Università
degli Studi
di Padova

Dipartimento
di Ingegneria
dell'Informazione

A Consensus Approach to Distributed Convex Optimization in Multi-Agent Systems

Ph.D. candidate
Filippo Zanella

Advisor
prof. Angelo Cenedese

Director
prof. Matteo Bertocco

Coordinator
prof. Carlo Ferrari

Ph.D. School in
Information Engineering

University of Padova
Department of Information Engineering
2013

Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Sede Amministrativa: Università degli Studi di Padova

Dipartimento: Ingegneria dell'Informazione

Scuola di dottorato di ricerca in: Ingegneria dell'Informazione

Indirizzo: Scienza e Tecnologia dell'Informazione

Ciclo: XXV

A Consensus Approach to Distributed Convex Optimization in Multi-Agent Systems

Coordinatore
Ch.mo Prof. Carlo Ferrari

Direttore della Scuola
Ch.mo Prof. Matteo Bertocco

Supervisore
Ch.mo Prof. Angelo Cenedese

Dottorando
Filippo Zanella

Contents

Abstract

In this thesis we address the problem of distributed unconstrained convex optimization under separability assumptions, i.e., the framework where a network of agents, each endowed with local private convex cost and subject to communication constraints, wants to collaborate to compute the minimizer of the sum of the local costs. We propose a design methodology that combines average consensus algorithms and separation of time-scales ideas. This strategy is proven, under suitable hypotheses, to be globally convergent to the true minimizer. Intuitively, the procedure lets the agents distributedly compute and sequentially update an approximated Newton-Raphson direction by means of suitable average consensus ratios. We consider both a scalar and a multidimensional scenario of the Synchronous Newton-Raphson Consensus, proposing some alternative strategies which trade-off communication and computational requirements with convergence speed. We provide analytical proofs of convergence and we show with numerical simulations that the speed of convergence of this strategy is comparable with alternative optimization strategies such as the Alternating Direction Method of Multipliers, the Distributed Subgradient Method and Distributed Control Method.

Moreover, we consider the convergence rates of the Synchronous Newton-Raphson Consensus and the Gradient Descent Consensus under the simplificative assumption of quadratic local cost functions. We derive sufficient conditions which guarantee the convergence of the algorithms. From these conditions we then obtain closed form expressions that can be used to tune the parameters for maximizing the rate of convergence. Despite these formulas have been derived under quadratic local cost functions assumptions, they can be used as rules-of-thumb for tuning the parameters of the algorithms.

Finally, we propose an asynchronous version of the Newton-Raphson Consensus. Beside having low computational complexity, low communication requirements and being interpretable as a distributed Newton-Raphson algorithm, the technique has also the beneficial properties of requiring very little coordination and naturally supporting time-varying topologies. Again, we analytically prove that under some assumptions it shows either local or global convergence properties. Through numerical simulations we corroborate these results and we compare the performance of the Asynchronous Newton-Raphson Consensus with other distributed optimization methods.

Sommario

In questa tesi viene affrontato il problema dell'ottimizzazione distribuita non vincolata di funzioni convesse. Lo scenario è costituito da una rete di agenti interconnessi, ognuno dei quali è dotato di una funzione costo locale convessa ed è soggetto a vincoli di comunicazione. Ogni agente vuole collaborare per calcolare il minimo della somma dei costi locali.

Viene proposta una soluzione che combina algoritmi di *average consensus* con concetti di separazione delle scale temporali, propri della teoria del controllo non lineare. Tale strategia, denotata come Newton-Raphson Consensus, si dimostra convergere globalmente al minimo richiesto, sotto opportune ipotesi. Intuitivamente, l'algoritmo permette agli agenti di calcolare in maniera distribuita e di aggiornare sequenzialmente una direzione approssimata alla Newton-Raphson, tramite specifici rapporti di average consensus. Viene proposta una versione sincrona del Newton-Raphson Consensus, validata sia per funzioni scalari che vettoriali, proponendo nel secondo caso alcune strategie alternative volte a bilanciare le prestazioni, in termini di requisiti computazionali e di comunicazione, con una adeguata velocità di convergenza. Vengono presentate prove analitiche di convergenza e simulazioni numeriche che evidenziano come la velocità di convergenza del Synchronous Newton-Raphson Consensus è comparabile con strategie di ottimizzazione alternative quali l'Alternating Direction Method of Multipliers, il Distributed Subgradient Method e il Distributed Control Method.

La trattazione si completa con l'analisi della velocità di convergenza del Synchronous Newton-Raphson Consensus, comparata con quella di un Gradient Descent Consensus (GDC), sotto l'ipotesi semplificativa di funzioni costo quadratiche. Vengono derivate condizioni sufficienti che garantiscono la conver-

genza di tali algoritmi. Da queste condizioni si ottengono espressioni in forma chiusa che possono essere utilizzate per regolare i parametri che caratterizzano gli algoritmi e per massimizzare la velocità di convergenza. Si evidenzia che nonostante queste formule siano derivate assumendo funzioni di costo (locali) quadratiche, esse possono essere usate come metodologie di riferimento per la regolazione dei parametri degli algoritmi in situazioni generali.

Infine, viene proposta una versione asincrona del Newton-Raphson Consensus. Oltre ad avere una ridotta complessità computazionale e minimi requisiti di comunicazione, questa tecnica richiede poca coordinazione tra gli agenti e si mantiene valida in topologie tempo-varianti. Ancora una volta, viene dimostrato analiticamente, sotto opportune ipotesi, che l'Asynchronous Newton-Raphson Consensus ha proprietà di convergenza locali o globali. Mediante simulazioni numeriche vengono corroborati tali risultati e vengono confrontate le prestazioni di tale algoritmo con altri metodi di ottimizzazione distribuita quali l'Asynchronous Fast Newton-Raphson Consensus, l'Asynchronous Distributed Subgradient Method, l'Asynchronous Alternating Direction Method of Multipliers e il Pairwise Equalizing Method.

Introduction

Optimization is a pervasive concept underlying many aspects of modern life (Shor (1985); Bertsekas et al. (2003); Boyd and Vandenberghe (2004)), and it also includes the management of distributed systems, i.e., artifacts composed by a multitude of interacting entities often referred to as "agents". Examples are transportation systems, where the agents are both the vehicles and the traffic management devices, smart electrical grids (see, e.g., Bolognani and Zampieri (2011)), where the agents are the energy producers-consumers and the power transformers-transporters, and distributed estimation and learning techniques. But this variety of applications fostered the development of several different strategies, each tailored for a specific niche of assumptions.

The interests on systems where agents collaborate to pursue a common goal, which is difficult or impossible to an individual agent, are driven by the possibility of synergies, i.e., coordinated actions whose total effects are bigger than the ones achievable without coordination. An entire class of control problems has been casted into the framework of the Networked Control Systems (NCSs), agent-based system in which the controller elements are not centralized but distributed over the entire system via parallel and distributed processing (Antsaklis and Baillieul (2007)).

In particular, in the past years an increasing attention from various research communities has been posed on *distributed optimization*, being it a building block for all the distributed decision making processes.

Here we consider exactly the problem of distributed optimization, i.e., the class of algorithms suitable for networked systems and characterized by the absence of a centralized coordination unit (see Tsitsiklis (1984); Bertsekas and Tsitsiklis (1997); Bertsekas (1998)). The interest on distributed optimization tools has become wider and wider, concurrently with the research on networked control systems, because the former methods let the networks self-organize and adapt to surrounding and changing environments, and that they are necessary to manage extremely complex systems in an autonomous way with only limited human intervention. In particular we focus on unconstrained convex optimization, although there is a rich literature also on distributed constrained optimization such as Linear Programming Bürger et al. (2012). Nonetheless there has been a general trend in trying to find algorithms with, at the same time, faster and faster convergence properties, and milder and milder requirements on the coordination among the agents. In this thesis we contribute to this trend by showing how a certain fast optimization technique can be based on completely asynchronous communication schemes. To position our findings in the existing literature we now briefly survey the existing optimization schemes and their properties.

Literature review

The literature on distributed unconstrained convex optimization is extremely vast. We specifically consider *distributed derivative-based optimization methods*, which roots can be traced back to the seminal work Tsitsiklis (1984). We thus assume small complexity and high accuracy in sampling the Jacobians and Hessians of the objectives.

The literature on derivative-based methods can be factorized in various macro-categories (e.g., methods based on contraction mappings or not, methods exploiting Lagrangian formalism or not, methods based on heuristic or ad-hoc methods or not) and micro-categories (e.g., methods exploiting Lagrangian formalisms *and* based on dual averaging techniques or not, methods not exploiting Lagrangian formalisms *and* based on second-order derivatives or not). A complete analysis of all these methods is beyond the scope of this thesis, and we thus limit in describing just the most famous ones. In particular, we focus on distributed nonlinear optimization technique and thus do not consider distributed linear programming algorithms; for recent advances in this framework see Bürger et al. (2012) and references therein.

Distributed Subgradient Methods (DSMs): among the distributed methods not exploiting Lagrangian formalisms, the most popular ones are the DSMs (Dem'yanov and Vasil'ev (1985)). Here the optimization of non smooth costs is performed by means of descent/ascent directions based on first order derivatives information and suitable projection operations to maintain feasibility during the evolution. These methods arise in both primal and dual formulations, since sometimes it is better to perform dual optimization.

DSM can be implemented using the same communication protocols involved in this thesis, and its convergence properties have been deeply investigated by several authors (e.g., Nedić and Bertsekas (2001); Nedić et al. (2001); Nedić and Ozdaglar (2008)), see Kiwiel (2004) for a unified view of many convergence results. There exist several slightly different versions, mainly with variations on the stepsizes, on the communication protocol, and on the management of the local constraint sets (Johansson et al. (2009); Lobel et al. (2011); Nedić et al. (2010); Zhu and Martínez (2012)). We also notice a renewed effort in increasing its convergence speed properties by exploiting augmented memory (Ghadimi et al. (2012)) and in lessening the dependence of the convergence properties on the choice of the stepsizes, see Lu et al. (2011).

Subgradient methods have been exploited for several practical purposes, e.g., to optimally allocate resources in Wireless Sensors-Actuators Networks (see Johansson (2008)), to maximize the convergence speeds of gossip algorithms as in Boyd et al. (2006), to manage optimality criteria defined in terms of ergodic limits (Ribeiro (2010)). We can also find analyses for several extensions of the original idea, e.g., directions that are computed combining information from other agents Blatt et al. (2007); Xiao and Boyd (2006) and stochastic errors in the evaluation of the subgradients Ram et al. (2009). Explicit characterizations can also show trade-offs between desired accuracy and number of iterations Nedić and Ozdaglar (2009).

Alternating Direction Method of Multipliers (ADMM): the most widely known dual method, i.e., method that exploits decompositions of the dual problem and that operates explicitly both on primal and dual variables, is ADMM, whose roots can be traced back to Hestenes (1969). It is based on an opportune modification of the Lagrangian of the original problem that gives rise to the possibility of alternate and distributedly execute primal descent and dual ascent steps (Bertsekas and Tsitsiklis, 1997; Bertsekas, 1982, pp. 253-261).

Its efficacy in several practical scenarios is undoubted, see, e.g., Boyd et al. (2010) and references therein. A notable size of the dedicated literature focuses on the analysis of its convergence performance and on the tuning of its parameters for optimal convergence speed, see, e.g., Erseghe et al. (2011) for least-squares estimation scenarios or He and Yuan (2011) for linearly constrained convex programs. It is known for having in general (but not always, see Ghadimi et al. (2012)) faster convergence properties than DSMs, and to be employable in various different frameworks (see Mota et al. (2012)). Even if proven to be an effective algorithm, ADMM suffers from requiring a careful implementation and synchronized communication protocols, although some recent attempts for asynchronous and distributed implementations have appeared Wei and Ozdaglar (2012); Mota et al. (2012); Jakovetić et al. (2011).

Newton-like methods: it is known that second order information, if used, can improve the convergence properties of optimization methods. Recently some distributed algorithms have exploited inexact Newton directions, resulting from the Laplacian of the communication graph, to compute dual ascent steps (see Jadbabaie et al. (2009); Zargham et al. (2011); Wei et al. (2010)). Since the Laplacian cannot be computed exactly, the convergence rates of these schemes rely on the analysis of inexact Newton methods Dembo et al. (1982). These methods, tailored and applicable only to network flow problems, have super-linear convergence under specific assumptions. Nonetheless they are embedded into primal-dual algorithms and can not be considered general-purpose distributed optimization tools.

Contraction mappings: these methods exploit consecutive projections of the local guesses into opportune sets. An example is Fischione (2011), leveraging some particular assumptions on the local costs to construct a distributed contraction mapping that converges superlinearly to the global optimum.

Other approaches: other important techniques, only partially falling in the previous categories, are the one proposed in Wang and Elia (2010), that reformulate distributed optimization as a controlled system with the local gradients as inputs, and the one in Duchi et al. (2012), based on dual averaging and with the noticeable contribution of linking directly the convergence properties of the algorithm with the spectral properties of the communication network. It is worth to mention the distributed randomized Kaczmarz method Freris and Zouzias (2012) for quadratic cost functions and the Pairwise Equalizing Method (PEM) and Pairwise Bisectioning (Lu et al. (2011)), two gossip algorithms that are easy to implement, bypass limitations of the subgradient algorithms, and produce switched, nonlinear, networked dynamical systems that asymptotically converge to a global minima. We also notice the presence of various heuristic or ad-hoc methods, e.g., Van Ast et al. (2008), which convergence properties are unfortunately difficult to be characterized.

Statement of contributions

The previously described algorithms require different degrees of coordination among the agents. For example, DSMs may be implemented without requiring synchronized communications (see Nedić and Ozdaglar (2009)), while ADMM generally requires the preservation of the order of the operations.

Considering that a benchmark for the applicability of distributed algorithms relates to how much agents must coordinate, our aim is to consider the Newton-Raphson Consensus (NRC), a promising primal-based distributed optimizer originally proposed in Zanella et al. (2011, 2012b,d,c), and lessen its coordination requirements. More specifically, we propose an asynchronous version of it and prove its convergence properties (Zanella et al. (2012a)).

The distributed Newton-Raphson optimization procedure we propose addresses the exact minimization of smooth multidimensional convex separable problems, where the global function is a sum of private local costs. With respect to the categorization proposed before, the strategy exploits neither Lagrangian formalisms nor Laplacian estimation steps. More specifically, it is based on average consensus techniques Garin and Schenato (2011) and on the principle of separation of time-scales (Khalil, 2001, Chap. 11). The main idea is that agents compute and keep updated, by means of average consensus protocols, an approximated Newton-Raphson direction that is built from suitable Taylor expansions of the local costs. Simultaneously, agents move their local guesses towards the Newton-Raphson direction. It is proven that, if the rate of change of the local update steps is sufficiently slow to allow the consensus algorithm to converge, then the Synchronous Newton-Raphson Consensus (SNRC) algorithm exponentially converges to global minimizer.

The interest on the NRC technique can be motivated as follows. First of all, at the best of our knowledge it is the unique primal-based distributed algorithm whose estimates evolve as driven by a Newton-Raphson optimization scheme, and that can be implemented without requiring a-priori knowledge about the topology of the network (see, e.g., Jadbabaie et al. (2009)). Secondly, the NRC exploits average consensus algorithms (Fagnani and Zampieri, 2008; Garin and Schenato, 2011, and references therein). Thus it inherits all their favorable properties, like immediate adaptation to time-varying topologies and extreme simplicity of implementation (Hadjicostis and Charalambous (2012, 2011)).

Structure

The thesis is organized as follows.

Chapter 1 collects the notation used through the whole paper, gives some background theory useful for the dissertation, formulates the problem and reports some preliminary results necessary for the proofs.

Chapter 2 proposes the main optimization algorithm in a scalar scenario based on sensible intuitions and provides convergence and robustness results. Furthermore it generalizes this algorithm to multidimensional domains and offers some strategies to reduce communication and computational complexity. Nonetheless, it analytically characterize the convergence rates of the proposed algorithm and of a novel gradient-descent implementation under the posed simplificative assumptions. Lastly it compares the performance of the proposed algorithm with several distributed optimization strategies available in the literature via numerical simulations.

Chapter 3 describe an asynchronous version of the original algorithm. Again, proofs of convergence are given considering different kind of communication circumstances. To completion, some numerical comparisons are performed between well-known asynchronous methods, assessing the performance of the proposed strategy.

List of acronyms

WSAN	Wireless Sensors-Actuators Network
NCS	Networked Control System
LTI	Linear Time Invariant
NR	Newton-Raphson
NRC	Newton-Raphson Consensus
SNRC	Synchronous Newton-Raphson Consensus
JC	Jacobi Consensus
GDC	Gradient Descent Consensus
SFNRC	Synchronous Fast Newton-Raphson Consensus
DSM	Distributed Subgradient Method
DCM	Distributed Control Method
ADMM	Alternating Direction Method of Multipliers
ANRC	Asynchronous Newton-Raphson Consensus
AFNRC	Asynchronous Fast Newton-Raphson Consensus
ADSM	Asynchronous Distributed Subgradient Method
PEM	Pairwise Equalizing Method
AADMM	Asynchronous Alternating Direction Method of Multipliers

1 Basics

1.1 Background

In this section we introduce the fundamentals of the theories that will be extensively exploited throughout the thesis.

We briefly review some concepts about consensus and singular perturbation theories. Interested readers can more details in Garin and Schenato (2011); Saber and Murray (2003); Olfati-Saber et al. (2007a); Olshevsky and Tsitsiklis (2009a) and Kokotović et al. (1999); Skinner (2011). We refer to the thesis work of Bolognani (2011) for the concepts of consensus theory and to standard textbooks like Khalil (2001) for the concepts of nonlinear dynamical systems.

1.1.1 Consensus theory

In recent years, the academic community has devoted a huge research effort on distributed and network controlled systems, especially those employing mobile and wireless devices. While on the one side, this context has been nourished by the growing popularity of Wireless Sensors-Actuators Networks (WSANs) as a common tool for both the academic and the industrial worlds, on the other it has pushed expectations and perspective for intensive diffusion and application of this technology (see, e.g., Akyildiz and Kasimoglu (2004)).

A WSAN is commonly modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the ordered set $\mathcal{V} = \{1, \dots, N\}$ of the vertexes depicts N agents that communicate along the edges, i.e. the communication links specified by the set $\mathcal{E}$. This simple model for agent-to-agent communication does not take into account many of the phenomena that characterize the communication channels that are available in real-life NCS: communication noise, packet losses and data errors, quantization, just to mention few. In some specific cases (e.g. when agents move in the space and communication is limited to a given maximum range) the communication graph can also depend on the state of the agents.

A graph $\mathcal{G}$ is called *undirected* when $(i, j) \in \mathcal{E} \Rightarrow (j, i) \in \mathcal{E}$. An undirected graph is *complete* when it has an edge between every pair of vertexes. We denote with $\mathcal{N}_i := \{j \mid (i, j) \in \mathcal{E}, i \neq j\} \subseteq \mathcal{V}$ the set of *neighbors* of agent i and with $d_i = |\mathcal{N}_i|$ its cardinality, usually known as the *degree* of the agent.

In a directed graph, a *walk* on a graph is an alternating series of vertexes and edges, beginning and ending with a vertex, in which each edge is incident with the vertex immediately preceding it and the vertex immediately following it. A *path* is a walk in which all vertexes are distinct. We say that a graph is *connected* if every pair of agents (i, j) is connected by a path. In particular, a directed graph is called *strongly connected* if there is a path from each vertex in the graph to every other vertex.

The access of the agents to the communication media[1] can be regulated according to different strategies. The communication strategy among the agents in the network, which is also intended as an activation policy, can fall into two categories: *broadcast* communication, where one agent i transmits a message to all its neighbors $\mathcal{N}_i$ (receiving nothing in response), and *gossip* communication, where a agent i transmits a message to a specific agent $j \in \mathcal{N}_i$ (while the rest of the system is unaware of the exchange of data).

The gossip communication can be either *symmetric* (if the graph is undirected) or *asymmetric*, respectively meaning that the transmitting agent awaits for an answer from the receiver or not (i.e. the exchange of information is always bidirectional). Symmetry in the data transmission can be required by some algorithms to guarantee some particular features (e.g. average consensus schemes). However ensuring symmetry of operation when the channel suffers from non zero packet drop rate or error probability is a challenging goal to achieve. Broadcast communications, instead, can require more resources (e.g. energy, bandwidth, time) than the gossip strategy, because a larger number of agents is involved.

In addition, the communication (broadcast or gossip) can be *synchronous*, in which all agents communicate at the same time, or *asynchronous*, when agents are triggered one at a time. In this case the resulting sequence of agents can be *randomized* or *deterministic*, with respect to the activation of a agent

[1] Note that all the consensus algorithms rely upon the assumption that each agent transmits to its neighboring agents the precise value of its state; this means that the communication network is not constituted of quantized or noisy links.

to send messages to the neighbors. The synchronous approach usually returns global behaviors that are easier to analyze, but can be unpractical to achieve, as it requires a great level of coordination among the agents. In asynchronous algorithms, on the other hand, agents are triggered one at a time. Building a deterministic sequence on the set of agents can require some supervision in the system, but it can simplify the algorithm analysis. In randomized algorithms, instead, the sequence of activation of the agents is the realization of a random process. In this situation, the remarkable feature is that no coordination is necessary. However, the design (and analysis) of algorithms based on randomized asynchronous operation is generally more challenging.

Now suppose the WSAN endowed with a state[2] $x_i(t) : \mathbb{R} \to \mathbb{R}$, $\forall i \in \mathcal{V}$.

The *consensus problem* can be summarized as the problem of allowing the agents to reach an agreement regarding a quantity of common interest that is function of the state of all agents. The theoretical framework for posing and solving consensus problems for networked dynamic systems was formerly introduced in Fax and Murray (2004) and Olfati-Saber et al. (2007b).

A *consensus algorithm* is an interaction rule that specifies the information exchange between an agent $i \in \mathcal{V}$ and its neighbors $\mathcal{N}_i$ in the communication graph. Specifically, a consensus algorithm is presented in the form.

$$\boldsymbol{x}(t+1) = P(t)\,\boldsymbol{x}(t), \tag{1.1}$$

where $\boldsymbol{x}(0)$ is given, $\boldsymbol{x}(t) = [x_1(t), \ldots, x_N(t)]^T \in \mathbb{R}^N$ and $P(t) \in \mathbb{R}^{N\times N}$. In general, it is assumed that the matrix P is compatible with the graph $\mathcal{G}$, meaning that its associated graph $\mathcal{G}_P = (\mathcal{V}, \mathcal{E}_P)$ is such that $\mathcal{E}_P \subseteq \mathcal{E}$, with $\mathcal{E}_P := \{(j,i) \mid [P]_{ij} > 0\}$. Considering the formulation in (1.1), the consensus problem can be posed more formally as the following:

Definition 1 (consensus problem) Consider algorithm (1.1). $P(t)$ solves the consensus problem if $\lim_{t\to\infty} x_i(t) = \tau$, where τ is a generic function of the system initial state $x(0)$. $P(t)$ solves the *average consensus problem* if $\tau = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$. If $P(t)$ is a random variable, (1.1) solves the *probabilistic (average) consensus problem* if the limit above exists almost surely.

[2] In the more general case of a multidimensional state, the system state is obtained by stacking all the individual agents' states.

This definition includes a wide class of consensus strategies: strategies with a time-invariant matrix $P(t) = P$, deterministic time-varying strategies $P(t)$, and randomized strategies where $P(t)$ is drawn from some distributions on a set of stochastic[3] matrices.

The next theorems describe some sufficient conditions which guarantee deterministic and probabilistic (average) consensus.

Theorem 2 Consider a sequence of constant matrices $P(t) = P$. If the graph $\mathcal{G}_P$ is strongly connected, then P solves the consensus problem. If in addition P is doubly stochastic, then $\mathcal{G}_P$ is strongly connected and P solves the average consensus problem. Moreover, the convergence rate in both cases is exponential and it is given by second largest eigenvalue in absolute value of the matrix P.

Theorem 2 is concerned with constant consensus matrix, $P(t) = P, \forall t \geq 0$, i.e. the update strategy is the same at each time step t. In particular, we say that $P \in \mathbb{R}^{N \times N}$ is an *average consensus matrix* if it is stochastic, symmetric and it includes all edges, i.e., $p_{ij} > 0$ if and only if $(i, j) \in \mathcal{E}$. In fact, under the aforementioned hypotheses, the Perron-Frobenius theorem guarantees that $\lim_{t\to\infty} P^t = \frac{1}{N}\mathbb{1}\mathbb{1}^T$, which means that $\lim_{t\to\infty} \boldsymbol{x}(t) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}(0)$.

The next theorem addresses the consensus problem in a probabilistic context that arises from randomized communication strategies, as in Fagnani and Zampieri (2008), or networks subject to random external disturbances, such as link or agent failure.

Theorem 3 Consider a random independent and identically distributed sequence of stochastic matrices $\{P(t)\}_{t\geq 0}$ drawn according to some distribution from the set of all stochastic matrices, and the stochastic matrix $\overline{P} = \mathbb{E}[P(t)]$. If $\mathcal{G}_{\overline{P}}$ is strongly connected, and $[P]_{ii} > 0$ almost surely for any $i \in \mathcal{V}$, then the sequence $\{P(t)\}_{t\geq 0}$ achieves probabilistic consensus. If in addition all the $P(t)$ are doubly stochastic, they solve the probabilistic average consensus.

[3] We recall that $P \in \mathbb{R}^{N \times N}$ is a stochastic matrix if $[P]_{ij} \geq 0$ and $\sum_{j=1}^{N}[P]_{ij} = 1, \forall j$, i.e. each row sums to unity. Equivalently, a matrix P is stochastic if its elements are non-negative, it is such that $P\mathbb{1} = \mathbb{1}$ where $\mathbb{1} := [1\ 1\ \cdots\ 1]^T \in \mathbb{R}^N$, and it is consistent with the graph $\mathcal{G}$, in the sense that each entry p_{ij} of P is $p_{ij} > 0$ only if $(i, j) \in \mathcal{E}$. A stochastic matrix P is said doubly stochastic if also $\sum_{i=1}^{N}[P]_{ij} = 1, \forall i$, i.e. each column sums to unity. Clearly if a stochastic matrix is symmetric then it is also doubly stochastic.

According to the communication strategies previously presented, various consensus algorithms can be implemented, resulting in different $\{P(t)\}_{t\geq 0}$.

In the broadcast strategy the consensus matrix $P(t)$ when an agent i transmits at time t is given by:

$$[P(t)]_{mn} = \begin{cases} 1 & \text{if } m = n \notin \mathcal{N}_i \\ 1 - w & \text{if } m = n \in \mathcal{N}_i \\ w & \text{if } m \in \mathcal{N}_i,\ n = i \\ 0 & \text{otherwise} \end{cases}$$

where $w \in (0, 1)$ is a tuning parameter.

In the symmetric gossip, when the edge (i, j) is selected, the consensus matrix $P(t)$ is given by:

$$[P(t)]_{mn} = \begin{cases} 1 & \text{se } m = n \neq j \text{ and } m = n \neq i \\ 1 - w & \text{if } m = n = j \text{ or } m = n = i \\ w & \text{if } (m, n) = (i, j) \text{ or } (m, n) = (j, i) \\ 0 & \text{otherwise} \end{cases}$$

Typically $w = 1/2$, and the consensus matrices defined above are based on the assumption that there is no link failure during the communication. Since agents usually communicate using a wireless channel, the transmission is not reliable, i.e. there is a non-zero packet loss probability. Let this communication unreliability modeled with the *connectivity matrix* $C \in \mathbb{R}^{N\times N}$, where $[C]_{ij} = c_{ij} \in [0\,1]$ is the probability that agent j can successfully transmit a message to agent i. Since the wireless channel is approximately symmetric, we assume that $C = C^T$ and $c_{ii} = 1$, $\forall i$. We define the *c-connectivity graph* $\mathcal{G}_C = (\mathcal{N}, \mathcal{E}_C)$ associated to the connectivity matrix C as the graph s.t. $(i, j) \in \mathcal{E}_C$ if and only if $c_{ij} \geq c$. This graph is undirected (i.e. $\forall i, j \in \mathcal{N}\ (i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$) since the matrix C is symmetric.

When link failure happens in broadcast communication, the matrix $P(t)$ needs to be modified with $[P(t)]_{jj} = 1$, $[P(t)]_{ji} = 0$, for some t. Instead, when it happens in symmetric gossip, there is no communication at all, and then no update is performed (i.e. $P(t) = I$ for some $t \geq 0$).

Based on the randomized communication modeling with link failure probability, it results that the expected consensus matrix $\overline{P} = \mathbb{E}[P(t)]$ generated for the broadcast strategy is given by:

$$[\overline{P}]_{mn} = \begin{cases} 1 - \dfrac{c\,w\,d_n}{N} & \text{if } \ m = n \\ \dfrac{c\,w}{N} & \text{if } m \in \mathcal{N}_n \\ 0 & \text{otherwise} \end{cases}$$

Note that $\overline{P} = (\overline{P})^T$ is symmetric and hence doubly stochastic, although the matrices P_i are never symmetric. Moreover $\mathcal{G}_{\overline{P}} = \mathcal{G}_C$, i.e. the graph associated with the expected consensus matrix $\overline{P}$, coincides with the underlying communication graph $\mathcal{G}_C$. As assured by Theorem 3, if $\mathcal{G}_{\overline{P}}$ is strongly connected then the algorithm achieves consensus with probabilty 1. If moreover the matrices $P(t)$ are all doubly stochastic then the algorithm achieves average consensus with probability 1. Therefore, if $\mathcal{G}_C$ is strongly connected, then this implies that the randomized broadcast guarantees probabilistic consensus. Although, it does not guarantee average consensus for all possible realizations of $P(t)$. Even if the gossip matrices are not doubly stochastic, the expected consensus matrix $\overline{P}$ is doubly stochastic, therefore the elements converge to the average of the initial conditions in mean sense.

The expected consensus matrix for the symmetric gossip is given by:

$$[\overline{P}]_{mn} = \begin{cases} 1 - \sum_{i \in \mathcal{N}_i} \dfrac{2\,c\,w}{N(d_n + d_i)} & \text{if } \ m = n \\ \dfrac{2\,c\,w}{N(d_m + d_n)} & \text{if } (m,n) \in \mathcal{E}_C, \ m \neq n \\ 0 & \text{otherwise} \end{cases}$$

Obviously $\overline{P} = (\overline{P})^T$ since all the gossip matrices $P(t)$, $t \geq 0$, from which the distribution is drawn are symmetric by construction. Similarly to the broadcast, we have $\mathcal{G}_{\overline{P}} = \mathcal{G}_C$. Thus, if $\mathcal{G}_C$ is strongly connected, the randomized symmetric gossip guarantees probabilistic average consensus, i.e., almost surely

$$\lim_{t \to \infty} x_i(t) = \frac{1}{N} \sum_{i=1}^{N} x_i(0) = \frac{1}{N} \mathbb{1} \boldsymbol{x}(0) \quad \forall i \in \mathcal{V}.$$

Compared to the randomized broadcast, the randomized symmetric gossip guarantees average consensus for all realizations, but it is more expensive from a communication point of view. Indeed, at least two packets with reception acknowledge need to be exchanged at every step of the consensus iteration, while for the broadcast only one is needed (with no acknowledge). Furthermore, with the symmetric gossip just two agents receive informations while with the broadcast strategy all the neighbor agents of the broadcaster do.

An extensive analysis has been done in the literature about the convergence rate of the consensus algorithms (see, e.g., Olshevsky and Tsitsiklis (2009b); Blondel et al. (2005); Nedić and Ozdaglar (2010)). For the scope of this thesis, it is sufficient to know that, for a (strongly) connected undirected graph, the consensus is globally exponentially reached with a speed that is faster or equal to the second largest eigenvalue of $(P + P^T)/2$.

1.1.2 Singular perturbation theory

Suppose that a given nonlinear system is posed in the form $\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \varepsilon)$, where $\boldsymbol{x} \in \mathbb{R}^N$ represents the global state, $t \in \mathbb{R}_+$ the time, and $\varepsilon \in \mathbb{R}$ is a parameter chosen opportunely small. Suppose that, under certain conditions, the equation has an exact solution $\boldsymbol{x}(t, \varepsilon)$. It is well known that, dealing with nonlinear differential equations, there are only a limited number of special classes of differential equations that allow to have (exact) analytic solutions in closed-form. In general, we need to resort to approximate solutions.

Asymptotic methods are one of the well-known approximation methodology for the analysis of this kind of nonlinear system. An asymptotic method aims to obtain a solution $\widetilde{\boldsymbol{x}}(t, \varepsilon)$, which is an approximation of $\boldsymbol{x}(t, \varepsilon)$, such that the approximation error $\boldsymbol{x}(t, \varepsilon) - \widetilde{\boldsymbol{x}}(t, \varepsilon)$ is small (for small $|\varepsilon|$ and in some norm) and that $\widetilde{\boldsymbol{x}}(t, \varepsilon)$ is expressed in terms of equations simpler than the original ones. Usually, the solution of this nonlinear system is characterized by the fact that some variables move in time faster than other variables. The *singular perturbation method* deals with the interaction of this two types of variables, namely *slow* and *fast* behaviors, in a discontinuous dependence of system properties on the perturbation parameter ε.

The singular perturbation method faces the so-called *standard singular perturbation model*

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{z}, \varepsilon), \quad \boldsymbol{x}(t_0) = \boldsymbol{\xi}(\varepsilon) \tag{1.2}$$

$$\varepsilon \dot{\boldsymbol{z}} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, \varepsilon), \quad \boldsymbol{z}(t_0) = \boldsymbol{\eta}(\varepsilon) \tag{1.3}$$

where $\boldsymbol{\xi}(\varepsilon)$ and $\boldsymbol{\eta}(\varepsilon)$ depend smoothly on ε and $t_0 \in [0, t_1)$. In this context, setting $\varepsilon = 0$ causes an abrupt change in the dynamic properties of the system, as the differential equation $\varepsilon \dot{\boldsymbol{z}} = \boldsymbol{g}$ degenerates into $\boldsymbol{0} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, 0)$. The meaning behind the singular perturbation theory is that analyzing these perturbations in separate time scales let to avoid the discontinuity of solutions generated by the singularities.

In the following, recalling slavishly the concepts in Khalil (2001), we define the standard singular perturbation model, presenting the two time-scale properties of this model. Moreover, we give a trajectory approximation, that is based on the decomposition into two different models related to the slow and fast dynamics: the *reduced model* and *boundary-layer model.* Lastly, we mention a a result of conceptual importance with regard to the exponential stability of the nonautonomous system described by (1.2)-(1.3).

We have already introduced the singular perturbation model with the equations (1.2) and (1.3), where the dynamical system is denoted by the fact that the derivatives of some states are multiplied by a small positive parameter ε, assumed to be small. Starting from that, we assume that the functions $\boldsymbol{f}$ and $\boldsymbol{g}$ are continuously differentiable for $(t, \boldsymbol{x}, \boldsymbol{z}, \varepsilon) \in [0, t_1] \times D_{\boldsymbol{x}} \times D_{\boldsymbol{z}} \times [0, \varepsilon_0]$, where $D_{\boldsymbol{x}} \subset \mathbb{R}^N$ and $D_{\boldsymbol{z}} \subset \mathbb{R}^M$ are open connected sets. Setting $\varepsilon = 0$ in (1.2) and (1.3), the differential equation (1.3) degenerates into

$$\mathbf{0} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, 0), \tag{1.4}$$

which means that the dimension of the state equation is reduced from $N + M$ to N. Now, the model (1.2)-(1.3) is said to be in *standard* form if (1.4) has $r \geq 1$ isolated real roots

$$\boldsymbol{z} = \boldsymbol{h}_i(t, \boldsymbol{x}), \quad i = l, 2, \dots, r \tag{1.5}$$

for each $(t, \boldsymbol{x}) \in [0, t_1] \times D_{\boldsymbol{x}}$. This fact ensures that each i-th root of (1.4) will correspond to a N-dimensional i-th reduced model

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{h}_i(t, \boldsymbol{x}), 0) \tag{1.6}$$

obtained by substitution from (1.5) and (1.2). The model (1.6) is known as the *slow model* and it is also called a *quasi-steady-state model*, because $\boldsymbol{z}$, whose velocity $\dot{\boldsymbol{z}} = \boldsymbol{g}/\varepsilon$ can be large when ε is small and $\boldsymbol{g} \neq \mathbf{0}$, may rapidly converge to a root of (1.4) (which is the equilibrium of (1.3)).

Singular perturbations cause a multitime-scale behavior of dynamical systems characterized by the presence of slow and fast transients in the system's response to external stimuli. In other words, the slow response corresponds approximatively to the reduced model (1.6), while the fast response is given by the discrepancy between the behaviour of the reduced model and that of the full model (1.2)-(1.3). To better understand this point, consider to solve the full problem[4] (1.2)-(1.3). Naturally, we retain the initial state for $\boldsymbol{x}$ to obtain the reduced problem[5]

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{h}(t, \boldsymbol{x}), 0), \quad \boldsymbol{x}(t_0) = \boldsymbol{\xi}_0 := \boldsymbol{\xi}(0) \tag{1.7}$$

[4] When we define the corresponding problem for the reduced model (1.6), we can only specify N initial conditions.

[5] From now on we drop the subscript i from $\boldsymbol{h}$. It will be clear from the context which root of (1.4) we are considering.

Since $\boldsymbol{z}$ has been substituted by its quasi-steady-state $\boldsymbol{h}(t, \boldsymbol{x})$, the only obtainable information about it by solving (1.7) is to compute $\overline{\boldsymbol{z}}(t) := \boldsymbol{h}(t, \overline{\boldsymbol{x}}(t))$, where $\overline{\boldsymbol{x}}(t)$ denote the solution of (1.7). $\overline{\boldsymbol{z}}(t)$ describes the quasi-steady-state behavior of $\boldsymbol{z}$ when $\boldsymbol{x} = \overline{\boldsymbol{x}}$. The quasi-steady-state $\overline{\boldsymbol{z}}$ is not free to start from a prescribed value, as instead occurs to the original variable $\boldsymbol{z}$ starting at t_0 from a prescribed $\boldsymbol{\eta}(\varepsilon)$, and there may be a large gap between its initial value $\overline{\boldsymbol{z}}(t_0) = \boldsymbol{h}(t_0, \boldsymbol{\xi}_0)$ and the initial state $\boldsymbol{\eta}(\varepsilon)$. Hence, $\overline{\boldsymbol{z}}(t)$ cannot be a uniform approximation of $\boldsymbol{z}(t, \varepsilon)$.

The best we can expect is that the estimate $\boldsymbol{z}(t, \varepsilon) - \overline{\boldsymbol{z}}(t) = O(\varepsilon)$ will hold on an interval excluding t_0, that is, for $t \in [t_b, t_1]$, where $t_b > t_0$. Unfortunately, we cannot translate the order of magnitude[6] statement into a numerical bound on the error. Knowing that the error is $O(\varepsilon)$ means that its norm is less than $k|\varepsilon|$ for some positive constant k that is independent of ε, therefore, for sufficiently small $|\varepsilon|$, the error will be small.

On the other hand, it is reasonable to expect $\boldsymbol{x}(t, \varepsilon) - \overline{\boldsymbol{x}}(t) = O(\varepsilon)$ to hold uniformly for all $t \in [t_0, t_1]$, since $\boldsymbol{x}(t_0, \varepsilon) - \overline{\boldsymbol{x}}(t_0) = \boldsymbol{\xi}(\varepsilon) - \boldsymbol{\xi}(0) = O(\varepsilon)$. If the error $\boldsymbol{z}(t, \varepsilon) - \overline{\boldsymbol{z}}(t)$ is indeed $O(\varepsilon)$ over $[t_b, t_1]$, then it must be true that during the initial interval $[t_0, t_b]$, corresponding to the boundary-layer dynamics, the variable $\boldsymbol{z}$ approaches $\overline{\boldsymbol{z}}$. Since the velocity of $\boldsymbol{z}$ can be high[7], because $\dot{\boldsymbol{z}} = \boldsymbol{g}/\varepsilon$, it should be clear that we cannot expect $\boldsymbol{z}$ to converge to its quasi-steady-state $\bar{\boldsymbol{z}}$, unless certain stability conditions are satisfied. We will see through the forthcoming analysis what kind of conditions are necessary for the stability. To do that, it is more convenient to perform the change of variables

$$\boldsymbol{y} = \boldsymbol{z} - \boldsymbol{h}(t, \boldsymbol{x}) \tag{1.8}$$

that shifts the quasi-steady-state of $\boldsymbol{z}$ to the origin. In the new variables $(\boldsymbol{x}, \boldsymbol{y})$, the full problem is

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon) \tag{1.9}$$

$$\varepsilon\dot{\boldsymbol{y}} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon) - \varepsilon\frac{\partial \boldsymbol{h}}{\partial t} - \varepsilon\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon) \tag{1.10}$$

with initial conditions

$$\begin{aligned} \boldsymbol{x}(t_0) &= \boldsymbol{\xi}(\varepsilon) \\ \boldsymbol{y}(t_0) &= \boldsymbol{\eta}(\varepsilon) - \boldsymbol{h}(t_0, \boldsymbol{\xi}(\varepsilon)) \end{aligned}$$

[6] $\delta_1(\varepsilon) = O(\delta_2(\varepsilon))$ if there exist constants $k, c > 0$ such that $|\delta_1(\varepsilon)| \leq k|\delta_2(\varepsilon)|$, $\forall|\varepsilon| < c$.

[7] Indeed, setting $\varepsilon = 0$ in (1.3), the transient of $\boldsymbol{z}$ becomes instantaneous whenever $\boldsymbol{g} \neq \boldsymbol{0}$.

The quasi-steady-state of (1.10) is now $\boldsymbol{y} = \mathbf{0}$, which when substituted into (1.9) results in the reduced model (1.7). To analyze (1.10), let us note that $\varepsilon\dot{\boldsymbol{y}}$ may remain finite even when ε tends to zero and $\dot{\boldsymbol{y}}$ tends to infinity. We set

$$\varepsilon\frac{d\boldsymbol{y}}{dt} = \frac{d\boldsymbol{y}}{d\tau};$$

hence, $d\tau/dt = 1/\varepsilon$ and use $\tau = 0$ as the initial value at $t = t_0$. The new time variable $\tau = (t - t_0)/\varepsilon$ is *stretched*, meaning that if ε tends to zero, τ tends to infinity even for finite t, only slightly larger than t_0, by a fixed difference. In this new time scale, (1.10) is given by

$$\frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon) - \varepsilon\frac{\partial \boldsymbol{h}}{\partial t} - \varepsilon\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon) \tag{1.11}$$

with $\boldsymbol{y}(0) = \boldsymbol{\eta}(\varepsilon) - \boldsymbol{h}(t_0, \boldsymbol{\xi}(\varepsilon))$. What happens is that t and $\boldsymbol{x}$ in the foregoing equation will be slowly varying since, in the τ time scale, they are given by

$$t = t_0 + \varepsilon\tau, \quad \boldsymbol{x} = \boldsymbol{x}(t_0 + \varepsilon\tau, \varepsilon)$$

Requiring $\varepsilon = 0$ freezes these variables at $t = t_0$ and $x = \xi_0$, and reduces (1.11) to the autonomous system

$$\frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(t_0, \boldsymbol{\xi}_0, \boldsymbol{y}+\boldsymbol{h}(t_0, \boldsymbol{\xi}_0), 0),\ \boldsymbol{y}(0) = \boldsymbol{\eta}(0)-\boldsymbol{h}(t_0, \boldsymbol{\xi}_0) := \boldsymbol{\eta}_0-\boldsymbol{h}(t_0, \boldsymbol{\xi}_0) \tag{1.12}$$

which has equilibrium at $\boldsymbol{y} = \mathbf{0}$. If this equilibrium point is asymptotically stable and $\boldsymbol{y}(0)$ belongs to its region of attraction, it is reasonable to expect that the solution of (1.12) will reach an $O(\varepsilon)$ neighborhood of the origin during the boundary-layer interval. Beyond this interval, it is necessary a stability property that guarantees that $\boldsymbol{y}(\tau)$ will remain close to zero, while the slowly varying parameters $(t, \boldsymbol{x})$ move away from their initial conditions $(t_0, \boldsymbol{\xi}_0)$. This overall trend can be analyze allowing the frozen parameters to take values in the region of the slowly varying parameters $(t, \boldsymbol{x})$. Assume that the solution $\overline{\boldsymbol{x}}(t)$ of the reduced problem is defined for $t \in [0, t_1]$ and $\overline{\boldsymbol{x}}(t) \in D_{\boldsymbol{x}} \subset \mathbb{R}^N$, for some domain $D_{\boldsymbol{x}}$. Rewrite (1.12) in the form of the boundary-layer model (or boundary-layer system)

$$\frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), 0) \tag{1.13}$$

where $(t, \boldsymbol{x}) \in [0, t_1] \times D_{\boldsymbol{x}}$ are treated as fixed parameters.

The crucial stability property for (1.13) is the exponential stability of its origin, uniformly in the frozen parameters, as stated in the next definition.

Definition 4 The equilibrium point $\boldsymbol{y} = \mathbf{0}$ of the boundary-layer system (1.13) is exponentially stable, uniformly in $(t, \boldsymbol{x}) \in [0, t_1] \times D_{\boldsymbol{x}}$, if there exist positive constants k, γ, and ρ_0 such that the solutions of (1.13) satisfy

$$\|\boldsymbol{y}(\tau)\| \leq k\|\boldsymbol{y}(0)\|e^{-\gamma\tau},\ \forall\|\boldsymbol{y}(0)\| \leq \rho_0,\ \forall(t, \boldsymbol{x}) \in [0, t_1] \times D_{\boldsymbol{x}}\,\forall\tau \geq 0 \quad (1.14)$$

The following theorem, widely known as Tikhonov's theorem, is of major importance in nonlinear singular perturbation theory, because it implies that whether the solution of system (1.2)-(1.3) can be approximated by the solutions of the reduced system and the boundary-layer system depends on the exponential stability of the boundary-layer system, and (or) the stability of the reduced system.

Theorem 5 Consider the singular perturbation problem of (1.2) and (1.3) and let $\boldsymbol{z} = \boldsymbol{h}(t, \boldsymbol{x})$ be an isolated root of (1.13). Assume that the following conditions are satisfied far all $[t, \boldsymbol{x}, \boldsymbol{z} - \boldsymbol{h}(t, \boldsymbol{x}), \varepsilon] \in [0, t_1] \times D_{\boldsymbol{x}} \times D_{\boldsymbol{y}} \times [0, \varepsilon_0]$ for some domains $D_{\boldsymbol{x}} \subset \mathbb{R}^N$ and $D_{\boldsymbol{y}} \subset Reals^M$, in which $D_{\boldsymbol{x}}$ is convex and $D_{\boldsymbol{y}}$ contains the origin:

- The functions $\boldsymbol{f}, \boldsymbol{g}$, their first partial derivatives with respect t o $(\boldsymbol{x}, \boldsymbol{z}, \varepsilon)$, and the first partial derivative of $\boldsymbol{g}$ with respect to t are continuous; the function $\boldsymbol{h}(t, \boldsymbol{x})$ and the Jacobian $[\partial \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, 0)/\partial \boldsymbol{z}]$ have continuous first partial derivatives with respect t o their arguments; the initial data $\boldsymbol{\xi}(\varepsilon)$ and $\boldsymbol{\eta}(\varepsilon)$ are smooth functions of ε.

- The reduced problem (1.7) has a unique solution $\overline{\boldsymbol{x}}(t) \in S$, for $t \in [t_0, t_1]$ where S is a compact subset of $D_{\boldsymbol{x}}$.

- The origin is an exponentially stable equilibrium point of the boundary-layer model (1.13), uniformly in $(t, \boldsymbol{x})$; let $\mathcal{R}_{\boldsymbol{y}} \subset D_{\boldsymbol{y}}$ be the region of attraction of 1.12 and $\Omega_{\boldsymbol{y}}$ be a compact subset of $\mathcal{R}_{\boldsymbol{y}}$.

Then, there exists a positive constant ε^* such that for all $\boldsymbol{\eta}_0 - \boldsymbol{h}(t_0, \varepsilon_0) \in \Omega_{\boldsymbol{y}}$ and $0 < \varepsilon < \varepsilon^*$, the singular perturbation problem of (1.2) and (1.3) has a unique solution $\boldsymbol{x}(t, \varepsilon)$, $\boldsymbol{z}(t, \varepsilon)$ on $[t_0, t_1]$, and

$$\boldsymbol{x}(t, \varepsilon) - \overline{\boldsymbol{x}}(t) = O(\varepsilon) \quad (1.15)$$

$$\boldsymbol{z}(t, \varepsilon) - \boldsymbol{h}(t, \overline{\boldsymbol{x}}(t)) - \hat{\boldsymbol{y}}(t/\varepsilon) = O(\varepsilon) \quad (1.16)$$

hold uniformly for $t \in [t_0, t_1]$, where $\hat{\boldsymbol{y}}(\tau)$ is the solution of the boundary-layer model (1.13). Moreover, given any $t_b > t_0$, there is $\varepsilon^{**} \le \varepsilon^*$ such that $\boldsymbol{z}(t,\varepsilon) - \boldsymbol{h}(t, \overline{\boldsymbol{x}}(t)) = O(\varepsilon)$ holds uniformly for $t \in [t_b, t_1]$ whenever $\varepsilon < \varepsilon^{**}$.

Theorem 6, that follows, is important because it establishes robustness of exponential stability to unmodeled fast dynamics. The technicalities of assuming exponential stability instead of only asymptotic stability, or assuming that exponential stability holds uniformly, are quite reasonable in most applications. When the origin of the reduced model is exponentially stable, Theorem 6 assures us that the origin of the actual system will be exponentially stable (provided the neglected fast dynamics are sufficiently fast).

Theorem 6 Consider the singularly perturbed system

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{z}, \varepsilon) \tag{1.17}$$

$$\varepsilon\dot{\boldsymbol{z}} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, \varepsilon) \tag{1.18}$$

Assume that the following assumptions are satisfied for all

$$(t, \boldsymbol{x}, \varepsilon) \in [0, \infty) \times B_r \times [0, \varepsilon_0]$$

- $\boldsymbol{f}(t, \mathbf{0}, \mathbf{0}, \varepsilon) = \mathbf{0}$ and $\boldsymbol{g}(t, \mathbf{0}, \mathbf{0}, \varepsilon) = \mathbf{0}$.
- The equation $\mathbf{0} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{z}, 0)$ has an isolated root $\boldsymbol{z} = \boldsymbol{h}(t, \boldsymbol{x})$ such that $\boldsymbol{h}(t, \mathbf{0}) = \mathbf{0}$.
- The functions $\boldsymbol{f}$, $\boldsymbol{g}$, $\boldsymbol{h}$, and their partial derivatives up to the second order are bounded far $\boldsymbol{z} - \boldsymbol{h}(t, \boldsymbol{x}) \in B_P\rho$.
- The origin of the reduced system

  $$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{h}(t, \boldsymbol{x}), 0)$$

  is exponentially stable.
- The origin of the boundary-layer system

  $$\frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{h}(t, \boldsymbol{x}), 0)$$

  is exponentially stable, uniformly in $(t, \boldsymbol{x})$.

Then, there exists $\varepsilon^* > 0$ such that for all $\varepsilon < \varepsilon^*$, the origin of (1.17)-(1.18) is exponentially stable.

1.2 Notation

We model the communication network as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose vertexes $\mathcal{V} := \{1, 2, \dots, N\}$ represent the agents and whose edges $(i, j) \in \mathcal{E}$ represent the available communication links. We denote with $\mathcal{N}_i = \{j \mid (i, j) \in \mathcal{E}, i \neq j\}$ the set of neighbors of node i. Unless stated differently, we assume that any graph G is undirected, connected, not time-varying and includes all self-arcs, i.e. $(i, i) \in \mathcal{E}, \forall i \in \mathcal{V}$.

We use to indicate with $P \in \mathbb{R}^{N \times N}$ an average consensus matrix. We recall that if P is stochastic, symmetric and includes all edges, i.e., $p_{ij} > 0$ if and only if $(i, j) \in \mathcal{E}$, it is guaranteed that $\lim_{k \to \infty} P^k = \frac{1}{N} \mathbb{1} \mathbb{1}^T$. We assume that the spectrum of P, $\operatorname{eig}(P) = \{\lambda_1 = 1, \lambda_2, \dots, \lambda_N\}$, is known, and that the eigenvalues are sorted in decreasing order. To compact the notation, we let $\Lambda := \operatorname{diag}[\lambda_1, \dots, \lambda_N]$. We will indicate with

$$\sigma := \max_{i, \lambda_i \neq 1} |\lambda_i|$$

the *essential spectral radius* of P, which under the connectivity hypothesis of the communication graph is s.t. $\sigma < 1$. $\rho := 1 - \sigma$ denotes instead its *spectral gap*. We also assume that no communication or quantization errors occur (i.e., information can be exchanged with infinite precision).

We use plain italic lower case fonts to indicate scalar quantities or functions whose range is a scalar (e.g., x, y, z), bold italic lower case fonts to indicate vectorial quantities or functions whose range is vectorial (e.g., $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$), plain italic capital letters to denote matrices (sometimes even scalar parameters).

Nonetheless, plain italic capital letters refer to outcomes of Kronecker products (e.g., X, Y, Z). We use Kronecker products also to indicate component-wise consensus steps. That is, if

$$A_i = \begin{bmatrix} a_{11}^{(i)} & \cdots & a_{1L}^{(i)} \\ \vdots & & \vdots \\ a_{M1}^{(i)} & \cdots & a_{ML}^{(i)} \end{bmatrix} \qquad i = 1, \dots, N$$

is a generic $M \times L$ matrix associated to agent i, $i = 1, \dots, N$, and if these agents want to distributedly compute $\frac{1}{N} \sum_{i=1}^{N} A_i$ by means of the communication matrix P, then to indicate the whole set of the single component-wise steps

$$\begin{bmatrix} a_{ml}^{(1)}(k+1) \\ \vdots \\ a_{ml}^{(N)}(k+1) \end{bmatrix} = P \begin{bmatrix} a_{ml}^{(1)}(k) \\ \vdots \\ a_{ml}^{(N)}(k) \end{bmatrix} \qquad \begin{array}{c} m = 1, \dots, M \\ l = 1, \dots, L \end{array} \tag{1.19}$$

we use the equivalent matrix notation

$$\begin{bmatrix} A_1(k+1) \\ \vdots \\ A_N(k+1) \end{bmatrix} = (P \otimes I_M) \begin{bmatrix} A_1(k) \\ \vdots \\ A_N(k) \end{bmatrix} \tag{1.20}$$

where I_M is the identity in $\mathbb{R}^{M \times M}$ and $\otimes$ is the Kronecker product. Notice that this notation is suited also for vectorial quantities.

The notation $\operatorname{diag}[\boldsymbol{v}]$, where $\boldsymbol{v} = [v_1 \ \cdots \ v_N]$ is a generic vector, denotes a diagonal matrix with $v_1, \dots, v_N$ on its diagonal. $I := \operatorname{diag}[\mathbb{1}]$.

We use the usual symbol $\odot$ to indicate the component-wise Hadamard product, and the fraction bars to indicate Hadamard divisions, i.e. the component-wise division of vectors of vectorial functions (e.g., if $\boldsymbol{a} = [a_1, \dots, a_N]^T$ and $\boldsymbol{b} = [b_1, \dots, b_N]^T$ then $\frac{\boldsymbol{a}}{\boldsymbol{b}} = \begin{bmatrix} \frac{a_1}{b_1} & \dots & \frac{a_N}{b_N} \end{bmatrix}^T$). Also, M indicates the dimensionality of the domain, k a discrete time index, t a continuous time index.

If f is a scalar function, we denote differentiation with

$$f' := \frac{df}{dx} \quad \text{and} \quad f'' := \frac{d^2 f}{dx^2}$$

when the domain is scalar, and with ∇ operators when it is not. The symbol $\| \cdot \|$ will denote the Euclidean norm in $\mathbb{R}^N$, and the induced norm in $\mathbb{R}^{N \times N}$.

All the additional notation is collected in the following:

(scalar case)

$$
\begin{aligned}
g_i\big(x_i(k)\big) &:= f_i''\big(x_i(k)\big)x_i(k) - f_i'\big(x_i(k)\big)\\
h_i\big(x_i(k)\big) &:= f_i''\big(x_i(k)\big)\\
\boldsymbol{x}(k) &:= [x_1(k) \ \cdots \ x_N(k)]^T \qquad \textit{(data from all the agents)}\\
\boldsymbol{y}(k) &:= [y_1(k) \ \cdots \ y_N(k)]^T\\
\boldsymbol{z}(k) &:= [z_1(k) \ \cdots \ z_N(k)]^T\\
\boldsymbol{g}\big(\boldsymbol{x}(k)\big) &:= \big[g_1\big(x_1(k)\big) \ \cdots \ g_N\big(x_N(k)\big)\big]^T\\
\boldsymbol{h}\big(\boldsymbol{x}(k)\big) &:= \big[h_1\big(x_1(k)\big) \ \cdots \ h_N\big(x_N(k)\big)\big]^T\\
\boldsymbol{f}'(\boldsymbol{x}) &:= [f_1'(x_1), \ldots, f_N'(x_N)]^T
\end{aligned}
$$

(vectorial case)

$$
\begin{aligned}
\boldsymbol{x}_i(k) &:= [x_{i,1}(k) \ \cdots \ x_{i,M}(k)]^T \in \mathbb{R}^M \qquad \textit{(data from a single agent)}\\
X(k) &:= \big[\boldsymbol{x}_1(k)^T \ \cdots \ \boldsymbol{x}_N(k)^T\big]^T \in \mathbb{R}^{MN} \qquad \textit{(data from all the agents)}\\
\nabla f\big(\boldsymbol{x}_i(k)\big) &:= \left[\left.\frac{\partial f}{\partial x_1}\right|_{\boldsymbol{x}_i(k)} \ \cdots \ \left.\frac{\partial f}{\partial x_M}\right|_{\boldsymbol{x}_i(k)} \right]^T \in \mathbb{R}^M\\
\nabla^2 f\left(\boldsymbol{x}_i(k)\right) &:= \begin{bmatrix} \ddots & \vdots & \\ \cdots & \left.\dfrac{\partial^2 f}{\partial x_m \partial x_n}\right|_{\boldsymbol{x}_i(k)} & \cdots \\ & \vdots & \ddots \end{bmatrix} \in \mathbb{R}^{M\times M}\\
Y(k) &:= \big[\boldsymbol{y}_1(k)^T \ \cdots \ \boldsymbol{y}_N(k)^T \ \big]^T \in \mathbb{R}^{MN}\\
\boldsymbol{Z}(k) &:= \big[Z_1(k)^T \ \cdots \ Z_N(k)^T \ \big]^T \in \mathbb{R}^{MN\times M}\\
H_i\big(\boldsymbol{x}_i(k)\big) &:= \textit{Equations } (2.60) \textit{ or } (2.61) \textit{ or } (2.62) \in \mathbb{R}^{M\times M}. \textit{ See Algorithm 3}\\
\boldsymbol{H}\big(X(k)\big) &:= \Big[H_1\big(\boldsymbol{x}_1(k)\big)^T \ \cdots \ H_N\big(\boldsymbol{x}_N(k)\big)^T\Big]^T \in \mathbb{R}^{MN\times M}\\
\boldsymbol{g}_i\big(\boldsymbol{x}_i(k)\big) &:= H_i\big(\boldsymbol{x}_i(k)\big)\boldsymbol{x}_i(k) - \nabla f_i\big(\boldsymbol{x}_i(k)\big) \in \mathbb{R}^N\\
G\big(X(k)\big) &:= \Big[\boldsymbol{g}_1\big(\boldsymbol{x}_1(k)\big)^T \ \cdots \ \boldsymbol{g}_N\big(\boldsymbol{x}_N(k)\big)^T\Big]^T \in \mathbb{R}^{MN}
\end{aligned}
$$

1.3 Problem formulation

We study the problem of unconstrained distributed optimization in the context of multi-agents systems (subject to limited communication connectivity). We focus on the minimization of a sum of cost functions, where each component of the global function is available only to a specific agent and can thus be seen as a private local cost. The agents need to cooperate to compute the minimizer of the sum of all costs. More formally, we deal with the following problem:

Problem 7 Assume that N agents are part of a WSAN modeled as a a graph that is undirected, connected and not time-varying, in a (randomized) symmetric gossip type communication network. The agents are endowed with cost functions $\boldsymbol{f}_i : \mathbb{R}^M \mapsto \mathbb{R}$ so that

$$\overline{\boldsymbol{f}} : \mathbb{R} \mapsto \mathbb{R}, \qquad \overline{\boldsymbol{f}}\left(\boldsymbol{x}\right) := \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{f}_i\left(\boldsymbol{x}\right) \tag{1.21}$$

is a well-defined global cost.

The aim of the agents is to cooperate and distributedly compute the minimizer of $\overline{\boldsymbol{f}}$, namely

$$\boldsymbol{x}^* := \arg\min_{\boldsymbol{x}} \overline{\boldsymbol{f}}\left(\boldsymbol{x}\right). \tag{1.22}$$

We now enforce the following simplificative assumptions in the scalar domain, stated in general for the multidimensional case, see, e.g., Xiao and Boyd (2006); Ho et al. (1980), and valid throughout the rest of the paper:

Assumption 8 (Convexity) The function $\overline{f} : \mathbb{R} \mapsto \mathbb{R}$ defined in (1.21) is of class $\mathcal{C}^2$, coercive, and strictly convex, i.e., $\overline{f}''(x) := \dfrac{d^2\overline{f}(x)}{dx^2} > 0$, $\forall x \in \mathbb{R}$ and so that x^* in (1.22) exists and is unique.

The positive second derivative is a mild sufficient condition to guarantee that the minimum x^* defined in (1.22) will be exponentially stable under the continuous Newton-Raphson dynamics described in the following Theorem 10. Notice that in principle just the average function $\overline{f}$ needs to have specific properties, and thus no conditions for the single f_i's are required: in fact they might even be non convex.

Practical examples

The need of solving large-scale problems exploiting several parallel processors has pioneered the research in the area of distributed optimization. This subject has historically been intended as the problem of dispatching part of a large scale optimization algorithm to different computational units (see the seminal work Tsitsiklis et al. (1986), and Bertsekas and Tsitsiklis (1997)). In the last decade, distributed optimization has been applied to NCSs, posing new issues to be addressed. The key point is that each agent has to implement an optimization procedure that must depend both on local data and on the information that each agent can gather from its neighboring agents, let say the agents in its communication range. Nonetheless the agents have to take into account the local measurements that they can perform.

Plenty of algorithms have been designed for convex optimization problems, and a strong and useful theory has been derived (see Boyd et al. (2010); Hastie et al. (2008); Boyd and Vandenberghe (2004) and the many references therein). Hereafter we propose two typical scenarios, significative examples of the many different problems that can be casted into this framework.

Regression

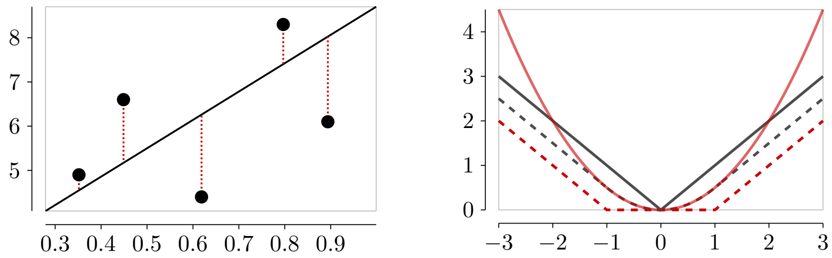
Consider a linear model with *measurements* y_i of the form $y_i = \boldsymbol{u}_i^T \boldsymbol{x} + v_i$, where $\boldsymbol{u}_i$ is the i-th *feature* vector and the measurement noises v_i are independent with log-concave or Gaussian densities. The goal is to find an optimal convex estimate of the regression function, i.e. solving the residuals minimization

$$\min_{\boldsymbol{x}} \quad \textstyle\sum_{i=1}^N \phi(y_i - \boldsymbol{u}_i^T \boldsymbol{x})$$

where $\boldsymbol{x} \in \mathbb{R}^M$ is the minimizer parameter, $\boldsymbol{u}_i \in \mathbb{R}^M$ is the feature (i.e. independent variable) vector for agent i, $y_i \in \mathbb{R}$ is the output or *response* of i, $\phi : \mathbb{R} \to \mathbb{R}$ is a convex loss function, as the following:

$$\begin{aligned}
&\phi(r) = |r|^2 && (\textit{least squares})\\
&\phi(r) = |r| && (\textit{least absolute deviations})\\
&\phi(r) = \begin{cases} 0 & \text{if } |r| < 1 \\ |r| - 1 & \text{otherwise} \end{cases} && (\textit{Vapnik}) \qquad (1.23)\\
&\phi(r) = \begin{cases} |r|^2 & \text{if } |r| < 1 \\ 2(|r| - 1) & \text{otherwise} \end{cases} && (\textit{Huber})
\end{aligned}$$

Notice that $\phi(\cdot)$ is the same for each agent, which is usually the case in practice, although it is not excluded it may be different for each specific agent. See Figure 1.1 for an example of estimated linear regression and cost functions.



(a) residuals and regression

(b) loss functions

Figure 1.1: On the left, example of a linear regression derived from random data points; on the right, plots of the different loss functions as in (1.23).

Classification

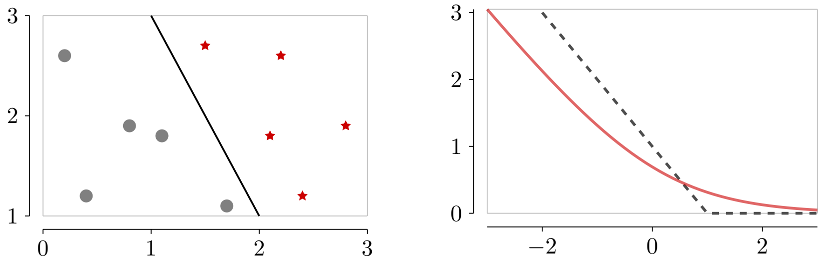
Let $\boldsymbol{u}_i \in \mathbb{R}^M$ denote the feature vector of the i-th example and let $y_i \in \{-1, 1\}$ denote the binary outcome, for $i = 1, \ldots, N$. The goal is to find a weight vector $\boldsymbol{x} \in \mathbb{R}^M$ that satisfies

$$\min_{\boldsymbol{x}} \sum_{i=1}^{N} l_i \left(y_i \boldsymbol{u}_i^T \boldsymbol{x} \right) + \lambda \left\| \boldsymbol{x} \right\|^2$$

where $l_i : \mathbb{R} \to \mathbb{R}$ is the convex loss for the i-th training example and $\lambda \left\| \boldsymbol{x} \right\|^2$ is a separable regularization function known as *Tikhonov regularization*. Common loss functions are the *hinge loss* $[1 - y_i \boldsymbol{u}_i^T \boldsymbol{x}]_+$, denoting with $[\,\cdot\,]_+$ the positive part of the corresponding term, and the *exponential loss* $e^{-y_i \boldsymbol{u}_i^T \boldsymbol{x}}$.

In general, the *support vector machine* corresponds to hinge loss with a quadratic penalty, while exponential loss yields *boosting*.

See Figure 1.2 for an example of two-class classification accompanied by the aforementioned loss functions.



(a) classification

(b) loss functions

Figure 1.2: On the left, example of a linear two-class classification; on the right, plots of the hinge and exponential loss functions.

1.4 Preliminary results

The following theorems will be used to prove the convergence properties of the algorithms proposed hereafter.

Consider the non autonomous system

$$\dot{\boldsymbol{x}} = \boldsymbol{u}(t, \boldsymbol{x}) \tag{1.24}$$

where $\boldsymbol{u} : [0, \infty) \times D \to \mathbb{R}^N$, is piecewise continuous in t and locally Lipschitz in $\boldsymbol{x}$ on $[0, \infty) \times D$, and $D \subset \mathbb{R}^N$ is a domain that contains the origin $\boldsymbol{x} = \boldsymbol{0}$. The origin is an equilibrium point for (1.24) at $t = 0$ if $\boldsymbol{u}(t, \boldsymbol{0}) = \boldsymbol{0}$, $\forall t \geq 0$.

Theorem 9 recalls a stability theorem offered in (Khalil, 2001, p. 154), that is exploited hereafter in Theorem 10.

Theorem 9 Let $\boldsymbol{x} = \boldsymbol{0}$ be an equilibrium point for (1.24) and $D \subset \mathbb{R}^N$ be a domain containing $\boldsymbol{x} = \boldsymbol{0}$. Let $V : [0, \infty) \times D \to \mathbb{R}$ be a continuously differentiable function such that

$$c_1 \|\boldsymbol{x}\|^c \leq V(t, \boldsymbol{x}) \leq c_2 \|\boldsymbol{x}\|^c \tag{1.25}$$

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{u}(t, \boldsymbol{x}) \leq -c_3 \|\boldsymbol{x}\|^c \tag{1.26}$$

$\forall t \geq 0$ and $\forall \boldsymbol{x} \in D$, where c, c_1, c_2, c_3 are positive constants. Then, $\boldsymbol{x} = \boldsymbol{0}$ is exponentially stable. If the assumptions hold globally, then $\boldsymbol{x} = \boldsymbol{0}$ is globally exponentially stable.

Theorem 10 is a preliminary result on the applicability of Newton-Raphson (NR) optimization procedures[8].

Moreover, along with the $\mathcal{C}^2$ requirements in Assumption 8, the theorem will allow us to apply standard singular perturbation analysis techniques.

Theorem 10 For every $r > \overline{f}(x^*)$, let $D_r := \left\{ x \in \mathbb{R} \;\middle|\; \overline{f}(x) \leq r \right\}$. Let moreover

$$\dot{x}(t) = -\frac{\overline{f}'\big(x(t)\big)}{\overline{f}''\big(x(t)\big)} =: \psi\big(x(t) - x^*\big), \quad x(0) \in D_r \tag{1.27}$$

[8] Other asymptotic properties of continuous time NR methods can be found, e.g., in Tanabe (1985); Hauser and Nedić (2005).

describe a continuous-time Newton-Raphson algorithm with $\overline{f}$ satisfying Assumption 8. Then x^* is an exponentially stable equilibrium, i.e., $|x(t)-x^*| \leq ce^{-\gamma t}|x(0)-x^*|$, $\forall$ t's and $\forall x(0) \in D_r$, for suitable positive constants c and γ possibly depending on r.

Proof We proceed showing that $\overline{f}$ is itself a suitable Lyapunov function for (1.27). We now show that $\overline{f}$ is suitable Lyapunov function for (1.27). Then, since $\overline{f}$ is smooth, closed, proper and convex, the set D_r is closed, convex and compact. Moreover $x^* \in D_r$. Let then $a_1 := \min_{x \in D_r} \overline{f}''(x)$ and $a_2 := \max_{x \in D_r} \overline{f}''(x)$, whose existence is assured being D_r closed and compact. Moreover $0 < a_1 \leq a_2$, since $\overline{f}''(x) > 0$ by hypothesis.

Consider then a generic $x \in D_r$, and the Taylor expansion of $\overline{f}$ around x^* with remainder in Lagrange form, i.e.,

$$\overline{f}(x) = \overline{f}(x^*) + \overline{f}'(x^*)(x-x^*) + \frac{\overline{f}''(\widetilde{x})}{2}(x-x^*)^2 \tag{1.28}$$

for a suitable $\widetilde{x}$ between x and x^* (thus $\widetilde{x} \in D_r$ by convexity). Since $\overline{f}'(x^*) = 0$, we can transform (1.28) into $\overline{f}(x) - \overline{f}(x^*) = \dfrac{\overline{f}''(\widetilde{x})}{2}(x-x^*)^2$, i.e.,

$$\frac{a_1}{2}(x-x^*)^2 \leq \overline{f}(x) - \overline{f}(x^*) \leq \frac{a_2}{2}(x-x^*)^2, \qquad \forall x \in D_r. \tag{1.29}$$

Moreover, differentiating (1.28) we obtain $\overline{f}'(x) = \overline{f}'(x^*) + \overline{f}''(\widetilde{x})(x-x^*)$, which implies

$$a_1 |x-x^*| \leq \left|\overline{f}'(x)\right| \leq a_2 |x-x^*|, \qquad \forall x \in D_r. \tag{1.30}$$

Consider then system (1.27). Exploiting (1.30) it follows, $\forall x(t) \in D_r - \{x^*\}$,

$$\dot{\overline{f}}\big(x(t)\big) = \overline{f}'\big(x(t)\big)\dot{x}(t) = -\frac{\left(\overline{f}'\big(x(t)\big)\right)^2}{\overline{f}''\big(x(t)\big)} \leq -\frac{a_1^2}{a_2}\big(x(t)-x^*\big)^2 < 0. \tag{1.31}$$

Consider then Theorem 9. Here (1.29) corresponds to (1.25), (1.31) corresponds to (1.25), and all the other hypotheses are satisfied. Thus $\overline{f}$ is a valid Lyapunov function and x^* is exponentially stable for all $x(0) \in D_r$. ♢

We notice that Theorem 10 states that x^* is practically globally stable. Thus one can start from any point and have an exponential convergence, although a convergence rate that is uniform for all initial conditions might not exist. Nonetheless we can notice that, locally and around the optimum, the rate of convergence of the Newton-Raphson dynamics is $\gamma = 1$ independently of the convex function $\overline{f}$. In fact, if we linearize ψ around 0 (i.e., the dynamics of (1.27) around x^*) we obtain

$$\begin{aligned}\psi(x) &= \psi(0) + \psi'(0)x + o(x)\\ &= -\frac{\overline{f}'(x^*)}{\overline{f}''(x^*)} - \frac{\overline{f}''(x^*)\overline{f}''(x^*) - \overline{f}'(x^*)\overline{f}'''(x^*)}{\left(\overline{f}''(x^*)\right)^2} x + o(x)\\ &= -x + o(x)\end{aligned}$$

since $\overline{f}'(x^*) = 0$ and $\overline{f}''(x^*) \neq 0$.

The hypotheses of Theorem 11.4 in (Khalil, 2001, p. 456) (i.e., Theorem 6 in Section 1.1) and of the converse Lyapunov Theorem 4.14 of (Khalil, 2001, pp. 162) are critical conditions to assess the convergence of the NRC, in the synchronous and asynchronous case respectively. They will be recalled later in Sections 2.1 and 3.1. Theorem 4.14 of (Khalil, 2001, pp. 162) is reported here in its original form, in the guise of Theorem 11.

Theorem 11 Let $\boldsymbol{x} = 0$ be an equilibrium point for the nonlinear system (1.24), where $\boldsymbol{u} : [0, \infty) \times D \to \mathbb{R}^N$ is continuously differentiable, $D = \{\boldsymbol{x} \in \mathbb{R}^N \,|\, \|\boldsymbol{x}\| < r\}$, and the Jacobian matrix $[\partial \boldsymbol{u}/\partial \boldsymbol{x}]$ is bounded on D, uniformly in t. Let k, λ, and r_0 be positive constants with $r_0 < r/k$. Let $D_0 = \{\boldsymbol{x} \in \mathbb{R}^N \,|\, \|\boldsymbol{x}\| < r_0\}$. Assume that the trajectories of the system satisfy

$$\|\boldsymbol{x}(t)\| \leq k\|\boldsymbol{x}(t_0)\|e^{-\lambda(t-t_0)}, \quad \forall \boldsymbol{x}(t_0) \in D_0, \quad \forall t \geq t_0 \geq 0$$

Then there is a function $V : [0, \infty) \times D_0 \to \mathbb{R}$ that satisfies the inequalities

$$\begin{aligned} c_1\|\boldsymbol{x}\|^2 \leq V(t, \boldsymbol{x}) &\leq c_2\|\boldsymbol{x}\|^2\\ \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \boldsymbol{x}}\boldsymbol{u}(t, \boldsymbol{x}) &\leq -c_3\|\boldsymbol{x}\|^2\\ \left\|\frac{\partial V}{\partial \boldsymbol{x}}\right\| &\leq c_4\|\boldsymbol{x}\| \end{aligned}$$

for some positive constants c_1,c_2,c_3,c_4. Moreover, if $r = \infty$ and the origin is globally exponentially stable, then $V(t, \boldsymbol{x})$ is defined and satisfies the aforementioned inequalities on $\mathbb{R}^N$. Furthermore, if the system is autonomous, V can be chosen independent of t.

Lemma 12, that recalls Lemma 9.1 of (Khalil, 2001, pp. 341), will be exploited to state the the global stability of the Asynchronous Newton-Raphson Consensus (ANRC).

Lemma 12 Consider the system

$$\dot{\boldsymbol{x}} = \boldsymbol{u}(t, \boldsymbol{x}) + \boldsymbol{d}(t, \boldsymbol{x}) \tag{1.32}$$

where $\boldsymbol{u} : [0, \infty) \times D \to \mathbb{R}^N$ and $\boldsymbol{d} : [0, \infty) \times D \to \mathbb{R}^N$ are piecewise continuous in t and locally Lipschitz in $\boldsymbol{x}$ on $[0, \infty) \times D$, and $D \subset \mathbb{R}^N$ is a domain that contains the origin $\boldsymbol{x} = \mathbf{0}$. Let $\boldsymbol{x} = \mathbf{0}$ be an exponentially stable equilibrium point of the nominal system $\dot{\boldsymbol{x}} = \boldsymbol{u}(t, \boldsymbol{x})$. Let $V : [0, \infty) \times D \to \mathbb{R}$ be a Lyapunov function of the nominal system that satisfies

$$c_1 \|\boldsymbol{x}\|^2 \leq V(t, \boldsymbol{x}) \leq c_2 \|\boldsymbol{x}\|^2$$

for some constants c_1, c_2 through

$$\left\| \frac{\partial V}{\partial \boldsymbol{x}} \right\| \leq c_4 \|\boldsymbol{x}\|$$

in $[0, \infty) \times D$. Suppose the perturbation term $\boldsymbol{d}(t, \boldsymbol{x})$ satisfies

$$\begin{aligned} \|\boldsymbol{d}(t, \boldsymbol{x})\| \leq \gamma \|\boldsymbol{x}\|, \quad \forall \boldsymbol{x} \in D \\ \gamma < \frac{c_3}{c_4} \end{aligned}$$

for some constants c_3, c_4. Then, the origin is an exponentially stable equilibrium point of the perturbed system (1.32). Moreover, if all the assumptions hold globally, then the origin is globally exponentially stable.

Theorem 1 and Theorem 2 in Sundarapandian (2002) will be extensively adopted into the proofs of convergence of the NRC methods. These theorems give state the global asymptotic stability of continuous-time and discrete-time nonlinear cascade systems. They are here recalled as Theorem 13 and Theorem 14 respectively.

Theorem 13 Consider the continuous-time nonlinear cascade system of the form

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega}) \\ \dot{\boldsymbol{\omega}} &= \boldsymbol{s}(\boldsymbol{\omega}) \end{aligned} \tag{1.33}$$

where $\boldsymbol{x} \in \mathbb{R}^N$, $\boldsymbol{\omega} \in \mathbb{R}^W$. Assume that $\boldsymbol{u} : \mathbb{R}^N \times \mathbb{R}^W \rightarrow \mathbb{R}^N$ and $\boldsymbol{s} : \mathbb{R}^W \rightarrow \mathbb{R}^W$ are both of class $\mathcal{C}^1$. Moreover, assume that $\boldsymbol{u}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$, $\boldsymbol{s}(\mathbf{0}) = \mathbf{0}$, so that $(\boldsymbol{x}, \boldsymbol{\omega}) = (\mathbf{0}, \mathbf{0})$ is an equilibrium of the cascade system (1.33).

Suppose that $\boldsymbol{x} = \mathbf{0}$ is a globally asymptotically stable equilibrium of the subsystem $\dot{\boldsymbol{x}} = \boldsymbol{u}(\boldsymbol{x}, \mathbf{0})$, $\boldsymbol{\omega} = \mathbf{0}$ is a globally asymptotically stable equilibrium of the subsystem $\dot{\boldsymbol{\omega}} = \boldsymbol{s}(\boldsymbol{\omega})$, and that the trajectories $(\boldsymbol{x}(t), \boldsymbol{\omega}(t))$ of (1.33) are bounded for $t > 0$. Then $(\boldsymbol{x}, \boldsymbol{\omega}) = (\mathbf{0}, \mathbf{0})$ is a globally asymptotically stable equilibrium of the cascade system (1.33).

Theorem 14 Consider the discrete-time nonlinear cascade system of the form

$$\begin{array}{rcl} \boldsymbol{x}(k+1) & = & \boldsymbol{u}(\boldsymbol{x}(k), \boldsymbol{\omega}(k)) \\ \boldsymbol{\omega}(k+1) & = & \boldsymbol{s}(\boldsymbol{\omega}(k)) \end{array} \tag{1.34}$$

where $\boldsymbol{x} \in \mathbb{R}^N$, $\boldsymbol{\omega} \in \mathbb{R}^W$. Assume that $\boldsymbol{u} : \mathbb{R}^N \times \mathbb{R}^W \rightarrow \mathbb{R}^N$ and $\boldsymbol{s} : \mathbb{R}^W \rightarrow \mathbb{R}^W$ are both of class $\mathcal{C}^1$. Moreover, assume that $\boldsymbol{u}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$, $\boldsymbol{s}(\mathbf{0}) = \mathbf{0}$, so that $(\boldsymbol{x}, \boldsymbol{\omega}) = (\mathbf{0}, \mathbf{0})$ is an equilibrium of the cascade system (1.34).

Suppose that $\boldsymbol{x} = \mathbf{0}$ is a globally asymptotically stable equilibrium of the subsystem $\boldsymbol{x}(k+1) = \boldsymbol{u}(\boldsymbol{x}(k), \mathbf{0})$, $\boldsymbol{\omega} = \mathbf{0}$ is a globally asymptotically stable equilibrium of the subsystem $\boldsymbol{\omega}(k+1) = \boldsymbol{s}(\boldsymbol{\omega}(k))$, and that the trajectories $(\boldsymbol{x}(k), \boldsymbol{\omega}(k))$ of (1.34) are bounded for $k \in \mathbb{N}$. Then $(\boldsymbol{x}, \boldsymbol{\omega}) = (\mathbf{0}, \mathbf{0})$ is a globally asymptotically stable equilibrium of the cascade system (1.34).

The Implicit Function Theorem, here presented as Theorem 15, will be used in Section 2.1 to show that non-null, but sufficiently small initial conditions on the NRC let the algorithm converge to a neighborhood of the optimum.

Theorem 15 (Basic Implicit Function Theorem) Suppose $\boldsymbol{\psi} : D \rightarrow \mathbb{R}$, $D \subset \mathbb{R}^3$, has $\boldsymbol{\psi}(\alpha, \beta, x) = c$, for some $(\alpha, \beta, x) \in D$ and $c \in \mathbb{R}$, and satisfies $\left(\frac{\partial \boldsymbol{\psi}}{\partial x_1}, \frac{\partial \boldsymbol{\psi}}{\partial x_2}, \frac{\partial \boldsymbol{\psi}}{\partial x_3}\right) \neq (0, 0, 0)$. Then the following hold.

i) There is a function $\xi(x_1, x_2)$, defined near $(\alpha, \beta) \in D \cap (\mathbb{R}^2 \times \{x\})$, such that $\boldsymbol{\psi}(x_1, x_2, \xi(x_1, x_2)) = c$.

ii) Near (α, β, x) the given equation has no solutions other than the ones described in (i).

iii) Near (α, β, x), the level set $\{(x_1, x_2, x_3) \in D \mid \boldsymbol{\psi}(x_1, x_2, x_3) = c\}$ is a bidimensional manifold, and its tangent plane at (α, β, x) is perpendicular to $\nabla \boldsymbol{\psi}(\alpha, \beta, x)$.

iv) The derivative of ξ at (α, β) is given by

$$\xi'(\alpha, \beta) = \begin{bmatrix} \frac{\partial \xi}{\partial x_1} & \frac{\partial \xi}{\partial x_2} \end{bmatrix} (\alpha, \beta) = \begin{bmatrix} -\dfrac{\dfrac{\partial \boldsymbol{\psi}}{\partial x_1}(\alpha, \beta, x)}{\dfrac{\partial \boldsymbol{\psi}}{\partial x_3}(\alpha, \beta, x)} & -\dfrac{\dfrac{\partial \boldsymbol{\psi}}{\partial x_2}(\alpha, \beta, x)}{\dfrac{\partial \boldsymbol{\psi}}{\partial x_3}(\alpha, \beta, x)} \end{bmatrix}.$$

2 Synchronous Newton-Raphson Consensus

2.1 The scalar case

For a better understanding of the algorithm we are going to propose, we add some additional assumptions and we later generalize these ideas into the general framework.

We start analyzing the following simplified scenario: the local costs are quadratic and scalar, i.e., $f_i(x) = \frac{1}{2} a_i (x - b_i)^2$, with $a_i > 0$ and $x \in \mathbb{R}$. It is known that, in this case, $x^* = \arg\min_{x \in \mathbb{R}} \overline{f}(x)$ can be computed using two average consensus algorithms in parallel, see, e.g., Xiao et al. (2005); Bolognani et al. (2010). In fact

$$x^* = \frac{\sum_{i=1}^{N} a_i b_i}{\sum_{i=1}^{N} a_i} = \frac{\frac{1}{N} \sum_{i=1}^{N} a_i b_i}{\frac{1}{N} \sum_{i=1}^{N} a_i}, \tag{2.1}$$

i.e., x^* corresponds to the ratio of two arithmetic means. Thus, *under quadratic costs assumptions*, if each agent defines the local variables $y_i(0) := a_i b_i$ and $z_i(0) := a_i$ and updates them cycling the steps

$$\begin{aligned} \boldsymbol{y}(k+1) &= P \boldsymbol{y}(k) \\ \boldsymbol{z}(k+1) &= P \boldsymbol{z}(k) \\ \boldsymbol{x}(k+1) &= \frac{\boldsymbol{y}(k+1)}{\boldsymbol{z}(k+1)}, \end{aligned} \tag{2.2}$$

it follows that, given the fact that P is an average consensus matrix,

$$\lim_{k \to \infty} \boldsymbol{x}(k) = x^* \mathbb{1}$$

(under mild assumptions, with the same convergence speed of the consensus algorithm induced by P). Since $x_i(k) = y_i(k)/z_i(k) \to x^*$ for all i's, the $x_i(k)$'s computed through (2.2) can be thought as the local estimates at time k of the global minimizer x^*.

We can now generalize (2.2) to the case where the local cost functions f_i are *not* quadratic. To this point we observe that, for all x, in the quadratic case it holds that $a_i b_i = f_i''(x)x - f_i'(x) =: g_i(x)$ and that $a_i = f_i''(x) =: h_i(x)$. As a consequence, we could let each agent choose an $x_i(0)$ for all i, then set $y_i(0) = f_i''\big(x_i(0)\big)x_i(0) - f_i'\big(x_i(0)\big)$ and $z_i(0) = f_i''\big(x_i(0)\big)$, apply (2.2) up to convergence and thus compute

$$\widehat{x}^* = \frac{\frac{1}{N}\sum_{i=1}^{N}\Big(f_i''\big(x_i(0)\big)x_i(0) - f_i'\big(x_i(0)\big)\Big)}{\frac{1}{N}\sum_{i=1}^{N} f_i''\big(x_i(0)\big)} = \frac{\frac{1}{N}\sum_{i=1}^{N} g_i\big(x_i(0)\big)}{\frac{1}{N}\sum_{i=1}^{N} h_i\big(x_i(0)\big)}. \tag{2.3}$$

Figure 2.1 gives a pictorial hint that $\widehat{x}^*$ may be an accurate guess of x^*.

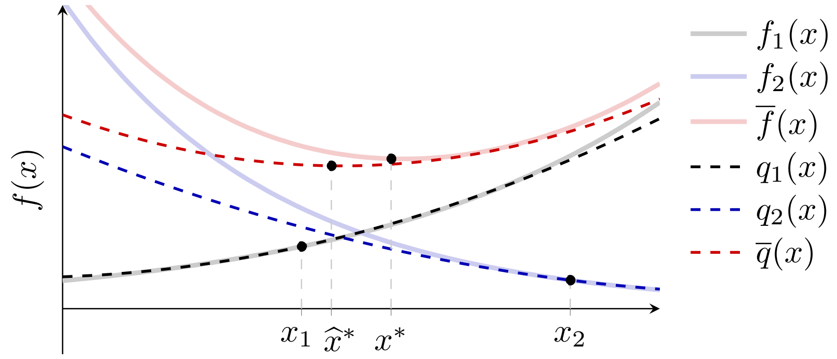


Figure 2.1: Graphical intuition underlying the SNRC. Starting from the local guesses x_1 and x_2, the agents locally compute the second order Taylor expansions of the local costs $f_1(x)$ and $f_2(x)$, namely $q_1(x)$ and $q_2(x)$. The minimum $\widehat{x}^*$ of $\overline{q}(x) := \sum_i q_i(x)/2$ can then be computed with (2.3). Interpreting $\overline{q}(x)$ as an approximation of $\overline{f}(x)$, $\widehat{x}^*$ is an approximation of x^*.

Because of the intuitions given before, we expect $\widehat{x}^*$ to be a sensible estimation for the true minimizer x^*. However, $\widehat{x}^*$ depends on the initial conditions $x_i(0)$ and, in general, if the f_i's are not quadratic then $\widehat{x}^* \neq x^*$. Therefore, (2.2) cannot be applied directly. Nonetheless we notice that if all the $x_i(0)$'s are equal, i.e., $x_i(0) = x$, $\forall i$, then

$$\widehat{x}^* = x - \frac{\overline{f}'(x)}{\overline{f}''(x)}$$

which is a standard NR update step. Thus, if all the agents agree on the $x_i(0)$'s and the consensus step (2.2) is given enough time to converge, then $\widehat{x}^*$

provides the right descent direction. If instead the agents do not agree on the $x_i(0)$'s, then $\widehat{x}^*$ provides just an approximation of the right descent direction.

To design the main algorithm we then observe the following:

- (2.2) shall be modified so that it computes meaningful Newton directions even if the $x_i(k)$'s change over time. Since time-varying $x_i(k)$'s imply time-varying $g_i\big(x_i(k)\big)$'s and $h_i\big(x_i(k)\big)$'s, the $y_i(k)$'s and $z_i(k)$'s must track the changing averages $\dfrac{1}{N}\displaystyle\sum_{i=1}^{N} g_i\big(x_i(k)\big)$ and $\dfrac{1}{N}\displaystyle\sum_{i=1}^{N} h_i\big(x_i(k)\big)$;

- the computation of the averages of the various $g_i\big(x_i(k)\big)$'s and $h_i\big(x_i(k)\big)$'s must have a convergence rate that is sufficiently faster than the rate of change of the different $x_i(k)$'s.

These ideas are captured in the following Algorithm 1, where the vectorial notation also for the functions $\boldsymbol{g}\big(\boldsymbol{x}(k)\big)$ and $\boldsymbol{h}\big(\boldsymbol{x}(k)\big)$ is introduced.

Algorithm 1 Synchronous Newton-Raphson Consensus (SNRC) – scalar case

```
    (storage allocation and constraints on the parameters)
 1: $\boldsymbol{x}(k),\ \boldsymbol{y}(k),\ \boldsymbol{z}(k) \in \mathbb{R}^N, \forall k \in \mathbb{N}$
 2: $\varepsilon \in (0,1)$
    (initialization)
 3: $\boldsymbol{x}(0) = \boldsymbol{0}$
 4: $\boldsymbol{y}(0) = \boldsymbol{g}\big(\boldsymbol{x}(-1)\big) = \boldsymbol{0}, \boldsymbol{z}(0) = \boldsymbol{h}\big(\boldsymbol{x}(-1)\big) = \mathbb{1}$
    (main algorithm)
 5: for $k = 1, 2, \ldots$ do
 6:     $\boldsymbol{x}(k) = (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon \dfrac{\boldsymbol{y}(k-1)}{\boldsymbol{z}(k-1)}$    (local update)
 7:     $\widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k-1) + \boldsymbol{g}\big(\boldsymbol{x}(k-1)\big) - \boldsymbol{g}\big(\boldsymbol{x}(k-2)\big)$    (local update)
 8:     $\widetilde{\boldsymbol{z}}(k) = \boldsymbol{z}(k-1) + \boldsymbol{h}\big(\boldsymbol{x}(k-1)\big) - \boldsymbol{h}\big(\boldsymbol{x}(k-2)\big)$    (local update)
 9:     $\boldsymbol{y}(k) = P\,\widetilde{\boldsymbol{y}}(k)$    (consensus step)
10:     $\boldsymbol{z}(k) = P\,\widetilde{\boldsymbol{z}}(k)$    (consensus step)
```

The following remarks highlight the peculiar structure of Algorithm 1:

- the initialization in line 4 is *critical* for convergence to the global minimizer. However robustness analysis on possible numerical errors in the initial conditions or quantization noise is discussed below;

- lines 7-8 let the various agents modify their local y_i and z_i in order to take into account the effects induced by changing $x_i(k)$'s *before* the consensus

steps on the y_i's and z_i's in lines 9-10.
Basically, lines 7-8 correspond to a high-pass filter with respect to time to track the changing averages $\frac{1}{N}\sum_{i=1}^{N} g_i(x_i)$ and $\frac{1}{N}\sum_{i=1}^{N} h_i(x_i)$;

- line 6 substitutes the local guess update step $x_i(k) = y_i(k)/z_i(k)$ in (2.2) with a low-pass filter dominated by the parameter ε. This is necessary because the consensus process on y_i's and z_i's must be faster than the tendency of $x_i(k)$'s to spread apart while the various y_i's and z_i's are not close. If this spreading mechanism is not dominated by the consensus on the y_i's and z_i's, the algorithm could eventually diverge. In other words, line 6 softens possible too aggressive updates of the local estimates. It is also similar to what is usually done in NR approaches where only a small step is taken towards the newly estimated global minimum;
- if $\varepsilon = 1$ and the functions f_i are quadratic, then Algorithm 1 reduces to the system (2.2).

Before providing a formal proof of the convergence properties of Algorithm 1 we give some intuitions on its behavior. The dynamics of Algorithm 1 can be written in state space as follows:

$$\begin{cases} \boldsymbol{v}(k) &= \boldsymbol{g}\big(\boldsymbol{x}(k-1)\big) \\ \boldsymbol{w}(k) &= \boldsymbol{h}\big(\boldsymbol{x}(k-1)\big) \\ \boldsymbol{y}(k) &= P\Big[\boldsymbol{y}(k-1) + \boldsymbol{g}\big(\boldsymbol{x}(k-1)\big) - \boldsymbol{v}(k-1)\Big] \\ \boldsymbol{z}(k) &= P\Big[\boldsymbol{z}(k-1) + \boldsymbol{h}\big(\boldsymbol{x}(k-1)\big) - \boldsymbol{w}(k-1)\Big] \\ \boldsymbol{x}(k) &= (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon\dfrac{\boldsymbol{y}(k-1)}{\boldsymbol{z}(k-1)}\,, \end{cases} \tag{2.4}$$

which can be interpreted as the forward-Euler discrete-time version of continuous-time system

$$\begin{cases} \varepsilon\dot{\boldsymbol{v}}(t) &= -\boldsymbol{v}(t) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big) \\ \varepsilon\dot{\boldsymbol{w}}(t) &= -\boldsymbol{w}(t) + \boldsymbol{h}\big(\boldsymbol{x}(t)\big) \\ \varepsilon\dot{\boldsymbol{y}}(t) &= -K\boldsymbol{y}(t) + (I-K)\Big[\boldsymbol{g}\big(\boldsymbol{x}(t)\big) - \boldsymbol{v}(t)\Big] \\ \varepsilon\dot{\boldsymbol{z}}(t) &= -K\boldsymbol{z}(t) + (I-K)\Big[\boldsymbol{h}\big(\boldsymbol{x}(t)\big) - \boldsymbol{w}(t)\Big] \\ \dot{\boldsymbol{x}}(t) &= -\boldsymbol{x}(t) + \dfrac{\boldsymbol{y}(t)}{\boldsymbol{z}(t)} \end{cases} \tag{2.5}$$

where ε is the discretization time interval and $K := I - P$. As a consequence, for sufficiently small ε the dynamic behavior of (2.4) is approximated by the one of (2.5), i.e., $x(t/\varepsilon) \approx x(k)$, where, with a little abuse of notation, we used the same symbols for the continuous and discrete time variables.

It is now possible to recognize the existence of a two-time scales dynamical system regulated by the parameter ε. Therefore, we can split the dynamics in the two time scales and study them separately for sufficiently small ε.

As for the *fast dynamics*, by construction K is positive semidefinite with kernel spanned by $\mathbb{1}$ and with eigenvalues $0 = \lambda_1 < \mathrm{Re}\left[\lambda_2\right] \leq \cdots \leq \mathrm{Re}\left[\lambda_N\right] < 2$. That is, the first four equations of system (2.5) imply that

$$\boldsymbol{y}(t) \approx \left(\frac{1}{N}\mathbb{1}^T\boldsymbol{g}\big(\boldsymbol{x}(t)\big)\right)\mathbb{1} \quad \text{and} \quad \boldsymbol{z}(t) \approx \left(\frac{1}{N}\mathbb{1}^T\boldsymbol{h}\big(\boldsymbol{x}(t)\big)\right)\mathbb{1}.$$

If these equations are inserted into the *slow dynamics*, i.e., into the last equation of system (2.5), then it follows that $\boldsymbol{x}(t) \approx \overline{x}(t)\mathbb{1}$, where $\overline{x}(t)$ is a scalar quantity that approximately evolves following the continuous-time Newton-Raphson update

$$\dot{\overline{x}}(t) = -\frac{\overline{f}'\big(\overline{x}(t)\big)}{\overline{f}''\big(\overline{x}(t)\big)} \ . \tag{2.6}$$

Summarizing, for sufficiently small values of ε Algorithm 1 can be described by (2.5), it is such that $x_i(k) \approx \overline{x}(t/\varepsilon)$ for all i's, and thus by Theorem 10 it converges to the global optimum x^*.

We now move from intuitions to a formal proof of convergence. We start by considering the robustness of the algorithm in terms of possible different initial conditions $\boldsymbol{x}(0)$.

Theorem 16 Consider Algorithm 1 with arbitrary initial conditions $\boldsymbol{x}(0)$ and let Assumption 8 hold true. For every ball $B_r^{x^*} := \{\boldsymbol{x} \mid \|\boldsymbol{x} - x^*\mathbb{1}\| < r\}$ there exist two positive constants $\overline{\varepsilon}_r$, c_r such that if $\varepsilon < \overline{\varepsilon}_r$, then there exists $\gamma_\varepsilon > 0$ such that, for all $\boldsymbol{x}(0) \in B_r^{x^*}$,

$$\|\boldsymbol{x}(k) - x^*\mathbb{1}\| \leq c_r e^{-\gamma_\varepsilon k} \|\boldsymbol{x}(0) - x^*\mathbb{1}\|$$

for all $k \in \mathbb{N}$.

Proof Since assumptions for Theorem 2 in Teel et al. (1998)[1] are satisfied, we are ensured that if ε is sufficiently small then the discretized system (2.4)

[1] Teel et al. (1998) show that if a continuous time input-to-state stabilizing controller is sampled fast enough, the obtained sampled-data nonlinear system still has the same properties as the continuous time system.

inherits the same stability properties of (2.5). Therefore we focus on proving that Theorem 16 holds true considering system (2.5) rather than Algorithm 1.

The proof is based on characterizing system (2.5) through classical multi-time-scales approaches for standard singular perturbation model analysis (Kokotović et al., 1999; Khalil, 2001, Chap. 11). It is divided in the following steps:

a) perform some suitable changes of variables;

b) analyze the boundary layer system (fast dynamics);

c) analyze the reduced system (slow dynamics).

In the following we will use the additional notation

$$\Pi^{\parallel} := \frac{\mathbb{1}\mathbb{1}^T}{N} \qquad \Pi^{\perp} := I - \frac{\mathbb{1}\mathbb{1}^T}{N}$$

$$\overline{x} := \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \boldsymbol{x}^{\parallel} := \Pi^{\parallel}\boldsymbol{x} = \overline{x}\mathbb{1} \qquad \boldsymbol{x}^{\perp} := \Pi^{\perp}\boldsymbol{x}.$$

Moreover, to ease the proof of the following Theorem 17 we initially consider the more general case where $\boldsymbol{g}\big(\boldsymbol{x}(-1)\big)$, $\boldsymbol{h}\big(\boldsymbol{x}(-1)\big)$, $\boldsymbol{y}(0)$, $\boldsymbol{z}(0)$ are generic elements in $\mathbb{R}^N$. To this aim we use the additional notation

$$\alpha := \frac{1}{N}\mathbb{1}^T\big(\boldsymbol{y}(0) - \boldsymbol{v}(0)\big) \qquad \beta := \frac{1}{N}\mathbb{1}^T\big(\boldsymbol{z}(0) - \boldsymbol{w}(0)\big). \tag{2.7}$$

a) *changes of variables* (to show that the boundary layer system has a single isolated root): let $\boldsymbol{d}(t) := \boldsymbol{y}(t) - \boldsymbol{v}(t)$, so that $\dot{\boldsymbol{y}}(t) = \dot{\boldsymbol{d}}(t) + \dot{\boldsymbol{v}}(t)$. This implies

$$\varepsilon\left(\dot{\boldsymbol{d}}(t) + \dot{\boldsymbol{v}}(t)\right) = -K\big[\boldsymbol{d}(t) + \boldsymbol{v}(t)\big] + (I - K)\left[\boldsymbol{g}\big(\boldsymbol{x}(t)\big) - \boldsymbol{v}(t)\right]$$

and thus, since $\varepsilon\dot{\boldsymbol{v}}(t) = -\boldsymbol{v}(t) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big)$,

$$\varepsilon\dot{\boldsymbol{d}}(t) = -K\Big[\boldsymbol{d}(t) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big)\Big]. \tag{2.8}$$

We now decompose $\boldsymbol{d}(t)$ into $\boldsymbol{d}(t) = \boldsymbol{d}^{\parallel} + \boldsymbol{d}^{\perp}$, $\boldsymbol{d}^{\parallel} := \Pi^{\parallel}\boldsymbol{d}$, $\boldsymbol{d}^{\perp} := \Pi^{\perp}\boldsymbol{d}$, i.e., into "mean component plus deviations from mean". Since $\Pi^{\parallel}K = \boldsymbol{0}$ and $\Pi^{\perp}K = K\Pi^{\perp} = K$, then (2.8) can be decomposed as

$$\begin{cases} \varepsilon\dot{\boldsymbol{d}}^{\parallel}(t) = \boldsymbol{0} & (2.9) \\ \varepsilon\dot{\boldsymbol{d}}^{\perp}(t) = -K\Big[\boldsymbol{d}^{\perp}(t) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big)\Big] & (2.10) \end{cases}$$

with (2.9) implying $\boldsymbol{d}^{\parallel}(t) = \boldsymbol{d}^{\parallel}(0) = \Pi^{\parallel}\big(\boldsymbol{y}(0) - \boldsymbol{v}(0)\big) = \alpha\mathbb{1}$, α defined in (2.7).

The same derivations can be applied to the variables $\boldsymbol{b}(t) := \boldsymbol{z}(t) - \boldsymbol{w}(t)$, $\boldsymbol{z}$ and $\boldsymbol{w}$, so that system (2.5) becomes

$$\begin{cases} \varepsilon\dot{\boldsymbol{v}}(t) &= -\boldsymbol{v}(t) + \boldsymbol{g}\left(\boldsymbol{x}(t)\right) \\ \varepsilon\dot{\boldsymbol{w}}(t) &= -\boldsymbol{w}(t) + \boldsymbol{h}\left(\boldsymbol{x}(t)\right) \\ \varepsilon\dot{\boldsymbol{d}}^{\perp}(t) &= -K\left[\boldsymbol{d}^{\perp}(t) + \boldsymbol{g}\left(\boldsymbol{x}(t)\right)\right] \\ \varepsilon\dot{\boldsymbol{b}}^{\perp}(t) &= -K\left[\boldsymbol{b}^{\perp}(t) + \boldsymbol{h}\left(\boldsymbol{x}(t)\right)\right] \\ \dot{\boldsymbol{x}}(t) &= -\boldsymbol{x}(t) + \dfrac{\boldsymbol{d}^{\perp}(t) + \alpha\mathbb{1} + \boldsymbol{v}(t)}{\boldsymbol{b}^{\perp}(t) + \beta\mathbb{1} + \boldsymbol{w}(t)} \end{cases} \tag{2.11}$$

Notice that the values for α and β reflect the numerical values of the initial conditions of the variables $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{y}$, $\boldsymbol{z}$.

b) *analysis of the boundary layer system*: in this case $\boldsymbol{x}(t)$ is considered to be constant in time, so that $\boldsymbol{g}\big(\boldsymbol{x}(t)\big) = \boldsymbol{g}\left(\boldsymbol{x}\right)$ and $\boldsymbol{h}\big(\boldsymbol{x}(t)\big) = \boldsymbol{h}\left(\boldsymbol{x}\right)$. To analyze the stability properties of system

$$\begin{cases} \varepsilon\dot{\boldsymbol{v}}(t) &= -\boldsymbol{v}(t) + \boldsymbol{g}\left(\boldsymbol{x}\right) \\ \varepsilon\dot{\boldsymbol{w}}(t) &= -\boldsymbol{w}(t) + \boldsymbol{h}\left(\boldsymbol{x}\right) \\ \varepsilon\dot{\boldsymbol{d}}^{\perp}(t) &= -K\big[\boldsymbol{d}^{\perp}(t) + \boldsymbol{g}\left(\boldsymbol{x}\right)\big] \\ \varepsilon\dot{\boldsymbol{b}}^{\perp}(t) &= -K\big[\boldsymbol{b}^{\perp}(t) + \boldsymbol{h}\left(\boldsymbol{x}\right)\big] \end{cases} \tag{2.12}$$

we start by applying the changes of variables and timescales

$$\begin{array}{ll} \widetilde{\boldsymbol{v}}(t) := \boldsymbol{v}(t) - \boldsymbol{g}\left(\boldsymbol{x}\right) & \qquad \widetilde{\boldsymbol{d}}^{\perp}(t) := \boldsymbol{d}^{\perp}(t) + \Pi^{\perp}\boldsymbol{g}\left(\boldsymbol{x}\right) \\ \widetilde{\boldsymbol{w}}(t) := \boldsymbol{w}(t) - \boldsymbol{h}\left(\boldsymbol{x}\right) & \qquad \widetilde{\boldsymbol{b}}^{\perp}(t) := \boldsymbol{b}^{\perp}(t) + \Pi^{\perp}\boldsymbol{h}\left(\boldsymbol{x}\right) \end{array}$$

$$\tau := \frac{t}{\varepsilon} \;\Rightarrow\; \frac{d\tau}{dt} = \frac{1}{\varepsilon}$$

the former induced by the isolated root of (2.12) and the latter by the low-pass filtering parameter ε. We thus obtain the equivalent boundary layer system

$$\begin{cases} \dot{\widetilde{\boldsymbol{v}}}(\tau) &= -\widetilde{\boldsymbol{v}}(\tau) \\ \dot{\widetilde{\boldsymbol{w}}}(\tau) &= -\widetilde{\boldsymbol{w}}(\tau) \\ \dot{\widetilde{\boldsymbol{d}}}^{\perp}(\tau) &= -K\widetilde{\boldsymbol{d}}^{\perp}(\tau) \\ \dot{\widetilde{\boldsymbol{b}}}^{\perp}(\tau) &= -K\widetilde{\boldsymbol{b}}^{\perp}(\tau) \end{cases} \tag{2.13}$$

where the stability properties are equivalent to the ones of (2.12).

We now show that (2.13) is exponentially stable. This is clear for the dynamics of the first two equations, while for the last two we claim that $V\left(\boldsymbol{\zeta}\right) = \frac{1}{2}\left\|\boldsymbol{\zeta}\right\|^2$ is a valid Lyapunov function for both $\widetilde{\boldsymbol{d}}^{\perp}$ and $\widetilde{\boldsymbol{b}}^{\perp}$. For, consider that

$$\dot{V}(\widetilde{\boldsymbol{d}}^{\perp}) = -(\widetilde{\boldsymbol{d}}^{\perp})^T K\widetilde{\boldsymbol{d}}^{\perp} \leq -\lambda_2\|\widetilde{\boldsymbol{d}}^{\perp}\|^2 \leq -\lambda_2 V(\widetilde{\boldsymbol{d}}^{\perp}),$$

where $\lambda_2 > 0$ is the smallest non-zero eigenvalue of the matrix K ($\widetilde{\boldsymbol{d}}^{\perp}$ by construction lives in $\ker(K)^{\perp}$; similar reasonings hold for $\widetilde{\boldsymbol{b}}^{\perp}$).

Applying this result back to (2.12), we are ensured that (2.12) is globally exponentially stable with equilibrium given by

$$\lim_{t\to\infty}\begin{bmatrix}\boldsymbol{v}(t)\\ \boldsymbol{w}(t)\\ \boldsymbol{d}^{\perp}(t)\\ \boldsymbol{b}^{\perp}(t)\end{bmatrix}=\begin{bmatrix}\boldsymbol{g}(\boldsymbol{x})\\ \boldsymbol{h}(\boldsymbol{x})\\ -\Pi^{\perp}\boldsymbol{g}(\boldsymbol{x})\\ -\Pi^{\perp}\boldsymbol{h}(\boldsymbol{x})\end{bmatrix} \tag{2.14}$$

for every initial condition and $\boldsymbol{x}$.

c) *analysis of the reduced system*: substituting (2.14) into the last equation of system (2.11) we obtain the reduced system

$$\dot{\boldsymbol{x}}(t) = -\boldsymbol{x}(t) + \frac{\alpha\mathbb{1} - \Pi^{\perp}\boldsymbol{g}\big(\boldsymbol{x}(t)\big) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big)}{\beta\mathbb{1} - \Pi^{\perp}\boldsymbol{h}\big(\boldsymbol{x}(t)\big) + \boldsymbol{h}\big(\boldsymbol{x}(t)\big)} .$$

Let now $\overline{g}\big(\boldsymbol{x}(k)\big) := \frac{1}{N}\sum_{i=1}^{N} g_i\big(x_i(k)\big)$, $\overline{h}\big(\boldsymbol{x}(k)\big) := \frac{1}{N}\sum_{i=1}^{N} h_i\big(x_i(k)\big)$, so that we can exploit the equivalence

$$-\Pi^{\perp}\boldsymbol{g}\big(\boldsymbol{x}(t)\big) + \boldsymbol{g}\big(\boldsymbol{x}(t)\big) = \Pi^{\parallel}\boldsymbol{g}\big(\boldsymbol{x}(t)\big) = \overline{g}\big(\boldsymbol{x}(t)\big)\mathbb{1}$$

in the numerator and a similar equivalence for $\boldsymbol{h}\big(\boldsymbol{x}(t)\big)$ in the denominator. We can thus rewrite the reduced system as

$$\dot{\boldsymbol{x}}(t) = -\boldsymbol{x}(t) + \frac{\alpha + \overline{g}\big(\boldsymbol{x}(t)\big)}{\beta + \overline{h}\big(\boldsymbol{x}(t)\big)}\mathbb{1} . \tag{2.15}$$

This eventually implies that (2.15) can then be rewritten as

$$\dot{\boldsymbol{x}}(t) = \Psi\big(\boldsymbol{x}(t), \alpha, \beta\big) \tag{2.16}$$

where Ψ is a smooth function of its arguments.

To address the stability of (2.16) we then decompose its dynamics along the projections given by $\Pi^{\perp}$ and $\Pi^{\parallel}$, obtaining the continuous time non-linear cascade system

$$\begin{cases}\dot{\boldsymbol{x}}^{\parallel}(t) &= -\boldsymbol{x}^{\parallel}(t) + \dfrac{\overline{g}\Big(\boldsymbol{x}^{\parallel}(t) + \boldsymbol{x}^{\perp}(t)\Big) + \alpha}{\overline{h}\Big(\boldsymbol{x}^{\parallel}(t) + \boldsymbol{x}^{\perp}(t)\Big) + \beta}\mathbb{1}\\[2ex] \dot{\boldsymbol{x}}^{\perp}(t) &= -\boldsymbol{x}^{\perp}(t)\end{cases} \tag{2.17}$$

where $\boldsymbol{x}^{\perp}(t)$ is independent on $\boldsymbol{x}^{\parallel}(t)$ and exponentially decaying to zero.

We now notice that, by construction, $\boldsymbol{x}^{\parallel}(t) = \overline{x}(t)\mathbb{1}$, i.e., $\boldsymbol{x}^{\parallel}(t)$ is a vector with identical entries. Therefore the dynamic behavior of the first equation in (2.17) is summarized by

$$\dot{\overline{x}}(t) = -\overline{x}(t) + \frac{\overline{g}\Big(\boldsymbol{x}^{\parallel}(t) + \boldsymbol{x}^{\perp}(t)\Big) + \alpha}{\overline{h}\Big(\boldsymbol{x}^{\parallel}(t) + \boldsymbol{x}^{\perp}(t)\Big) + \beta}. \tag{2.18}$$

By Theorem 13 (i.e., Theorem 1 in Sundarapandian (2002))[2], proof of exponential stability of system (2.17) can be reduced to proof of exponential stability of system (2.18) with $\boldsymbol{x}^{\perp}(t) = 0$, so that our case reduces to analyze

$$\dot{\overline{x}}(t) = -\overline{x}(t) + \frac{\overline{g}\big(\overline{x}(t)\big) + \alpha}{\overline{h}\big(\overline{x}(t)\big) + \beta} = \psi\big(\overline{x}(t), \alpha, \beta\big). \tag{2.19}$$

We now consider the case $\alpha = \beta = 0$, which is guaranteed by the initialization of line 4 in Algorithm 1. Given $\alpha = \beta = 0$ and considering the definitions of $\overline{g}(x)$ and $\overline{h}(x)$, (2.19) reduces to the continuous-time Newton-Raphson method

$$\dot{\overline{x}}(t) = -\frac{\overline{f}'\big(\overline{x}(t)\big)}{\overline{f}''\big(\overline{x}(t)\big)} \tag{2.20}$$

that, due to Theorem 10, exponentially converges to x^*. Moreover, given $\alpha = \beta = 0$, the strict positivity of $\overline{h}(\cdot)$ and the properness of $\overline{g}(\cdot)$, the trajectories of system (2.18) are all bounded. Thus the hypotheses of Theorem 13 are all satisfied and we can claim that (2.18) exponentially converges to $x^*\mathbb{1}$.

It is immediate now to check that the hypotheses of Theorem 6 are satisfied, and this guarantees our claims. $\diamondsuit$

In the previous theorem, the critical value for the parameter $\overline{\varepsilon}_r$ depends on r and the function $\overline{f}$. The explicit computation of the critical value based on Lyapunov theory is in general very pessimistic and therefore of no practical use. However, the proof shows that if r and ε are sufficiently small, the rate of convergence of the algorithm tends to $\gamma_\varepsilon = \varepsilon$, and the dynamics of the estimates is approximately given by $x_i(k) \approx x^* + \big(\overline{x}(0) - x^*\big)e^{-\varepsilon k}$, if $\overline{x}(0) := \frac{1}{N}\sum_{i=1}^{N} x_i(0)$.

As noticed before, Theorem 16 holds only for the specific initial conditions given in line 4 of Algorithm 1. Although these initial conditions can be arbitrarily designed, nonetheless it is important to evaluate the robustness of

[2] Formally, Theorem 1 in Sundarapandian (2002) considers just simple stability. Nonetheless it is immediate to check that its proof is such that if the subsystems are exponentially stable then the overall system is again exponentially stable.

the algorithm with respect to them, since small numerical errors and quantization noise might lead to some perturbations. The following theorem shows that non-null but sufficiently small initial conditions on the variables $\boldsymbol{v}(0)$, $\boldsymbol{w}(0)$, $\boldsymbol{y}(0)$, $\boldsymbol{z}(0)$ let the solution of the algorithm exponentially converge to a neighborhood of the true optimum:

Theorem 17 Consider Algorithm 1 with arbitrary initial conditions, $\boldsymbol{x}(0)$ $\boldsymbol{v}(0)$, $\boldsymbol{w}(0)$, $\boldsymbol{y}(0)$, $\boldsymbol{z}(0)$. Let Assumption 8 hold true and

$$\alpha := \frac{1}{N}\mathbb{1}^T\big(\boldsymbol{y}(0) - \boldsymbol{v}(0)\big), \quad \beta := \frac{1}{N}\mathbb{1}^T\big(\boldsymbol{z}(0) - \boldsymbol{w}(0)\big).$$

Then there exist two positive constants $\overline{\alpha}$, $\overline{\beta} \in \mathbb{R}_+$ and a scalar smooth function $\xi(\alpha, \beta)$ with $\xi(0,0) = x^*$ such that if $|\alpha| < \overline{\alpha}$ and $|\beta| < \overline{\beta}$ then Theorem 16 holds true with x^* substituted with $\xi(\alpha, \beta)$.

Proof The proof follows exactly as in Theorem 16 up to equation (2.19), where ψ is a smooth function of its arguments and $\dot{\overline{x}}(t) = \psi\big(\overline{x}(t), 0, 0\big)$ globally and exponentially converges to x^*.

Given our smoothness assumptions, since $\psi(x^*, 0, 0) = 0$ we can apply Theorem 15 (i.e. the implicit function theorem) and be ensured that there must exist, in a neighborhood of $\alpha = \beta = 0$, a smooth function $\xi(\alpha, \beta)$ such that $\xi(0,0) = x^*$ and $\psi\big(\xi(\alpha, \beta), \alpha, \beta\big) = 0$, i.e., $\xi(\alpha, \beta)$ returns the equilibria of system (2.19) for sufficiently small values of α, β.

Then, by performing the change of variables $\overline{\chi} = \overline{x} - \xi(\alpha, \beta)$ and following the same derivations to prove the stability of slowly varying systems in (Khalil, 2001, Section 9.6), it readily follows that the equilibrium points $\xi(\alpha, \beta)$ are exponentially stable in a neighborhood of $\alpha = \beta = 0$. ♢

The previous theorem shows that the initialization of the variables $\boldsymbol{v}(0)$, $\boldsymbol{w}(0)$, $\boldsymbol{y}(0)$, $\boldsymbol{z}(0)$ is critical to the convergence of the correct minimum, but it also assures that sufficiently small errors will have no dramatic effects such as instability. Numerical simulations in fact suggest that the algorithm is robust with respect to numerical errors and quantization noise.

Before turning to the multidimensional scenario, we notice that Theorem 16 guarantees the existence of a critical value $\overline{\varepsilon}_r$ but does not provide indications on its value. This is a known issue in all the systems dealing with separation of time scales. A standard rule of thumb is then to let the rate of convergence of the fast dynamics be sufficiently faster than the one of the slow dynamics, typically 2-10 times faster. In our algorithm the fast dynamics inherits the rate of convergence of the consensus matrix P, given by its spectral gap ρ, i.e., its spectral radius $\sigma = 1 - \rho$. The rate of convergence of the slow dynamics is instead governed by (2.6), which is nonlinear and therefore possibly depending on the initial conditions. However, close the equilibrium point the dynamic behavior is approximately given by $\dot{\overline{x}}(t) \approx -\big(\overline{x}(t) - x^*\big)$, thus, since $x_i(k) \approx \overline{x}(t/\varepsilon)$, then the convergence rate of the algorithm approximately given by $1-\varepsilon$. Thus we aim to let $1-\sigma \gg 1-(1-\varepsilon)$, which provides the rule of thumb

$$\varepsilon \ll \rho \, . \tag{2.21}$$

We then notice that, although the spectral gap ρ might not be known in advance, it is possible to distributedly estimate it, see, e.g., Sahai et al. (2012). However, such rule of thumb might be very conservative. In fact, as pointed out above, if all the f_i's are quadratic, then the algorithm would converge even for $\varepsilon = 1$. As a consequence, the best rate of convergence of the whole algorithm is likely to be achieved by values of ε much larger than those dictated by the rule, although it is very dependent on the functions f_i and therefore very difficult to estimate in advance.

2.2 Convergence Analysis

In this section we analytically characterize the stability and the rate of convergence of the SNRC optimization approach (Algorithm 1) and a novel gradient-descent implementation, say Gradient Descent Consensus (GDC), that is obtained from a simplification of the previous Newton-Raphson approach (Algorithm 2). This analysis is done in a opportune simplificative framework, where the local cost functions are generic quadratic costs, the communications are synchronous and the communication matrix P is an irreducible symmetric stochastic matrix. Although these restrictions are substantial, nonetheless they allow for analytic characterization and stability conditions and optimization of the rate of convergence, which are in general not possible for general convex functions.

The intent is thus to derive general rule-of-thumbs based on these analytical results that could be useful for the design and tuning of the algorithm in the context of general convex cost functions.

We consider an undirected, connected and static network composed by N

agents, each endowed with a local scalar quadratic cost function

$$\psi_i : \mathbb{R} \mapsto \mathbb{R} \qquad \psi_i(x) = \frac{1}{2} a_i (x - b_i)^2$$

with $a_i > 0$ (this implies ψ_i to be strictly convex). The global cost function

$$\overline{\psi} : \mathbb{R} \mapsto \mathbb{R} \qquad \overline{\psi}\,(x) := \frac{1}{N} \sum_{i=1}^{N} \psi_i\,(x)$$

is thus still a quadratic cost. The goal of the agents is to collaborate in order to compute the minimizer x^* of the global cost function $\overline{\psi}$, which is given by (2.1). In this section we will use the following shorthands:

$$\begin{aligned}
g_i\,(x_i\,(k)) &:= \psi_i''\,(x_i\,(k))\,x_i\,(k) - \psi_i'\,(x_i\,(k)) \\
\widetilde{g}_i\,(x_i\,(k)) &:= x_i\,(k) - \psi_i'\,(x_i\,(k)) \\
h_i\,(x_i\,(k)) &:= \psi_i''\,(x_i\,(k)) \\
\boldsymbol{x}\,(k) &:= [x_1\,(k)\ \cdots\ x_N\,(k)]^T \\
\boldsymbol{g}\,(\boldsymbol{x}(k)) &:= [g_1\,(x_1\,(k))\ \cdots\ g_N\,(x_N\,(k))]^T \\
\widetilde{\boldsymbol{g}}\,(\boldsymbol{x}(k)) &:= [\widetilde{g}_1\,(x_1\,(k))\ \cdots\ \widetilde{g}_N\,(x_N\,(k))]^T \\
\boldsymbol{h}\,(\boldsymbol{x}(k)) &:= [h_1\,(x_1\,(k))\ \cdots\ h_N\,(x_N\,(k))]^T \\
\boldsymbol{a} &:= [a_1\ \cdots\ a_N]^T \\
\boldsymbol{b} &:= [b_1\ \cdots\ b_N]^T .
\end{aligned}$$

Distributed Newton-Raphson

We analyze the Newton-Raphson consensus algorithm reported in Algorithm 1.

Introduce the variables $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ to account respectively for $\boldsymbol{g}\big(\boldsymbol{x}(k-1)\big)$ and $\boldsymbol{h}\big(\boldsymbol{x}(k-1)\big)$. Algorithm 1 can be rewritten as

$$\begin{cases}
\boldsymbol{v}(k) &= \boldsymbol{a} \odot \boldsymbol{b} \\
\boldsymbol{w}(k) &= \boldsymbol{a} \\
\boldsymbol{y}(k) &= P\big(\boldsymbol{y}(k-1) + \boldsymbol{a} \odot \boldsymbol{b} - \boldsymbol{v}(k-1)\big) \\
\boldsymbol{z}(k) &= P\big(\boldsymbol{z}(k-1) + \boldsymbol{a} - \boldsymbol{w}(k-1)\big) \\
\boldsymbol{x}(k) &= (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon \dfrac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)}
\end{cases}$$

with initial conditions $\boldsymbol{v}(0) = \boldsymbol{w}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{x}(0) = \boldsymbol{0}$. From

$$\begin{cases} \boldsymbol{v}(k) & = \boldsymbol{a} \odot \boldsymbol{b} \\ \boldsymbol{w}(k) & = \boldsymbol{a} \\ \boldsymbol{y}(k) & = P^k \left(\boldsymbol{a} \odot \boldsymbol{b}\right) \\ \boldsymbol{z}(k) & = P^k \boldsymbol{a} \end{cases}$$

and

$$\boldsymbol{p}(k) := \frac{P^{k+1}\left(\boldsymbol{a} \odot \boldsymbol{b}\right)}{P^{k+1}\boldsymbol{a}} \;;\quad \boldsymbol{p}^* := \frac{\frac{1}{N}\sum_{i=1}^N a_i b_i}{\frac{1}{N}\sum_{i=1}^N a_i}\mathbb{1}$$

we obtain the simplified system

$$\boldsymbol{x}(k+1) = (1-\varepsilon)\boldsymbol{x}(k) + \varepsilon\boldsymbol{p}(k)\;, \qquad \boldsymbol{x}(0) = \boldsymbol{0}\;.$$

We notice that Theorem 10 assures the existence of an $\overline{\varepsilon} \in \mathbb{R}_+$ such that if $\varepsilon < \overline{\varepsilon}$ then Algorithm 1 distributedly and asymptotically computes the global optimum x^*, i.e., $\lim_{k\to+\infty} \boldsymbol{x}(k) = x^*\mathbb{1}$.

In order to compute the rate of convergence, we want to express the systems dynamics in terms of the local error of each agent with respect to the global minimum, i.e., $|x^* - x_i(k)|$, therefore we consider the error vector $\boldsymbol{\xi}(k) := \boldsymbol{x}(k) - \boldsymbol{p}^*$ whose dynamics can be written as

$$\boldsymbol{\xi}(k+1) = (1-\varepsilon)\boldsymbol{\xi}(k) + \varepsilon\big(\boldsymbol{p}^* - \boldsymbol{p}(k)\big),$$

or equivalently as

$$\boldsymbol{\xi}(k) = \sum_{\ell=1}^{k} \varepsilon(1-\varepsilon)^{k-\ell}\big(\boldsymbol{p}^* - \boldsymbol{p}(\ell-1)\big)\;.$$

Consider now that, from consensus theory, it holds that

$$\boldsymbol{y}(k) \xrightarrow{\sigma^k} \frac{1}{N}\sum_{i=1}^N a_i b_i \mathbb{1} \quad \text{and} \quad \boldsymbol{z}(k) \xrightarrow{\sigma^k} \frac{1}{N}\sum_{i=1}^N a_i \mathbb{1}.$$

Recall then that $\boldsymbol{p}(k) = \dfrac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)}$, and that the Hadamard division operator is continuous and differentiable around the point $\boldsymbol{p}^*$. This implies then that there exist a positive constant $c \in \mathbb{R}_+$ which might depend on the initial condition $\boldsymbol{\xi}(0)$ such that

$$\|\boldsymbol{p}^* - \boldsymbol{p}(\ell)\| \leq c\sigma^\ell, \quad \forall \ell$$

where σ is the essential spectral radius of P. Thus, we have that

$$\begin{aligned}\|\boldsymbol{\xi}(k)\| &\leq c\varepsilon\frac{(1-\varepsilon)^k}{\sigma}\sum_{\ell=1}^{k}\left(\frac{\lambda_2}{1-\varepsilon}\right)^{\ell} \\ &= c\varepsilon\frac{(1-\varepsilon)^k-\sigma^k}{1-\varepsilon-\sigma}\,.\end{aligned}$$

Considering then that

$$\|\boldsymbol{\xi}(k)\| \leq \left|\frac{c\varepsilon}{1-\varepsilon-\sigma}\right|(1-\varepsilon)^k + \left|\frac{c\varepsilon}{1-\varepsilon-\sigma}\right|\sigma^k$$

it follows that the convergence rate is dominated by the biggest between $(1-\varepsilon)$ and σ. The previous thus states that it is possible, by setting $\varepsilon = 1$, to directly obtain $\|\boldsymbol{\xi}(k)\| \leq c\sigma^k$, i.e., for the quadratic case the unique factor limiting the convergence rate of the algorithm is given by the speed of the consensus algorithm induced by P.

We can summarize the results obtained above in the following:

Theorem 18 Under the assumption of local quadratic cost functions, Algorithm 1 is ensured to converge for all $\varepsilon \in (0, 2)$ for any positive vector $\boldsymbol{a}$. Moreover the fastest rate of convergence of the algorithm is given by the essential spectral radius of P, namely σ, and it is achieved for any $|\varepsilon| \leq 1-\sigma$.

Distributed Gradient Descent

In this section we consider a modified version of Algorithm 1, with the advantage of requiring a smaller number of local variables and therefore a reduced communication load. As we will see, this trades off with a more restricted interval of ε's guaranteeing the convergence to the global optimum, and thus eventually with a slower convergence rate. The algorithm, initially proposed in Zanella et al. (2012b) and here reported in Algorithm 2, is reminiscent of a distributed gradient descent strategy based on a consensus algorithm.

The analytical derivations of the stability and convergence rate of Algorithm 2 are more involved that those of the previous algorithm and rely on two main steps:

1. the transformation of the algorithm into a Linear Time Invariant (LTI) system, characterized by an additional parameter;

Algorithm 2 Gradient Descent Consensus (GDC) - scalar case

```
    (storage allocation and constraints on parameters)
1:  $\boldsymbol{x}(k), \boldsymbol{y}(k) \in \mathbb{R}^N$ for $k = 0, 1, \ldots$
2:  $\varepsilon \in (0, 1)$

    (initialization)
3:  $\boldsymbol{x}(0) = \boldsymbol{0}$
4:  $\boldsymbol{y}(0) = \widetilde{\boldsymbol{g}}\left(\boldsymbol{x}(-1)\right) = \boldsymbol{0}$

    (main algorithm)
5:  for $k = 1, 2, \ldots$ do
6:      $\widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k-1) + \widetilde{\boldsymbol{g}}\big(\boldsymbol{x}(k-1)\big) - \widetilde{\boldsymbol{g}}\big(\boldsymbol{x}(k-2)\big)$
7:      $\boldsymbol{y}(k) = P\widetilde{\boldsymbol{y}}(k)$
8:      $\boldsymbol{x}(k) = (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon\boldsymbol{y}(k)$
```

2. the adoption of small-gain theory to derive analytical rules to compute the optimal ε and the convergence rate based on σ and the vector $\boldsymbol{a}$.

Be aware that in this context we let ε existing in the interval $(0, 2)$.

Transformation of GDC into an LTI system

Define the new variable $\boldsymbol{v}(k) := \widetilde{\boldsymbol{g}}\big(\boldsymbol{x}(k-1)\big)$, so that Algorithm 2 can be rewritten as

$$\begin{cases} \boldsymbol{v}(k) = \mathrm{diag}\left[\mathbb{1} - \boldsymbol{a}\right] \boldsymbol{x}(k-1) + \mathrm{diag}\left[\boldsymbol{a}\right] \boldsymbol{b} & (2.22) \\ \boldsymbol{y}(k) = P\big(\boldsymbol{y}(k-1) + \boldsymbol{v}(k) - \boldsymbol{v}(k-1)\big) & (2.23) \\ \boldsymbol{x}(k) = (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon\boldsymbol{y}(k) & (2.24) \end{cases}$$

with initial conditions $\boldsymbol{v}(0) = \boldsymbol{y}(0) = \boldsymbol{x}(0) = \boldsymbol{0}$. Substituting (2.22) into (2.23) we obtain

$$\boldsymbol{y}(k) = P\boldsymbol{y}(k-1) + P\mathrm{diag}\left[\mathbb{1} - \boldsymbol{a}\right]\big(\boldsymbol{x}(k-1) - \boldsymbol{x}(k-2)\big)$$

that, substituted into (2.24), gives

$$\begin{aligned} \boldsymbol{x}(k) = \ & \Big((1-\varepsilon)I + \varepsilon P\mathrm{diag}\left[\mathbb{1} - \boldsymbol{a}\right]\Big)\boldsymbol{x}(k-1) + \\ & + \varepsilon P\boldsymbol{y}(k-1) - \varepsilon P\mathrm{diag}\left[\mathbb{1} - \boldsymbol{a}\right]\boldsymbol{x}(k-2) \end{aligned} \qquad (2.25)$$

Rearranging the update rule (2.24) we obtain[3]:

$$\boldsymbol{y}(k-1) = \frac{1}{\varepsilon}\boldsymbol{x}(k-1) - \frac{1-\varepsilon}{\varepsilon}\boldsymbol{x}(k-2)\ . \tag{2.26}$$

Then by substituting (2.26) into (2.25) we eventually rewrite (2.24) as

$$\begin{aligned} \boldsymbol{x}(k) = & \Big((1+\varepsilon)P + (1-\varepsilon)I - \varepsilon P \mathrm{diag}\left[\boldsymbol{a}\right]\Big)\boldsymbol{x}(k-1) + \\ & + \Big(-P + \varepsilon P \mathrm{diag}\left[\boldsymbol{a}\right]\Big)\boldsymbol{x}(k-2) \end{aligned} \tag{2.27}$$

with initial conditions $\boldsymbol{x}(-1) = \boldsymbol{x}(0) = \boldsymbol{0}$.

Let us define the diagonal matrix $\Delta := \mathrm{diag}\left[1-a_1, 1-a_2, \ldots, 1-a_N\right]$, summarizing the deviations from the ideal condition where all parabolic cost functions are identical and with unitary curvature. Clearly

$$\delta := \max_i |1-a_i| \quad \Rightarrow \quad \|\Delta\| = \delta.$$

Let us now define the new state vector $\boldsymbol{\chi}(k) := \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{x}(k-1) \end{bmatrix}$ and the following matrices:

$$A := \begin{bmatrix} (1+\varepsilon)P + (1-\varepsilon)I - \varepsilon P & \varepsilon P - P \\ I & \boldsymbol{0} \end{bmatrix} \tag{2.28}$$

$$B := \begin{bmatrix} \varepsilon P \\ \boldsymbol{0} \end{bmatrix} \qquad C := \begin{bmatrix} -I & I \end{bmatrix}. \tag{2.29}$$

With these we can transform (2.27) into

$$\boldsymbol{\chi}(k+1) = (A + B\Delta C)\boldsymbol{\chi}(k), \tag{2.30}$$

i.e., into the LTI feedback system

$$\begin{cases} \boldsymbol{\chi}(k+1) &=& A\boldsymbol{\chi}(k) + B\boldsymbol{u}(k) \\ \boldsymbol{\nu}(k) &=& C\boldsymbol{\chi}(k) \\ \boldsymbol{u}(k) &=& \Delta\boldsymbol{\nu}(k) \end{cases} \tag{2.31}$$

Therefore $\boldsymbol{x}(k)$ converges to $x^*\mathbb{1}$ if and only if $\boldsymbol{\chi}(k)$ converges to $x^*[\mathbb{1}^T\ \mathbb{1}^T]^T$.

Let us consider the unitary matrix U that diagonalizes the communication matrix P, i.e., $U^T P U = \Lambda$, and let us introduce $\overline{\boldsymbol{\chi}} := \begin{bmatrix} U & 0 \\ 0 & U \end{bmatrix} \boldsymbol{\chi}$. With these we can obtain the equivalent LTI system

$$\overline{\boldsymbol{\chi}}(k+1) = (\overline{A} + \overline{B}\,\overline{\Delta}\,\overline{C})\overline{\boldsymbol{\chi}}(k) \tag{2.32}$$

[3] We notice that there is a causal connection between $\boldsymbol{y}(k)$ and $\boldsymbol{x}(k)$ since (2.24) can be computed only after the computation of (2.23). Nonetheless we can exploit (2.26) being it a relation between quantities that, at time k, are all known.

where

$$\overline{A} = \begin{bmatrix} \Lambda + (1-\varepsilon)I & (\varepsilon - 1)\Lambda \\ I & \mathbf{0} \end{bmatrix} \qquad \overline{B} = \begin{bmatrix} \varepsilon\Lambda \\ \mathbf{0} \end{bmatrix} \tag{2.33}$$

$$\overline{C} = C \qquad\qquad \overline{\Delta} = U^T \Delta U \ . \tag{2.34}$$

The previous system can be rewritten in block-diagonal form by adopting an opportune change of variables $\widetilde{\boldsymbol{\chi}} := V\overline{\boldsymbol{\chi}}$ where V is a simple permutation matrix. More precisely, let $\overline{\boldsymbol{\chi}}$ be $\overline{\boldsymbol{\chi}} = [\overline{\chi}'_1\, \overline{\chi}'_2 \ldots \overline{\chi}'_N\, \overline{\chi}''_1\, \overline{\chi}''_2 \ldots \overline{\chi}''_N]^T$. Then V can be chosen such that $\widetilde{\boldsymbol{\chi}} = [\overline{\chi}'_1\, \overline{\chi}''_1 \overline{\chi}'_2\, \overline{\chi}''_2 \ldots \overline{\chi}'_N\, \overline{\chi}''_N]^T$. In this way we obtain

$$\widetilde{\boldsymbol{\chi}}(k+1) = (\widetilde{A} + \widetilde{B}\,\widetilde{\Delta}\,\widetilde{C})\widetilde{\boldsymbol{\chi}}(k) \tag{2.35}$$

where

$$\widetilde{A} = \begin{bmatrix} A_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & A_N \end{bmatrix} \qquad \widetilde{B} = \begin{bmatrix} B_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & B_N \end{bmatrix} \tag{2.36}$$

$$\widetilde{C} = \begin{bmatrix} C_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & C_N \end{bmatrix} \qquad \widetilde{\Delta} = V^T U^T \Delta U V \tag{2.37}$$

and where $C_i := [-1, 1]$,

$$A_i := \begin{bmatrix} \lambda_i + (1-\varepsilon) & (\varepsilon - 1)\lambda_i \\ 1 & 0 \end{bmatrix} \qquad B_i := \begin{bmatrix} \varepsilon\lambda_i \\ 0 \end{bmatrix} . \tag{2.38}$$

From the previous equation it can be seen that the global dynamics can be decomposed into N parallel subsystem of dimension 2, which are coupled by the uncertainty matrix $\widetilde{\Delta}$. Notice that the dynamics of the global system is thus affine in the uncertainty matrix $\widetilde{\Delta}$, the latter thus amenable for the stability and convergence properties of the whole algorithm.

To analyze these stability properties we now exploit classical small-gain theory results (Desoer and Vidyasagar, 2009, Chap. 5), that guarantee the stability of the global system if the following N perturbed subsystems (with an abuse of notation on $\boldsymbol{x}$, $\boldsymbol{u}$ and $\boldsymbol{y}$)

$$\begin{cases} \boldsymbol{x}(k+1) &=& A_i \boldsymbol{x}(k) + B_i \boldsymbol{u}(k) \\ y(k) &=& C_i \boldsymbol{x}(k) \\ \boldsymbol{u}(k) &=& \Delta_i y(k), \end{cases} \tag{2.39}$$

with Δ_i such that $\|\Delta_i\| \leq \|\widetilde{\Delta}\| = \|\Delta\| = \delta$, are stable. We remark that this kind of results provide in general conservative bounds, since they do not take into account the structure of $\widetilde{\Delta}$, but the knowledge of its Euclidean norm.

Small-gain theory can also be used to analyze the rate of convergence by recasting the computation of the rate of convergence as a stability problem. In fact, by considering the transformation $\overline{\boldsymbol{x}}(k) = \theta^k \boldsymbol{x}(k)$, it is immediate to check that the convergence rate of (2.39) is at least as θ^{-k}, $\theta > 1$ if and only if all the systems

$$\begin{cases} \overline{\boldsymbol{x}}(k+1) &= \theta A_i \overline{\boldsymbol{x}}(k) + \theta B_i \boldsymbol{u}(k) \\ y(k) &= C_i \overline{\boldsymbol{x}}(k) \\ \boldsymbol{u}(k) &= \Delta_i y(k) \end{cases} \tag{2.40}$$

are asymptotically stable.

To this regard, the transfer functions of the input-output systems (2.40) are given by

$$\mathfrak{F}_i(z) = \begin{cases} \dfrac{-\varepsilon\theta}{z - \theta(1-\varepsilon)} & \text{if } i = 1 \\ \dfrac{-\varepsilon\lambda_i\theta(z-\theta)}{(z-\theta\lambda_i)(z-\theta(1-\varepsilon))} & \text{if } i = 2, \ldots, N \end{cases} \tag{2.41}$$

i.e., θ modulates the position of the natural poles λ_i and $(1-\varepsilon)$. Note that the transfer function relative to the average component, i.e., the transfer function relative to the subsystem $i = 1$, has order one due a zero-pole cancellation relative to the eigenvalue $z = \theta$. This is to be expected since it corresponds to the unitary eigenvalue (multiplied by θ) of the global dynamics (2.30) which guarantees that the consensus $\boldsymbol{\chi} = \mathbb{1}$ is an equilibrium point of the global system. Such eigenvalue and its relative eigenspace $\boldsymbol{\chi} = \mathbb{1}$ should be excluded from the stability analysis. This is indeed the case since this eigenvector is independent of Δ, therefore it does not appear in the transfer functions above.

Stability analysis

Now our aim is to apply the small-gain theory, see, e.g., (Desoer and Vidyasagar, 2009, Chap. 5), to prove that the closed loop system 2.31 is finite gain stable. Before doing that we need to introduce the following definition.

Definition 19 A function $\boldsymbol{f} : \mathbb{N}_+ \mapsto \mathbb{R}^N$ is said to belong to the Banach space $l_2(\mathbb{N}_+)$ if it is measurable and in addition $\sum_{n=0}^{\infty} ||\boldsymbol{f}(n)||^2 < \infty$.

The l_2 norm of a function $\boldsymbol{f} \in l_2(\mathbb{N}_+)$ is defined to be $\sqrt{\sum\limits_{n=0}^{\infty} ||\boldsymbol{f}(n)||^2}$.

Based on small-gain theory, the various systems (2.40) are ensured to be asymptotically stable if the products of the l_2 gains of the direct chains and the feedbacks are strictly smaller than 1, i.e., if

$$\max_{\omega\in[0,2\pi)} \left\|\mathfrak{F}_i\left(e^{j\omega}\right)\right\| \left\|\Delta_i\right\| < 1, \qquad i = 1,\dots,N. \tag{2.42}$$

To check whether (2.42) holds we exploit the following:

Lemma 20 Given $\theta \geq 1$ and $\sigma \geq 0$,

$$\max_{\omega\in[0,2\pi)} \left\|\mathfrak{F}_1\left(e^{j\omega}\right)\right\| = \mathcal{F}(\theta,\varepsilon) = \begin{cases} |\mathfrak{F}_1(1)| & = \dfrac{\varepsilon\theta}{|1-\theta(1-\varepsilon)|} \quad \text{if } \varepsilon \in (0,1] \\ |\mathfrak{F}_1(-1)| & = \dfrac{\varepsilon\theta}{|1+\theta(1-\varepsilon)|} \quad \text{if } \varepsilon \in [1,2) \end{cases} \tag{2.43}$$

$$\max_{\omega\in[0,2\pi)} \|\mathfrak{F}_i(e^{j\omega})\| \leq \mathcal{G}(\theta,\varepsilon), \quad i = 2,\dots,N \tag{2.44}$$

where $\mathcal{G}(\theta,\varepsilon)$ is defined as

$$\begin{cases} \dfrac{\varepsilon\sigma\theta}{|1-\theta\sigma|}\dfrac{\big(\theta-1\big)}{|1-\theta(1-\varepsilon)|} \quad \text{if } \varepsilon \in \left(0, 1-\dfrac{1}{\theta^2}\right] \\[2ex] \dfrac{\varepsilon\sigma\theta}{|1-\theta\sigma|}\dfrac{\big(\theta+1\big)}{|1+\theta(1-\varepsilon)|} \quad \text{if } \varepsilon \in \left(1-\dfrac{1}{\theta^2}, 2\right). \end{cases} \tag{2.45}$$

Proof Consider the transfer functions of the input-output systems (2.40) for $i = 1$. Its gain is given by

$$|\mathfrak{F}_1(e^{j\omega})| = \frac{\varepsilon\theta}{\sqrt{1-2\theta(1-\varepsilon)\cos\omega+\theta^2(1-\varepsilon)^2}}.$$

If $\varepsilon \leq 1$, it is clear from (4) that $|\mathfrak{F}_1(e^{j\omega})|$ is maximized for $\omega = 0$, and if $\varepsilon > 1$ the maximum is reached when $\omega = \pi$, yielding easily to (2.43).

Consider the transfer function

$$\frac{\big(z-\theta\big)}{\big(z-\theta(1-\varepsilon)\big)}\ , \tag{2.46}$$

and the quantities x, $a(x)$ and $b(x)$ defined in Figure 2.2. From simple geometric considerations it follows that

$$\big(a(x)\big)^2 = (\theta-x)^2 + (1-x^2)\ , \tag{2.47}$$

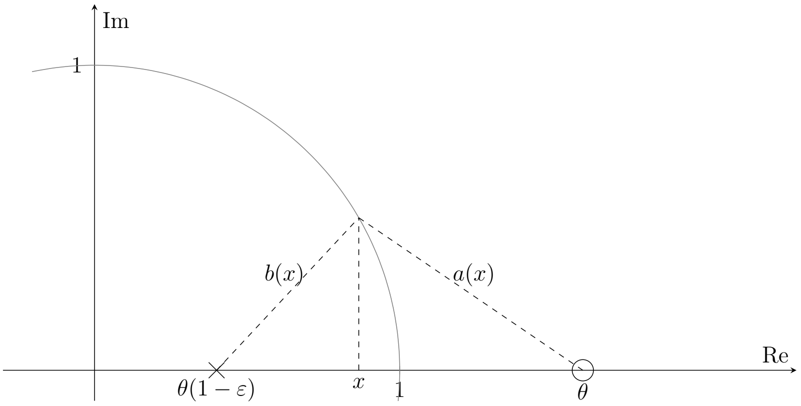


Figure 2.2: Geometrical interpretation of the quantities involved in the computation of the gain of the transfer function in (2.46).

$$\big(b(x)\big)^2 = \big(x - \theta(1-\varepsilon)\big)^2 + (1-x^2)\ . \tag{2.48}$$

The l_2 gain of the transfer function in (2.46) can thus be computed as

$$\max_{x\in[-1,1]} \frac{a(x)}{b(x)} \tag{2.49}$$

where, after trivial simplifications,

$$\frac{a(x)}{b(x)} = \sqrt{\frac{1+\theta^2-2\theta x}{1+\theta^2(1-\varepsilon)^2-2\theta(1-\varepsilon)x}}\ . \tag{2.50}$$

It follows that if $\theta \geq \frac{1}{\sqrt{1-\varepsilon}}$, $\varepsilon \in (0,1)$, then $\frac{a(x)}{b(x)}$ is non-decreasing, otherwise, it is non-increasing if $\theta < \frac{1}{\sqrt{|1-\varepsilon|}}$, $\varepsilon \in (0,2)$, or $\theta \geq \frac{1}{\sqrt{1-\varepsilon}}$ for $\varepsilon \in (1,2)$. Thus, depending on the values of θ and ε, the gain of (2.46) is obtained either for $z=1$ or $z=-1$, i.e., $\frac{\big(\theta+1\big)}{\big|1+\theta(1-\varepsilon)\big|}$ when it is non-increasing and $\frac{\big(\theta-1\big)}{\big|1-\theta(1-\varepsilon)\big|}$ when it has a non-decreasing behavior. Equation (2.45) is proven considering that

$$\min_{|z|=1} |z-\theta\lambda_2| = |1-\theta\sigma|\ . \tag{2.51}$$

Notice now that the argument minimizing $|z-\theta\lambda_2|$ is either $z=+1$ or $z=-1$, depending on the sign of λ_2. Thus (2.44) holds as an inequality, since it may happen that the z maximizing the modulus of (2.46) is $+1$, while the one solving (2.51) is -1, or vice versa. ♢

With Lemma 20 and condition (2.42) we then bound the stability region of the proposed algorithm (in terms of ε) and upper and lower bound its rate of convergence. We start with the smallest ε guaranteeing stability, that can be defined via the following optimization problem:

$$\begin{aligned} \varepsilon_c(\sigma, \delta) := \quad & \sup & & \varepsilon \\ & \text{such that} & & \mathcal{F}(1, \varepsilon)\delta < 1 \\ & & & \mathcal{G}(1, \varepsilon)\delta < 1. \end{aligned} \tag{2.52}$$

The smallest ε guaranteeing stability is then described by the following:

Theorem 21 Let

$$\varepsilon_1 := \frac{2}{1+\delta}, \quad \varepsilon_2 := \frac{2(1-\sigma)}{1-\sigma+2\delta\sigma}, \quad \varepsilon_c := \min\{\varepsilon_1, \varepsilon_2\}. \tag{2.53}$$

If $\varepsilon \in (0, \varepsilon_c)$ then Algorithm 2 converges to the global optimum.

Proof Assume that Lemma 20 holds, and notice that in this case $\theta = 1$.

Solving (2.52) only considering the condition $\mathcal{G}(1, \varepsilon)\delta < 1$ it follows that the algorithm is asymptotically stable if $\dfrac{2\varepsilon\sigma\delta}{\big(1-\sigma\big)\big(2-\varepsilon\big)} < 1$ that easily leads to $\varepsilon = \varepsilon_2$. For $\mathcal{F}(1, \varepsilon)\delta < 1$, trivial computations, under the assumptions that $\varepsilon \in (0, 2)$, yield to $\varepsilon = \varepsilon_1$ if $\delta > 1$, and for all ε otherwise.

Combining together all the constraints the implication is that (2.52) is solved by $\varepsilon_c = \min\{\varepsilon_1, \varepsilon_2\}$. For the convergence, $\varepsilon \in (0, \varepsilon_c)$ is immediate. $\diamondsuit$

It is also easy to verify that if $\sigma > \frac{1}{2}$ then $\varepsilon_2 < \varepsilon_1$ and thus also $\varepsilon_c = \varepsilon_2$. In general, large networks are such that $\sigma \approx 1$, thus in large networks the typical limiting factor is ε_2 (recall that ε_1 is associated to the dynamics of the average component). To complete the characterization in large networks we also notice that, in the same situation, if $\delta\sigma \gg 1 - \sigma$ then $\varepsilon_c \approx \dfrac{1-\sigma}{\delta}$. That is, in this case the critical ε_c is directly proportional to the spectral gap and inversely proportional to the deviation from the ideal condition where all the costs are jointly curved parabolas.

Convergence rate analysis

Letting $\eta := 1/\theta$, we can bound the rate of convergence by means of the optimization problem

$$
\begin{aligned}
(\varepsilon^*, \eta^*) := \;\; & \arg\inf_{\varepsilon,\eta} \quad \eta \\
& \text{such that} \quad \mathcal{F}\left(\tfrac{1}{\eta}, \varepsilon\right)\delta < 1 \\
& \qquad\qquad\;\; \mathcal{G}\left(\tfrac{1}{\eta}, \varepsilon\right)\delta < 1,
\end{aligned}
\tag{2.54}
$$

To solve (2.54) we divide it in two subproblems that are then analyzed separately. In particular, defining

$$
\begin{aligned}
\eta_1(\varepsilon) := \;\; & \inf \quad \eta \\
& \text{such that} \quad \mathcal{F}\left(\tfrac{1}{\eta}, \varepsilon\right)\delta < 1
\end{aligned}
\tag{2.55}
$$

$$
\begin{aligned}
\eta_2(\varepsilon) := \;\; & \inf \quad \eta \\
& \text{such that} \quad \mathcal{G}\left(\tfrac{1}{\eta}, \varepsilon\right)\delta < 1
\end{aligned}
\tag{2.56}
$$

$$
\eta(\varepsilon) := \;\; \max\{\eta_1(\varepsilon), \eta_2(\varepsilon)\}
\tag{2.57}
$$

It follows that the solution of (2.54) can be rewritten as

$$
\varepsilon^* = \arg\inf_\varepsilon \eta(\varepsilon), \quad \eta^* = \eta(\varepsilon^*).
$$

The solutions of (2.55) and (2.56) are then given by:

$$
\eta_1(\varepsilon) = \begin{cases} 1 - \varepsilon(1-\delta) & \text{if } 0 < \varepsilon \le 1 \\ -1 + \varepsilon(1+\delta) & \text{if } 1 < \varepsilon \le \varepsilon_1 \end{cases}
\tag{2.58}
$$

$$
\eta_2(\varepsilon) = \begin{cases} \kappa_1(\varepsilon) & \text{if } 0 < \varepsilon \le \overline{\varepsilon} \\ \kappa_2(\varepsilon) & \text{if } \overline{\varepsilon} < \varepsilon \le \varepsilon_2. \end{cases}
\tag{2.59}
$$

where

$$
\kappa_1(\varepsilon) := \frac{\sigma + 1 - \varepsilon(1+\sigma\delta) + \sqrt{\big(\sigma + 1 - \varepsilon(1+\sigma\delta)\big)^2 - 4\sigma\big(1 - \varepsilon(1+\delta)\big)}}{2}
$$

$$
\kappa_2(\varepsilon) := \frac{\sigma - 1 + \varepsilon(1+\sigma\delta) - \sqrt{\big(\sigma - 1 + \varepsilon(1+\sigma\delta)\big)^2 + 4\sigma\big(1 - \varepsilon(1-\delta)\big)}}{2}
$$

$$
\overline{\varepsilon} := \frac{-\sigma^2 + 2\sigma\delta + 2 - \sigma\sqrt{\sigma^2 + 4\sigma\delta(1+\sigma\delta)}}{2\big(\sigma\delta + 1\big)^2}
$$

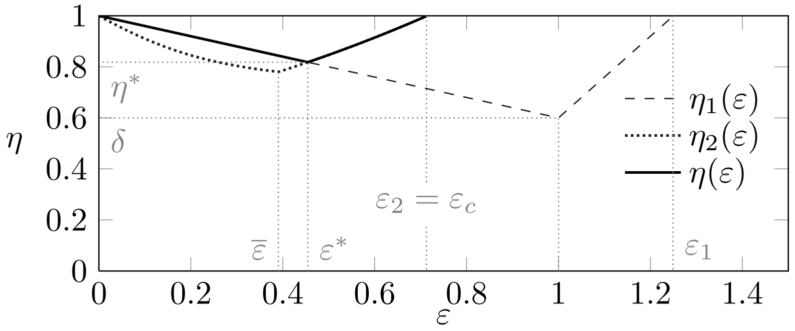


Figure 2.3: Numerical evaluation of function η_1, η_2, η as given in (2.58), (2.59) and problem (2.57) for $\sigma = \delta = 0.6$.

A graphical representation of the functions and variables defined above is given in Figure 2.3 for a specific choice of the parameters δ and σ.

The optimal rate of converge η^* as a function of the parameters σ and δ cannot be given in closed form, however can be readily computed based on the previous theorem as shown in Figure 2.4. This figure clearly shows that rate of convergence decreases as either δ or σ are close to one.

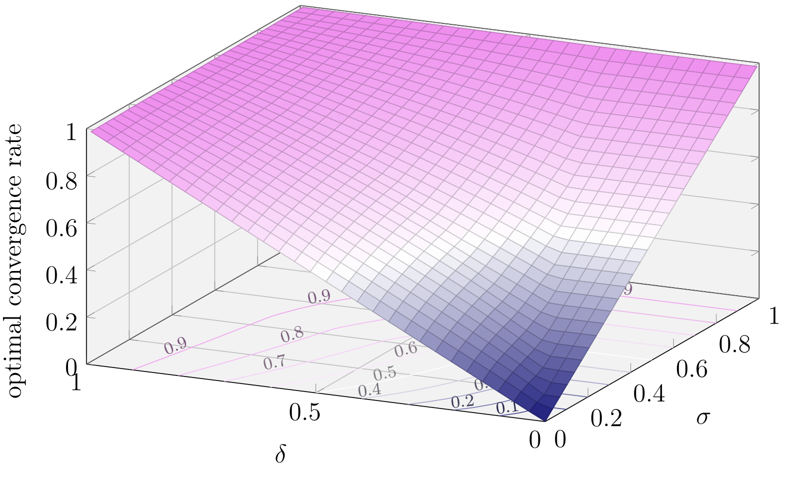


Figure 2.4: Optimal rate of convergence η^* as a function of σ and δ parameters.

Considering that $\eta_2(\overline{\varepsilon}) := \dfrac{\sigma + \sqrt{\sigma^2 + 4\sigma\delta(1+\sigma\delta)}}{2(1+\sigma\delta)}$, as a completion of the remark proposed after Theorem 21, (2.59) implies that in large networks and for $\delta \neq 0$, $\overline{\varepsilon} = \dfrac{2(1-\sigma)}{1+2\delta} + o(1-\sigma),$, $\eta_2(\overline{\varepsilon}) = 1 - \dfrac{1-\sigma}{1+2\delta} + o(1-\sigma)$. As expected, the convergence rate η is lower with small spectral gaps $1-\sigma$ and higher with large deviations of the curvature of the local costs given by a_i.

2.3 The multidimensional case

In the previous sections we derived the algorithm for the scalar case considering that, for scalar quadratic local costs, the optimum is given by (2.1). We could derive the algorithm for the multidimensional case using exactly the same intuitions: in fact considering multidimensional quadratic local costs $f_i(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{b}_i)^T A_i (\boldsymbol{x} - \boldsymbol{b}_i)$, with $\boldsymbol{x} := [x_1 \ \cdots \ x_M]^T$ and $\boldsymbol{b}_i, A_i$ of suitable dimensions, it follows immediately that

$$\boldsymbol{x}^* = \left(\frac{1}{N} \sum_{i=1}^{N} A_i \right)^{-1} \left(\frac{1}{N} \sum_{i=1}^{N} A_i \boldsymbol{b}_i \right).$$

A sensible extension of the scalar algorithm to a multidimensional scenario is to replace $f_i'(x_i)$ with the gradient $\nabla f_i(\boldsymbol{x}_i)$ and the $f_i''(x_i)$ with the full Hessian $\nabla^2 f_i(\boldsymbol{x}_i)$. However, this is not the only possible choice, and indeed, by suitablely defining functions $\boldsymbol{g}_i(\boldsymbol{x}_i)$ and $H_i(\boldsymbol{x}_i)$, which play the role of $g_i(\boldsymbol{x}_i)$ and $h_i(\boldsymbol{x}_i)$ of the scalar case, one can obtain different procedures with different convergence properties and different computational/communication requirements. The following are (some) plausible choices for $H_i \in \mathbb{R}^{M \times M}$:

$$H_i\big(\boldsymbol{x}_i(k)\big) = \nabla^2 f_i\big(\boldsymbol{x}_i(k)\big) \tag{2.60}$$

$$H_i\big(\boldsymbol{x}_i(k)\big) = \operatorname{diag}\Big[\nabla^2 f_i\big(\boldsymbol{x}_i(k)\big)\Big] = \begin{bmatrix} \left.\dfrac{\partial^2 f_i}{\partial x_1^2}\right|_{\boldsymbol{x}_i(k)} & & 0 \\ & \ddots & \\ 0 & & \left.\dfrac{\partial^2 f_i}{\partial x_M^2}\right|_{\boldsymbol{x}_i(k)} \end{bmatrix} \tag{2.61}$$

$$H_i\big(\boldsymbol{x}_i(k)\big) = I_M. \tag{2.62}$$

The multidimensional version of Algorithm 1 is given by Algorithm 3, where H_i is left undefined and depending on its choice, it leads to a different version of the algorithm. The three proposed choices lead to the following algorithms:

- Equation (2.60) → **Synchronous Newton-Raphson Consensus (SNRC)**: in this case it is possible to rewrite Algorithm 3 as done in Section 2.1 and show that, for sufficiently small ε, $\boldsymbol{x}_i(k) \approx \overline{\boldsymbol{x}}(t/\varepsilon)$, where $\overline{\boldsymbol{x}}(t)$ evolves according to the continuous-time Newton-Raphson dynamics

$$\dot{\overline{\boldsymbol{x}}}(t) = -\Big[\nabla^2 \overline{f}\big(\overline{\boldsymbol{x}}(t)\big)\Big]^{-1} \nabla \overline{f}\big(\overline{\boldsymbol{x}}(t)\big) ,$$

which, analogously to its scalar version, can be shown to converge to the global optimum $\boldsymbol{x}^*$.

Algorithm 3 Synchronous Newton-Raphson Consensus (SNRC), Jacobi Consensus (JC), Gradient Descent Consensus (GDC) – multidimensional case

```
(storage allocation and constraints on the parameters)
$\boldsymbol{x}_i(k), \boldsymbol{y}_i(k) \in \mathbb{R}^M$, $Z_i(k) \in \mathbb{R}^{M \times M}$, $\forall i \in \mathcal{V}$
$\varepsilon \in (0,1)$
(initialization)
$\boldsymbol{x}_i(0) = \mathbf{0}$, $\forall i \in \mathcal{V}$
$\boldsymbol{y}_i(0) = \boldsymbol{g}_i\big(\boldsymbol{x}_i(-1)\big) = \mathbf{0}$, $Z_i(0) = H_i\big(\boldsymbol{x}_i(-1)\big) = I$, $\forall i \in \mathcal{V}$
(main algorithm)
for $k = 1, 2, \ldots$ do
    for $i = 1, \ldots, N$ do
        $\boldsymbol{x}_i(k) = (1-\varepsilon)\boldsymbol{x}_i(k-1) + \varepsilon\big(Z_i(k-1)\big)^{-1}\boldsymbol{y}_i(k-1)$   (local update)
        $\widetilde{\boldsymbol{y}}_i(k) = \boldsymbol{y}_i(k-1) + \boldsymbol{g}_i\big(\boldsymbol{x}_i(k-1)\big) - \boldsymbol{g}_i\big(\boldsymbol{x}_i(k-2)\big)$   (local update)
        $\widetilde{Z}_i(k) = Z_i(k-1) + H_i\big(\boldsymbol{x}_i(k-1)\big) - H_i\big(\boldsymbol{x}_i(k-2)\big)$   (local update)
    $Y(k) = \big(P \otimes I_N\big)Y(k)$   (multidimensional consensus step)
    $\mathbf{Z}(k) = \big(P \otimes I_N\big)\mathbf{Z}(k)$   (multidimensional consensus step)
```

• Equation (2.61) → **Jacobi Consensus (JC)**: choice (2.60) requires agents to exchange information on $O\left(M^2\right)$ scalars, and this could pose problems under heavy communication bandwidth constraints and large M's. Choice (2.61) instead reduces the amount of information to be exchanged via the underlying diagonalization process, also called Jacobi approximation[4]. In this case, for sufficiently small ε, $\boldsymbol{x}_i(k) \approx \overline{\boldsymbol{x}}(t/\varepsilon)$, where $\overline{\boldsymbol{x}}(t)$ evolves according to the continuous-time dynamics

$$\dot{\overline{\boldsymbol{x}}}(t) = -\left(\mathrm{diag}\Big[\nabla^2 \overline{f}\big(\overline{\boldsymbol{x}}(t)\big)\Big]\right)^{-1} \nabla \overline{f}\big(\overline{\boldsymbol{x}}(t)\big)\ ,$$

which can be shown to converge to the global optimum $\boldsymbol{x}^*$ with a convergence rate that in general is slower than the Newton-Raphson when the global cost function is skewed.

• Equation (2.62) → **Gradient Descent Consensus (GDC)**: this choice is motivated in frameworks where the computation of the local second derivatives $\left.\dfrac{\partial^2 f_i}{\partial x_m^2}\right|_{\boldsymbol{x}_i(k)}$ is expensive, or where the second derivatives simply might not be continuous. With this choice Algorithm 3 reduces to a distributed gradient-descent procedure. In fact, for sufficiently small ε, $\boldsymbol{x}_i(k) \approx \overline{\boldsymbol{x}}(t/\varepsilon)$ with $\overline{\boldsymbol{x}}(t)$

[4] In centralized approaches, nulling the Hessian's off-diagonal terms is a well-known procedure, see, e.g., Becker and Le Cun (1988). See also, e.g., see Athuraliya and Low (2000); Zargham et al. (2011), for other Jacobi algorithms with different communication structures.

evolving according to the continuous-time dynamics

$$\dot{\overline{\boldsymbol{x}}}(t) = -\nabla \overline{f}\big(\overline{\boldsymbol{x}}(t)\big) \,,$$

which one again is guaranteed to converge to the global optimum $\boldsymbol{x}^*$.

The following Table 2.1 summarizes the various costs of the previously proposed strategies.

Choice	SNRC (2.60)	JC (2.61)	GDC (2.62)
Computational Cost	$O\left(M^3\right)$	$O\left(M\right)$	$O\left(M\right)$
Communication Cost	$O\left(M^2\right)$	$O\left(M\right)$	$O\left(M\right)$
Memory Cost	$O\left(M^2\right)$	$O\left(M\right)$	$O\left(M\right)$

Table 2.1: Computational, communication and memory costs of SNRC, JC, GDC per single unit and single step (lines 5 to 7 of Algorithm 3).

The following theorem characterizes the convergence properties of Algorithm 3 (see definitions in page 31) and it is the multidimensional version of Theorem 16:

Theorem 22 Consider Algorithm 3 with arbitrary initial conditions $\boldsymbol{x}_i(0)$, H_i defined in (2.60) or (2.61) or (2.62), and Assumption 8 holding true. Then for every open ball $B_r^{\boldsymbol{x}^*} := \left\{X \in \mathbb{R}^{MN} \mid \|X - \mathbb{1} \otimes \boldsymbol{x}^*\| < r\right\}$ there exist two positive constants $\overline{\varepsilon}_r$, c_r such that if $\varepsilon < \overline{\varepsilon}_r$ then there exists $\gamma_\varepsilon > 0$ such that

$$X(0) \in B_r^{\boldsymbol{x}^*} \quad \Rightarrow \quad \|X(k) - \mathbb{1} \otimes \boldsymbol{x}^*\| \leq c_r e^{-\gamma_\varepsilon k} \, \|X(0) - \mathbb{1} \otimes \boldsymbol{x}^*\| \,.$$

for all $k \in \mathbb{N}$.

Proof The proof follows closely the proof of Theorem 16 thus in the interest of space we provide just a simple sketch. Notice that it involves the following alternative notation:

$$\Pi^{\|} := \frac{\mathbb{1}\mathbb{1}^T}{N} \otimes I_M, \quad \Pi^{\perp} := \left(I_N - \frac{\mathbb{1}\mathbb{1}^T}{N}\right) \otimes I_M,$$

$$X^{\|}(k) := \Pi^{\|} X(k), \quad X^{\perp}(k) := \Pi^{\perp} X(k).$$

To prove the theorem we start recognizing that, for sufficiently small ε, the convergence properties of the algorithm are the same as the continuous time system

$$\begin{cases} \varepsilon\dot{V}(t) &= -V(t) + G\big(X(t)\big) \\ \varepsilon\dot{\boldsymbol{W}}(t) &= -\boldsymbol{W}(t) + \boldsymbol{H}\big(X(t)\big) \\ \varepsilon\dot{Y}(t) &= -KY(t) + (I-K)\Big[G\big(X(t)\big) - V(t)\Big] \\ \varepsilon\dot{\boldsymbol{Z}}(t) &= -K\boldsymbol{Z}(t) + (I-K)\Big[\boldsymbol{H}\big(X(t)\big) - \boldsymbol{W}(t)\Big] \\ \dot{\boldsymbol{x}}_i(t) &= -\boldsymbol{x}_i(t) + \big(Z_i(t)\big)^{-1}\boldsymbol{y}_i(t) \qquad i = 1, \ldots, N \end{cases} \tag{2.63}$$

where $K := I_{MN} - (P \otimes I_M)$ (in gray indications for the dimensions of the identity matrices) is again positive semidefinite. Then, with the substitutions $D(t) := Y(t) - V(t)$, $\boldsymbol{B}(t) := \boldsymbol{Z}(t) - \boldsymbol{W}(t)$ one can prove as before that the boundary layer system of (2.63) admits the globally exponentially stable equilibrium

$$\lim_{t\to\infty} \begin{bmatrix} V(t) \\ \boldsymbol{W}(t) \\ D^{\perp}(t) \\ \boldsymbol{B}^{\perp}(t) \end{bmatrix} = \begin{bmatrix} G(X) \\ \boldsymbol{H}(X) \\ -\Pi^{\perp}G(X) \\ -\Pi^{\perp}\boldsymbol{H}(X) \end{bmatrix}. \tag{2.64}$$

The stability of the reduced system can instead be analyzed decomposing again its dynamics along the projections given by $\Pi^{\perp}$ and $\Pi^{\parallel}$, obtaining a continuous time non-linear cascade system equivalent to (2.17) whose global stability properties are ensured by Theorem 13. Similarly to the scalar version of the algorithm, the dynamics of the average $\overline{\boldsymbol{x}}(t)$ follow

$$\dot{\overline{\boldsymbol{x}}}(t) = -\overline{\boldsymbol{x}}(t) + \Big(\overline{H}\big(\overline{\boldsymbol{x}}(t)\big)\Big)^{-1}\overline{\boldsymbol{g}}\big(\overline{\boldsymbol{x}}(t)\big)\,. \tag{2.65}$$

For all the cases (2.60), (2.61) and (2.62), then, it follows that $V(\boldsymbol{x}) := \overline{f}(\boldsymbol{x}) - \overline{f}(\boldsymbol{x}^*)$ is a Lyapunov function for the reduced system which guarantees exponential converges to $\boldsymbol{x}^*$. $\diamondsuit$

We remark that $\overline{\varepsilon}_r$ in Theorem 22 depends also on the particular choice for H_i. The list of choices for H_i given by (2.60), (2.61) and (2.62) is not exhaustive. For example, future directions are to implement distributed quasi-Newton procedures. To this regard, we recall that approximations of the Hessians that do not maintain symmetry and positive definiteness or are bad conditioned require additional modification steps, e.g., through Cholesky factorizations Golub and Van Loan (1996). Finally, we notice that in scalar scenarios JC and SNRC are equivalent, while GDC corresponds to algorithms requiring just the knowledge of first derivatives.

2.4 Numerical Examples

In this section we analyze the effects of different choices of ε on the scalar SNRC and we numerically compare the performance of the multidimensional SNRC, JC and GDC. Finally, the scalar SNRC is compared with some of the most important distributed convex optimization algorithms available in literature.

In all the simulations we consider a ring network of N agents that communicate only to their left and right neighbors through the consensus matrix

$$P = \begin{bmatrix} 0.5 & 0.25 & & & 0.25 \\ 0.25 & 0.5 & 0.25 & & \\ & \ddots & \ddots & \ddots & \\ & & 0.25 & 0.5 & 0.25 \\ 0.25 & & & 0.25 & 0.5 \end{bmatrix}. \tag{2.66}$$

Here, $\sigma \approx 0.99$ so that the spectral gap is $\rho \approx 0.01$.

2.4.1 Scalar scenario

Here we consider scalar costs of the form

$$f_i(x) = c_i e^{a_i x} + d_i e^{-b_i x},$$

$i = 1, \ldots, N$, $N = 30$, with $a_i, b_i \sim \mathcal{U}[0, 0.2]$, $c_i, d_i \sim \mathcal{U}[0, 1]$ ($\mathcal{U}$ indicates the uniform distribution).

Figure 2.5 compares the evolution of the local states x_i of the continuous system (2.5) for different values of ε. When ε is not sufficiently small, then the trajectories of $x_i(t)$ are different even if they all start from the same initial condition $x_i(0) = 0$. As ε decreases, the difference between the two time scales becomes more evident and all the trajectories $x_i(k)$ become closer to the trajectory given by the slow NR dynamics $\overline{x}(t/\varepsilon)$ given in (2.6) and guaranteed to converge to the global optimum x^*.

In Figures 2.6-2.7 we address the robustness of the proposed algorithm with respect to the choice of the initial conditions. In particular, Figure 2.6 shows that if $\alpha = \beta = 0$ then the local states $x_i(t)$ converge to the optimum x^* for arbitrary initial conditions $x_i(0)$, as guaranteed by Theorem 16. Figure 2.7 considers, besides different initial conditions $x_i(0)$, also perturbed initial conditions $\boldsymbol{v}(0)$, $\boldsymbol{w}(0)$, $\boldsymbol{y}(0)$, $\boldsymbol{z}(0)$ leading to non null α's and β's. More precisely we apply Algorithm 1 to different random initial conditions such that

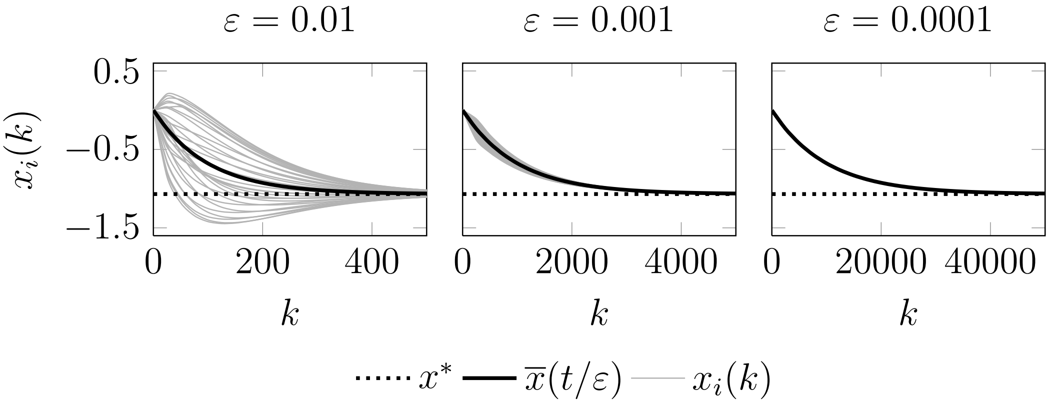


Figure 2.5: Temporal evolution of system (2.5) for different values of ε, with $N = 30$. The black dotted line indicates x^*. The black solid line indicates the slow dynamics $\overline{x}(t/\varepsilon)$ of Equation (2.6). As ε decreases. the difference between the time scale of the slow and fast dynamics increases, and the local states $x_i(k)$ converge to the manifold of $\overline{x}(t/\varepsilon)$.

$\alpha, \beta \sim \mathcal{U}[-\upsilon, \upsilon]$. Figure 2.7 shows the boxplots of the errors $x_i(+\infty) - x^*$ for different υ's based on 300 Monte Carlo runs with $\varepsilon = 0.01$ and $N = 30$.

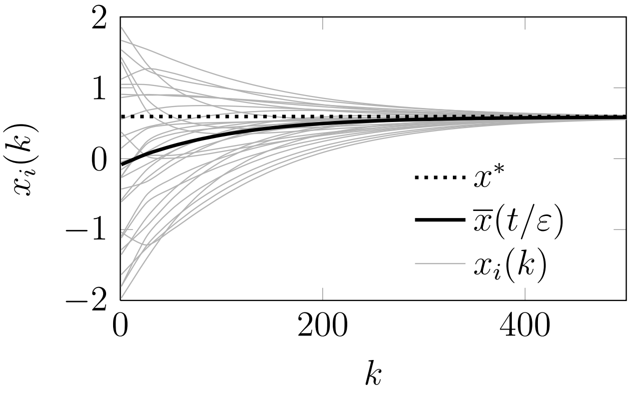


Figure 2.6: Time evolution of the local states $x_i(k)$ with $\boldsymbol{v}(0) = \boldsymbol{w}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{0}$ and $x_i(0) \sim \mathcal{U}[-2, 2]$. In all the experiments $\varepsilon = 0.01$ and $N = 30$.

2.4.2 Multidimensional scenario

Here we consider bi-dimensional costs of the form

$$f_i(\boldsymbol{x}) = e^{(\boldsymbol{x}-\boldsymbol{b}_i)^T A_i (\boldsymbol{x}-\boldsymbol{b}_i)},$$

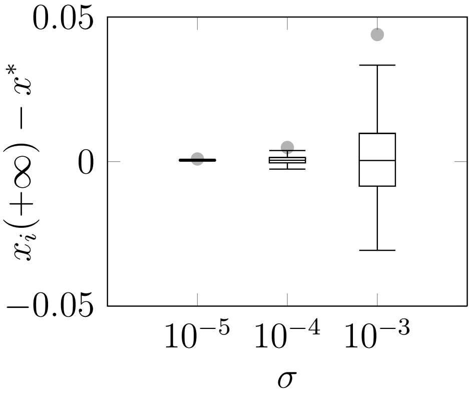


Figure 2.7: Empirical distribution of the errors $x_i(+\infty) - x^*$ under artificially perturbed initial conditions $\alpha(0), \beta(0) \sim \mathcal{U}[-v, v]$ for different values of v. In all the experiments $\varepsilon = 0.01$ and $N = 30$.

$i = 1, \ldots, 15$, with $\boldsymbol{b}_i \sim \left[\mathcal{U}[-5,5] \;\; \mathcal{U}[-5,5]\right]^T$, $A_i = D_i D_i^T > 0$, and

$$D_i := \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2} \,. \tag{2.67}$$

Define the $\mathcal{R}$-distribution as:

$$\mathcal{R}[c, d] := \begin{cases} c & \text{with probability} \quad 0.5 \\ d & \text{with probability} \quad 0.5 \end{cases} .$$

We compare the performances of the previous algorithms in the following three different scenarios:

$$\mathrm{S}_1 : \qquad \begin{cases} d_{11} & = d_{22} \sim \mathcal{U}[-0.08, 0.08] \; \mathcal{R}[-1, 1] \\ d_{12} & \sim \mathcal{U}[-0.08, 0.08] \; \mathcal{R}[-0.25, 0.5] \\ d_{21} & \sim \mathcal{U}[-0.08, 0.08] \; \mathcal{R}[-0.5, 0.25] \end{cases} \tag{2.68}$$

where the axes of each contour plot are randomly oriented in the bi-dimensional plane.

$$\mathrm{S}_2 : \qquad \begin{cases} d_{11} & \sim \mathcal{U}[-0.08, 0.08] \\ d_{12} & = d_{21} = 0 \\ d_{22} & = 2\, d_{11} \end{cases} \tag{2.69}$$

where the axes of all the contour plots of the f_i surfaces are aligned with the axes of the natural reference system.

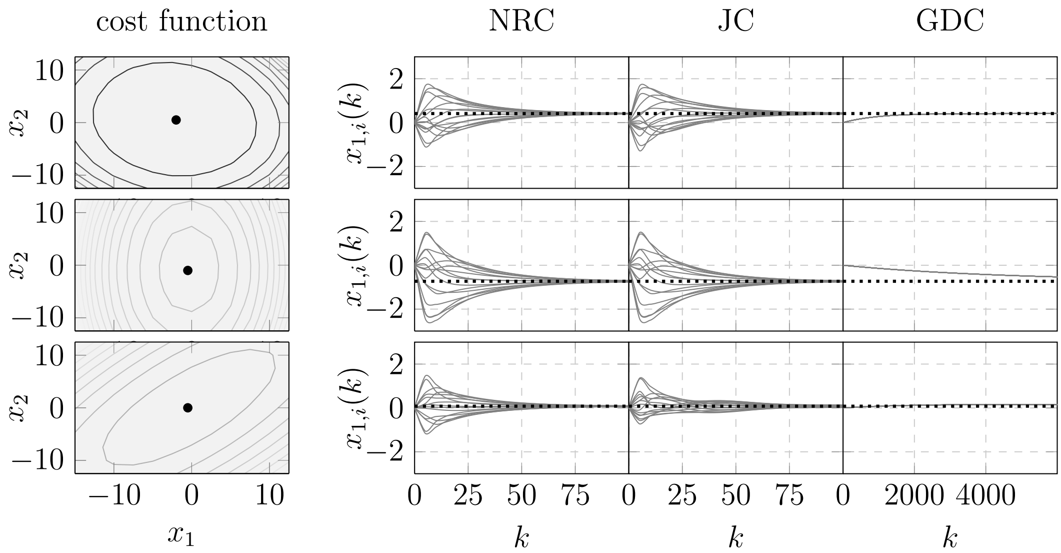


Figure 2.8: First column on the left, contours plot of global function $\bar{f}$'s for scenarios S_1, S_2, S_3, respectively (from top to bottom). Black dots indicate the positions of the global minima $\mathbf{x}^*$. Second, third and fourth columns, temporal evolutions of the first components of the local states $\mathbf{x}_1$, for the case $\varepsilon = 0.25$ and $N = 15$. In particular: second column, Synchronous Newton-Raphson Consensus. Third column, Jacobi Consensus. Fourth column, Gradient Descent Consensus. The black dashed lines indicate the first components of the global optima $\mathbf{x}^*$. Notice that we show a bigger number of time steps for the GDC (fourth column).

$$\mathrm{S}_3: \quad \begin{cases} d_{11} & \sim \mathcal{U}\left[-0.08, 0.08\right] \\ d_{12} & = d_{21} = -0.01 \\ d_{22} & \sim \mathcal{R}[0.9, 1.1]\, d_{11} \end{cases} \tag{2.70}$$

i.e. the axes of each contour plot are randomly oriented along the bisection of the first and third quadrant.

These cost functions are skewed, i.e., their level curves are close to ellipsoids whose axes are oriented along the bisection of the first and third quadrant.

The contour plots of the global cost functions $\bar{f}$'s generated using (2.68), (2.69) and (2.70), and the evolution of the local states $\mathbf{x}_i$ for the three algorithms are shown in Figure 2.8. We notice that SNRC and JC have qualitatively the same behavior for the scenarios (2.68) and (2.69). This is because the approximation introduced in JC is actually a good approximation of the analytical Hessians $\nabla^2 f_i\left(\mathbf{x}_i(k)\right)$. Conversely, GDC presents a remarkably slower convergence rate. Since the computational time of JC and GDC are comparable, JC seems to represent the best choice among all the presented solutions.

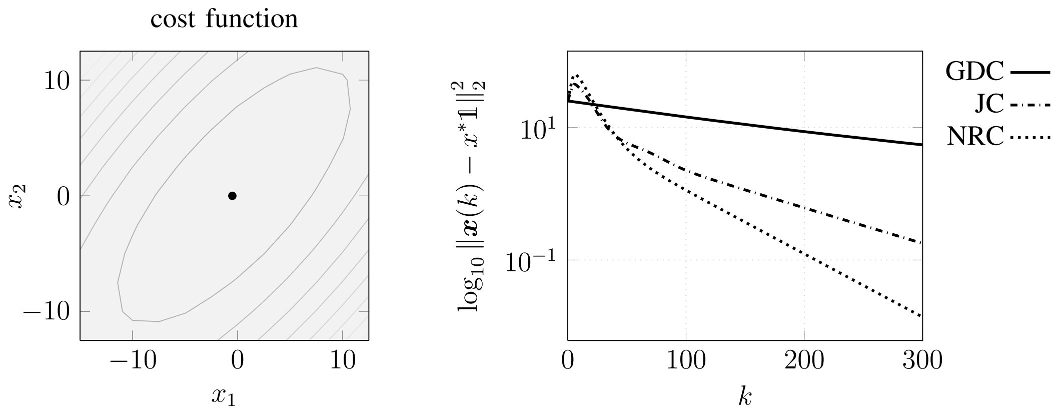


Figure 2.9: Contour plot of the global cost $\overline{f}$ and evolution of the errors choosing (2.60), (2.61) or (2.62) in Algorithm 3. In this experiment, for SNRC and JC $\varepsilon = 0.25$ while for GDC $\varepsilon = 1$. $N = 15$.

Figure 2.9 shows the contour plot of the global cost $\overline{f}$ for a typical realization of the parameters and the corresponding evolution of the errors choosing SNRC (definition (2.60)), JC (definition (2.61)), and GDC (definition (2.62)) in Algorithm 3. The parameter ε was manually tuned for each algorithm to obtain the best performance. We notice that the differences between SNRC and JC are evident but not resounding, due to the fact that the Jacobi approximations are in this case a good approximation of the analytical Hessians. Conversely, GDC presents a slower convergence rate which is a known drawback of gradient descent algorithms. It is also interesting to notice that, initially, the errors increase. This is not surprising since agents at the beginning diverge from each other (see left panel of Figure 2.5), but eventually consensus takes place and the errors reduce.

2.4.3 Comparisons

We compare Algorithm 1 and its accelerated version, denoted as Synchronous Fast Newton-Raphson Consensus (SFNRC) and described in detail in Algorithm 4, with three distributed convex optimization methods, namely the DSM, the Distributed Control Method (DCM) and the ADMM, described respectively in Algorithm 5, 6 and 7. The following discussion provides some details about these strategies.

SFNRC The Synchronous Fast Newton-Raphson Consensus is an accelerated version of Algorithm 1: it inherits the structure of the so called *second* *order diffusive schedules*, see, e.g., Muthukrishnan et al. (1998), that exploits an additional level of memory (here the states $\boldsymbol{y}(k+1)$, $\boldsymbol{z}(k+1)$ of the various agents) to speed up the convergence properties of the consensus strategy.

This method has been applied initially for a load balancing problem on an undirected connected graph characterized by a weight distribution $\boldsymbol{w}(0)$ on the nodes. In this scenario, the problem consists in determining a schedule to move weights in each step across edges so as to balance the weights on the nodes. The second order diffusive schedules are modeled as $\boldsymbol{w}(k+1) = \varphi P\boldsymbol{w}(k) + (1-\varphi)\boldsymbol{w}(k-1)$, $\boldsymbol{w}(1) = P\boldsymbol{w}(0)$, for some appropriate φ. It is well known that if φ lies in the intervals $(-\infty, 0]$ and $[2, +\infty)$, the second order schedule does not even converge. Hence, we focus on $\varphi \in (0, 2)$. Specifically, the parameter φ, that weights the gradient and the memory, is set to

$$\varphi = \frac{2}{1+\sqrt{1-\lambda_2^2}},$$

$1 < \varphi < 2$, to guarantee the SFNRC to be faster than the SNRC. In our simulations we assumed to know λ_2, although it would be possible to estimate it, see, e.g., Sahai et al. (2012). It is proven in Muthukrishnan et al. (1998) that if φ is fixed, independent of P, to be $0 < \varphi < 1$, memory provably does not help, i.e., for each such fixed φ, there are P's such that SFNRC takes longer to converge than SNRC.

Algorithm 4 SFNRC Muthukrishnan et al. (1998)

```
storage allocation, constraints on the parameters and initialization as in
Algorithm 1, plus $\widetilde{\boldsymbol{y}}(0) = \boldsymbol{g}(\boldsymbol{x}(0))$ and $\widetilde{\boldsymbol{z}}(0) = \boldsymbol{h}(\boldsymbol{x}(0))$
(main algorithm)
for $k = 1, 2, \dots$ do
    $\widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k-1) + \frac{1}{2-\varphi}\left(\boldsymbol{g}(\boldsymbol{x}(k-1)) - \boldsymbol{g}(\boldsymbol{x}(k-2))\right)$   (local update)
    $\widetilde{\boldsymbol{z}}(k) = \boldsymbol{z}(k-1) + \frac{1}{2-\varphi}\left(\boldsymbol{h}(\boldsymbol{x}(k-1)) - \boldsymbol{h}(\boldsymbol{x}(k-2))\right)$   (local update)
    $\boldsymbol{y}(k) = \varphi P\,\widetilde{\boldsymbol{y}}(k) + (1-\varphi)\,\widetilde{\boldsymbol{y}}(k-1)$   (consensus step)
    $\boldsymbol{z}(k) = \varphi P\,\widetilde{\boldsymbol{z}}(k) + (1-\varphi)\,\widetilde{\boldsymbol{z}}(k-1)$   (consensus step)
    $\boldsymbol{x}(k) = (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon\frac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)}$   (local update)
```

DSM As proposed in Nedić and Ozdaglar (2009), the Distributed Subgradient Method alternates consensus steps on the current estimated global minimum $x_i(k)$ with subgradient updates of each $x_i(k)$ towards the minimum of the local cost f_i. To guarantee the convergence, it is required to appropriately decrease the amplitude of the local subgradient steps. Algorithm 5 presents a synchronous DSM implementation, where ϱ is a tuning parameter (here manually tuned for best convergence rates) and P is defined in (2.66).

Algorithm 5 DSM Nedić and Ozdaglar (2009)

```
(storage allocation and constraints on parameters)
$\boldsymbol{x}^{(c)}(k), \boldsymbol{x}^{(\ell)}(k) \in \mathbb{R}^N$
$\varrho \in \mathbb{R}_+$
(initialization)
$\boldsymbol{x}^{(\ell)}(0) = \mathbf{0}$
(main algorithm)
for $k = 0, 1, \dots$ do
    $\boldsymbol{x}^{(c)}(k) = P\boldsymbol{x}^{(\ell)}(k)$                                  (consensus step)
    $\boldsymbol{x}^{(\ell)}(k+1) = \boldsymbol{x}^{(c)}(k) - \frac{\varrho}{k}\boldsymbol{f}'\left(\boldsymbol{x}^{(c)}(k)\right)$        (local update)
```

DCM As proposed in Wang and Elia (2010), the Distributed Control Method differentiates from the gradient searching because it forces the states to the global optimum by controlling the sum of the subgradients of individual convex cost functions. This model overcomes two limitations of the current algorithms, namely, the diminishing step size, which is a fundamental limitation of the performance of subgradient algorithms, and sensitivity to additive noise, which is an intrinsic property of consensus algorithms based on convex mixing. This approach views the subgradient as an input/output map and uses small gain theorems to guarantee the convergence property of the system. Again, each agents i locally computes and exchanges information with its neighbors, collected in the set $\mathcal{N}_i$. The state equations for all agents can be written as

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) - \mu(L \otimes I)\boldsymbol{x}(k) - \mu(L \otimes I)\boldsymbol{z}(k) - \mu\nu G(\boldsymbol{x}(k)) \tag{2.71}$$

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \mu(L \otimes I)\boldsymbol{x}(k) \tag{2.72}$$

where $G(\cdot)$ is a concatenation of the subgradients $g_i(\cdot)$, $i = 1, \dots, N$, and $L \in \mathbb{R}^{N \times N}$ is the *Laplacian* of the graph, defined by its elements as:

$$[L]_{ij} = \begin{cases} -1 & \text{if } i \in \mathcal{N}_j \\ d_i & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

According to the definition, all row-sums of L are zero, and therefore L always has a zero eigenvalue $\lambda_1 = 0$. The corresponding eigenvector is $\mathbb{1}$. Moreover, L is supposed to be symmetric (that is the case of the undirected graph), thus its eigenvalues satisfy $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N \leq 2 \max_{i \in \mathcal{V}} d_i$.

DCM is summarized in Algorithm 6, where $\mu, \nu > 0$ are parameters to be designed to ensure the stability property of the system. Note that in Algorithm 6, $\mu\nu$ is used as the constant scaling of the local subgradients. Specifically, μ is chosen in the interval

$$0 < \mu < \frac{2}{2 \max_{i \in \mathcal{V}} d_i + 1}, \tag{2.73}$$

where d_i is the degree of each node, to bound the induced gain of the subgradients. In Wang and Elia (2010) it has proven that choosing μ as in (2.73), the system (2.72) is bounded input bounded output (BIBO) stable. Also here the parameters have been manually tuned for best convergence rates.

Algorithm 6 DCM Wang and Elia (2010)

```
   (storage allocation and constraints on parameters)
1: $\boldsymbol{x}(k), \boldsymbol{g}(k), \boldsymbol{z}(k) \in \mathbb{R}^N$
2: $\mu, \nu \in \mathbb{R}_+$
   (initialization)
3: $\boldsymbol{x}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(0) = \mathbf{0}$
   (main algorithm)
4: for $k = 0, 1, \ldots$ do
5:     for $i = 1, \ldots, N$ do
6:         $z_i(k+1) = z_i(k) + \mu \sum_{j \in \mathcal{N}_i} \big(x_i(k) - x_j(k)\big)$
7:         $x_i(k+1) = x_i(k) + \mu \sum_{j \in \mathcal{N}_i} \big(x_j(k) - x_i(k)\big) + \mu \sum_{j \in \mathcal{N}_i} \big(z_j(k) - z_i(k)\big) - \mu\,\nu\, g_i\big(x_i(k)\big)$
```

ADMM The Alternating Direction Method of Multipliers requires the augmentation of the system through additional constraints that do not change the optimal solution but allow the Lagrangian formalism. Standard ADMM decomposes the original problem into two sub-problems, sequentially solves them and updates the dual variables associated with a coupling constraint at each iteration. The best known rate of convergence for the classic ADMM algorithm is $O(1/k)$. There exist different implementations of ADMM in distributed contexts, see, e.g., (Bertsekas and Tsitsiklis, 1997; Schizas et al., 2008; Boyd et al., 2010, pp. 253-261). Here we consider the following Algorithm 7, based on reformulating our optimization problem consistently with an undirected ring communication graph, i.e., as

$$\begin{aligned} &\min_{x_1,\dots,x_N,z_1,\dots,z_N} \quad \sum_{i=1}^{N} f_i(x_i) \\ &\text{such that } z_i = x_{i-1} = x_i = x_{i+1}, \qquad i = 1,\dots,N \end{aligned}$$

where $x_0 := x_N$ and $x_{N+1} := x_1$. With this reformulation the Lagrangian becomes

$$\begin{aligned} L_i(x_i,k) :=\, & f_i\left(x_i\right) + \\ & + y_i^{(\ell)}(k)\big(x_i - z_{i-1}(k)\big) + y_i^{(c)}(k)\big(x_i - z_i(k)\big) + y_i^{(r)}(k)\big(x_i - z_{i+1}(k)\big) + \\ & + \frac{\delta}{2}\big|x_i - z_{i-1}(k)\big|^2 + \frac{\delta}{2}\big|x_i - z_i(k)\big|^2 + \frac{\delta}{2}\big|x_i - z_{i+1}(k)\big|^2, \end{aligned}$$

with δ a tuning parameter.

Algorithm 7 ADMM (Bertsekas and Tsitsiklis, 1997, pp. 253-261)

```
    (storage allocation and constraints on parameters)
$\boldsymbol{x}(k), \boldsymbol{z}(k), \boldsymbol{y}^{(\ell)}(k), \boldsymbol{y}^{(c)}(k), \boldsymbol{y}^{(r)}(k) \in \mathbb{R}^N$
$\delta \in (0,1)$
    (initialization)
$\boldsymbol{x}(0) = \boldsymbol{0}$
$\boldsymbol{y}^{(\ell)}(0) = \boldsymbol{y}^{(c)}(0) = \boldsymbol{y}^{(r)}(0) = \boldsymbol{0}$
$\boldsymbol{z}(0) = \boldsymbol{0}$
    (main algorithm)
for $k = 0, 1, \dots$ do
    for $i = 1, \dots, N$ do
        $x_i(k+1) = \arg\min_{x_i} L_i(x_i,k)$
        $z_i(k+1) = \frac{1}{3\delta}\Big(y_{i+1}^{(\ell)}(k) + y_i^{(c)}(k) + y_{i-1}^{(r)}(k)\Big) + \frac{1}{3}\Big(x_{i-1}(k+1) + x_i(k+1) + x_{i+1}(k+1)\Big)$
        $y_i^{(\ell)}(k+1) = y_i^{(\ell)}(k) + \delta\Big(x_i(k+1) - z_{i-1}(k+1)\Big)$
        $y_i^{(c)}(k+1) = y_i^{(c)}(k) + \delta\Big(x_i(k+1) - z_i(k+1)\Big)$
        $y_i^{(r)}(k+1) = y_i^{(r)}(k) + \delta\Big(x_i(k+1) - z_{i+1}(k+1)\Big)$
```

Results

Figure 2.10 shows a comparison of the five strategies for the same ring graph (here all the tuning parameters of each algorithm are manually tuned for best convergence rates). We notice that, for this specific simulation, all the algorithms converge to the global optimum. While the DSM is the slowest to converge, DCM is significantly faster than DSM but slower than SNRC. Instead, the SFNRC and ADMM methods converge to the global optimum in a comparable amount of time. A virtue of our SNRC strategy is that it can be easily adapted in an asynchronous scenarios where the topology of graph is time-varying. Differently, even if ADMM can be implemented asynchronously with some effort, it can hardly cope with time-varying topologies since the dual variables y_i strongly depend on the specific constrain imposed between the variables z_i and x_i.

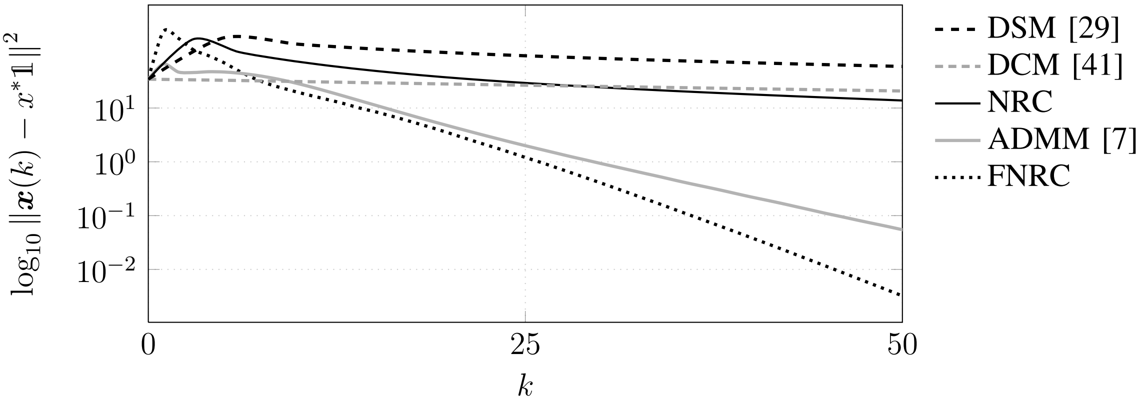


Figure 2.10: Comparison of the square error $\|\boldsymbol{x}(k) - x^* \mathbb{1}\|^2$ in $\log_{10}$-scale for the case $N = 30$ applied to the five algorithms: SNRC (Algorithm 1, $\varepsilon = 0.9$), SFNRC (Algorithm 4, $\varepsilon = 0.9$), DCM (Algorithm 6, $\mu = 0.25$, $\nu = 1.5$), DSM (Algorithm 5, $\varrho = 100$), ADMM (Algorithm 7, $\delta = 0.01$).

3

Asynchronous Newton-Raphson Consensus

3.1 Iterative algorithm

In the following we extend the ideas behind the SNRC to be amenable to more realistic asynchronous implementations.

As noticed before, steps 9 and 10 of Algorithm 1 rely on synchronous communications and on the updates of the various y_i's and z_i's. Thus this implementation requires a high degree of coordination among the agents, being consequently of limited practical applicability.

Here we propose an asynchronous version of the Newton-Raphson Consensus algorithm that is built upon the standard symmetric gossip consensus: at every time a single agent is activated, then this agent selects one of its neighbors and communicates with it. To describe precisely this process we use the following notation:

N1) $k = 1, 2, \ldots$ correspond to the time instants $t_1, t_2, \ldots$ where a generic agent i activates and communicates with one of its neighbors $j \in \mathcal{N}_i$ ($\mathcal{N}_i := \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E}\}$);

N2) $v(k) : \mathbb{N} \mapsto \mathcal{V}$ indicates which agent has been activated at time k;

N3) $e(k) : \mathbb{N} \mapsto \mathcal{E}$ indicates which edge have been activated at time k;

N4) $w_i(k)$ is a flag indicating whether agent i has been activated at time k or not, i.e., $w_i(k) = 1$ if $v(k) = i$, $w_i(k) = 0$ otherwise;

N5) $u_{(i,j)}(k)$ is a flag indicating whether edge (i,j) has been activated at time k or not, i.e., $u_{(i,j)}(k) = 1$ if $e(k) = (i,j)$, $u_{(i,j)}(k) = 0$ otherwise.

Notice that we thus allow the activation of just a single agent and single edge for each time instant k.

As for the agent activation process, we exploit either uniform or persistent agent activation hypotheses:

Assumption 23 (uniform activation) There exist a strictly decreasing function ζ and a positive integer B such that

$$\left|\frac{1}{T}\sum_{k=h}^{T+h-1} w_i(k) - \frac{1}{N}\right| \leq \zeta(T), \quad \forall i \in \mathcal{V}, \forall h, T \in \mathbb{N} \tag{3.1}$$

$$\sum_{k=h}^{B+h-1} u_{(i,j)}(k) \geq 1, \quad \forall (i,j) \in \mathcal{E}, \forall h \in \mathbb{N}. \tag{3.2}$$

Condition (3.1) basically states that, on the long run, all the agents are activated the same number of times. Condition (3.2) instead states that every edge is activated at least once in any window of length B, which can be arbitrarily large but finite.

Assumption 24 (persistent activation) There exists a positive integer B such that

$$\sum_{k=h}^{B+h-1} w_i(k) \geq 1, \quad \forall i \in \mathcal{V}, \forall h \in \mathbb{N} \tag{3.3}$$

and (3.2) simultaneously hold.

(3.3) is weaker than (3.1) in the sense that the former states just that each agent activates at least once in every sufficiently large time window.

Exploiting the previous definitions we introduce the *agent selection matrix* $S(k) \in \mathbb{R}^{N\times N}$, the *edge selection matrix* $E(k) \in \mathbb{R}^{N\times N}$ and the *symmetric gossip consensus matrix* $P(k) \in \mathbb{R}^{N\times N}$ as follows:

$$\begin{aligned}
\boldsymbol{b}_{(i,j)} &:= [0\cdots 0 \ \overset{(i)}{1} \ 0\cdots 0 \ \overset{(j)}{-1} \ 0\cdots 0]^T \in \mathbb{R}^N &(3.4)\\
S(k) &:= \operatorname{diag}\big(w_1(k),\ldots,w_N(k)\big) &(3.5)\\
E(k) &:= \big(\operatorname{diag}(\boldsymbol{b}_{e(k)})\big)^2 &(3.6)\\
P(k) &:= I - \alpha \boldsymbol{b}_{e(k)}\boldsymbol{b}_{e(k)}^T, \quad \alpha \in (0,1). &(3.7)
\end{aligned}$$

Basically, $S(k)$ is zero everywhere except for a one in the diagonal element (i, i) corresponding to the activated agent i. $E(k)$ is zero everywhere except for two ones in the diagonal elements (i, i) and (j, j), corresponding to the agents of the activated edge (i, j). $P(k)$ is the standard symmetric gossip consensus matrix with weight $1-\alpha$ on the diagonals elements (i, i) and (j, j), and α on the (i, j) and (j, i) elements. With this notation it is possible to derive the ANRC, presented in Algorithm 8, as a straightforward modification of the SNRC.

Algorithm 8 Asynchronous Newton-Raphson Consensus (ANRC)

```
       (storage allocation and constraints on parameters)
 1:  $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \widetilde{\boldsymbol{y}}, \widetilde{\boldsymbol{z}} \in \mathbb{R}^N$
 2:  $\varepsilon \in (0,1)$
       (initialization)
 3:  $\boldsymbol{x}(0) = \boldsymbol{x_0}$
 4:  $\boldsymbol{y}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{0}$, $\boldsymbol{z}(0) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \mathbb{1}$
 5:  $\widetilde{\boldsymbol{y}}(0) = \boldsymbol{g}(\boldsymbol{x}(0))$, $\widetilde{\boldsymbol{z}}(0) = \boldsymbol{h}(\boldsymbol{x}(0))$
       (main algorithm)
 6:  for $k = 1, 2, \dots$ do
         (update of the local guesses)
 7:      $\boldsymbol{x}(k) = \boldsymbol{x}(k-1) + \varepsilon S(k) \left( -\boldsymbol{x}(k-1) + \frac{\boldsymbol{y}(k-1)}{\boldsymbol{z}(k-1)} \right)$
         (consensus)
 8:      $\boldsymbol{y}(k) = P(k) \quad \widetilde{\boldsymbol{y}}(k-1)$
 9:      $\boldsymbol{z}(k) = P(k) \quad \widetilde{\boldsymbol{z}}(k-1)$
         (update of the auxiliary variables)
10:      $\widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k) + E(k)\Big(\boldsymbol{g}\big(\boldsymbol{x}(k)\big) - \boldsymbol{g}\big(\boldsymbol{x}(k-1)\big)\Big)$
11:      $\widetilde{\boldsymbol{z}}(k) = \boldsymbol{z}(k) + E(k)\Big(\boldsymbol{h}\big(\boldsymbol{x}(k)\big) - \boldsymbol{h}\big(\boldsymbol{x}(k-1)\big)\Big)$
```

Lines 8-11 in Algorithm 8 compactly represent the fact that all agents do not perform any action except for the selected ones i, j, updating their local variables y_i, y_j as

$$\begin{aligned} y_i(k+1) &= (1-\alpha)\Big(y_i(k) + g_i\big(x_i(k)\big) - g_i\big(x_i(k-1)\big)\Big) + \alpha y_j(k) \\ y_j(k+1) &= (1-\alpha)\Big(y_j(k) + g_j\big(x_j(k)\big) - g_j\big(x_j(k-1)\big)\Big) + \alpha y_i(k) \end{aligned}$$

and z_i, z_j in a similar way. We notice that, among x_i and x_j, just the former is updated: in the proposed version we require the local guess to be updated just for the agent that initiates the communication. Line 7 in Algorithm 8 thus reads as

$$x_i(k+1) = x_i(k) + \varepsilon\left(-x_i(k) + \frac{y_i(k+1)}{z_i(k+1)}\right).$$

The convergence properties of Algorithm 8 are summarized by the two main Propositions, namely 28 and 29. Before proving them we need to prove the ancillary results given by Lemma 27, in its turn relying on Lemmas 25 and 26. Throughout the rest of the treatment we will exploit the following definitions: $B_r^N := \{\boldsymbol{x} \in \mathbb{R}^N \mid \|\boldsymbol{x}\| < r\}$ and $B_{r_0}^N := \{\boldsymbol{x} \in \mathbb{R}^N \mid \|\boldsymbol{x}\| < r_0\}$.

Lemma 25 Consider the nonlinear time-varying discrete-time system

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon\boldsymbol{\phi}\big(k, \boldsymbol{x}(k)\big) \tag{3.8}$$

with $\boldsymbol{x}(k) \in B_r^N \subset \mathbb{R}^N$, $\boldsymbol{\phi} : \mathbb{N} \times B_r^N \mapsto \mathbb{R}^N$ and $\varepsilon \in (0, \varepsilon_0]$. Assume that, for every $k, h \in \mathbb{N}$ and $\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2 \in B_r^N$,

A1)

$$\boldsymbol{\phi}_{\text{ave}}(\boldsymbol{x}) := \lim_{T\to\infty} \frac{1}{T} \sum_{k=h+1}^{h+T} \boldsymbol{\phi}(k, \boldsymbol{x})$$

exists and is independent of h;

A2) $\boldsymbol{\phi}(k, \boldsymbol{0}) = \boldsymbol{\phi}_{\text{ave}}(\boldsymbol{0}) = \boldsymbol{0}$, i.e., the origin $\boldsymbol{x} = \boldsymbol{0}$ is an equilibrium point for

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon\boldsymbol{\phi}_{\text{ave}}\big(\boldsymbol{x}(k)\big) \tag{3.9}$$

and for (3.8);

A3) $\boldsymbol{\phi}(k, \cdot)$ and $\boldsymbol{\phi}_{\text{ave}}(\cdot)$ have continuous and bounded first-order derivatives for all k;

A4) $\boldsymbol{\phi}(k, \cdot)$ and $\boldsymbol{\phi}_{\text{ave}}(\cdot)$ are Lipschitz with respect to $\boldsymbol{x} \in B_r^N$ uniformly in k, i.e.,

$$\begin{aligned}&\|\boldsymbol{\phi}(k, \boldsymbol{x}_1) - \boldsymbol{\phi}(k, \boldsymbol{x}_2)\| \leq \ell\,\|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \quad \forall k \in \mathbb{N},\\ &\|\boldsymbol{\phi}_{\text{ave}}(\boldsymbol{x}_1) - \boldsymbol{\phi}_{\text{ave}}(\boldsymbol{x}_2)\| \leq \ell_{\text{ave}}\,\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|;\end{aligned}$$

A5) letting $\boldsymbol{\delta}(k, \boldsymbol{x}) := \boldsymbol{\phi}(k, \boldsymbol{x}) - \boldsymbol{\phi}_{\text{ave}}(\boldsymbol{x})$, $\boldsymbol{\delta}(k, \boldsymbol{x})$ is piecewise continuous in k, has bounded and continuous first partial derivatives in $\boldsymbol{x}$, and is s.t. $\boldsymbol{\delta}(k, \mathbf{0}) = \mathbf{0}$ for all $k \in \mathbb{N}$. Moreover there exists a strictly decreasing bounded function $\sigma : \mathbb{N} \mapsto \mathbb{R}_+$ satisfying $\lim_{T\to\infty} \sigma(T) = 0$ such that

$$\left\| \frac{1}{T} \sum_{k=h+1}^{h+T} \boldsymbol{\delta}(k, \boldsymbol{x}) \right\| \leq \|\boldsymbol{x}\| \, \sigma(T),$$
$$\left\| \frac{1}{T} \sum_{k=h+1}^{h+T} \frac{\partial \boldsymbol{\delta}(k, \boldsymbol{x})}{\partial \boldsymbol{x}} \right\| \leq \sigma(T),$$

for all $k, h \in \mathbb{N}$, $\boldsymbol{x} \in B_r^N$;

A6) the origin is exponentially stable in B_r^N for (3.9), i.e., there exist positive scalars c, γ such that for every $\boldsymbol{x}(0) \in B_r^N$ the solutions of system (3.9) satisfy

$$\big\|\boldsymbol{x}(k)\big\| \leq c \, \big\|\boldsymbol{x}(0)\big\| \gamma^k.$$

Then there exist $\varepsilon^* \in (0, \varepsilon_0]$ such that for all $\varepsilon \in (0, \varepsilon^*]$ system (3.8) admits a Lyapunov function $V : \mathbb{N} \times B_r^N \mapsto \mathbb{R}$ and positive constants $a_1, \ldots, a_4$ such that, for every $k \in \mathbb{N}$, $\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2 \in B_r^N$,

$$a_1 \|\boldsymbol{x}\|^2 \leq V(k, \boldsymbol{x}) \leq a_2 \|\boldsymbol{x}\|^2 \tag{3.10}$$
$$V\big(k+1, \boldsymbol{x} + \boldsymbol{\phi}(k, \boldsymbol{x})\big) - V(k, \boldsymbol{x}) \leq -\varepsilon a_3 \|\boldsymbol{x}\|^2 \tag{3.11}$$
$$\big|V(k, \boldsymbol{x}_1) - V(k, \boldsymbol{x}_2)\big| \leq a_4 \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \left(\|\boldsymbol{x}_1\| + \|\boldsymbol{x}_2\| \right) \tag{3.12}$$

Proof The lemma is a joint reformulation of the Converse Lyapunov Theorem 2.1.1 and the Basic Averaging Theorem 2.2.2 in Bai et al. (1986). ♢

Lemma 26 Consider the nonlinear time-varying discrete-time system

$$\boldsymbol{y}(k+1) = \boldsymbol{\varphi}\big(k, \boldsymbol{x}, \boldsymbol{y}(k)\big) \tag{3.13}$$

with $\boldsymbol{y}(k) \in B_r^M \subset \mathbb{R}^M$, $\boldsymbol{x} \in \Gamma \subset \mathbb{R}^N$ and $\boldsymbol{\varphi} : \mathbb{N} \times \Gamma \times B_r^M \mapsto \mathbb{R}^M$. Assume that, for every $k \in \mathbb{N}$, $\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2 \in \Gamma$ and $\boldsymbol{y}, \boldsymbol{y}_1, \boldsymbol{y}_2 \in B_r^M$,

A1) $\boldsymbol{\varphi}(k, \boldsymbol{x}, \mathbf{0}) = \mathbf{0}$, i.e., the origin $\mathbf{0}$ is an equilibrium point;

A2) $\boldsymbol{\varphi}(k, \cdot, \cdot)$ has continuous and bounded first-order derivatives in $\boldsymbol{y} \in B_r^M$ for every k and $\boldsymbol{x} \in \Gamma$;

A3) $\varphi(k,\cdot,\cdot)$ is Lipschitz w.r.t. $\boldsymbol{y} \in B_r^M$ and $\boldsymbol{x} \in \Gamma$ uniformly in k, i.e.,

$$\begin{aligned}
&\big\|\varphi(k,\boldsymbol{x},\boldsymbol{y}_1) - \varphi(k,\boldsymbol{x},\boldsymbol{y}_2)\big\| \le \ell_1 \big\|\boldsymbol{y}_1 - \boldsymbol{y}_2\big\|,\\
&\big\|\varphi(k,\boldsymbol{x}_1,\boldsymbol{y}) - \varphi(k,\boldsymbol{x}_2,\boldsymbol{y})\big\| \le \ell_2 \,\|\boldsymbol{y}\|\, \big\|\boldsymbol{x}_1 - \boldsymbol{x}_2\big\|;
\end{aligned}$$

A4) the origin is exponentially stable in an opportune $B_{r'}^M$, i.e., there exist positive scalars c, γ and r' satisfying $\gamma < 1$, and $r' < \dfrac{r}{c}$ such that for every $\boldsymbol{y}(0) \in B_{r'}^M$ the solutions of the system satisfy

$$\big\|\boldsymbol{y}(k)\big\| \le c\,\big\|\boldsymbol{y}(0)\big\|\gamma^k.$$

Then there exists a Lyapunov function $W : \mathbb{N} \times \Gamma \times B_{r'}^M \mapsto \mathbb{R}$ and positive constants a_1, a_2, a_3, a_4, a_5 such that, for every $k \in \mathbb{N}$, $\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2 \in \Gamma$ and $\boldsymbol{y}, \boldsymbol{y}_1, \boldsymbol{y}_2 \in B_{r'}^M$,

$$a_1\|\boldsymbol{y}\|^2 \le W(k,\boldsymbol{x},\boldsymbol{y}) \le a_2\|\boldsymbol{y}\|^2 \tag{3.14}$$

$$W\big(k+1,\boldsymbol{x},g(k,\boldsymbol{x},\boldsymbol{y})\big) - W(k,\boldsymbol{x},\boldsymbol{y}) \le -a_3\|\boldsymbol{y}\|^2 \tag{3.15}$$

$$\big|W(k,\boldsymbol{x},\boldsymbol{y}_1) - W(k,\boldsymbol{x},\boldsymbol{y}_2)\big| \le a_4\big\|\boldsymbol{y}_1 - \boldsymbol{y}_2\big\|\Big(\big\|\boldsymbol{y}_1\big\| + \big\|\boldsymbol{y}_2\big\|\Big) \tag{3.16}$$

$$\big|W(k,\boldsymbol{x}_1,\boldsymbol{y}) - W(k,\boldsymbol{x}_2,\boldsymbol{y})\big| \le a_5\big\|\boldsymbol{y}\big\|^2\,\big\|\boldsymbol{x}_1 - \boldsymbol{x}_2\big\|. \tag{3.17}$$

Proof The claim follows closely the ones of converse Lyapunov theorems for continuous time perturbed systems, see (Khalil, 2001, Theorem 4.14, page 162), and is similar to the claims in (Khalil, 2001, Exercise 4.68, page 194) and in (Jiang and Wang, 2002, Theorem 2). However, since some inequalities are different, we report the detailed proof for sake of completeness.

The proof is divided in the following steps, marked by bullets:

a) introduce a candidate Lyapunov function,

b) verify the previous inequalities.

• *introduction of a candidate Lyapunov function:* let $\boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y})$ denote the solution of the system at time τ, obtained starting at time k with fixed parameter $\boldsymbol{x}$ and initial condition $\boldsymbol{y}$. That is, $\boldsymbol{\psi}(k:k,\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{y}$. Then the candidate Lyapunov function is given by

$$W(k,\boldsymbol{x},\boldsymbol{y}) := \sum_{\tau=k}^{k+T} \big\|\boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y})\big\|^2 \tag{3.18}$$

where T is a nonnegative integer *to be chosen.*

• *proof of* (3.14)*:* In the first inequality, for every nonnegative integer T it follows immediately

$$W(k,\boldsymbol{x},\boldsymbol{y}) \geq \left\|\boldsymbol{\psi}(k:k,\boldsymbol{x},\boldsymbol{y})\right\|^2 = \left\|\boldsymbol{y}\right\|^2 = a_1\left\|\boldsymbol{y}\right\|^2.$$

Regards the second inequality, Assumption A4 implies

$$W(k,\boldsymbol{x},\boldsymbol{y}) \;\leq\; \sum_{\tau=k}^{k+T} c^2\gamma^{2(\tau-k)}\left\|\boldsymbol{y}\right\|^2 = c^2\frac{1-\gamma^{2(T+1)}}{1-\gamma^2}\left\|\boldsymbol{y}\right\|^2 = a_2\left\|\boldsymbol{y}\right\|^2.$$

• *proof of* (3.15)*:* assume $T \geq -\dfrac{\log(c)}{\log(\gamma)}$, and consider that

$$\boldsymbol{\psi}\big(k+1:\tau,\boldsymbol{x},\boldsymbol{\varphi}(k,\boldsymbol{x},\boldsymbol{y})\big) = \boldsymbol{\psi}\big(k:\tau,\boldsymbol{x},\boldsymbol{y}\big).$$

Then

$$\begin{aligned}
&W\big(k+1,\boldsymbol{x},\boldsymbol{\varphi}(k,\boldsymbol{x},\boldsymbol{y})\big) - W(k,\boldsymbol{x},\boldsymbol{y}) = \\
&= \sum_{\tau=k+1}^{k+T+1}\left\|\boldsymbol{\psi}\big(k+1:\tau,\boldsymbol{x},\boldsymbol{\varphi}(k,\boldsymbol{x},\boldsymbol{y})\big)\right\|^2 - \sum_{\tau=k}^{k+T}\left\|\boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y})\right\|^2 \\
&= \sum_{\tau=k+1}^{k+T+1}\left\|\boldsymbol{\psi}\big(k:\tau,\boldsymbol{x},\boldsymbol{y}\big)\right\|^2 - \sum_{\tau=k}^{k+T}\left\|\boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y})\right\|^2 \\
&= \left\|\boldsymbol{\psi}\big(k:k+T+1,\boldsymbol{x},\boldsymbol{y}\big)\right\|^2 - \left\|\boldsymbol{\psi}\big(k:k,\boldsymbol{x},\boldsymbol{y}\big)\right\|^2 \\
&\leq c^2\gamma^{2(T+1)}\left\|\boldsymbol{y}\right\|^2 - \left\|\boldsymbol{y}\right\|^2 \leq -\frac{1}{2}\left\|\boldsymbol{y}\right\|^2 = -a_3\left\|\boldsymbol{y}\right\|^2.
\end{aligned}$$

• *proof of* (3.16)*:* consider that, thanks to assumption A4,

$$\boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y}) \leq c\gamma^{\tau-k}\left\|\boldsymbol{y}\right\|. \tag{3.19}$$

Let moreover $\boldsymbol{\psi}_1^\tau := \boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y}_1)$, $\boldsymbol{\psi}_2^\tau := \boldsymbol{\psi}(k:\tau,\boldsymbol{x},\boldsymbol{y}_2)$ be shorthands to compact the notation. Notice that $\boldsymbol{\psi}_1^\tau = \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x},\boldsymbol{\psi}_1^{\tau-1}\right)$. Thus, by assumption A3,

$$\begin{aligned}
\left\|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\right\| &= \left\|\boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x},\boldsymbol{\psi}_1^{\tau-1}\right) - \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x},\boldsymbol{\psi}_2^{\tau-1}\right)\right\| \\
&\leq \ell_1\left\|\boldsymbol{\psi}_1^{\tau-1} - \boldsymbol{\psi}_2^{\tau-1}\right\|.
\end{aligned}$$

Consider then the reverse triangle inequality

$$\Big|\left\|\boldsymbol{a}\right\| - \left\|\boldsymbol{b}\right\|\Big| \leq \left\|\boldsymbol{a}-\boldsymbol{b}\right\| \quad \forall \boldsymbol{a},\boldsymbol{b}.$$

It thus follows, applying also assumption A4,

$$\begin{aligned}\Big|\|\boldsymbol{\psi}_1^\tau\|^2 - \|\boldsymbol{\psi}_2^\tau\|^2\Big| &= \Big|\|\boldsymbol{\psi}_1^\tau\| + \|\boldsymbol{\psi}_2^\tau\|\Big| \cdot \Big|\|\boldsymbol{\psi}_1^\tau\| - \|\boldsymbol{\psi}_2^\tau\|\Big| \\ &\leq c\gamma^{\tau-k}\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\| \\ &\leq c\gamma^{\tau-k}\ell_1\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{\psi}_1^{\tau-1} - \boldsymbol{\psi}_2^{\tau-1}\| \\ &\leq c\gamma^{\tau-k}\ell_1^2\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{\psi}_1^{\tau-2} - \boldsymbol{\psi}_2^{\tau-2}\| \\ &\ \vdots \\ &\leq c\gamma^{\tau-k}\ell_1^{\tau-k}\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{y}_1 - \boldsymbol{y}_2\|.\end{aligned}$$

Consider then that, applying definition (3.18),

$$\big|W(k,\boldsymbol{x},\boldsymbol{y}_1) - W(k,\boldsymbol{x},\boldsymbol{y}_2)\big| = \left|\sum_{\tau=k}^{k+T}\Big(\|\boldsymbol{\psi}_1^\tau\|^2 - \|\boldsymbol{\psi}_2^\tau\|^2\Big)\right|.$$

Combining with the previous result, this eventually implies

$$\begin{aligned}\big|W(k,\boldsymbol{x},\boldsymbol{y}_1) - W(k,\boldsymbol{x},\boldsymbol{y}_2)\big| &\leq \sum_{\tau=k}^{k+T}\Big|\|\boldsymbol{\psi}_1^\tau\|^2 - \|\boldsymbol{\psi}_2^\tau\|^2\Big| \\ &\leq \left(c\sum_{\tau=k}^{k+T}(\gamma\ell_1)^{(\tau-k)}\right)\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{y}_1 - \boldsymbol{y}_2\| \\ &= a_4\Big(\|\boldsymbol{y}_1\| + \|\boldsymbol{y}_2\|\Big) \cdot \|\boldsymbol{y}_1 - \boldsymbol{y}_2\|\end{aligned}$$

- *proof of* (3.17)*:* let now $\boldsymbol{\psi}_1^\tau$, $\boldsymbol{\psi}_1^\tau$ be used as $\boldsymbol{\psi}_1^\tau := \boldsymbol{\psi}(k:\tau,\boldsymbol{x}_1,\boldsymbol{y})$, $\boldsymbol{\psi}_2^\tau := \boldsymbol{\psi}(k:\tau,\boldsymbol{x}_2,\boldsymbol{y})$, so that $\boldsymbol{\psi}_1^\tau = \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_1,\boldsymbol{\psi}_1^{\tau-1}\right)$, that implies

$$\begin{aligned}\|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\| &= \|\boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_1,\boldsymbol{\psi}_1^{\tau-1}\right) - \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_2,\boldsymbol{\psi}_2^{\tau-1}\right)\| \\ &= \|\boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_1,\boldsymbol{\psi}_1^{\tau-1}\right) - \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_2,\boldsymbol{\psi}_1^{\tau-1}\right) + \\ &\quad +\boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_2,\boldsymbol{\psi}_1^{\tau-1}\right) - \boldsymbol{\varphi}\left(\tau-1,\boldsymbol{x}_2,\boldsymbol{\psi}_2^{\tau-1}\right)\|.\end{aligned}$$

Thus by the (direct) triangle inequality and assumption A3,

$$\begin{aligned}\|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\| &\leq \ell_2\|\boldsymbol{\psi}_1^{\tau-1}\|\|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \\ &\quad +\ell_1\|\boldsymbol{\psi}_1^{\tau-1} - \boldsymbol{\psi}_2^{\tau-1}\|.\end{aligned}$$

Exploiting then (3.19) it follows

$$\begin{aligned}\|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\| &\leq c\ell_2\gamma^{\tau-k}\|\boldsymbol{y}\|\|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \\ &\quad +\ell_1\|\boldsymbol{\psi}_1^{\tau-1} - \boldsymbol{\psi}_2^{\tau-1}\|.\end{aligned}$$

Applying this inequality recursively it thus follows

$$\|\boldsymbol{\psi}_1^\tau - \boldsymbol{\psi}_2^\tau\| \leq \bar{\ell}\|\boldsymbol{y}\|\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$$

with

$$\overline{\ell} = c\ell_2\gamma^{\tau-k} + c\ell_1\ell_2\gamma^{\tau-1-k} + \ldots + c\ell_1^{\tau-k}\ell_2.$$

Consider then that definition (3.18) implies

$$\left|W(k,\boldsymbol{x}_1,\boldsymbol{y}) - W(k,\boldsymbol{x}_2,\boldsymbol{y})\right| = \left|\sum_{\tau=k}^{k+T}\left(\left\|\boldsymbol{\psi}_1^\tau\right\|^2 - \left\|\boldsymbol{\psi}_2^\tau\right\|^2\right)\right|.$$

Focusing for simplicity on the single term $\left\|\boldsymbol{\psi}_1^\tau\right\|^2 - \left\|\boldsymbol{\psi}_2^\tau\right\|^2$, assumption A3, the reverse triangle inequality and the previous results thus imply

$$\begin{aligned}\left|\left\|\boldsymbol{\psi}_1^\tau\right\|^2 - \left\|\boldsymbol{\psi}_2^\tau\right\|^2\right| &= \left|\left\|\boldsymbol{\psi}_1^\tau\right\| + \left\|\boldsymbol{\psi}_2^\tau\right\|\right| \cdot \left|\left\|\boldsymbol{\psi}_1^\tau\right\| - \left\|\boldsymbol{\psi}_2^\tau\right\|\right| \\ &\leq \left(2c\gamma^{\tau-k}\left\|\boldsymbol{y}\right\|\right)\left(\overline{\ell}\left\|\boldsymbol{y}\right\|\left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|\right) = \overline{\ell}'\left\|\boldsymbol{y}\right\|^2\left\|\boldsymbol{x}_1 - \boldsymbol{x}_2\right\|. \quad \diamondsuit\end{aligned}$$

Lemma 27 Consider the nonlinear time-varying perturbed system

$$\begin{cases}\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon\boldsymbol{\phi}\big(k,\boldsymbol{x}(k),\boldsymbol{z}(k)\big) & (3.20a)\\ \boldsymbol{z}(k+1) = \boldsymbol{\varphi}\big(k,\boldsymbol{x}(k),\boldsymbol{z}(k)\big) & (3.20b)\end{cases}$$

with

- $\boldsymbol{x}(k) \in B_{r'}^N \subset \mathbb{R}^N$;
- $\boldsymbol{z}(k) \in B_{r''}^M \subset \mathbb{R}^M$;
- $\boldsymbol{\phi} : \mathbb{N} \times B_{r'}^N \times B_{r''}^M \mapsto \mathbb{R}^N$;
- $\boldsymbol{\varphi} : \mathbb{N} \times B_{r'}^N \times B_{r''}^M \mapsto \mathbb{R}^M$;
- $\varepsilon \in (0, \varepsilon_0]$.

Let then $\boldsymbol{0}$ indicate both the origins of $\mathbb{R}^N$ and $\mathbb{R}^M$. Assume that, for every $k \in \mathbb{N}$, $\boldsymbol{x} \in B_{r'}^N$ and $\boldsymbol{z} \in B_{r''}^M$,

A1) $\boldsymbol{\phi}(k,\boldsymbol{0},\boldsymbol{0}) = \boldsymbol{0}$ and $\boldsymbol{\varphi}(k,\boldsymbol{0},\boldsymbol{0}) = \boldsymbol{0}$, i.e., the origin $\boldsymbol{0}$ is an equilibrium for (3.20a) and for (3.20b);

A2) the equation $\boldsymbol{\varphi}(k,\boldsymbol{x},\boldsymbol{z}) = \boldsymbol{z}$ has an isolated root $\boldsymbol{z} = \boldsymbol{\vartheta}(\boldsymbol{x})$ with $\boldsymbol{\vartheta} : B_{r'}^N \mapsto B_{r''}^M$ independent of k and satisfying $\boldsymbol{\vartheta}(\boldsymbol{0}) = \boldsymbol{0}$, and such that the change of variables

$$\boldsymbol{y}(k) = \boldsymbol{z}(k) - \boldsymbol{\vartheta}\big(\boldsymbol{x}(k)\big) \tag{3.21}$$

is always well defined;

A3) the functions $\boldsymbol{\phi}, \boldsymbol{\varphi}$ and $\boldsymbol{\vartheta}$ have continuous and bounded first-order derivatives;

A4) the functions $\boldsymbol{\phi}, \boldsymbol{\varphi}$ and $\boldsymbol{\vartheta}$ are Lipschitz with respect to $\boldsymbol{x} \in B_{r'}^N$ and $\boldsymbol{z} \in B_{r''}^M$ uniformly in k;

A5) the reduced system

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon \boldsymbol{\phi}\Big(k, \boldsymbol{x}(k), \boldsymbol{\vartheta}\big(\boldsymbol{x}(k)\big)\Big) \tag{3.22}$$

satisfies the thesis of Lemma 25 – where, with a little abuse of notation, $\boldsymbol{\phi}(k, \boldsymbol{x}) = \boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{\vartheta}(\boldsymbol{x})\big)$;

A6) the boundary-layer system

$$\boldsymbol{y}(k+1) = \boldsymbol{\varphi}\big(k, \boldsymbol{x}, \boldsymbol{y}(k) + \boldsymbol{\vartheta}(\boldsymbol{x})\big) - \boldsymbol{\vartheta}(\boldsymbol{x}) \tag{3.23}$$

satisfies the thesis of Lemma 26 – where, with a little abuse of notation, $\boldsymbol{\varphi}(k, \boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\varphi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big) - \boldsymbol{\vartheta}(\boldsymbol{x})$.

Then, there exists $\varepsilon^* \in (0, \varepsilon_0]$ such that for all $\varepsilon \in (0, \varepsilon^*]$ the origin $(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{0}, \boldsymbol{0})$ is an exponentially stable equilibrium for the whole system (3.20a)-(3.20b).

Proof The proof is divided in the following steps, marked by bullets:

a) introduce some notation and basic bounds, to be used as building blocks.

b) introduce two Lyapunov functions and some of their properties: one for the reduced system (3.22) and one for the boundary layer (3.23).

d) build a Lyapunov function for the complete system (3.20a)-(3.20b).

• *notation and basic bounds:* consider the change of variables (3.21) and the shortcuts $\boldsymbol{x} = \boldsymbol{x}(k)$ and $\boldsymbol{x}^+ = \boldsymbol{x}(k+1)$, so that the complete system (3.20a)-(3.20b) becomes

$$\begin{cases} \boldsymbol{x}^+ &= \boldsymbol{x} + \varepsilon \boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big) \\ \boldsymbol{y}^+ &= \boldsymbol{\varphi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big) - \boldsymbol{\vartheta}\Big(\boldsymbol{x} + \varepsilon \boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big)\Big). \end{cases}$$

For notational brevity we moreover let

$$\boldsymbol{\chi}^+ := \boldsymbol{x} + \varepsilon \boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{\vartheta}(\boldsymbol{x})\big) \tag{3.24}$$

so that $\boldsymbol{\chi}^+$ is equal to $\boldsymbol{x}^+$ as soon as $\boldsymbol{y} = \boldsymbol{0}$. The following basic bounds follow immediately from the Lipschitz and vanishing properties of the various functions:

$$
\begin{aligned}
\big\|\boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{\vartheta}(\boldsymbol{x})\big)\big\| &\leq \ell_1 \|\boldsymbol{x}\| && (3.25)\\
\big\|\boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big)\big\| &\leq \ell_2 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) && (3.26)\\
\big\|\boldsymbol{x}^+ - \boldsymbol{x}\big\| &\leq \varepsilon \ell_2 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) && (3.27)\\
\big\|\boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big) - \boldsymbol{\phi}\big(k, \boldsymbol{x}, \boldsymbol{\vartheta}(\boldsymbol{x})\big)\big\| &\leq \ell_3 \|\boldsymbol{y}\| && (3.28)\\
\big\|\boldsymbol{x}^+ - \boldsymbol{\chi}^+\big\| &\leq \varepsilon \ell_3 \|\boldsymbol{y}\| && (3.29)\\
\big\|\boldsymbol{\varphi}\big(k, \boldsymbol{x}, \boldsymbol{y} + \boldsymbol{\vartheta}(\boldsymbol{x})\big) - \boldsymbol{\vartheta}(\boldsymbol{x})\big\| &\leq \ell_4 \|\boldsymbol{y}\| && (3.30)\\
\big\|\boldsymbol{\vartheta}(\boldsymbol{x}^+) - \boldsymbol{\vartheta}(\boldsymbol{x})\big\| &\leq \varepsilon \ell_5 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) && (3.31)\\
\big\|\boldsymbol{x}^+\big\| &\leq \ell_6 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) && (3.32)\\
\big\|\boldsymbol{\chi}^+\big\| &\leq \ell_7 \big(\|\boldsymbol{x}\|\big) && (3.33)\\
\big\|\boldsymbol{\vartheta}(\boldsymbol{x})\big\| &\leq \ell_8 \|\boldsymbol{x}\| && (3.34)
\end{aligned}
$$

for suitable positive constants $\ell_1, \ldots, \ell_8$.

• *Lyapunov function for the reduced system* (3.22)*:* assumption A5 ensures the existence of a Lyapunov function $V(k, \boldsymbol{x})$ and a $r'_0 \leq r'$ such that, $\forall \boldsymbol{x} \in B^N_{r'_0}$,

$$
\begin{aligned}
c_1 \|\boldsymbol{x}\|^2 \leq V(k, \boldsymbol{x}) &\leq c_2 \|\boldsymbol{x}\|^2 && (3.35)\\
V\big(k+1, \boldsymbol{\chi}^+\big) - V(k, \boldsymbol{x}) &\leq -\varepsilon c_3 \|\boldsymbol{x}\|^2 && (3.36)\\
\big|V(k, \boldsymbol{x}_1) - V(k, \boldsymbol{x}_2)\big| &\leq c_4 \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \big(\|\boldsymbol{x}_1\| + \|\boldsymbol{x}_2\|\big) && (3.37)
\end{aligned}
$$

for some positive constants $c_1, \ldots, c_4$. As for the temporal evolution of $V(k, \boldsymbol{x})$, exploiting definition (3.24) and properties (3.36) and (3.37) it follows that

$$
\begin{aligned}
\Delta V(k, \boldsymbol{x}) &:= V\big(k+1, \boldsymbol{x}^+\big) - V(k, \boldsymbol{x}) =\\
&= V\big(k+1, \boldsymbol{x}^+\big) - V\big(k+1, \boldsymbol{\chi}^+\big) + V\big(k+1, \boldsymbol{\chi}^+\big) - V(k, \boldsymbol{x})\\
&\leq c_4 \big\|\boldsymbol{x}^+ - \boldsymbol{\chi}^+\big\| \Big(\big\|\boldsymbol{x}^+\big\| + \big\|\boldsymbol{\chi}^+\big\|\Big) - \varepsilon c_3 \|\boldsymbol{x}\|^2
\end{aligned}
$$

and thus, using properties (3.29), (3.32) and (3.33),

$$
\Delta V(k, \boldsymbol{x}) \leq \varepsilon c_4 \ell_3 \|\boldsymbol{y}\| \Big(\ell_6 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) + \ell_7 \|\boldsymbol{y}\|\Big) - \varepsilon c_3 \|\boldsymbol{x}\|^2.
$$

Letting then $\ell_9 = c_4 \ell_3 \ell_6$, $\ell_{10} = c_4 \ell_3 (\ell_6 + \ell_7)$ we obtain the quadratic bound

$$
\Delta V(k, \boldsymbol{x}) \leq -\varepsilon c_3 \|\boldsymbol{x}\|^2 + \varepsilon 2 \ell_9 \|\boldsymbol{x}\| \|\boldsymbol{y}\| + \varepsilon \ell_{10} \|\boldsymbol{y}\|^2. \tag{3.38}
$$

• *Lyapunov function for the boundary layer* (3.23)*:* let $\boldsymbol{x} \in B^N_{r'}$ be a constant. Then assumption A6 ensures the existence of a Lyapunov function

$W(k,\boldsymbol{x},\boldsymbol{y})$ and a $r_0'' < r''$ such that, for all $\boldsymbol{y} \in B^M_{r_0''}$,

$$b_1\|\boldsymbol{y}\|^2 \leq W(k,\boldsymbol{x},\boldsymbol{y}) \leq b_2\|\boldsymbol{y}\|^2 \tag{3.39}$$

$$W\Big(k+1,\boldsymbol{x},\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\Big) - W(k,\boldsymbol{x},\boldsymbol{y}) \leq -b_3\|\boldsymbol{y}\|^2 \tag{3.40}$$

$$\big|W(k,\boldsymbol{x},\boldsymbol{y}_1) - W(k,\boldsymbol{x},\boldsymbol{y}_2)\big| \leq b_4\|\boldsymbol{y}_1-\boldsymbol{y}_2\|\big(\|\boldsymbol{y}_1\|+\|\boldsymbol{y}_2\|\big) \tag{3.41}$$

$$\big|W(k,\boldsymbol{x}_1,\boldsymbol{y}) - W(k,\boldsymbol{x}_2,\boldsymbol{y})\big| \leq b_5\|\boldsymbol{y}\|^2\|\boldsymbol{x}_1-\boldsymbol{x}_2\| \tag{3.42}$$

As for the temporal evolution of $W(k,\boldsymbol{x},\boldsymbol{y})$, consider that

$$\begin{aligned}\Delta W(k,\boldsymbol{x},\boldsymbol{y}) :=\ & W(k+1,\boldsymbol{x}^+,\boldsymbol{y}^+) - W(k,\boldsymbol{x},\boldsymbol{y})\\
=\ & W\Big(k+1,\boldsymbol{x}^+,\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x}^+)\Big) - W(k,\boldsymbol{x},\boldsymbol{y})\\
=\ & W\Big(k+1,\boldsymbol{x}^+,\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x}^+)\Big)\\
& - W\Big(k+1,\boldsymbol{x}^+,\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\Big)\\
& + W\Big(k+1,\boldsymbol{x}^+,\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\Big)\\
& - W\Big(k+1,\boldsymbol{x},\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\Big)\\
& + W\Big(k+1,\boldsymbol{x},\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\Big) - W(k,\boldsymbol{x},\boldsymbol{y})\ .\end{aligned} \tag{3.43}$$

We then exploit: (3.41) to bound the first two rows of the last r.h.s.; (3.42) to bound the third and fourth rows; (3.40) to bound for the last two rows. This implies that $\Delta W(k,\boldsymbol{x},\boldsymbol{y}) \leq \beta_1+\beta_2+\beta_3$, where the last three symbols are the following shorthands:

$$\begin{aligned}\beta_1 :=\ & b_4\big\|\boldsymbol{\vartheta}(\boldsymbol{x}^+)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|\Big(\big\|\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x}^+)\big\|\\
& \qquad + \big\|\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|\Big)\\
\beta_2 :=\ & b_5\big\|\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|^2\big\|\boldsymbol{x}^+-\boldsymbol{x}\big\|\\
\beta_3 :=\ & -b_3\|\boldsymbol{y}\|^2\ .\end{aligned}$$

To bound β_1 we apply the triangular inequality so that

$$\begin{aligned}\beta_1 \leq\ & b_4\big\|\boldsymbol{\vartheta}(\boldsymbol{x}^+)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|\Big(\ \big\|\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|\\
& + \big\|\boldsymbol{\vartheta}(\boldsymbol{x}^+)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\| + \big\|\boldsymbol{\varphi}\big(k,\boldsymbol{x},\boldsymbol{y}+\boldsymbol{\vartheta}(\boldsymbol{x})\big)-\boldsymbol{\vartheta}(\boldsymbol{x})\big\|\ \Big)\end{aligned}$$

and thus, exploiting (3.30) and (3.31), that

$$\begin{aligned}\beta_1 \leq\ & \varepsilon b_4\ell_5\big(\|\boldsymbol{x}\|+\|\boldsymbol{y}\|\big)\Big(2\ell_4\|\boldsymbol{y}\|+\varepsilon\ell_5\big(\|\boldsymbol{x}\|+\|\boldsymbol{y}\|\big)\Big)\\
\leq\ & 2\varepsilon b_4\ell_5\big(\ell_4+\varepsilon\ell_5\big)\|\boldsymbol{x}\|\|\boldsymbol{y}\| + \varepsilon^2 b_4\ell_5^2\|\boldsymbol{x}\|^2 + \varepsilon b_4\ell_5(2\ell_4+\varepsilon\ell_5)\|\boldsymbol{y}\|^2.\end{aligned}$$

Concerning β_2, consider that from $\|\boldsymbol{x}\| \leq r'$, $\|\boldsymbol{z}\| \leq r_0$ and inequality 3.34 it follows $\|\boldsymbol{y}\| \leq \|\boldsymbol{z}\| + \big\|\boldsymbol{\vartheta}(\boldsymbol{x})\big\|$. Moreover, using (3.27) and (3.30) we obtain

$$\begin{aligned}\beta_2 \leq \; & \varepsilon b_5 \ell_2 \ell_5 \|\boldsymbol{y}\|^2 \big(\|\boldsymbol{x}\| + \|\boldsymbol{y}\|\big) \\ \leq \; & \varepsilon b_5 \ell_2 \ell_5 \big(r_0 + \ell_8 r'\big) \|\boldsymbol{y}\|^2 .\end{aligned}$$

Given the previous, we can thus write

$$\Delta W(k, \boldsymbol{x}, \boldsymbol{y}) \leq \big(\varepsilon \ell_{11} - b_3\big)\|\boldsymbol{y}\|^2 + \varepsilon 2 \ell_{12} \|\boldsymbol{x}\| \|\boldsymbol{y}\| + \varepsilon^2 \ell_{13} \|\boldsymbol{x}\|^2 \tag{3.44}$$

for suitable positive constants $\ell_{11}, \ell_{12}, \ell_{13}$.

- *Lyapunov function for the whole system* (3.20a)-(3.20b)*:* let the candidate be

$$U(k, \boldsymbol{x}, \boldsymbol{y}) = V(k, \boldsymbol{x}) + W(k, \boldsymbol{x}, \boldsymbol{y}).$$

We must check whether, for all the plausible trajectories in $B^N_{r'_0} \times B^M_{r''_0}$, the condition $(\boldsymbol{x}, \boldsymbol{y}) \neq (0, 0)$ implies

$$\Delta U(k, \boldsymbol{x}, \boldsymbol{y}) := U\big(k+1, \boldsymbol{x}^+, \boldsymbol{y}^+\big) - U\big(k, \boldsymbol{x}, \boldsymbol{y}\big) < 0.$$

Consider then that inequalities (3.38) and (3.44) form a quadratic form that can be rewritten as

$$\Delta U(k, \boldsymbol{x}, \boldsymbol{y}) \leq - \begin{bmatrix} \|\boldsymbol{x}\| & \|\boldsymbol{y}\| \end{bmatrix} A \begin{bmatrix} \|\boldsymbol{x}\| \\ \|\boldsymbol{y}\| \end{bmatrix} \tag{3.45}$$

where

$$A := \begin{bmatrix} \varepsilon c_3 - \varepsilon^2 \ell_{13} & -\varepsilon(\ell_9 + \ell_{12}) \\ -\varepsilon(\ell_9 + \ell_{12}) & b_3 - \varepsilon(\ell_{10} + \ell_{11}) \end{bmatrix}.$$

Consider now that the leading principal minors of A are, in Landau notation and for $\varepsilon \to 0$,

$$\varepsilon c_3 + O\big(\varepsilon^2\big), \quad \varepsilon c_3 b_3 + O\big(\varepsilon^2\big).$$

Thus there must exist a sufficiently small $\varepsilon^* \in (0, \varepsilon_0]$ such that for every $\varepsilon \in (0, \varepsilon^*]$ A is positive definite, i.e.,

$$\varepsilon \in (0, \varepsilon^*] \;\; \Rightarrow \;\; A \geq \varepsilon \ell_{14} I, \quad \varepsilon \in (0, \varepsilon^*) \tag{3.46}$$

for an opportune positive scalar ℓ_{14}.

We can now prove the exponential convergence to the equilibrium as soon as $\varepsilon \in (0, \varepsilon^*]$. In this case, in fact, from (3.35), (3.45), (3.39) and (3.46) it follows that:

$$\begin{aligned}\Delta U(k, \boldsymbol{x}, \boldsymbol{y}) \;\; & \leq \;\; -\varepsilon \ell_{14} \big(\|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2\big) \\ & \leq \;\; -\varepsilon \ell_{14} \left(\frac{1}{c_2} V(\boldsymbol{x}) + \frac{1}{b_2} W(k, \boldsymbol{x}, \boldsymbol{y}) \right) \\ & \leq \;\; -\varepsilon \gamma U(k, \boldsymbol{x}, \boldsymbol{y})\end{aligned}$$

where $\gamma := \ell_{14} \min \left\{ \frac{1}{b_2}, \frac{1}{c_2} \right\}$. This eventually implies that

$$\begin{bmatrix} \|\boldsymbol{x}(k)\| \\ \|\boldsymbol{y}(k)\| \end{bmatrix} \leq \ell \Big(\sqrt{1 - \varepsilon\gamma} \Big)^k \begin{bmatrix} \|\boldsymbol{x}(0)\| \\ \|\boldsymbol{y}(0)\| \end{bmatrix} ,$$

where ℓ is an appropriate constant, and this concludes the proof. ♢

Now we are ready to prove the two propositions that ensure respectively the global and local stability of Algorithm 8.

Proposition 28 (global stability) Consider Algorithm 8 and let Assumptions 8 and 23 hold true. Then for every $r > 0$ there exists two positive constants $\overline{\varepsilon}_r$, c_r such that if $\varepsilon < \overline{\varepsilon}_r$, then there exists $\gamma_\varepsilon > 0$ such that

$$\left\| \boldsymbol{x}(k) - x^* \mathbb{1} \right\| \leq c_r e^{-\gamma_\varepsilon k} \left\| \boldsymbol{x}_0 - x^* \mathbb{1} \right\|$$

for all $\boldsymbol{x}_0 \in B_r^{x^*} := \left\{ \boldsymbol{x} \mid \left\| \boldsymbol{x} - x^* \mathbb{1} \right\| < r \right\}$.

Proof The proof consists in different steps: rewriting Algorithm 8 in a convenient form, then separately analyze the boundary layer and the reduced systems to check whether the assumptions of Lemma 27 are satisfied. In the following every bullet corresponds to one of the previous steps. For ease of notation we will not shift the equilibria to the origins.

We also notice that, to be able to claim global convergence, the proof must let the open balls containing the initial conditions and the trajectories of the boundary layer and reduced systems have arbitrary radii.

As before we use the shortcuts $\boldsymbol{x} = \boldsymbol{x}(k)$, $\boldsymbol{x}^+ = \boldsymbol{x}(k+1)$, $\boldsymbol{x}^- = \boldsymbol{x}(k-1)$ and similar notations also for the other quantities. Moreover we also let

$$\Pi^{\|} := \frac{\mathbb{1}\mathbb{1}^T}{N} \quad \Pi^{\perp} := I - \frac{\mathbb{1}\mathbb{1}^T}{N}$$

$$\boldsymbol{x}^{\|} := \Pi^{\|} \boldsymbol{x} = \overline{x} \mathbb{1} \quad \overline{x} := \frac{1}{N} \sum_{i=1}^N x_i \quad \boldsymbol{x}^{\perp} := \Pi^{\perp} \boldsymbol{x}$$

$$\overline{g}(\boldsymbol{x}) := \frac{1}{N} \sum_{i=1}^N \Big(f_i''(x_i)\, x_i - f_i'(x_i) \Big) \quad \overline{h}(\boldsymbol{x}) := \frac{1}{N} \sum_{i=1}^N f_i''(x_i)$$

so that

$$\overline{g}(\boldsymbol{x}^{\|}) = \overline{g}(\overline{x}\mathbb{1}) = \overline{f}''(\overline{x})\overline{x} - \overline{f}'(\overline{x}), \tag{3.47}$$

$$\overline{h}(\boldsymbol{x}^{\|}) = \overline{h}(\overline{x}\mathbb{1}) = \overline{f}''(\overline{x}). \tag{3.48}$$

• *rewriting Algorithm 8:* aiming to obtain a system like (3.20a)-(3.20b), we rewrite Algorithm 8 as follows:

$$\begin{cases} \boldsymbol{v}(k+1) = \boldsymbol{g}\big(\boldsymbol{x}(k)\big) & (3.49a) \\ \boldsymbol{w}(k+1) = \boldsymbol{h}\big(\boldsymbol{x}(k)\big) & (3.49b) \\ \boldsymbol{y}(k+1) = P(k)\Big[\boldsymbol{y}(k) + E(k)\Big(\boldsymbol{g}\big(\boldsymbol{x}(k)\big) - \boldsymbol{v}(k)\Big)\Big] & (3.49c) \\ \boldsymbol{z}(k+1) = P(k)\Big[\boldsymbol{z}(k) + E(k)\Big(\boldsymbol{h}\big(\boldsymbol{x}(k)\big) - \boldsymbol{w}(k)\Big)\Big] & (3.49d) \\ \boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon S(k)\left(-\boldsymbol{x}(k) + \dfrac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)}\right) & (3.49e) \end{cases}$$

Considering Lemma 27, (3.49a)–(3.49d) constitute (3.20b), while (3.49e) constitute (3.20a). The corresponding functions $\boldsymbol{\phi}(\cdot)$ and $\boldsymbol{\varphi}(\cdot)$ thus satisfy assumptions A3 and A4.

• *analysis of the boundary layer system:* let $\varepsilon = 0$, so that $\boldsymbol{x}$ (assumed in $B_{r'}^{N}$) is constant and thus (3.49) reduces to

$$\begin{cases} \boldsymbol{v}(k) & = \boldsymbol{g}(\boldsymbol{x}) \\ \boldsymbol{w}(k) & = \boldsymbol{h}(\boldsymbol{x}) \end{cases}, \qquad k \geq 0$$

$$\begin{cases} \boldsymbol{y}(k+1) & = P(k)\boldsymbol{y}(k) \\ \boldsymbol{z}(k+1) & = P(k)\boldsymbol{z}(k) \end{cases}, \qquad \begin{array}{l} \boldsymbol{y}(0) = \boldsymbol{g}(\boldsymbol{x}) \\ \boldsymbol{z}(0) = \boldsymbol{h}(\boldsymbol{x}) \end{array}.$$

It thus follows that the function $\boldsymbol{\vartheta}(\cdot)$ in Lemma 27 is defined by stacking $\overline{g}(\boldsymbol{x})\mathbb{1}$ and $\overline{h}(\boldsymbol{x})\mathbb{1}$. $\boldsymbol{\vartheta}(\boldsymbol{x})$ thus satisfies assumptions A2, A3 and A4.

Consider now that assumption 23 implies assumption 24, and that the latter ensures

$$\lim_{k\to\infty} \boldsymbol{y}(k) = \frac{1}{N}\sum_{i=1}^{N} g_i(x_i)\mathbb{1} = \overline{g}(\boldsymbol{x})\mathbb{1}$$

$$\lim_{k\to\infty} \boldsymbol{z}(k) = \frac{1}{N}\sum_{i=1}^{N} h_i(x_i)\mathbb{1} = \overline{h}(\boldsymbol{x})\mathbb{1}$$

with a convergence that is exponentially fast independently on r'', see, e.g., Olshevsky and Tsitsiklis (2009a) (even though the convergence rate might be not uniform in r''). This convergence is moreover global, and this implies that: *a*) assumption A6 in Lemma 27 is satisfied; *b*) assumption A4 in Lemma 26 is satisfied with $c = 1$ (important because it implies that we can continue considering the entire $B_{r''}^{M}$ and not subsets of it).

• *analysis of the reduced system:* let $\boldsymbol{y}(k) = \overline{g}\big(\boldsymbol{x}(k)\big)\mathbb{1}$ and $\boldsymbol{z}(k) = \overline{g}\big(\boldsymbol{x}(k)\big)\mathbb{1}$, so that (3.49) reduces to

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \varepsilon S(k)\left(-\boldsymbol{x}(k) + \frac{\overline{g}\big(\boldsymbol{x}(k)\big)}{\overline{h}\big(\boldsymbol{x}(k)\big)}\mathbb{1}\right). \tag{3.50}$$

Following Lemma 25 we start studying the time average of (3.50) (namely $\boldsymbol{\phi}_{\text{ave}}(\cdot)$ in Lemma 25).

Considering sufficiently small ε, so that Theorem 14 (i.e. Theorem 2 in Sundarapandian (2002)) holds, the discretized average system inherits the same stability properties of the equivalent continuous time average system

$$\dot{\boldsymbol{x}} = f_{\text{ave}}(\boldsymbol{x}) = S_{\text{ave}} \left(-\boldsymbol{x} + \frac{\overline{g}(\boldsymbol{x})}{\overline{h}(\boldsymbol{x})} \mathbb{1} \right), \tag{3.51}$$

where

$$S_{\text{ave}} := \lim_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T} S(k).$$

Given Assumption 23 it then follows that

$$S_{\text{ave}} = \frac{1}{N} I,$$

so that assumptions A1, A3, A4 and A5 in Lemma 25 are satisfied.

We now verify that assumptions A2 and A6 in Lemma 25 hold, i.e., that (3.50) and (3.51) share the same unique equilibrium, and that the system is exponentially convergent to it.

Consider the decomposition $\boldsymbol{x} = \boldsymbol{x}^{\perp} + \boldsymbol{x}^{\parallel}$, so that (3.51) can be written as

$$\begin{cases} \dot{\boldsymbol{x}}^{\perp} = -\dfrac{1}{N} \boldsymbol{x}^{\perp} & \text{(3.52a)} \\ \dot{\boldsymbol{x}}^{\parallel} = -\dfrac{1}{N} \boldsymbol{x}^{\parallel} + \dfrac{1}{N} \dfrac{\overline{g}\left(\boldsymbol{x}^{\perp} + \boldsymbol{x}^{\parallel}\right)}{\overline{h}\left(\boldsymbol{x}^{\perp} + \boldsymbol{x}^{\parallel}\right)} \mathbb{1}. & \text{(3.52b)} \end{cases}$$

It follows that (3.52a) is such that $\left\| \boldsymbol{x}^{\perp}(k) \right\| \leq e^{-\frac{k}{N}} \left\| \boldsymbol{x}(0) \right\|$, i.e., $\boldsymbol{x}^{\perp}(k)$ converges to zero exponentially fast and independently of $\boldsymbol{x}^{\parallel}$.

To prove the exponential convergence properties of (3.52b) we thus intended it as a *perturbed version* of

$$\dot{\boldsymbol{x}}^{\parallel} = -\frac{1}{N} \boldsymbol{x}^{\parallel} + \frac{1}{N} \frac{\overline{g}\left(\boldsymbol{x}^{\parallel}\right)}{\overline{h}\left(\boldsymbol{x}^{\parallel}\right)} \mathbb{1}. \tag{3.53}$$

From (3.47) and (3.48) it then follows that system (3.53) is equivalent to

$$\begin{aligned} \dot{\overline{x}} &= -\frac{1}{N} \overline{x} + \frac{1}{N} \frac{\overline{g}\left(\overline{x} \mathbb{1}\right)}{\overline{h}\left(\overline{x} \mathbb{1}\right)} \\ &= \frac{1}{N} \left(-\overline{x} + \frac{\overline{f}''(\overline{x}) \overline{x} - \overline{f}'(\overline{x})}{\overline{f}''(\overline{x})} \right) \\ &= -\frac{1}{N} \frac{\overline{f}'(\overline{x})}{\overline{f}''(\overline{x})} \end{aligned} \tag{3.54}$$

that, under Assumption 8, satisfies the hypotheses of Theorem 10 and it is thus globally exponentially converging to its unique equilibrium point $x^* = \arg\max_x \overline{f}(x)$.

To prove that (3.54) exponentially implies also (3.52b) to be exponentially convergent we use standard perturbation analysis tools. Rewrite (3.52b) as

$$\dot{\boldsymbol{x}}^{\parallel} = -\frac{1}{N}\boldsymbol{x}^{\parallel} + \frac{1}{N}\frac{\overline{g}\big(\boldsymbol{x}^{\parallel}\big)}{\overline{h}\big(\boldsymbol{x}^{\parallel}\big)}\mathbb{1} + \frac{1}{N}\Upsilon\big(\boldsymbol{x}^{\perp}, \boldsymbol{x}^{\parallel}\big)\mathbb{1}$$

with

$$\Upsilon\big(\boldsymbol{x}^{\perp}, \boldsymbol{x}^{\parallel}\big) := \frac{\overline{g}\big(\boldsymbol{x}^{\perp} + \boldsymbol{x}^{\parallel}\big)}{\overline{h}\big(\boldsymbol{x}^{\perp} + \boldsymbol{x}^{\parallel}\big)} - \frac{\overline{g}\big(\boldsymbol{x}^{\parallel}\big)}{\overline{h}\big(\boldsymbol{x}^{\parallel}\big)}.$$

Consider then that Assumption 8 implies Υ to be Lipschitz over $B_{r'}^N$, and that system (3.54) satisfies the hypotheses of Theorem 11. Thus it is possible to apply Lemma 12 and state that (3.52b) converges exponentially to $x^*\mathbb{1}$ for each $\boldsymbol{x}(0) \in B_{r'}^N$. Hence assumption A5 of Lemma 27 is satisfied, and this proves the claim of proposition. $\diamondsuit$

Proposition 29 (local stability) Consider Algorithm 8 and let Assumptions 8 and 24 hold true. Then there exist $\overline{r} > 0$ such that for every $r \in (0, \overline{r})$ there exists two positive constants $\overline{\varepsilon}_r$, c_r such that if $\varepsilon < \overline{\varepsilon}_r$, then there exists $\gamma_\varepsilon > 0$ such that

$$\big\|\boldsymbol{x}(k) - x^*\mathbb{1}\big\| \leq c_r e^{-\gamma_\varepsilon k}\big\|\boldsymbol{x}_0 - x^*\mathbb{1}\big\|$$

for all $\boldsymbol{x}_0 \in B_r^{x^*} := \big\{\boldsymbol{x} \mid \big\|\boldsymbol{x} - x^*\mathbb{1}\big\| < r\big\}$.

Proof The proof is analogous to the one of Proposition 28, except for the analysis of the reduced system. Consider then (3.50), and the fact that $\boldsymbol{x} = x^*\mathbb{1}$ is still an equilibrium point.

Notice then that Assumption (24) does not guarantee the existence of an average system as in the proof of Proposition 28.

To prove the claim we thus linearize the dynamics of (3.50) around the equilibrium $\boldsymbol{x} = x^*\mathbb{1}$. Consider that, given the definitions of $\overline{g}(\cdot)$ and $\overline{h}(\cdot)$,

$$\left[\nabla\left(\frac{\overline{g}(\boldsymbol{x})}{\overline{h}(\boldsymbol{x})}\right)\right]_j = \frac{f_j'''(x_j)\displaystyle\sum_{i=1}^{N}\Big(f_i''(x_i)(x_j - x_i) + f_i'(x_i)\Big)}{\left(\displaystyle\sum_{i=1}^{N} f_i''(x_i)\right)^2}.$$

This implies that

$$\nabla \left(\frac{\overline{g}(\boldsymbol{x})}{\overline{h}(\boldsymbol{x})} \right) \Bigg|_{\boldsymbol{x}=x^*\mathbb{1}} = \mathbf{0},$$

and thus that the linearized version of (3.50) is

$$\boldsymbol{x}^{\mathrm{lin}}(k+1) = \boldsymbol{x}^{\mathrm{lin}}(k) + \varepsilon S(k) \Big(-\boldsymbol{x}^{\mathrm{lin}}(k) + x^* \mathbb{1} \Big).$$

Since $S(k)$ is diagonal, c.f. (3.5), it follows that the dynamics of each local variable x_i^{lin} is decoupled from the dynamics of the others. That is, each x_i^{lin} evolves as

$$x_i^{\mathrm{lin}}(k+1) = x_i^{\mathrm{lin}}(k) - \varepsilon w_i(k) \Big(x_i^{\mathrm{lin}}(k) - x^* \Big),$$

$i = 1, \ldots, N$, that eventually implies

$$x_i^{\mathrm{lin}}(k) = x^* + \Big(x_i^{\mathrm{lin}}(0) - x^* \Big) \cdot e^{-\sum\limits_{h=0}^{k-1} w_i(h)}.$$

Consider then that Assumption 24 ensures the existence of an opportune $B \in \mathbb{N}$ such that $\sum_{k=h}^{B+h-1} w_i(k) \geq 1$ for every $h \in \mathbb{N}$. This implies that

$$\Big| x_i^{\mathrm{lin}}(k) - x^* \Big| \leq \Big| x_i^{\mathrm{lin}}(0) - x^* \Big| \cdot e^{-\left\lfloor \frac{k}{B} \right\rfloor}$$

where $\lfloor \cdot \rfloor$ is the floor operator. This shows that $\mathbf{0}$ is exponentially stable for the linearized system, and thus that it is locally exponentially stable for (3.50).

This proves our claim, because this implies the thesis of Lemma 25 to be satisfied, i.e., all the hypotheses of Lemma 27 to be satisfied. ♢

Under the hypothesis of both the previous propositions, for sufficiently small ε and initial points $x_i(0)$ sufficiently close to the equilibrium point x^*, the dynamics can be summarized in

$$\big(x_i(k) - x^* \big) \approx \big(\overline{x}(0) - x^* \big) e^{-\varepsilon \sum_{t=0}^{k-1} w_i(t)}$$

for all $i \in \mathcal{V}$, i.e., all the local estimates show a linear convergence to the global optimum, with rates depending both on ε and on the number of local updates. As a consequence, one would like to increase ε as much as possible, however large ε might lead the system to instability if the initial conditions are not sufficiently close to the global optimum x^*.

Remark 30 In Algorithm 8 we considered linear iterative average consensus schemes for notational simplicity reasons. However, *the proof of Proposition 28 does not rely on linear consensus updates on a undirected graph.* The only requirement is to update the y_i's and z_i's with an algorithm that achieves average consensus exponentially uniformly fast. Therefore it is possible to exploit also convergence acceleration methods (see, e.g., Aysal et al. (2009)) or average consensus algorithms for directed graphs Franceschelli et al. (2011); Cai and Ishii (2012).

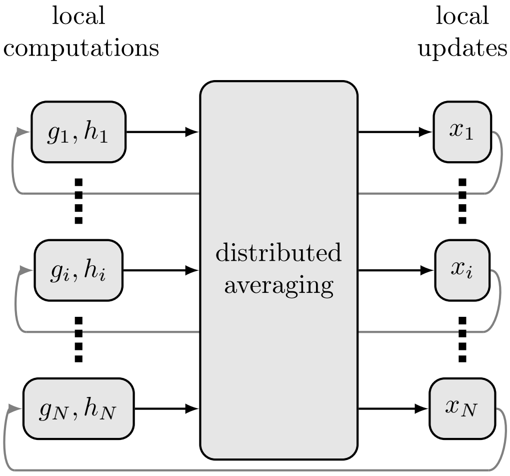


Figure 3.1: Graphical representation of ANRC which emphasize the local and global operations based on an arbitrary average consensus algorithm.

Remark 31 The previous propositions are based on Assumptions 23 and 24, that are deterministic worst-case hypotheses on the agents and edges activations processes. We conjecture that substituting the previous deterministic assumptions with randomized ones where i.i.d. agents and edges activations satisfy

$$\mathbb{E}[w_i(k)] \geq \overline{w} > 0, \quad \forall i \in \mathcal{V}, \forall k \in \mathbb{N}$$
$$\mathbb{E}[u_{(i,j)}] \geq \overline{u} > 0, \quad \forall (i,j) \in \mathcal{E}, \forall k \in \mathbb{N}$$

for some positive constants $\overline{w}$, $\overline{u}$ will lead to exponential bounds holding almost surely.

3.2 Performance

We compare the performance of the ANRC and its accelerated version, denoted as Asynchronous Fast Newton-Raphson Consensus (AFNRC) and described in detail in Algorithm 12, with three distributed convex optimization methods, namely the Asynchronous Distributed Subgradient Method (ADSM), the PEM and the Asynchronous Alternating Direction Method of Multipliers (AADMM), described respectively in Algorithm 9, 10 and 11. The following discussion provides some details about these strategies.

The aim is to show that, for the considered experiments, the convergence rates of Algorithm 8 are generally faster than the ones of Algorithm ADSM and PEM and comparable with Algorithm 11. We now present the quantities involved in the simulations and the kind of experiments performed. Then we describe the results.

We consider two particular graphs, both of $N = 25$ agents: the random geometric graph of Figure 3.2 and a complete one. We generate the costs as

$$f_i(x) = c_i e^{a_i x} + d_i e^{-b_i x}, \quad i = 1, \ldots, N \tag{3.55}$$

where $a_i, b_i \sim \mathcal{U}\,[0, 0.2]$ and $c_i, d_i \sim \mathcal{U}\,[0, 1]$. Examples are shown in Figure 3.2.

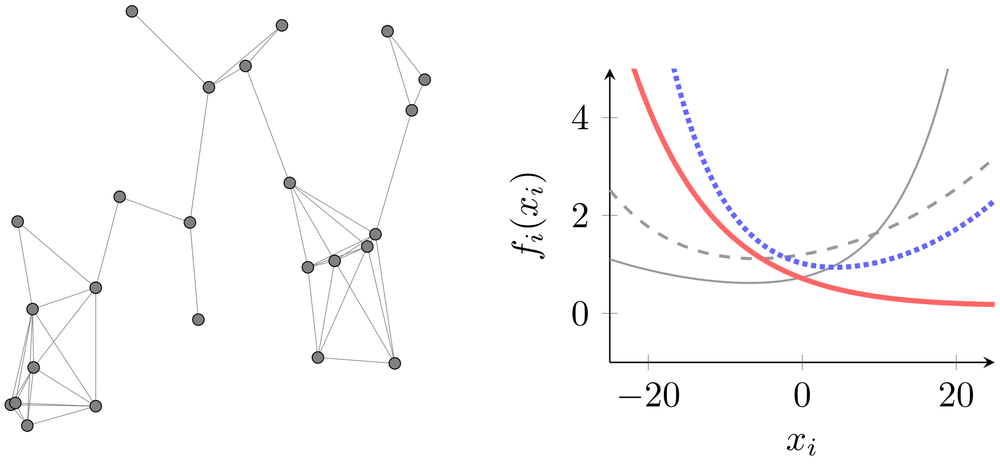


Figure 3.2: The random geometric graph used in the simulations and some examples of local cost functions (3.55).

We use symmetric gossip for the consensus protocol as in (3.7) with $\alpha = 0.5$. To make a fair comparison between the different approaches, we pay attention to respect the requirements of each algorithm. This means that the ordered agents in $\mathcal{V}$ are sequentially activated, as requested by Algorithm 11. Instead, the edges activation sequence is obtained concatenating independent permutations of the elements of $j \in \mathcal{N}_i$ once a agent i is selected, so that Assumption 23 is ensured (with $B = Nd_{max}$ where d_{max} is the largest agent degree of the network). Moreover, the initial conditions are $x_i(0) = \arg\min_x f_i(x) =: x_i^*$ for all the algorithms, as requested by Algorithm 10. The tuning parameters of the five algorithms have been manually selected in order to achieve the fastest convergence rates possible while preventing divergence effects. In the case of the incomplete random geometric graph, the empirically selected parameters are $\varepsilon = 0.9$ for the ANRC, $\varepsilon = 0.1$ and $\varphi = 1.55$ for the AFNRC, $\varrho = 30$ for the ADSM and $\delta = 0.05$ for the AADMM. In the case of the complete rgraph, the parameters are $\varepsilon = 0.9$ for the ANRC, $\varepsilon = 0.9$ and $\varphi = 1$ for the AFNRC, $\varrho = 55$ for the ADSM and $\delta = 0.001$ for the AADMM.

ADSM

ADSM as proposed in Nedić (2010); Nedić et al. (2001), considers the classical asynchronous incremental subgradient, alternating consensus steps on the current estimated global minimum $x_i(k)$ with subgradient updates of each $x_i(k)$ towards the minimum of the local cost f_i. To guarantee the convergence, the amplitude of the local subgradient steps is dynamically decreased through diminishing stepsize. Algorithm 9 presents an ADSM implementation, where ϱ is a tuning parameter.

Algorithm 9 ADSM Nedić (2010)

```
    (storage allocation and constraints on parameters)
$\boldsymbol{x} \in \mathbb{R}^N$ for $k = 0, 1, \ldots$
$c_i$ = counter associated to agent $i$, $i = 1, \ldots, N$ ($\boldsymbol{c} := [c_1, \ldots, c_N]^T$)
$\varrho \in \mathbb{R}_+$
    (initialization)
$\boldsymbol{x}(0) = \boldsymbol{x_0}$
$\boldsymbol{c}(0) = \boldsymbol{0}$
    (main algorithm)
for $k = 0, 1, \ldots$ do
    $\boldsymbol{x}(k+1) = P(k)\left[\boldsymbol{x}(k) - \varrho E(k)\frac{\boldsymbol{f}'(\boldsymbol{x}(k))}{\boldsymbol{c}(k)}\right]$
    $\boldsymbol{c}(k+1) = \boldsymbol{c}(k) + E(k)\mathbb{1}$
```

PEM

As proposed in Lu et al. (2011), PEM is a gossip-style, distributed asynchronous iterative algorithm that solves unconstrained, separable, convex consensus optimization problems over undirected networks with time varying topologies, where each local function is strictly convex, continuously differentiable, and has a minimizer. PEM utilizes non-gradient-based update rules similar to Pairwise Averaging (Tsitsiklis (1984)) that involve no stepsize.

PEM requires one-time sharing of the f_i's between gossiping agents (which may be costly or impermissible in some applications). At each time step k an agent i with one or more one-hop neighbors initiates the iteration and selects a neighbor j, to gossip. Agent i transmits $x_i(k)$ to j and it also transmits f_i, if agents j does not know f_i. Agent j sets $x_j(k+1) = (f_i' + f_j')^\dagger(f_i'(x_i(k)) + f_j'(x_j(k)))$, where $(f_i' + f_j')^\dagger$ denotes the inverse of $f_i' + f_j'$. Then j transmits $x_j(k+1)$ to i that sets $x_i(k+1) = x_j(k+1)$. PEM essentially needs that every agent is capable of applying a root-finding method, maintaining a list of its one-hop neighbors, and remembering the functions it learns along the way. PEM may be expressed in a compact form as in Algorithm 10 where $\Psi : (\mathcal{F}', k) \to \mathcal{F}'$ is the linear operator onto the vectorial space $\mathcal{F}'$ of $\boldsymbol{f}'$, such that $\Psi[\boldsymbol{f'}, k] := 2(I - P_{1/2}(k))\boldsymbol{f'}$ and $\Psi[\boldsymbol{f'}, k]^\dagger$ denote the inverse of the injective function $\Psi[\boldsymbol{f'}, k]$, with its codomain restricted to its range.

Algorithm 10 PEM Lu et al. (2011)

```
    (storage allocation and constraints on parameters)
$\boldsymbol{x}(k) \in \mathbb{R}^N$
    (initialization)
$\boldsymbol{x}(0) = \arg\min_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x})$
    (main algorithm)
for $k = 0, 1, \dots$ do
    $\boldsymbol{x}(k+1) = \boldsymbol{x}(k) - S(k)\Big(\boldsymbol{x}(k-1) + \Psi[\boldsymbol{f'}, k]^\dagger\left(2(I - P_{1/2}(k))\boldsymbol{f}'(\boldsymbol{x}(k))\right)\Big)$
```

AADMM

AADMM instead, is a distributed ADMM algorithm in which the updates of the agents are done in a sequential order. A particular implementation of an asynchronous ADMM belongs to the incremental distributed algorithms where each agent takes turn to update the system wide decision variable and passes the updated variable to the network (see Ram et al. (2009)). Here we consider the following Algorithm 11, that distinguishes itself among the other studies in the literature because each agent maintains and updates its local estimate. Specifically, AADMM uses an augmented Lagrangian approach, using scalar $\delta > 0$ as the penalty parameter. A dual variable λ_{ij} is associated with the constraint $x_i = x_j$ on edge (i, j). Each agent i keeps a local decision estimate x_i and a vector of dual variables λ_{ki} with $k < i$. $\boldsymbol{\lambda}_i$ denotes the vector of dual variables owned by agent i, i.e.

$$\boldsymbol{\lambda}_i := \begin{cases} \lambda_{ji} & \text{if } j < i \\ 0 & \text{if } j \geq i. \end{cases}$$

$L_i := \{j \,|\, (i,j) \in \mathcal{E},\, j < i\}$, $R_i := \{j \,|\, (i,j) \in \mathcal{E},\, i < j\}$ denote the sets of the predecessors and successors of i, i.e. the sets of neighbors whose index is smaller and larger than i respectively. At each iteration k agent i updates its state x_i and dual variables $\boldsymbol{\lambda}_i$ before agent j if $i < j$, i.e. the ordered agents in $\mathcal{V}$ are sequentially activated.

Algorithm 11 AADMM Wei and Ozdaglar (2012)

```
(storage allocation and constraints on parameters)
$\boldsymbol{x}(k) \in \mathbb{R}^N, \boldsymbol{\lambda}_i(k) \in \mathbb{R}^N, \forall i \in \mathcal{V}$
$\delta \in (0,1)$
(initialization)
$\boldsymbol{x}(0) = \boldsymbol{x_0}$
$\boldsymbol{\lambda}_i(0) = \mathbf{0}, \forall i \in \mathcal{V}$
(main algorithm)
for $k = 0, 1, \ldots$ do
    if $w_i(k)$, $i \in \mathcal{V}$ then
        $x_i(k+1) = \arg\min_x \Big[ f_i(x) + \frac{\delta}{2} \sum_{j \in L_i} \Big\| x_j(k) - x - \frac{\delta}{2} \lambda_{ji}(k) \Big\|^2 + \frac{\delta}{2} \sum_{j \in R_i} \Big\| x - x_j(k) - \frac{\delta}{2} \lambda_{ij}(k) \Big\|^2 \Big]$
        $\lambda_{ji}(k+1) = \lambda_{ji}(k) - \delta(x_j(k) - x_i(k+1))$
    else
        $x_i(k+1) = x_i(k)$
        $\lambda_{ji}(k+1) = \lambda_{ji}(k), \forall j \in L_i$
```

AFNRC

AFNRC is an accelerated version of Algorithm 8 that recalls the accelerated SNRC presented in Zanella et al. (2012c). It exploits an additional level of memory (here the states $\widetilde{\boldsymbol{y}}(k+1)$, $\widetilde{\boldsymbol{z}}(k+1)$) to speed up the convergence properties of the consensus strategy. The parameter φ, weights the gradient, the memory and the consensus update. To be effective in an asynchronous perspective, a further level of memory is introduced (the states $\widehat{\boldsymbol{y}}(k+1)$, $\widehat{\boldsymbol{z}}(k+1)$), whose update, as long as for the states $\boldsymbol{y}(k)$ and $\boldsymbol{z}(k)$, is determined by the matrix

$$Q(k) := \begin{bmatrix} (1-\varphi)(I-E(k)) + \varphi P(k) & (1-\varphi)E(k) \\ E(k) & I-E(k) \end{bmatrix}.$$

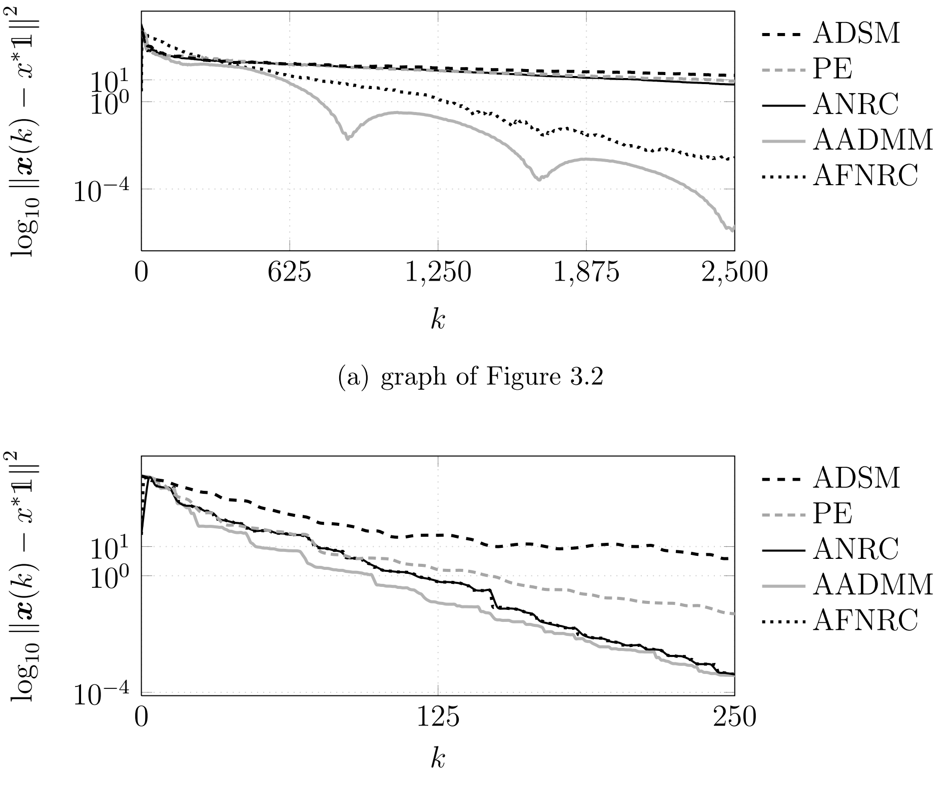
To guarantee the AFNRC to be faster than the ANRC, φ is taken in the interval $(1, 2)$, see Muthukrishnan et al. (1998).

Algorithm 12 AFNRC

```
    (storage allocation and constraints on parameters)
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \widetilde{\boldsymbol{y}}, \widetilde{\boldsymbol{z}}, \widehat{\boldsymbol{y}}, \widehat{\boldsymbol{z}} \in \mathbb{R}^N$
$\varepsilon \in (0,1)$, $\varphi \in (1,2)$
    (initialization)
$\boldsymbol{x}(0) = \boldsymbol{x_0}$
$\boldsymbol{y}(0) = \boldsymbol{z}(0) = \widehat{\boldsymbol{y}}(0) = \widehat{\boldsymbol{z}}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0}$
$\widetilde{\boldsymbol{y}}(0) = \frac{1}{2-\varphi}\boldsymbol{g}(\boldsymbol{x}(0))$, $\widetilde{\boldsymbol{z}}(0) = \frac{1}{2-\varphi}\boldsymbol{h}(\boldsymbol{x}(0))$
    (main algorithm)
for $k = 1, 2, \ldots$ do
    (consensus)
    $\begin{bmatrix} \boldsymbol{y}(k) \\ \widehat{\boldsymbol{y}}(k) \end{bmatrix} = Q(k) \begin{bmatrix} \widetilde{\boldsymbol{y}}(k-1) \\ \widehat{\boldsymbol{y}}(k-1) \end{bmatrix}$
    $\begin{bmatrix} \boldsymbol{z}(k) \\ \widehat{\boldsymbol{z}}(k) \end{bmatrix} = Q(k) \begin{bmatrix} \widetilde{\boldsymbol{z}}(k-1) \\ \widehat{\boldsymbol{z}}(k-1) \end{bmatrix}$
    (update of the local guesses)
    $\boldsymbol{x}(k) = \boldsymbol{x}(k-1) + \varepsilon S(k) \left( -\boldsymbol{x}(k-1) + \frac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)} \right)$
    (update of the auxiliary variables)
    $\widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k) + \frac{1}{2-\varphi} E(k) \left[ \boldsymbol{g}\big(\boldsymbol{x}(k)\big) - \boldsymbol{g}\big(\boldsymbol{x}(k-1)\big) \right]$
    $\widetilde{\boldsymbol{z}}(k) = \boldsymbol{z}(k) + \frac{1}{2-\varphi} E(k) \left[ \boldsymbol{h}\big(\boldsymbol{x}(k)\big) - \boldsymbol{h}\big(\boldsymbol{x}(k-1)\big) \right]$
```

Results

Figure 3.3 shows the evolutions of the square errors $\|\boldsymbol{x}(k) - x^*\mathbb{1}\|^2$ relative to the outcomes of the five strategies considering respectively the graph of Figure 3.2 and the complete graph (some of the local costs are shown in Figure 3.2). We notice that, for this specific simulation, all the algorithms converge to the global optimum x^*. We can notice that the effect of the topology of the network can play a crucial role on the convergence properties of these algorithms (the intuition being that the faster the consensus is, the faster the optimization converges). While the ADSM is the slowest to converge, PEM is faster than ADSM but slower than ANRC. Instead, the AFNRC and AADMM methods converge to the global optimum in a comparable amount of time.



(a) graph of Figure 3.2

(b) complete graph

Figure 3.3: Comparison of the square error $\|\boldsymbol{x}(k) - x^*\mathbb{1}\|^2$ in $\log_{10}$-scale for the case $N = 25$ applied to the five algorithms: ANRC (Algorithm 8, (a) $\varepsilon = 0.9$, (b) $\varepsilon = 0.9$), AFNRC (Algorithm 12, (a) $\varepsilon = 0.1$ and $\varphi = 1.55$, (b) $\varepsilon = 0.9$ and $\varphi = 1$), PEM (Algorithm 10), ADSM (Algorithm 9, (a) $\varrho = 30$, (b) $\varrho = 55$), AADMM (Algorithm 11, (a) $\delta = 0.05$, (b) $\delta = 0.001$).

Conclusions

In thesis we proposed a novel distributed optimization strategy suitable for convex, unconstrained, multidimensional, smooth and separable cost functions. The algorithm does not rely on Lagrangian formalisms and acts as a distributed Newton-Raphson optimization strategy by repeating the following steps: agents first locally compute and update second order Taylor expansions around the current local guesses and then they suitably combine them by means of average consensus algorithms to obtain a sort of approximated Taylor expansion of the global cost. This allows each agent to infer a local Newton direction, used to locally update the guess of the global minimum.

First we offered a synchronous scalar version of the Newton-Raphson Consensus strategy. We then proposed a multidimensional perspective, accompanied by two approximated versions of the main algorithm to take into account the possible computational, communication and memory constraints that may arise in practical scenarios.

We produced proofs of convergence and analysis of robustness in terms of initial conditions under some simplifying assumptions like the use of scalar smooth convex functions and synchronous implementations. Importantly, the average consensus protocols and the local updates steps have different time-scales, and the whole algorithm is proven to be convergent only if the updates are sufficiently slow with respect to the consensus.

We provided some numerical simulations confirming the properties of the proposed algorithm and we compared it with popular distributed optimization strategies: if suitably tuned, the algorithm is generally faster than Distributed Subgradient Methods and, from practical perspectives, it can have the same performance of Alternating Direction Method of Multipliers. We showed that the proposed algorithm uses average consensus as a building block, thus it naturally supports the use of accelerated consensus techniques, that can further improve its convergence properties.

We studied the rates of convergence of the Synchronous Newton-Raphson Consensus and the Gradient Descent Consensus, the simple assumption of using quadratic costs functions, with the aim of building the path for characterizations valid in general frameworks.

The results have shown that convergence properties heavily rely on the amount of coordination required to the agents. Especially for the distributed gradient descent, the degree of diversity of the local cost functions, i.e. their curvature, impacts on the rate of convergence: the intuition is that the optimum can be reached more easily if agents have similar curvatures. In a certain sense, similar costs reflect to similar behaviors, and similar behaviors require less coordination to reach consensus.

Finally, we proposed an asynchronous version of the Synchronous Newton-Raphson Consensus algorithm, here referred as Asynchronous Newton-Raphson Consensus. By proposing this extension we showed that Synchronous Newton-Raphson Consensus may play an important role among the distributed optimization algorithms. It has in fact a natural niche, composed by the situations where the network topology is unknown and possibly time-varying (for which Asynchronous Alternating Direction Method of Multipliers may suffer of extremely complex implementations), and where the local cost functions are sufficiently smooth (for which the Asynchronous Newton-Raphson Consensus converges faster than Asynchronous Distributed Subgradient Methods, due to the fact that the former uses also second-order derivatives). Nonetheless numerical investigations lead to conjecture that the algorithm preserves convergence properties for certain opportune stochastic protocols.

Future works, that need firstly to address this issue, should also analyze the effects of numerical errors and packet losses, the convergence speed under specific graphs and local cost functions scenarios, and also extend the technique to constrained problems. Moreover, an important avenue is to study how the agents can dynamically and locally tune the speed of the local updates with respect to the consensus process, namely how to tune their parameter ε. In fact large values of ε gives faster convergence but might lead to instability. Lastly, the analytical characterization of the rate of convergence of the proposed strategies and the extensions to non-smooth convex functions need to be addressed. Natural questions are how the rate of convergence is affected by time-varying consensus protocols and non-quadratic costs functions.

Bibliography

Akyildiz, I. F. and I. H. Kasimoglu (2004). Wireless sensor and actor networks: research challenges. *Ad Hoc Networks 2*(4), 351–367.

Antsaklis, P. and J. Baillieul (2007, jan.). Special issue on technology of networked control systems. *Proceedings of the IEEE 95*(1), 5–8.

Athuraliya, S. and S. H. Low (2000). Optimization flow control with Newton like algorithm. *Telecommunication Systems 15*, 345–358.

Aysal, T., B. Oreshkin, and M. Coates (2009, April). Accelerated Distributed Average Consensus via Localized Node State Prediction. *IEEE Transactions on Signal Processing 57*(4), 1563–1576.

Bai, E.-W., L.-C. Fu, and S. S. Sastry (1986, September). Averaging analysis for discrete time sample data. Technical report, University of California, Berkeley.

Becker, S. and Y. Le Cun (1988, September). Improving the convergence of back-propagation learning with second order methods. Technical report, University of Toronto.

Bertsekas, D. P. (1982). *Constrained Optimization and Lagrange Multiplier Methods.* Boston, MA: Academic Press.

Bertsekas, D. P. (1998). *Network Optimization: Continuous and Discrete Models.* Belmont, Massachusetts: Athena Scientific.

Bertsekas, D. P., A. Nedić, and A. E. Ozdaglar (2003). *Convex Analysis and Optimization.* Athena Scientific.

Bertsekas, D. P. and J. N. Tsitsiklis (1997). *Parallel and Distributed Computation: Numerical Methods.* Athena Scientific.

Blatt, D., A. Hero, and H. Gauchman (2007). A convergent incremental gradient method with a constant step size. *SIAM Journal on Optimization 18*(1), 29–51.

Blondel, V. D., J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis (2005). Convergence in Multiagent Coordination, Consensus, and Flocking. pp. 2996–3000.

Bolognani, S. (2011). *Methods and applications in networked control and Feedback control design for quantum systems.* Ph.d. thesis, Università degli Studi di Padova.

Bolognani, S., S. Del Favero, L. Schenato, and D. Varagnolo (2010, January). Consensus-based distributed sensor calibration and least-square parameter identification in WSNs. *International Journal of Robust and Nonlinear Control 20*(2), 176–193.

Bolognani, S. and S. Zampieri (2011, December). Distributed control for optimal reactive power compensation in smart microgrids. In *Proc. 50th Control and Decision Conference and European Control Conference (CDC-ECC'11)*, Orlando, FL.

Boyd, S., A. Ghosh, B. Prabhakar, and D. Shah (2006). Randomized Gossip Algorithms. *IEEE Transactions on Information Theory 52*(6), 2508–2530.

Boyd, S., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2010). Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. Technical report, Stanford Statistics Dept.

Boyd, S. and L. Vandenberghe (2004). *Convex Optimization.* Cambridge University Press.

Bürger, M., G. Notarstefano, F. Bullo, and F. Allgöwer (2012). A distributed simplex algorithm for degenerate linear programs and multi-agent assignments. *Automatica 48*(9), 2298–2304.

Cai, K. and H. Ishii (2012, November). Average Consensus on General Strongly Connected Digraphs. *Automatica 48*(11), 2750–2761.

Dembo, R. S., S. C. Eisenstat, and T. Steihaug (1982). Inexact newton methods. *SIAM Journal on Numerical 19*(2), 400–408.

Dem'yanov, V. F. and L. V. Vasil'ev (1985). *Nondifferentiable Optimization.* Springer - Verlag.

Desoer, C. A. and M. Vidyasagar (2009). *Feedback Systems.* SIAM.

Duchi, J. C., A. Agarwal, and M. J. Wainwright (2012, March). Dual averaging for distributed optimization: convergence analysis and network scaling. *IEEE Transactions on Automatic Control 57*(3), 592–606.

Erseghe, T., D. Zennaro, E. Dall'Anese, and L. Vangelista (2011, November). Fast Consensus by the Alternating Direction Multipliers Method. *IEEE Transactions on Signal Processing 59*(11), 5523–5537.

Fagnani, F. and S. Zampieri (2008, May). Randomized consensus algorithms over large scale networks. *IEEE Journal on Selected Areas in Communications 26*(4), 634–649.

Fax, J. and R. Murray (2004). Information flow and cooperative control of vehicle formations. *Automatic Control, IEEE Transactions on 49*(9), 1465–1476.

Fischione, C. (2011). F-Lipschitz Optimization with Wireless Sensor Networks Applications. *IEEE Transactions on Automatic Control 56*(10), 2319–2331.

Franceschelli, M., A. Giua, and C. Seatzu (2011). Distributed Averaging in Sensor Networks Based on Broadcast Gossip Algorithms. *IEEE Sensors Journal 11*(3), 808–817.

Freris, N. and A. Zouzias (2012). Fast Distributed Smoothing for Network Clock Synchronization. In *IEEE Conference on Decision and Control.*

Garin, F. and L. Schenato (2011). *A survey on distributed estimation and control applications using linear consensus algorithms*, Volume 406, pp. 75–107. Springer.

Ghadimi, E., I. Shames, and M. Johansson (2012). Accelerated Gradient Methods for Networked Optimization. *IEEE Transactions on Signal Processing (under review).*

Ghadimi, E., A. Teixeira, I. Shames, and M. Johansson (2012). On the Optimal Step-size Selection for the Alternating Direction Method of Multipliers. In *Necsys*.

Golub, G. H. and C. F. Van Loan (1996). *Matrix computations* (third ed.). John Hopkins University Press.

Hadjicostis, C. N. and T. Charalambous (2011, September). Asynchronous coordination of distributed energy resources for the provisioning of ancillary services. In *Allerton Conference on Communication, Control, and Computing*, pp. 1500–1507.

Hadjicostis, C. N. and T. Charalambous (2012, October). Average Consensus in the Presence of Delays and Dynamically Changing Directed Graph Topologies. *IEEE Transactions on Automatic Control (under review)*.

Hastie, T., R. Tibshirani, and J. Friedman (2008). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Second ed.). Springer.

Hauser, R. and J. Nedić (2005). The Continuous Newton-Raphson Method Can Look Ahead. *SIAM Journal on Optimization 15*, 915–925.

He, B. and X. Yuan (2011). On the O(1/t) convergence rate of alternating direction method. *SIAM Journal on Numerical Analysis (to appear)*.

Hestenes, M. R. (1969). Multiplier and gradient methods. *Journal of Optimization Theory and Applications 4*(5), 303–320.

Ho, Y. C., L. Servi, and R. Suri (1980). A class of center-free resource allocation algorithms. *Large Scale Systems 1*(1), 51–62.

Jadbabaie, A., A. Ozdaglar, and M. Zargham (2009). A Distributed Newton Method for Network Optimization. In *IEEE Conference on Decision and Control*, pp. 2736–2741.

Jakovetić, D., J. a. Xavier, and J. M. F. Moura (2011, August). Cooperative convex optimization in networked systems: Augmented lagrangian algorithms with directed gossip communication. *IEEE Transactions on Signal Processing 59*(8), 3889–3902.

Jiang, Z.-P. and Y. Wang (2002). A converse Lyapunov theorem for discrete-time systems with disturbances. *Systems & Control Letters 45*(1), 49–58.

Johansson, B. (2008). *On Distributed Optimization in Networked Systems*. Ph. D. thesis, KTH Royal Institute of Technology.

Johansson, B., M. Rabi, and M. Johansson (2009). A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization 20*(3), 1157–1170.

Khalil, H. K. (2001). *Nonlinear Systems* (Third ed.). Prentice Hall.

Kiwiel, K. C. (2004). Convergence of approximate and incremental subgradient methods for convex optimization. *SIAM Journal on Optimization 14*(3), 807–840.

Kokotović, P., H. K. Khalil, and J. O'Reilly (1999). *Singular Perturbation Methods in Control: Analysis and Design*. Number 25 in Classics in applied mathematics. SIAM.

Lobel, I., A. Ozdaglar, and D. Feijer (2011). Distributed multi-agent optimization with state-dependent communication. *Mathematical Programming 129*(2), 255–284.

Lu, J., C. Y. Tang, P. R. Regier, and T. D. Bow (2011, December). Gossip Algorithms for Convex Consensus Optimization Over Networks. *IEEE Transactions on Automatic Control 56*(12), 2917–2923.

Mota, J. a., J. Xavier, P. Aguiar, and M. Püschel (2012). Distributed ADMM for Model Predictive Control and Congestion Control. In *IEEE Conference on Decision and Control*.

Muthukrishnan, S., B. Ghosh, and M. H. Schultz (1998). First and Second Order Diffusive Methods for Rapid, Coarse, Distributed Load Balancing. *Theory of Computing Systems 31*(4), 331–354.

Nedić, A. (2010). Asynchronous Broadcast-Based Convex Optimization over a Network. *IEEE Transactions on Automatic Control 56*(6), 1337–1351.

Nedić, A., D. Bertsekas, and V. Borkar (2001). Distributed asynchronous incremental subgradient methods. *Studies in Computational Mathematics 8*, 381–407.

Nedić, A. and D. P. Bertsekas (2001). Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization 12*(1), 109–138.

Nedić, A. and A. Ozdaglar (2008). Approximate primal solutions and rate analysis for dual subgradient methods. *SIAM Journal on Optimization 19*(4), 1757–1780.

Nedić, A. and A. Ozdaglar (2009). Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE Transactions on Automatic Control 54*(1), 48–61.

Nedić, A. and A. Ozdaglar (2010, July). Convergence rate for consensus with delays. *Journal of Global Optimization 47*(3), 437–456.

Nedić, A., A. Ozdaglar, and P. A. Parrilo (2010, April). Constrained Consensus and Optimization in Multi-Agent Networks. *IEEE Transactions on Automatic Control 55*(4), 922–938.

Olfati-Saber, R., J. A. Fax, and R. M. Murray (2007a). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE 95*(1), 215–233.

Olfati-Saber, R., J. A. A. Fax, and R. M. M. Murray (2007b, January). Consensus and Cooperation in Multi-Agent Networked Systems. *Proceedings of IEEE 95*(1), 215–233.

Olshevsky, A. and J. Tsitsiklis (2009a). Convergence speed in distributed consensus and averaging. *SIAM journal on control and optimization 48*(1), 33–55.

Olshevsky, A. and J. N. Tsitsiklis (2009b, February). Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization 48*(1), 33–55.

Ram, S. S., A. Nedić, and V. V. Veeravalli (2009). Incremental stochastic subgradient algorithms for convex optimzation. *SIAM Journal on Optimization 20*(2), 691–717.

Ribeiro, A. (2010, December). Ergodic stochastic optimization algorithms for wireless communication and networking. *IEEE Transactions on Signal Processing 58*(12), 6369–6386.

Saber, R. O. and R. M. Murray (2003). Consensus protocols for networks of dynamic agents. In *American Control Conference, 2003. Proceedings of the 2003*, Volume 2, pp. 951–956.

Sahai, T., A. Speranzon, and A. Banaszuk (2012, January). Hearing the clusters of a graph: A distributed algorithm. *Automatica 48*(1), 15–24.

Schizas, I. D., A. Ribeiro, and G. B. Giannakis (2008). Consensus in Ad Hoc WSNs With Noisy Links - Part I: Distributed Estimation of Deterministic Signals. *IEEE Transactions on Signal Processing 56*, 350–364.

Shor, N. Z. (1985). *Minimization Methods for Non-Differentiable Functions.* Springer-Verlag.

Skinner, L. A. (2011). *Singular Perturbation Theory.* Springer.

Sundarapandian, V. (2002, April). Global asymptotic stability of nonlinear cascade systems. *Applied Mathematics Letters 15*(3), 275–277.

Tanabe, K. (1985). Global analysis of continuous analogues of the Levenberg-Marquardt and Newton-Raphson methods for solving nonlinear equations. *Annals of the Institute of Statistical Mathematics 37*(1), 189–203.

Teel, A. R., D. Nesic, and P. V. Kokotovic (1998, December). A note on input-to-state stability of sampled-data nonlinear systems. In *IEEE Conference on Decision and Control*, Volume 3, pp. 2473–2478.

Tsitsiklis, J. N. (1984). *Problems in decentralized decision making and computation.* Ph. D. thesis, Massachusetts Institute of Technology.

Tsitsiklis, J. N., D. P. Bertsekas, and M. Athans (1986, September). Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control 31*(9), 803–812.

Van Ast, J., R. Babška, and B. D. Schutter (2008). Particle swarms in optimization and control. In *IFAC World Congress.*

Wang, J. and N. Elia (2010, September). Control approach to distributed optimization. In *Forty-Eighth Annual Allerton Conference*, Volume 1, Allerton, Illinois, USA, pp. 557–561.

Wei, E. and A. Ozdaglar (2012). Distributed Alternating Direction Method of Multipliers. In *IEEE Conference on Decision and Control.*

Wei, E., A. Ozdaglar, and A. Jadbabaie (2010). A Distributed Newton Method for Network Utility Maximization. In *IEEE Conference on Decision and Control*, pp. 1816–1821.

Xiao, L. and S. Boyd (2006). Optimal scaling of a gradient method for distributed resource allocation. *Journal of optimization theory and applications 129*(3), 469–488.

Xiao, L., S. Boyd, and S. Lall (2005). A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus. In *International symposium on Information processing in sensor networks*, pp. 63–70.

Zanella, F., D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato (2011, December). Newton-Raphson consensus for distributed convex optimization. In *IEEE Conference on Decision and Control and European Control Conference*, pp. 5917–5922.

Zanella, F., D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato (2012a). Asynchronous Newton-Raphson Consensus for Distributed Convex Optimization. In *Necsys 2012*.

Zanella, F., D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato (2012b). Multidimensional Newton-Raphson consensus for distributed convex optimization. In *American Control Conference*.

Zanella, F., D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato (2012c). Newton-Raphson Consensus for Distributed Convex Optimization. *IEEE Transactions on Automatic Control (under review)*.

Zanella, F., D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato (2012d, December). The convergence rate of Newton-Raphson consensus optimization for quadratic cost functions. In *IEEE Conference on Decision and Control and European Control Conference*.

Zargham, M., A. Ribeiro, A. Ozdaglar, and A. Jadbabaie (2011). Accelerated Dual Descent for Network Optimization. In *American Control Conference*.

Zhu, M. and S. Martínez (2012). On Distributed Convex Optimization Under Inequality and Equality Constraints. *IEEE Transactions on Automatic Control 57*(1), 151–164.