such that

for every $\mathbf{x} \in B_r$ $\|\mathbf{R}(\mathbf{x})\| \leq R_{\max}$ and $\|J(\mathbf{x})\| \leq L_2$. The following Lemma includes a set of inequalities, proved in [3] that are a direct consequence of assumptions E2, E3 on the bounded subset B_r.

Lemma 6.2. *[3]If Assumptions E2-E3 hold, then for every* $\mathbf{x}, \mathbf{y} \in B_r$

i) $\|\mathbf{R}(x) - \mathbf{R}(y) - J(\mathbf{y})(\mathbf{x} - \mathbf{y})\| \leq \frac{1}{2} L \|\mathbf{x} - \mathbf{y}\|^2$

ii) $\|\mathbf{R}(\mathbf{x}) - \mathbf{R}(\mathbf{y})\| \leq L_2 \|\mathbf{x} - \mathbf{y}\|$

iii) $\|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})\| \leq L_3 \|\mathbf{x} - \mathbf{y}\|$ *with* $L_3 = L_2^2 + L_1 R_{\max}$

iv) denoting with $L_4 = \frac{1}{2} L_1 L_2$

$$\|\mathbf{g}(\mathbf{y}) - \mathbf{g}(\mathbf{x}) - J(\mathbf{x})^\top J(\mathbf{x})(\mathbf{y} - \mathbf{x})\| \leq L_4 \|\mathbf{x} - \mathbf{y}\|^2 + \|(J(\mathbf{x}) - J(\mathbf{y}))^\top \mathbf{R}(\mathbf{y})\|$$

v) for every $\bar{\mathbf{z}} \in B_r \cap S$

$$\begin{aligned} \|(J(\mathbf{x}) - J(\mathbf{y}))^\top \mathbf{R}(\mathbf{y})\| \leq\ & L_1 L_2 \|\mathbf{x} - \bar{\mathbf{z}}\| \|\mathbf{y} - \bar{\mathbf{z}}\| + L_1 L_2 \|\mathbf{y} - \bar{\mathbf{z}}\|^2 \\ & + \|J(\mathbf{x})^\top \mathbf{R}(\bar{\mathbf{z}})\| + \|J(\mathbf{y})^\top \mathbf{R}(\bar{\mathbf{z}})\| \end{aligned}$$

In the rest of the section we make the following additional assumptions, which are standard for the local convergence analysis of Levenberg-Marquardt method. In particular, Assumption E4, referred to in the literature as local error bound condition, is typically used in place of the nonsingularity of the Jacobian matrix, while Assumption E5 is common for non-zero residual problems.

Assumption E4. *There exists* $\omega > 0$ *such that for every* $\mathbf{x} \in B(\mathbf{x}^*, r)$

$$\omega dist(\mathbf{x}, S) \leq \|J(\mathbf{x})^\top R(\mathbf{x})\|$$

Assumption E5. *There exist* $\sigma, \delta > 0$ *such that for every* $\mathbf{x} \in B(\mathbf{x}^*, r)$ *and every* $\bar{\mathbf{z}} \in B(\mathbf{x}^*, r) \cap S$

$$\|(J(\mathbf{x}) - J(\bar{\mathbf{z}}))^\top R(\bar{\mathbf{z}})\| \leq \sigma \|\mathbf{x} - \bar{\mathbf{z}}\|^{1+\delta}.$$

Notice that since $\bar{\mathbf{z}}$ is a stationary point of $\|R(\mathbf{x})\|^2$, the inequality above is equivalent to

$$\|J(\mathbf{x})^\top R(\bar{\mathbf{z}})\| \leq \sigma \|\mathbf{x} - \bar{\mathbf{z}}\|^{1+\delta}.$$

Lemma 6.3. *Let us assume that E2-E5 hold and that $\{\mathbf{x}^k\}$ is the sequence generated by Algorithm PILM with $\alpha_k = 1$ for every k. Moreover, let us assume that $\mathbf{x}^k, \mathbf{x}^{k+1} \in B_r$ and $\|\mathbf{d}^k\| \leq c_1 dist(\mathbf{x}^k, S)$ for some constant $c_1 \geq 0$. Then the following inequality holds with $c_2 = L_4 c_1^2 + L_1 L_2 (1 + c_1) + L_1 L_2 (1 + c_1)^2$*

$$\begin{aligned}\omega \|\mathbf{x}^{k+1} - \bar{\mathbf{x}}^{k+1}\| \leq c_2 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2 + \left(\sigma + \sigma(1+c_1)^{1+\delta}\right) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta} \\ + (L_3 \rho_k^{\ell_k} + c_1 \mu_k) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|\end{aligned}$$

Proof. By the triangular inequality, the assumptions of the Lemma, and the definition of $\bar{\mathbf{x}}^k$, we have the following inequalities

$$\begin{aligned}&\|\mathbf{x}^{k+1} - \bar{\mathbf{x}}^k\| \leq \|\mathbf{d}^k\| + \|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \leq (1 + c_1) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \\ &\|(J^k)^\top \mathbf{R}(\bar{\mathbf{x}}^k)\| \leq \sigma \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta} \\ &\|(J^{k+1})^\top \mathbf{R}(\bar{\mathbf{x}}^k)\| \leq \sigma \|\mathbf{x}^{k+1} - \bar{\mathbf{x}}^k\|^{1+\delta} \leq \sigma (1 + c_1)^{1+\delta} \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta}.\end{aligned}$$

From part *iv*) and *v*) of Lemma 6.2, using the inequalities above, we have

$$\begin{aligned}&\|\mathbf{g}^{k+1} - \mathbf{g}^k - (J^k)^\top J^k (\mathbf{x}^{k+1} - \mathbf{x}^k)\| \\ &\leq L_4 \|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2 + \|(J^k - J^{k+1})^\top \mathbf{R}^{k+1}\| \\ &\leq L_4 \|\mathbf{d}^k\|^2 + L_1 L_2 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \|\mathbf{x}^{k+1} - \bar{\mathbf{x}}^k\| + L_1 L_2 \|\mathbf{x}^{k+1} - \bar{\mathbf{x}}^k\|^2 \\ &+ \|(J^k)^\top \mathbf{R}(\bar{\mathbf{x}}^k)\| + \|(J^{k+1})^\top \mathbf{R}(\bar{\mathbf{x}}^k)\| \\ &\leq c_2 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2 + \left(\sigma + \sigma(1+c_1)^{1+\delta}\right) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta}.\end{aligned} \tag{6.23}$$

From part *iii*) in Lemma 6.2, using the fact that

$$\left((J^k)^\top J^k + \mu_k I\right) \mathbf{d}^k = -\mathbf{g}^k + \mathbf{r}^k,$$

that $\bar{\mathbf{x}}^k$ is a stationary point of F, and the assumption over $\mathbf{d}^k$, we get

$$\begin{aligned}
&\|\mathbf{g}^k + (J^k)^\top J^k \mathbf{d}^k\| \leq \|\mathbf{g}^k + \left((J^k)^\top J^k + \mu_k I\right) \mathbf{d}^k\| + \mu_k \|\mathbf{d}^k\| \\
&\leq \|\mathbf{r}^k\| + \mu_k \|\mathbf{d}^k\| \leq \rho_k^{\ell_k} \|\mathbf{g}^k\| + \mu_k c_1 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \\
&\leq \rho_k^{\ell_k} \|\mathbf{g}^k - \mathbf{g}(\bar{\mathbf{x}}^k)\| + \mu_k c_1 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \leq \left(L_3 \rho_k^{\ell_k} + c_1 \mu_k\right) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|.
\end{aligned} \tag{6.24}$$

By Assumption E4, adding and subtracting $\mathbf{g}^k + (J^k)^\top J^k \mathbf{d}^k$ we have

$$\omega dist(\mathbf{x}^{k+1}, S) \leq \|\mathbf{g}^{k+1}\| \leq \|\mathbf{g}^{k+1} - \mathbf{g}^k - (J^k)^\top J^k \mathbf{d}^k\| + \|\mathbf{g}^k + (J^k)^\top J^k \mathbf{d}^k\|.$$

Replacing the two terms of the right-hand side with the bounds found in (6.23) and (6.24), we get the thesis. □

Lemma 6.4. *If Assumptions E2-E5 hold and $\{\mathbf{x}^k\}$ is the sequence generated by Algorithm PILM with $\alpha_k = 1$, $\mu_k = \max\{\mu_{\min}, C_\mu \|B^k\|\}$ and $\ell_k \geq \ell$ for a given $\ell \in \mathbb{N}$ then, if $\{\mathbf{x}^k\} \subset B_r$, there exists $c_1 > 0$ such that for every iteration index k*

$$\|\mathbf{d}^k\| \leq c_1 dist(\mathbf{x}^k, S).$$

Proof. Let us denote with p the rank of $J(\mathbf{x}^*)^\top J(\mathbf{x}^*)$ and let $\{\lambda_i^*\}_{i=1}^n = \text{eig}(J(\mathbf{x}^*)^\top J(\mathbf{x}^*))$, in nonincreasing order. For a given iteration index k, let us consider the eigendecomposition of $(J^k)^\top J^k$

$$(J^k)^\top J^k = \left(Q_1^k, Q_2^k\right) \begin{pmatrix} \Lambda_1^k & \\ & \Lambda_2^k \end{pmatrix} \left(Q_1^k, Q_2^k\right)^\top \tag{6.25}$$

with $\Lambda_1^k = \text{diag}(\lambda_1^k, \dots \lambda_p^k) \in \mathbb{R}^{p \times p}$ and $\Lambda_2^k = \text{diag}(\lambda_{p+1}^k, \dots \lambda_n^k) \in \mathbb{R}^{(n-p)\times(n-p)}$, where $\{\lambda_i^k\}_{i=1}^n = \text{eig}(J(\mathbf{x}^k)^\top J(\mathbf{x}^k))$ again in nonincreasing order, and $Q_1^k \in \mathbb{R}^{n \times p}$, $Q_2^k \in \mathbb{R}^{n \times (n-p)}$. By continuity of $J(\mathbf{x})$ and of the eigenvalues over the entries of the matrix, we have that for r small enough $\min_{i=1:p} \lambda_i^k \geq \lambda_p^*/2$.

By (6.25) we have, for $i = 1, 2$

$$(Q_i^k)^\top(\mathbf{r}^k - \mathbf{g}^k) = (Q_i^k)^\top \left((J^k)^\top J^k + \mu_k I\right) \mathbf{d}^k = \left(\Lambda_i^k + \mu_k I\right) (Q_i^k)^\top \mathbf{d}^k.$$

For $i = 1$, by definition of $\rho_k^{\ell_k}$ and the bound on λ_p^k we have

$$\begin{aligned} &\|(Q_1^k)^\top \mathbf{d}^k\| \le \| \left(\Lambda_1^k + \mu_k I\right)^{-1} (Q_1^k)^\top(-\mathbf{g}^k + \mathbf{r}^k)\| \\ &\le \frac{1}{\lambda_{\min}(\Lambda_1^k + \mu_k)} \| - \mathbf{g}^k + \mathbf{r}^k\| \\ &\le \frac{2}{\lambda_p^*}(1 + \rho_k^{\ell_k})\|\mathbf{g}^k\| \le \frac{4L_3}{\lambda_p^*}\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|. \end{aligned} \tag{6.26}$$

For $i = 2$, by Lemma 6.1, Lemma 6.2, Assumption E5, and the fact that $\| \left(\Lambda_2^k + \mu_k I\right)^{-1} \Lambda_2^k\| \le 1$, we have

$$\begin{aligned} &\|(Q_2^k)^\top \mathbf{d}^k\| \le \| \left(\Lambda_2^k + \mu_k I\right)^{-1} (Q_2^k)^\top(-\mathbf{g}^k + \mathbf{r}^k)\| \\ &\le \| \left(\Lambda_2^k + \mu_k I\right)^{-1} (Q_2^k)^\top(\mathbf{g}^k - \mathbf{g}(\bar{\mathbf{x}}^k) - (J^k)^\top J^k(\mathbf{x}^k - \bar{\mathbf{x}}^k))\| \\ &+ \| \left(\Lambda_2^k + \mu_k I\right)^{-1} (Q_2^k)^\top (J^k)^\top J^k(\mathbf{x}^k - \bar{\mathbf{x}}^k)\| + \frac{1}{\mu_k}\|\mathbf{r}^k\| \\ &\le \frac{1}{\mu_k}\|\mathbf{g}^k - \mathbf{g}(\bar{\mathbf{x}}^k) - (J^k)^\top J^k(\mathbf{x}^k - \bar{\mathbf{x}}^k)\| \\ &+ \| \left(\Lambda_2^k + \mu_k I\right)^{-1} \Lambda_2^k (Q_2^k)^\top(\mathbf{x}^k - \bar{\mathbf{x}}^k)\| + \frac{\rho_k^{\ell_k}}{\mu_k}\|\mathbf{g}^k\| \\ &\le \frac{1}{\mu_k}\left(L_4\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2 + \|(J^k)^\top \mathbf{R}(\bar{\mathbf{x}}^k)\|\right) + \left(1 + \frac{\rho_k^{\ell_k} L_3}{\mu_k}\right)\|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \\ &\le \frac{L_4}{\mu_k}\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2 + \frac{\sigma}{\mu_k}\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta} + \left(1 + \frac{\rho_k^{\ell_k} L_3}{\mu_k}\right)\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|. \end{aligned} \tag{6.27}$$

By assumption we have $\mu_k \ge \mu_{\min}$ and $\rho_k^{\ell_k} \le \rho_k^{\ell} \le C_\mu^{-\ell}$. Therefore,

proceeding in the previous chain of inequalities

$$\|(Q_2^k)^\top \mathbf{d}^k\| \leq \left(1 + \frac{L_4 + \sigma + C_\mu^{-\ell} L_3}{\mu_{\min}}\right) \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|. \tag{6.28}$$

By the fact that (Q_1^k, Q_2^k) is an orthonormal matrix, putting together (6.26) and (6.28) we get

$$\begin{aligned}
&\|\mathbf{d}^k\|^2 \leq \|(Q_1^k)^\top \mathbf{d}^k\|^2 + \|(Q_2^k)^\top \mathbf{d}^k\|^2 \\
&\leq \left(\frac{2L_3}{\lambda_p^*}\right)^2 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2 + \left(1 + \frac{L_4 + \sigma + C_\mu^{-\ell} L_3}{\mu_{\min}}\right)^2 \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2.
\end{aligned}$$

By definition of $\bar{\mathbf{x}}^k$, this implies the thesis with

$$c_1 = \left(\left(\frac{2L_3}{\lambda_p^*}\right)^2 + \left(1 + \frac{L_4 + \sigma + C_\mu^{-\ell} L_3}{\mu_{\min}}\right)^2\right)^{1/2}.$$

□

Before we state the following Lemma, we notice that by Assumption E1 and the definition of L_2, we have that $\|B(\mathbf{x})\| \leq C_B L_2^2$ for every $\mathbf{x} \in B_r$. In particular, if $\mathbf{x}^k \in B_r$ and $\mu_k = \max\{\mu_{\min}, C_\mu \|B^k\|\}$, then

$$\mu_k \leq C_\mu C_B L_2^2. \tag{6.29}$$

Lemma 6.5. *Let Assumptions E1-E5 hold and let us denote with* $\{\mathbf{x}^k\}$ *the sequence generated by Algorithm PILM with* $\alpha_k = 1$, $\mu_k = \max\{\mu_{\min}, C_\mu \|B^k\|\}$ *for* $C_\mu > 1$ *and* $\ell_k \geq \ell$ *for a given* $\ell \in \mathbb{N}$. *Moreover, let as assume that there exists* $\nu \in (0,1)$ *such that* $\omega\nu > c_4$ *with* $c_4 = \sigma + \sigma(1+c_1)^2 + L_3 C_\mu^{-\ell} + c_1 C_\mu L_2^2 C_B$. *If* $\mathbf{x}^0 \in B(\mathbf{x}^*, \varepsilon)$ *with*

$$\varepsilon \leq \min\left\{\frac{\omega\nu - c_4}{c_2}, \frac{r(1-\nu)}{1 + c_1 - \nu}\right\}$$

then we have that for every $k \in \mathbb{N}_0$

i) $\mathbf{x}^{k+1} \in B_r$

ii) $dist(\mathbf{x}^{k+1}, S) \leq \nu dist(\mathbf{x}^k, S)$

iii) $dist(\mathbf{x}^{k+1}, S) \leq \varepsilon$.

Proof. We proceed by induction over k. By Lemma 6.4 and the bound on ε

$$\begin{aligned} \|\mathbf{x}^1 - \mathbf{x}^*\| &\leq \|\mathbf{d}^0\| + \|\mathbf{x}^0 - \mathbf{x}^*\| \\ &\leq (1 + c_1) dist(\mathbf{x}^0, S) \leq (1 + c_1)\varepsilon \leq r, \end{aligned} \tag{6.30}$$

Which is *i)* for $k = 0$. Since $\mathbf{x}^0 \in B_r$ the bound (6.29) holds. By Lemma 6.3, and the fact that $\mathbf{x}^0 \in B_\varepsilon$ we then have

$$\begin{aligned} &\omega dist(\mathbf{x}^1, S) \leq c_2 \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^2 + \left(\sigma + \sigma(1 + c_1)^{1+\delta}\right) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^{1+\delta} \\ &+ (L_3 \rho_0^{\ell_0} + c_1 \mu_0) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\| \\ &\leq \left(c_2 \varepsilon + \sigma + \sigma(1 + c_1)^{1+\delta} + L_3 \rho_0^{\ell_0} + c_1 \mu_0\right) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\| \\ &\leq \left(c_2 \varepsilon + \sigma + \sigma(1 + c_1)^2 + L_3 C_\mu^{-\ell} + c_1 C_\mu C_B L_2^2\right) dist(\mathbf{x}^0, S) \\ &\leq \omega \nu dist(\mathbf{x}^0, S) \end{aligned} \tag{6.31}$$

and therefore *ii)* holds for $k = 0$. To prove *iii)* is now enough to notice that since $\nu < 1$ we have

$$dist(\mathbf{x}^1, S) \leq \nu dist(\mathbf{x}^0, S) \leq \nu\varepsilon \leq \varepsilon.$$

Let us now assume that for every $i = 1, \ldots, k$ we have $\mathbf{x}^i \in B_r$, $dist(\mathbf{x}^i, S) \leq \nu dist(\mathbf{x}^{i-1}, S)$ and $dist(\mathbf{x}^i, S) \leq \varepsilon$. We want to prove that the same holds for $i = k + 1$. From the definition of $\mathbf{x}^{k+1}$, the triangular inequality, the inductive assumptions and Lemma 6.4 we

have

$$\begin{aligned}\|\mathbf{x}^{k+1}-\mathbf{x}^*\| &\leq \sum_{i=0}^{k}\|\mathbf{d}^i\|+\|\mathbf{x}^0-\mathbf{x}^*\| \\ &\leq c_1\sum_{i=0}^{k} dist(\mathbf{x}^i,S)+\|\mathbf{x}^0-\mathbf{x}^*\| \\ &\leq c_1\sum_{i=0}^{k}\nu^i dist(\mathbf{x}^0,S)+\|\mathbf{x}^0-\mathbf{x}^*\| \\ &\leq \left(1+c_1\sum_{i=0}^{k}\nu^i\right)\varepsilon \leq \left(1+\frac{c_1}{1-\nu}\right)\varepsilon.\end{aligned} \tag{6.32}$$

Since $\varepsilon \leq \frac{r(1-\nu)}{1+c_1-\nu}$ we have that the right-hand side is smaller than r and therefore $\mathbf{x}^{k+1}\in B_r$.
Proceeding as in (6.31),

$$\begin{aligned}&\omega dist(\mathbf{x}^{k+1},S) \leq c_2\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|^2+\left(\sigma+\sigma(1+c_1)^{1+\delta}\right)\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|^{1+\delta} \\ &+(L_3\rho_k^{\ell_k}+c_1\mu_k)\|\mathbf{x}^k-\bar{\mathbf{x}}^k\| \\ &\leq \left(c_2\varepsilon+\sigma+\sigma(1+c_1)^2+L_3C_\mu^{-\ell}+c_1C_\mu C_B L_2^2\right)dist(\mathbf{x}^k,S) \\ &\leq \omega\nu dist(\mathbf{x}^k,S)\end{aligned} \tag{6.33}$$

which implies *ii*). Since $\nu<1$, part *iii*) of the thesis follows directly from *ii*) and the fact that $dist(\mathbf{x}^k,S)\leq\varepsilon$.

□

Theorem 6.2. *If the same Assumptions of Lemma 6.5 hold, then* $dist(\mathbf{x}^k,S)\to 0$ *linearly and* $\mathbf{x}^k\to\bar{\mathbf{x}}\in S\cap B(\mathbf{x}^*,r)$.

Proof. By part *ii*) of Lemma 6.5 we have that for every iteration index k

$$dist(\mathbf{x}^{k+1},S)\leq\nu dist(\mathbf{x}^k,S)$$

since $\nu < 1$ this implies that the sequence $dist(\mathbf{x}^k, S)$ converges linearly to 0. To prove the second part of the thesis, let us consider $l, s \in \mathbb{N}_0$ with $l \geq s$. From Lemma 6.5 we have

$$\|\mathbf{x}^l - \mathbf{x}^s\| \leq \sum_{i=s}^{l-1} \|\mathbf{d}^i\| \leq c_1 \varepsilon \sum_{i=s}^{l-1} \nu^i = c_1 \varepsilon \frac{\nu^s - \nu^l}{1 - \nu}$$

which implies that $\{\mathbf{x}^k\}$ is a Cauchy sequence and therefore is convergent. By Lemma 6.5 i) and the fact that $dist(\mathbf{x}^k, S) \to 0$, the limit point of the sequence has to be in $S \cap B(\mathbf{x}^*, r)$, which concludes the proof. □

We saw so far that the near-separability property of the problem influences the choice of the damping parameter μ_k. In order to ensure convergence of the fixed-point method in lines 3-6 of Algorithm PILM, one has to chose μ_k large enough, depending on the norm of the matrix $B(\mathbf{x})$. However, for classical Levenberg-Marquardt method, in order to achieve local superlinear convergence, the sequence of damping parameters typically has to vanish [3]. In the reminder of this section we show that under a stronger version of the near separability condition, the proposed method achieves superlinear and quadratic local convergence with assumptions, other than the near separability one, that are analogous to those of the classical LM.

Assumption E6. *For every $x \in B_r$ we have that $\|B(\mathbf{x})\| < \lambda_{\min}(P^k)$*

If this assumption holds, then $\rho_k = \|B^k(P^k + \mu_k I)^{-1}\| < 1$ for every choice of μ_k. The following Theorem shows that, whenever the previous Assumption holds and $\delta > 0$ in Assumption E5, one can find a suitable choice of the damping parameter μ_k that ensures local superlinear convergence, provided that the number of inner iterations ℓ_k is large enough.

Lemma 6.6. *Let Assumptions E2-E6 hold with $\delta > 0$ is E5, and let us denote with $\{\mathbf{x}^k\}$ the sequence generated by Algorithm PILM with $\alpha_k = 1$, and μ_k such that*

$$c_\mu \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta \leq \mu_k \leq C_\mu \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta$$

for $0 < c_\mu \leq C_\mu$. If for every k the number of inner iterations ℓ_k is such that $\rho_k^{\ell_k} \leq C_\eta \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta$ for some $C_\eta \geq 0$, then there exists $c_1 \geq 0$ such that

$$\|\mathbf{d}^k\| \leq c_1 dist(\mathbf{x}^k, S)$$

for every $k \in \mathbb{N}_0$.

Remark 6.3. *From Assumption E4 and part iii) in Lemma 6.2, we have that $\omega\|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \leq \|\mathbf{g}^k\| \leq L_3\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|$ therefore, $\mu_k = \bar{\mu}\|\mathbf{g}^k\|^\delta$ satisfies the assumption of the Lemma. Analogously, taking ℓ_k such that $\rho_k^{\ell_k} \leq \frac{C_\eta}{L_3^\delta}\|\mathbf{g}^k\|^\delta$ yields $\rho_k^{\ell_k} \leq C_\eta \|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta$.*

Proof. The proof is analogous to that of Lemma 6.4. Inequalities (6.26) and (6.27) still hold, as they are independent of the choice of μ_k and ℓ_k. With the assumptions of the current Lemma, from (6.27) we get

$$\begin{aligned} \|(Q_2^k)^\top \mathbf{d}^k\| &\leq \frac{L_4\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^2}{c_\mu\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta} + \frac{\sigma\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^{1+\delta}}{c_\mu\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta} \\ &+ \left(1 + \frac{C_\eta L_3\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta}{c_\mu\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|^\delta}\right)\|\mathbf{x}^k - \bar{\mathbf{x}}^k\| \leq \\ &\leq \left(1 + \frac{L_4 + \sigma + C_\eta L_3}{c_\mu}\right)\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|. \end{aligned} \tag{6.34}$$

Putting together (6.26), (6.34) and the fact that $\|\mathbf{d}^k\|^2 = \|(Q_1^k)^\top \mathbf{d}^k\|^2 + \|(Q_2^k)^\top \mathbf{d}^k\|^2$ we get the thesis with

$$c_1 = \left(\left(\frac{2L_3}{\lambda_p^*}\right)^2 + \left(1 + \frac{L_4 + \sigma + C_\eta L_3}{c_\mu}\right)^2\right)^{1/2}.$$

□

Lemma 6.7. *Let us assume that the same hypotheses of Lemma 6.6 hold, and let us define* $c_3 = c_2 + \sigma + \sigma(1+c_1)^{1+\delta} + L_3 C_\eta + c_1 C_\mu$ *and fix* $\nu \in (0,1)$. *If*

$$\varepsilon \leq \min\left\{ \left(\frac{\omega\nu}{c_3}\right)^{1/\delta}, \frac{r(1-\nu)}{1+c_1-\nu} \right\}$$

and $\mathbf{x}^0 \in B(\mathbf{x}^*, \varepsilon)$ *we have that for every* $k \in \mathbb{N}_0$

i) $\mathbf{x}^{k+1} \in B_r$

ii) $dist(\mathbf{x}^{k+1}, S) \leq \nu dist(\mathbf{x}^k, S)$

iii) $dist(\mathbf{x}^{k+1}, S) \leq \varepsilon$.

Proof. The proof proceeds analogously to that of Lemma 6.5. Let us first consider the case $k = 0$. Part *i)* of the thesis is proved as in (6.30). By Lemma 6.3, the assumptions on μ_k and $\rho_k^{\ell_k}$ and the fact that $\mathbf{x}^0 \in B_\varepsilon$ we then have

$$\begin{aligned}
&\omega dist(\mathbf{x}^1, S) \leq c_2 \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^2 + \left(\sigma + \sigma(1+c_1)^{1+\delta}\right) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^{1+\delta} \\
&+ (L_3 \rho_0^{\ell_0} + c_1 \mu_0) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\| \\
&\leq c_2 \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^2 + \left(\sigma + \sigma(1+c_1)^{1+\delta} L_3 C_\eta + c_1 C_\mu\right) \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^{1+\delta} \\
&\leq c_3 \|\mathbf{x}^0 - \bar{\mathbf{x}}^0\|^{1+\delta} \leq c_3 \varepsilon^\delta dist(\mathbf{x}^0, S),
\end{aligned} \tag{6.35}$$

and by the bound on ε we have that *ii)* holds for $k = 0$. Since $\nu < 1$, *iii)* follows immediately. Let us now assume that for every $i = 1, \ldots, k$ we have $\mathbf{x}^i \in B_r$, $dist(\mathbf{x}^i, S) \leq \nu dist(\mathbf{x}^{i-1}, S)$ and $dist(\mathbf{x}^i, S) \leq \varepsilon$. We want to prove that the same holds for $i = k+1$. From the definition of

$\mathbf{x}^{k+1}$, the triangular inequality, the inductive assumptions and Lemma 6.6 we have

$$\begin{aligned}\|\mathbf{x}^{k+1}-\mathbf{x}^*\| &\leq \sum_{i=0}^{k}\|\mathbf{d}^i\|+\|\mathbf{x}^0-\mathbf{x}^*\|\\ &\leq c_1\sum_{i=0}^{k} dist(\mathbf{x}^i,S)+\|\mathbf{x}^0-\mathbf{x}^*\|\\ &\leq c_1\sum_{i=0}^{k}\nu^i dist(\mathbf{x}^0,S)+\|\mathbf{x}^0-\mathbf{x}^*\|\\ &\leq \left(1+c_1\sum_{i=0}^{k}\nu^i\right)\varepsilon \leq \left(1+\frac{c_1}{1-\nu}\right)\varepsilon.\end{aligned}$$

Since $\varepsilon \leq \frac{r(1-\nu)}{1+c_1-\nu}$ we have that the right-hand side is smaller than r and therefore $\mathbf{x}^{k+1}\in B_r$.
Proceeding as in (6.35),

$$\begin{aligned}\omega dist(\mathbf{x}^{k+1},S) &\leq c_2\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|^2+\left(\sigma+\sigma(1+c_1)^{1+\delta}\right)\|\mathbf{x}^0-\bar{\mathbf{x}}^0\|^{1+\delta}\\ &\quad+(L_3\rho_k^{\ell_k}+c_1\mu_k)\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|\\ &\leq c_3\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|^{1+\delta}\leq c_3\varepsilon^{\delta}dist(\mathbf{x}^k,S),\end{aligned} \tag{6.36}$$

which implies *ii*). Since $\nu<1$, part *iii*) of the thesis follows directly from *ii*) and the fact that $dist(\mathbf{x}^k,S)\leq\varepsilon$. □

Theorem 6.3. *If the same Assumptions of Lemma 6.7 hold, then* $dist(\mathbf{x}^k,S)\to 0$ *and* $\mathbf{x}^k\to\bar{\mathbf{x}}\in S\cap B(\mathbf{x}^*,r)$. *The convergence of* $dist(\mathbf{x}^k,S)$ *is superlinear if* $\delta\in(0,1)$ *and quadratic if* $\delta=1$.

Proof. Convergence of $dist(\mathbf{x}^k,S)$ to zero follows directly from part *ii*) of the previous Lemma. Moreover, proceeding as in (6.36), we get

$$dist(\mathbf{x}^{k+1},S)\leq\frac{c_3}{\omega}\|\mathbf{x}^k-\bar{\mathbf{x}}^k\|^{1+\delta}=\frac{c_3}{\omega}dist(\mathbf{x}^k,S)^{1+\delta}$$

which implies superlinear convergence for $\delta \in (0,1)$ and quadratic convergence for $\delta = 1$. Convergence of the sequence $\{\mathbf{x}^k\}_{k=0}^{\infty}$ is proved as in Theorem 6.2. $\square$

6.3 Implementation and Numerical Results

In this section we discuss the parallel implementation of the proposed method and we present the results of a set of numerical experiments carried out to investigate the performance of method and the influence of the parameter K.

We consider the least squares problems that arise from a Network Adjustment problem [60]. Consider a set of points $\{P_i, \ldots, P_n\}$ in $\mathbb{R}^2$ with unknown coordinates, and assume that a set of observations of geometrical quantities involving the points are available. Least Squares adjustments consists into using the available measurements to find accurate coordinates of the points, by minimizing the residual with respect to the given observations in the least squares sense. We consider here network adjustment problems with three kinds of observations: point-point distance, angle formed by three points and point-line distance.

The problems is generated as follows, taking into account the information about average connectivity and structure of the network obtained from the analysis of real cadastral networks. Given the number of points $\hat{n}$ we take $\{P_1, \ldots, P_{\hat{n}}\}$ by uniformly sampling 25% of the points on a regular $2\sqrt{\hat{n}} \times 2\sqrt{\hat{n}}$ grid and we generate observations of the three kinds mentioned above until the average degree of the points is equal to 6. Each observation is defined by randomly selecting the points involved with probability depending on the distance between the points, and generating a random number from the Gaussian distri-

bution with mean equal to the true measurement and given standard deviation. We use a standard deviation equal to 0.01 and 1 degree for distance and angle observations respectively. For all points we also add coordinates observations: for 1% of the points we use standard deviation 0.01, while for the remaining 99% we use standard deviation 1. The problem is stated as a least squares adjustment problem [60]. That is, given the set of observations, the optimization problem is defined as a weighted least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\sum_{j=1}^{m} r_j(\mathbf{x})^2 = \min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{R}(\mathbf{x})\|_2^2 \tag{6.37}$$

where $n = 2\hat{n}$, m is the number of observations, and $r_j(\mathbf{x}) = w_j^{-1}\widehat{r}_j(\mathbf{x})$, with $\widehat{r}_j$ residual function of the j-th observation and w_j corresponding standard deviation.

In Figure 6.1 we present the spyplot of the matrix $J^\top J$ for a problem of size $n = 35,000$.

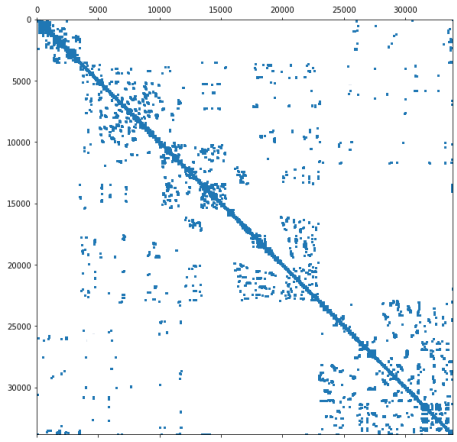


Figure 6.1: Sparsity plot of the coefficient matrix for n= 35,000

The proposed method is implemented in Python and all the tests are performed on the AXIOM computing facility consisting of 16 nodes (8 × Intel i7 5820k 3.3GHz and 8 × Intel i7 8700 3.2GHz CPU - 96 cores and 16GB DDR4 RAM/node) interconnected by a 10 Gbps network.

Algorithm 6.1 assumes that the number K and the subsets I_s, E_s, $s = 1, \ldots, K$ of the variables and the residuals are given. In the tests that follow, given the number K of workers, the server computes the partition of the variables using METIS [32], then defines the corresponding partition of the residuals as in (6.2) and transmits them to the nodes. The time necessary to carry out this preprocessing phase is included in the timings that we show below.

In the following, given $i = 1, \ldots, K$ we with denote with $\mathcal{N}_i$ the set of indices $j \neq i$ such that there exists an observation in $\hat{E}$ involving variables in both I_i and I_j. The Jacobian matrix and the derivatives of F are computed as follows. For every $i = 1, \ldots, n$ node i computes $J_{i\mathbf{R}_i}$ and $J_{i\rho}$ and shares $J_{i\rho}$ with the server, which then broadcasts $\{J_{i\rho}\}_{i=1}^{K}$ to the workers. Node i then computes g_i^k, P_i^k and $\{B_{ij}^k\}_{j\in\mathcal{N}_i}$ according to (6.8). Notice that B_{ij}^k is nonzero only if $j \in \mathcal{N}_j$. The local gradients $\mathbf{g}_i^k$ are transmitted by the workers to the server, that then broadcasts the aggregated gradient $\mathbf{g}^k$. We observed that, compared to the approach where the server computes the whole Jacobian and then transmits it to the rest of the nodes, the distributed approach that we use leads to 50% faster execution of this phase.

To compute the direction $\mathbf{d}^k$ (lines 3-7 in Algorithm 6.1) we proceed as follows. At the first inner iteration (line 3), node i computes $\mathbf{y}_i^1$ solution of

$$(P_i^k + \mu_k I_{n_i})\mathbf{y}_i^1 = -\mathbf{g}_i^k$$

which only involves quantities available to it. After solving this system, all nodes shares their local solution with server. The server defines the aggregated vector $\mathbf{y}^l = (\mathbf{y}_1^l, \ldots, \mathbf{y}_K^l)^\top$ and broadcasts it to the nodes. For all other inner iterations (line 8) each node i first computes the right hand side

$$(\mathbf{g}^k + B^k \mathbf{y}^l)_i = \mathbf{g}_i^k + \sum_{j \in \mathcal{N}_i} B_{ij}^k \mathbf{y}_j^l$$

using the aggregated vector received from the server, then computes the new local estimate $\mathbf{y}_i^{l+1}$ as the solution of

$$(P_i^k + \mu_k I_{n_i}) \mathbf{y}_i^{l+1} = -(\mathbf{g}^k + B^k \mathbf{y}^l)_i.$$

Each node then sends the local vector to the server, which defines and shares the aggregated vector, and a new inner iteration begins. All the linear systems are solved with PyPardiso [22].

For all the communication phases we considered three approaches. The one mentioned above where the server broadcasts the aggregated quantities to all the nodes, the case where it send to each node only the blocks that are necessary to them to perform their local computations, and the case where we define communicator between the workers, in such a way that node i can share relevant quantities directly to the nodes in $\mathcal{N}_i$. While in the second option the amount of exchanged data is smaller, we observed that the broadcasting approach results in practice in a significantly shorter communication time. Overall, the performance of the node-to-node approach was very similar to that of broadcasting and therefore we chose to continue with the broadcasting strategy, which is simpler from the point of view of the implementation.

We consider a network adjustment problem with $n = 10^6$ and $m = 2.5 \times 10^6$ generated as described above, and we solve the problem

for different values of the parameter K. The initial guess is defined as the coordinate observations available in the problem description while the execution is terminated when at least $68\%, 95\%$ and 99.5% of the residuals is smaller than 1, 2 and 3 times the standard deviation respectively. To understand the behavior of the method, in Figure 6.2 we plot the values of the three percentages above at each iteration, for K=60. In Figure 6.3 we plot the execution time to arrive a termination for $K \in [35, 85]$. The damping parameter μ_k is initialized as 10^5, which is the same order of magnitude as $\|\mathbf{R}\|$. At each iteration we take $\mu_{k+1} = \mu_k/2$ if the accepted step size is larger than 0.5 and $\mu_{k+1} = 2\mu_k$ otherwise, with safeguards $\mu_{\min} = 10^{-10}$ and $\mu_{\max} = 10^{10}$. The number of inner iteration is fixed to $\ell_k = 5$ for every k.

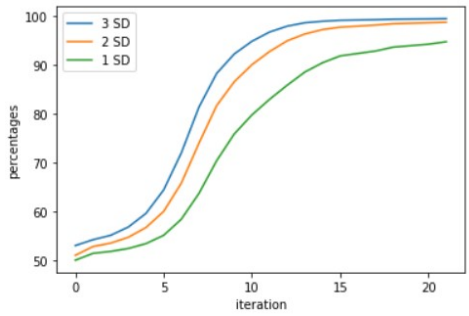


Figure 6.2: Percentage of residuals within 1, 2 and 3 standard deviations. Values of the percentages at each iteration

We can see that, starting from the smaller values of K, the execution time of the method decreases while K increase, until reaching a plateau, after which it begins to increase. The plot shows the good performance of the proposed method. Smaller values of K are omitted from the plot as the time necessary to arrive at termination becomes too large. In particular for $K = 1$, which correspond to the centralized method, the execution time is orders of magnitude larger than for the

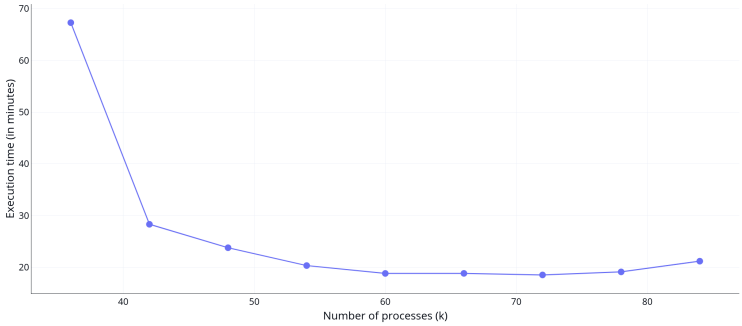


Figure 6.3: Execution time for different number of processors

values of K included in the plot, and hence not comparable.
There are two main reasons behind the increase for large values of K. The first is that for larger values of K the norm of the matrix B is larger and therefore the fixed point method converges more slowly to the solution of the LM system. Since we are running a fixed number of inner iterations ℓ that does not depend on the number of nodes K, large values of K result in a direction $\mathbf{d}^k$ that is a worse approximation of the LM direction and therefore the number of outer iterations needed by the method is larger. That is, after a certain point the overall computational cost increases because the saving induced by the fact that the linear systems solved by each node are smaller is not enough to balance the additional number of outer iterations. The second reason is common to all parallel methods: increasing the number of nodes K increases the communication traffic and, when K is too large, the time necessary to handle the additional communication overcomes the saving in terms of computation.
The fact that there is a plateau is also relevant from the practical point of view. The optimal value K depends on the size n but also on sparsity and the separability of the problem, and thus it may be hard to predict. However, the results show that the obtained timings

on the considered problems are similar and nearly-optimal for a wide range of values of K, suggesting that an accurate choice of the number of nodes could in general not be necessary in order for the method to achieve a good performance.
Notice that while the choice of μ_k does not ensure theoretically $\rho_k < 1$ at all iterations, it gives good results in practice, and does not require the computation of $\|B_k\|$, which may be expensive in the distributed framework.

As a comparison, Algorithm 6.1 was also implemented and tested on the same problem in a sequential fashion. That is, with only one machine performing the tasks for $i = 1, \ldots, K$ in sequence. The resulting timings were 228, 61.5 and 59.6 minutes for $K = 45, 80, 100$ respectively. Since these timings decrease for increasing K, this shows that the proposed method is effective, compared to classical LM method (equivalent to $K = 1$), even when a parallel implementation is not possible in practice. Moreover, they show that the saving in time induced by the parallelization of the computation is significantly larger than the time necessary to handle the communication.

Chapter 7

Conclusions

The results in this thesis cover several topics in distributed optimization. The considered problems are unconstrained and of large dimension, while the computational framework is assumed to be distributed. Two main types of considered computational networks are distributed networks, where computational nodes can communicate through communication links represented by a communication matrix and server/worker networks where one has a central node that communicates with all other nodes, but the worker nodes cannot communicate between themselves. The objective function is stated as a sum of a large number of the so-called local functions. The main motivation for the considered problem comes from machine learning and Big Data analytic. Theoretical analysis, followed by numerical results is presented for four classes of methods that are considered in the thesis.

In Chapter 3 we proved that a class of distributed first-order methods, including those proposed in [24, 25], is robust to time-varying and uncoordinated step-sizes and time-varying weight-balanced digraphs, without requiring the network to be connected at all iterations. The achieved results provide a solid improvement in understanding of the robustness of exact distributed first-order methods to time-varying

networks and uncoordinated time-varying step sizes. Most notably, we showed that the unification strategy in [24] and the spectral-like step size selection strategy in [25], as well as combination of those, exhibits a high degree of robustness.

In Chapter 4 we proposed a Distributed inexact Newton method that applies JOR method for the computation of the direction at each iteration and employs an adaptive step size that does not require a priori knowledge of the regularity constants of the objective function. Provided that the local functions are strongly convex with Lipschitz-continuous Hessian matrices, the strategy adopted by the method for the computation of the stepsize ensures convergence to the solution of the penalty problem (4.3) for any choice of the initial guess. Moreover, it can be proved that after a finite number of iterations the full stepsize is accepted, and that the method achieves local convergence with order depending on the choice of the forcing terms. The method is presented and analyzed in the decentralized framework, assuming that the underlying communication network is fixed during the execution of the method. It would be interesting from both the theoretical and the practical point of view is to extend the method to the case of time-varying network, as well as to consider different frameworks such as the federated learning framework and the asynchronous case mentioned in Section 2.1.

In Chapter 5 a class of novel, iterative, distributed methods for the solution of linear systems of equations, are derived upon classical fixed point methods. We proved linear convergence for strongly connected communication network and showed that the convergence rate depends on the diameter of the network and on the norm of the underlying iterative matrix. In particular, if the graph is strongly connected the obtained result is analogous to the classical, centralized case. The presented method is extended to the time-varying case and an analogous convergence result is proved under suitable joint connectivity assumptions, comparable with assumptions required by different methods in

literature. The algorithm is compared with the relevant optimization methods presented in [35, 47, 57, 59, 36]. The numerical results show good performance of DFIX in comparison with the mentioned methods. In particular, in the vast majority of the considered tests, DFIX outperformed all the methods in terms of both computational cost and communication traffic.

In Chapter 6 we presented an Inexact Levenberg-Marquardt method, suitable for paralellization in the server/worker framework, for the solution of nearly-separable least squares problems. The method relies on a fixed-point iteration for the computation of the direction at each iteration, and on a nonmonotone line search strategy for the selection of the stepsize. We proved that with suitable assumptions on the objective function and the separablity of the problem, the proposed method achieves global convergence and local linear, superlinear, or quadratic convergence, depending on the choice of the damping parameter and the number of fixed-point iterations. An interesting line of research could be to develop an analogous method for the decentralized case, possibly exploiting the main ideas underlying the distributed Inexact Newton method presented in Chapter 4, such as the focus on the penalty reformulation of the distributed problem and the adaptive stepsize.

Bibliography

[1] S. A. Alghunaim, E. K. Ryu, K. Yuan, and A. H. Sayed. Decentralized proximal gradient algorithms with linear convergence rates. *IEEE Transactions on Automatic Control*, 66(6):2787–2794, 2021.

[2] J. Barzilai and J. M. Borwein. Two point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.

[3] R. Behling, D. S. Gonçalves, and S. A. Santos. Local convergence analysis of the Levenberg-Marquardt framework for nonzero-residue nonlinear least-squares problems under an error bound condition. *Journal of Optimization Theory and Applications*, 183:1099–1122, 2019.

[4] S. Bellavia, S. Gratton, and E. Riccietti. A Levenberg-Marquardt method for large nonlinear least-squares problems with dynamic accuracy in functions and gradients. *Numerische Mathematik*, 140(3):791–825, 2018.

[5] A. S. Berahas, R. Bollapragada, N. S. Keskar, and E. Wei. Balancing communication and computation in distributed optimization. *IEEE Transactions on Automatic Control*, 64(8):3141–3155, 2019.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1), 2011.

[7] N. A. C. Cressie. Statistics for spatial data. *New York: Wiley*, 1993.

[8] Y. H. Dai, Y. Huang, and X. W. Liu. A family of spectral gradient methods for optimization. *Computational Optimization and Applications*, 74(1):43–65, 2019.

[9] Y. H. Dai and L. Z. Liao. R-linear convergence of the Barzilai and Borwein gradient method. *IMA Journal of Numerical Analysis*, 22(1):1–10, 2002.

[10] H. Dan, N. Yamashita, and M. Fukushima. Convergence properties of the inexact Levenberg-Marquardt method under local error bound conditions. *Optimization Methods and Software*, 17(4):605–626, 2002.

[11] M. H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[12] R. S. Dembo, S. C Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Optimization*, (2):400–408, 1982.

[13] C. Desoer and M. Vidyasagar. Feedback systems: Input-output properties. *SIAM - Classics in Applied Mathematics*, 2009.

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

[15] D. Di Serafino, V. Ruggiero, G. Toraldo, and L Zanni. On the steplength selection in gradient methods for unconstrained optimization. *Applied Mathematics and Computation*, 318:176–195, 2018.

[16] S. C. Eisenstat and H. F. Walker. Globally convergent inexact Newton methods. *SIAM Journal on Optimization*, 4:393–422, 1994.

[17] P. Erdős and A. Renyi. On random graphs I. *Publicationes Mathematicae*, 6:290–297, 1959.

[18] J. Fan and J. Pan. Convergence properties of a self-adaptive Levenberg-Marquardt algorithm under local error bound condition. *Computational Optimization and Applications*, 34(1):47–62, 2006.

[19] J. Fan and Y. Yuan. On the quadratic convergence of the Levenberg-Marquardt method without nonsingularity assumption. *Computing*, 74(1):23–39, 2005.

[20] L. Fodor, D. Jakovetić, N. Krejić, and G. Malaspina. Parallel inexact Levenberg-Marquardt method for sparse least squares problems. *(in preparation)*.

[21] J. Franken, W. Florijn, M. Hoekstra, , and E. Hagemans. Rebuilding the cadastral map of the netherlands: the artificial intelligence solution. *FIG working week 2021 proceedings*, 2021.

[22] A Haas. Pypardiso.

[23] J. M. Hendrickx, R. M. Jungers, A. Olshevsky, and G. Vankeerberghen. Graph diameter, eigenvalues, and minimum-time consensus. *Automatica*, 50:635–640, 2014.

[24] D. Jakovetić. A unification and generalization of exact distributed first-order methods. *IEEE Transactions on Signal and Information Processing over Networks*, 5(1):31–46, 2019.

[25] D. Jakovetić, N. Krejić, and N. Krklec Jerinkić. Exact spectral-like gradient method for distributed optimization. *Computational Optimization and Applications*, 74:703—-728, 2019.

[26] D. Jakovetić, N. Krejić, and N. Krklec Jerinkić. Efix: Exact fixed point methods for distributed optimization. *Journal of Global Optimization*, 2022.

[27] D. Jakovetić, N. Krejić, and N. Krklec Jerinkić. A Hessian inversion-free exact second order method for distributed consensus optimization. *arxiv preprint, arXiv:2204.02690*, 2022.

[28] D. Jakovetić, N. Krejić, N. Krklec Jerinkić, G. Malaspina, and A. Micheletti. Distributed fixed point method for solving systems of linear algebraic equations. *Automatica*, 134(8), 2021.

[29] D. Jakovetić, N. Krejić, and G. Malaspina. Distributed inexact Newton method with adaptive stepsize. *(in preparation)*.

[30] 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, 2012.

[31] E. W. Karas, S. A. Santos, and B. F. Svaiter. Algebraic rules for computing the regularization parameter of the levenberg–marquardt method. *Computational Optimization and Applications*, 65:723–751, 2016.

[32] G. Karypis and V. Kumar. Graph partitioning and sparse matrix ordering system. *University of Minnesota*, 2009.

[33] K. Konolige. Sparse bundle adjustment. *British Machine Vision Conference*, 2010.

[34] D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *ournal of the Southern African Institute of Mining and Metallurgy*, 52(6):119—-139, 1951.

[35] N. Li and G. Qu. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions Control of Network Systems*, 5(3):1245–1260, 2017.

[36] J. Liu, A. S. Morse, A. Nedić, and T. Başar. Exponential convergence of a distributed algorithm for solving linear algebraic equations. *Automatica*, 83:37–46, 2017.

[37] J. Liu, S. Mou, and A. S. Morse. A distributed algorithm for solving a linear algebraic equation. *Proceedings of the 51st Annual Allerton Conference on Communication, Control, and Computing*, 60(11):267—-274, 2013.

[38] J. Liu, S. Mou, and A. S. Morse. Asynchronous distributed algorithms for solving linear algebraic equations. *IEEE Transactions on Automatic Control*, 63(2):372–385, 2018.

[39] G. Malaspina, D. Jakovetić, and N. Krejić. Linear convergence rate analysis of a class of exact first-order distributed methods for time-varying directed networks and uncoordinated step sizes. *arxiv preprint: arXiv:2007.08837*.

[40] G. Malaspina, N. Krejić, and L. Swaenen. Splitted Levenberg-Marquardt method for large-scale sparse problems. *arxiv preprint: arXiv:2206.05188*.

[41] G. Mao, B. Fidan, and B. D. O. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10):2529–2553, 2007.

[42] G. Matheron. Traité de geostatistique appliquée vol. II, le krigeage. *Memoires du Bureau de Recherches Géologiques et Miniéres*, 24, 1963.

[43] A. Mokhtari, Q. Ling, and A. Ribeiro. Network Newton distributed optimization methods. *IEEE Transactions on Signal Processing*, 65(1):146–161, 2017.

[44] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro. A decentralized second-order method with exact linear convergence rate for consensus optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(4):507–522, 2016.

[45] J. Mota, J. Xavier, P. Aguiar, and M. Püschel. Distributed optimization with local domains: Applications in mpc and network flows. *IEEE Transactions on Automatic Control*, 60(7):2004–2009, 2015.

[46] S. Mou, Z. Lin, L. Wang, D. Fullmer, and A. S. Morse. A distributed algorithm for efficiently solving linear equations and its applications. *System & Control Letters*, 91:21–27, 2016.

[47] A. Nedić, A. Olshevsky, and W. Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597—-2633, 2017.

[48] A. Nedić, A. Olshevsky, W. Shi, and C. A. Uribe. Geometrically convergent distributed optimization with uncoordinated step sizes. *American Control Conference*, pages 3950–3955, 2017.

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

[50] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

[51] Bertsekas D. P. Nonlinear programming. *Athena Scientific, Belmont*, 1997.

[52] B. Polyak and A. Tremba. New versions of Newton method: Step-size choice, convergence domain and under-determined equations. *Optimization Methods and Software*, 35(6):1272—-1303, 2020.

[53] M. Raydan. On the Barzilai and Borwein choice of steplength for the gradient method. *IMA Journal of Numerical Analysis*, 13:321–326, 1993.

[54] M. Raydan. Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal on Optimization*, 7:26–33, 1997.

[55] F. Saadatniaki, R. Xin, and U. A. Khan. Decentralized optimization over time-varying directed graphs with row and column-stochastic matrices. *IEEE Transactions on Automatic Control*, 60(11):4769–4780, 2020.

[56] G. Scutari and Y. Sun. Distributed nonconvex constrained optimization over time-varying digraphs. *Mathematical Programming*, 176(1–2):497–544, 2019.

[57] W. Shi, Q. Ling, G. Wu, and W. Yin. Extra: an exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

[58] Y. Sun, A. Daneshmand, and G. Scutari. Convergence rate of distributed optimization algorithms based on gradient tracking. *arxiv preprint, arXiv:1905.02637*, 2019.

[59] A. Sundararajan, B. Van Scoy, and L. Lessard. Analysis and design of first-order distributed optimization algorithms over time-varying graphs. *arxiv preprint, arXiv:1907.05448*, 2019.

[60] P. J. G. Teunissen. Adjustment theory. *Series on Mathematical Geodesy and Positioning*, 2003.

[61] B. Touri and A. Nedić. On backward product of stochastic matrices. *Automatica*, 48(8):1477–1488, 2012.

[62] F. van den Heuvel, G. Vestjens, G. Verkuijl, and M. van den Broek. Rebuilding the cadastral map of the netherlands: the geodetic concept. *FIG working week 2021 proceedings*, 2021.

[63] P. Wang, S. Mou, J. Lian, and W. Ren. Solving a system of linear equations: From centralized to distributed algorithms. *Annual Reviews in Control*, 47:306–322, 2019.

[64] X. Wang, J. Zhou, S. Mou, and M. J. Corless. A distributed algorithm for least square solutions. *IEEE Transactions on Automatic Control*, 64(10):4217–4222, 2019.

[65] L. Xiao, S. Boyd, and S. Lall. Distributed average consensus with time-varying metropolis weights. *Automatica*, 2006.

[66] Y. Xiao and J. Hu. Distributed solutions of convex feasibility problems with sparsely coupled constraints. *IEEE 56th Annual Conference on Decision and Control*, pages 3386–3392, 2017.

[67] R. Xin and U. A. Khan. Distributed heavy-ball: a generalization and acceleration of first-order methods with gradient tracking. *IEEE Transactions on Automatic Control*, 65(6):2627–2633, 2020.

[68] R. Xin, C. Xi, and U. A. Khan. Frost—fast row-stochastic optimization with uncoordinated step-sizes. *EURASIP Journal on Advances in Signal Processing—Special Issue on Optimization, Learning, and Adaptation over Networks*, 1, 2019.

[69] J. Xu, Y. Tian, Y. Sun, and G. Scutari. Distributed algorithms for composite optimization: Unified framework and convergence analysis. *arxiv preprint, arXiv:2002.11534*, 2020.

[70] J. Xu, S. Zhu, Y. C. Soh, and L. Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant step sizes. *IEEE Conference on Decision and Control*, pages 2055–2060, 2015.

[71] N. Yamashita and M. Fukushima. On the rate of convergence of the Levenberg-Marquardt method. *Topics in Numerical Analysis*, 15:239–249, 2001.

[72] K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835—-1854, 2016.

[73] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed. Exact diffusion for distributed optimization and learning — part I: Algorithm development. *IEEE Transactions on Signal Processing*, 67(3):708–723, 2019.

[74] J. Zhang, Q. Ling, and A. M. C. So. A Newton tracking algorithm with exact linear convergence for decentralized consensus

optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 7:346–358, 2021.

[75] J. Zhang, K. You, and Başar T. Distributed adaptive Newton methods with global superlinear convergence. *Automatica*, 138, 2022.

Appendix A
Short Biography

Greta Malaspina was born on the 13th of June 1992 in Pietrasanta, Italy. In 2011 she enrolled in the Bachelor program in mathematics at the university of Pisa, which she completed in 2015. In 2018 she received her Master degree in mathematics from the University of Florence. In 2019 she started her PhD studies at the University of Novi Sad, and participated in BIGMATH, a PhD project that is part of the European Eunion's Horizon 2020 Marie Skłodowska-Curie actions. During her PhD she attended several international conferences, including ICIAM 2019 - International Congress on Industrial and Applied Mathematics, and EURO conference in 2021 and 2022.

Novi Sad, 2022 Greta Malaspina

Овај Образац чини саставни део докторске дисертације, односно докторског уметничког пројекта који се брани на Универзитету у Новом Саду. Попуњен Образац укоричити иза текста докторске дисертације, односно докторског уметничког пројекта.

План третмана података

Назив пројекта/истраживања
Методе дистрибуиране оптимизације за проблеме великих димензија без ограничења Distributed Optimization Methods for Large Scale Unconstrained Optimization Problems
Назив институције/институција у оквиру којих се спроводи истраживање
Универзитет у Новом Саду Природно-математички факултет
Назив програма у оквиру ког се реализује истраживање
Докторске студије математике
1. Опис података
1.1 Врста студије У овој студији нису прикупљани подаци
2. Прикупљање података
3. Третман података и пратећа документација
4. Безбедност података и заштита поверљивих информација
5. Доступност података
6. Улоге и одговорност