Multidimensional Newton-Raphson consensus for distributed convex optimization

Filippo Zanella, Damiano Varagnolo, Angelo Cenedese, Gianluigi Pillonetto, Luca Schenato

***Abstract*— In this work we consider a multidimensional distributed optimization technique that is suitable for multi-agents systems subject to limited communication connectivity. In particular, we consider a convex unconstrained additive problem, i.e. a case where the global convex unconstrained multidimensional cost function is given by the sum of local cost functions available only to the specific owning agents. We show how, by exploiting the separation of time-scales principle, the multidimensional consensus-based strategy approximates a Newton-Raphson descent algorithm. We propose two alternative optimization strategies corresponding to approximations of the main procedure. These approximations introduce tradeoffs between the required communication bandwidth and the convergence speed/accuracy of the results. We provide analytical proofs of convergence and numerical simulations supporting the intuitions developed through the paper.**

***Index Terms*— multidimensional distributed optimization, multidimensional convex optimization, consensus algorithms, multi-agent systems, Newton-Raphson methods**

I. Introduction

To cope with the growing mankind demands, humanity is building greater and greater systems. But, since big centralized systems suffer small structural flexibility and robustness to failures, nowadays trends are to shift towards distributed services and structures. Brilliant examples are the (current) principal source of information - Internet, and the (future) network of renewable energy sources - wind farms, wave parks and home solar systems. But, to operate at their best, these networks are required to distributedly solve complex optimization problems. Computations should thus require minimal coordination efforts, small computational and memory requirements, and do not rely on central processing units.

Development and study of such algorithms are major research topics in the area of control and system theory [1], [2], and have lead up to now to numerous contributions. These can be roughly divided into three main categories: methods based on primal decompositions, methods based on dual decomposition, and heuristic methods.

Primal decomposition methods operate manipulating the primal variables, often through subgradient methods, [3] and references therein. Despite they are widely applicable, they are easy to implement and they require mild assumptions on the objective functions, they may be rather slow and may not progress at each iteration [4, Chap. 6]. Implementations can be based on incremental gradients methods [5] with deterministic [6] or randomized [7] approaches, and they may use opportune projection steps to account for possible constraints [8].

Decomposition methods instead operate manipulating the dual problem, usually splitting it into simpler sub-tasks that require the agents to own local copies of the to-be-updated variables. Convergence to the global optimum is ensured constraining the local variables to converge to a common value [9]. In the class of dual decomposition methods, a particularly popular strategy is the Alternating Direction Method of Multipliers (ADMM) developed in [1, pp. 253-261] and recently proposed in various distributed contexts [10], [11].

An other interesting approach, suitable only for particular optimization problems, is to use the so-called Fast-Lipschitz methods [12], [13]. These exploit particular structures of the objective functions and constraints to increase the convergence speed. Alternative distributed optimization approaches are based on heuristics like swarm optimization [14] or genetic algorithms [15]. However their convergence and performance properties are difficult to be studied analytically.

Statement of contribution: here we focus on the unconstrained minimization of a sum of multidimensional convex functions, where each component of the global function is a private local cost available only to a specific agent. We thus offer a distributed algorithm that approximatively operates as a Newton-Raphson minimization procedure, and then derive two approximated versions that trade-off between the required communication bandwidth and the convergence speed / accuracy of the results. For these strategies we provide convergence proofs and analysis on the robustness on initial conditions of the algorithms, under the assumptions that local cost functions are convex and smooth, and that communication schemes are synchronous. The main algorithm is an extension of what has been proposed in [16], while the approximated versions are completely novel. We notice that communications between agents are based on classical average-consensus algorithms [17]. The offered algorithms inherit thus the good properties of consensus algorithms, namely their simplicity, their potential implementation with asynchronous communication schemes, and their ability to adapt to time-varying network topologies.

Structure of the paper: in Sec. II we formulate the problem from a mathematical point of view. In Sec. III we derive the main generic distributed algorithm, from which we derive three different and specific instances in Sections IV, V and VI. In Sec. VII we briefly discuss the properties of these algorithms, and then in Sec. VIII we show their effectiveness

The research leading to these results has received funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement n°257462 HYCON2 Network of excellence and n°223866 FeedNetBack, by Progetto di Ateneo CPDA090135/09 funded by the University of Padova, and by the Italian PRIN Project "New Methods and Algorithms for Identification and Adaptive Control of Technological Systems".

by means of numerical examples. Finally in Sec. IX we draw some concluding remarks[1].

II. Problem formulation

We assume that S agents, each endowed with the local N-dimensional and strictly convex cost function $f_i : \mathbb{R}^N \mapsto \mathbb{R}$, aim to collaborate in order to minimize the global cost function

$$\overline{f} : \mathbb{R}^N \mapsto \mathbb{R} \qquad \overline{f}(\mathbf{x}) = \frac{1}{S}\sum_{i=1}^{S} f_i(\mathbf{x}) \tag{1}$$

where $\mathbf{x} := [x_1 \ \cdots \ x_N]^T$ is the generic element in $\mathbb{R}^N$. Agents thus want to distributedly compute

$$\mathbf{x}^* := \arg\min_{\mathbf{x}} \overline{f}(\mathbf{x}) \tag{2}$$

exploiting low-complexity distributed optimization algorithms. As in [16], we model the communication network as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose vertexes $\mathcal{V} = \{1, 2, \ldots, S\}$ represent the agents and the edges $(i, j) \in \mathcal{E}$ represent the available communication links. We assume that the graph is undirected and connected. We say that a stochastic matrix $P \in \mathbb{R}^{S\times S}$, i.e. a matrix whose elements are non-negative and $P\mathbb{1}_S = \mathbb{1}_S$, where $\mathbb{1}_S := [1\ 1\ \cdots\ 1]^T \in \mathbb{R}^S$, is consistent with a graph $\mathcal{G}$ if $P_{ij} > 0$ only if $(i, j) \in \mathcal{E}$. If P is also symmetric and includes all edges, i.e. $P_{ij} > 0$ if $(i, j) \in \mathcal{E}$, then $\lim_{k\to\infty} P^k = \frac{1}{S}\mathbb{1}_S\mathbb{1}_S^T$. Such matrix P is also often referred as a *consensus matrix*.

In the following we use $\mathbf{x}_i(k) := [x_{i,1}(k) \ \cdots \ x_{i,N}(k)]^T$ to indicate the input location of agent i at time k, and operator ∇ to indicate differentiation w.r.t. $\mathbf{x}$, i.e.

$$\nabla f_i(\mathbf{x}_i(k)) := \left[\left.\frac{\partial f_i}{\partial x_1}\right|_{\mathbf{x}_i(k)} \ \cdots \ \left.\frac{\partial f_i}{\partial x_N}\right|_{\mathbf{x}_i(k)}\right]^T \tag{3}$$

$$\nabla^2 f_i(\mathbf{x}_i(k)) := \left[\left.\frac{\partial^2 f_i}{\partial x_m \partial x_n}\right|_{\mathbf{x}_i(k)}\right]. \tag{4}$$

In general we use the fraction bar to indicate the Hadamard division, i.e. the component-wise division of vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^N$

$$\frac{\mathbf{a}}{\mathbf{b}} := \left[\frac{a_1}{b_1}, \ \cdots, \frac{a_N}{b_N}\right]^T. \tag{5}$$

In general we use bold fonts to indicate vectorial quantities or functions which range is vectorial, plain italic fonts to indicate scalar quantities or functions which range is a scalar. We use capital italic fonts to indicate matrix quantities and capital bold fonts to indicate matrix quantities derived stacking other matrix quantities. As in [16], to simplify the proofs we exploit the following assumption, implying that $\mathbf{x}^*$ is unique:

Assumption 1. Local functions f_i belongs to $\mathcal{C}^2, \forall i$, i.e. they are continuous up to the second partial derivatives, their second partial derivatives are strictly positive, bounded, and they are defined for all $\mathbf{x} \in \mathbb{R}^N$. Moreover each scalar component of the global minimizer $\mathbf{x}^*$ does not take value on the extended values $\pm\infty$.

We notice that from the strict convexity assumptions it follows that $\mathbf{x}^*$ is unique. Moreover the assumption that each scalar component of $\mathbf{x}^*$ does not take value on the extended values is to obtain convergence proofs that do not require modifications of the standard multi-time-scales approaches for singular perturbation model analysis [18], [19, Chap. 11]. We also notice that these smoothness assumptions, despite restrictive, have been used also by other authors, see e.g. [20], [21].

A. Notation for Multidimensional Consensus Algorithms

Assume

$$A_i = \begin{bmatrix} a_{11}^{(i)} & \cdots & a_{1M}^{(i)} \\ \vdots & & \vdots \\ a_{N1}^{(i)} & \cdots & a_{NM}^{(i)} \end{bmatrix} \qquad i = 1, \ldots, S$$

to be S generic $N \times M$ matrices associated to agents $1, \ldots, S$, and that these agents want to distributedly compute $\frac{1}{S}\sum_{i=1}^{S} A_i$ by means of the double-stochastic communication matrix P. In the following sections, to indicate the whole set of the single component-wise steps

$$\begin{bmatrix} a_{pq}^{(1)}(k+1) \\ \vdots \\ a_{pq}^{(S)}(k+1) \end{bmatrix} = P \begin{bmatrix} a_{pq}^{(1)}(k) \\ \vdots \\ a_{pq}^{(S)}(k) \end{bmatrix} \qquad \begin{matrix} p = 1, \ldots, N \\ q = 1, \ldots, M \end{matrix} \tag{6}$$

we use the equivalent matricial notation

$$\begin{bmatrix} A_1(k+1) \\ \vdots \\ A_S(k+1) \end{bmatrix} = (P \otimes I_N) \begin{bmatrix} A_1(k) \\ \vdots \\ A_S(k) \end{bmatrix} \tag{7}$$

where I_N is the identity in $\mathbb{R}^{N\times N}$ and $\otimes$ is the Kronecker product. Notice that the notation is suited also for vectorial quantities, e.g. $A_i \in \mathbb{R}^N$.

III. Distributed multidimensional consensus-based optimization

Assume the local cost functions to be quadratic, i.e.

$$f_i(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{b}_i)^T A_i (\mathbf{x} - \mathbf{b}_i)$$

where $A_i > 0$. Straightforward computations show that the unique minimizer of $\overline{f}$ is given by

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

and can thus be computed using the output of two average consensus algorithms. Defining the local variables

$$\mathbf{y}_i(0) := A_i\mathbf{b}_i \in \mathbb{R}^N \qquad Z_i(0) := A_i \in \mathbb{R}^{N\times N}$$

[1]The proofs of the proposed propositions can be found in the homonymous technical report available on the authors' webpages.

and the corresponding compact forms

$$Y(k) := \begin{bmatrix} \mathbf{y}_1(k) \\ \vdots \\ \mathbf{y}_S(k) \end{bmatrix} \in \mathbb{R}^{NS} \qquad \mathbf{Z}(k) := \begin{bmatrix} Z_1(k) \\ \vdots \\ Z_S(k) \end{bmatrix} \in \mathbb{R}^{NS \times N}$$

then the algorithm

$$Y(k+1) = \left(P \otimes I_N\right) Y(k) \tag{8}$$
$$\mathbf{Z}(k+1) = \left(P \otimes I_N\right) \mathbf{Z}(k) \tag{9}$$
$$\mathbf{x}_i(k) = \left(Z_i(k)\right)^{-1} \mathbf{y}_i(k) \qquad i = 1, \ldots, S \tag{10}$$

alternates average-consensus steps (i.e. (8) and (9), given the considerations in Sec. II-A) with local updates (i.e. (10)), and is s.t. $\lim_{k\to\infty} \mathbf{x}_i(k) = \mathbf{x}^*$. The element $\mathbf{x}_i(k)$ can thus be considered the local estimate of the global minimizer $\mathbf{x}^*$ at time k. If the cost functions are not quadratic, then the previous strategy cannot be applied as it is but needs to be modified using the guidelines:

1) in general

   $$\mathbf{y}_i(0) = \nabla^2 f_i\left(\mathbf{x}_i(0)\right) \mathbf{x}_i(0) \qquad Z_i(0) = \nabla^2 f_i\left(\mathbf{x}_i(0)\right).$$

   For quadratic scenarios these two quantities are in fact independent of $\mathbf{x}_i$, but this does not happen in the general case. Consensus steps (8)-(9) should then be performed considering that the $\mathbf{x}_i(k)$'s change over time. This requires to appropriately design the update rules for $\mathbf{y}_i$ and Z_i;
2) (10) might lead to estimates that change too rapidly. This requires to take smaller steps towards the estimated minimum $\left(Z_i(k)\right)^{-1} \mathbf{y}_i(k)$.

To this aim, we propose the following general Alg. 1. Notice that *it depends on quantities that have not yet been defined*, namely $\mathbf{g}_i(k)$ and $H_i(k)$, $i = 1, \ldots, S$.

The importance of this algorithm is given by the fact that, under opportune hypotheses, the temporal evolution of the average state $\overline{\mathbf{x}} := \frac{1}{S}\sum_{i=1}^{S} \mathbf{x}_i$ approximatively follows the update rule

$$\dot{\overline{\mathbf{x}}}(t) = -\overline{\mathbf{x}}(t) + \left(\frac{1}{S}\sum_{i=1}^{S} H_i\left(\overline{\mathbf{x}}(t)\right)\right)^{-1} \left(\frac{1}{S}\sum_{i=1}^{S} \mathbf{g}_i\left(\overline{\mathbf{x}}(t)\right)\right)$$

(see proof of Prop. 2). In the following we show that this property is appealing since, exploiting proper choices of $\mathbf{g}_i(k)$ and $H_i(k)$, we can obtain distributed optimization algorithms with desirable properties such as convergence to the global optimum and small communication bandwidth requirements.

IV. Distributed Multidimensional Newton-Raphson

Consider the following Alg. 2, based on the general layout given by Alg. 1. We show now how it corresponds to the multidimensional extension of the distributed scalar optimizer described in [16], and that it distributedly computes the global optimum $\mathbf{x}^*$. We notice that initializations given in line 5 are critical for the convergence to the global minimizer; lines 8-9 are local operations assuring the Newton-Raphson computation to be based on the current local estimates $\mathbf{x}_i(k)$; lines 10-11 perform the consensus operations, and operations in line 13 are again local operations performing convex combinations between the past and new estimates.

Algorithm 1 Distributed Optimization - General Layout

(variables)
1: $\mathbf{x}_i(k), \mathbf{y}_i(k), \mathbf{g}_i(k) \in \mathbb{R}^N$; $Z_i(k), H_i(k) \in \mathbb{R}^{N\times N}$ for $i = 1, \ldots, S$ and $k = 1, 2, \ldots$
(notice: $\mathbf{g}_i$ and H_i defined in Alg. 2, Alg. 3, Alg. 4)

(parameters)
2: $P \in \mathbb{R}^{S\times S}$, consensus matrix
3: $\varepsilon \in (0, 1)$

(initialization)
4: **for** $i = 1, \ldots, S$ **do**
   set: $\mathbf{y}_i(0) = \mathbf{g}_i(-1) = \mathbf{0}$
5: $Z_i(0) = H_i(-1) = \mathbf{0}$
   $\mathbf{x}_i(0) = \mathbf{0}$

(main algorithm)
6: **for** $k = 1, 2, \ldots$ **do**
(local updates)
7: **for** $i = 1, \ldots, S$ **do**
8: $\mathbf{y}_i(k) = \mathbf{y}_i(k-1) + \mathbf{g}_i(k-1) - \mathbf{g}_i(k-2)$
9: $Z_i(k) = Z_i(k-1) + H_i(k-1) - H_i(k-2)$
(multidimensional average consensus step)
10: $Y(k) = \left(P \otimes I_N\right) Y(k)$
11: $\mathbf{Z}(k) = \left(P \otimes I_N\right) \mathbf{Z}(k)$
(local updates)
12: **for** $i = 1, \ldots, S$ **do**
13: $\mathbf{x}_i(k) = (1-\varepsilon)\mathbf{x}_i(k-1) + \varepsilon \left(Z_i(k)\right)^{-1} \mathbf{y}_i(k)$

The convergence properties can be proved exploiting a state augmentation, recognizing the existence of a two-time scales dynamical system regulated by the parameter ε, and then considering that, for small ε, the fluctuations induced by the fast subsystem exponentially vanish while the dynamics of the slow subsystem correspond to continuous-time Newton-Raphson algorithm that converges to the global optimum given the previously posed Assumption 1. For this purpose it is useful to define the shorthands

$$G(k) := \begin{bmatrix} \mathbf{g}_1(k) \\ \vdots \\ \mathbf{g}_S(k) \end{bmatrix} \in \mathbb{R}^{NS} \qquad \mathbf{H}(k) := \begin{bmatrix} H_1(k) \\ \vdots \\ H_S(k) \end{bmatrix} \in \mathbb{R}^{NS\times N}$$

Algorithm 2 Distributed Newton-Raphson

Execute Alg. 1 with definitions

$$\mathbf{g}_i(k) := \nabla^2 f_i\left(\mathbf{x}_i(k)\right) \mathbf{x}_i(k) - \nabla f_i\left(\mathbf{x}_i(k)\right) \in \mathbb{R}^N$$
$$H_i(k) := \nabla^2 f_i\left(\mathbf{x}_i(k)\right) \in \mathbb{R}^{N\times N}.$$

The first step is then to introduce the additional variables $V(k) = G(k-1)$ and $\mathbf{W}(k) = \mathbf{H}(k-1)$ and rewrite Alg. 2

as

$$\begin{aligned} V(k) &= G(k-1) \\ \mathbf{W}(k) &= \mathbf{H}(k-1) \\ Y(k) &= (P\otimes I_N)\left(Y(k-1)+G(k-1)-V(k-1)\right) \\ \mathbf{Z}(k) &= (P\otimes I_N)\left(\mathbf{Z}(k-1)+\mathbf{H}(k-1)-\mathbf{W}(k-1)\right) \\ \mathbf{x}_i(k) &= (1-\varepsilon)\mathbf{x}_i(k-1)+\varepsilon\left(Z_i(k)\right)^{-1}\mathbf{y}_i(k) \end{aligned} \tag{11}$$

from which it is possible to recognize the tracking of the quantities $\mathbf{x}_i(k)$ plus the consensus step (1^{st} to 4^{th} rows) and the local smooth updates (5^{th} row). (11) can be considered the Euler discretization, with time interval $T=\varepsilon$, of the continuous time system

$$\begin{aligned} \varepsilon\dot{V}(t) &= -V(t)+G(t) \\ \varepsilon\dot{\mathbf{W}}(t) &= -\mathbf{W}(t)+\mathbf{H}(t) \\ \varepsilon\dot{Y}(t) &= -KY(t)+(I_{NS}-K)\left[G(t)-V(t)\right] \\ \varepsilon\dot{\mathbf{Z}}(t) &= -K\mathbf{Z}(t)+(I_{NS}-K)\left[\mathbf{H}(t)-\mathbf{W}(t)\right] \\ \dot{\mathbf{x}}_i(t) &= -\mathbf{x}_i(t)+\left(Z_i(t)\right)^{-1}\mathbf{y}_i(t) \end{aligned} \tag{12}$$

with $K:=I_{NS}-(P\otimes I_N)$. It is immediate to show that K is positive semidefinite, its kernel is generated by $\mathbb{1}_{NS}$, and that its eigenvalues satisfy $0=\lambda_1<\mathrm{Re}\left[\lambda_2\right]\leq\cdots\leq\mathrm{Re}\left[\lambda_{NS}\right]<2$, where $\mathrm{Re}\left[\lambda\right]$ indicates the real part of λ. (12) is constituted by two dynamical subsystems with different time-scales, one of which is regulated by the parameter ε. Exploiting classical time-separation techniques [19, Chap. 11], splitting the dynamics in the two time scales and studying them separately for sufficiently small ε, it follows that the fast dynamics, i.e. the first four equations of (12), are s.t. $\mathbf{x}_i(t)\approx\overline{\mathbf{x}}(t)$, where $\overline{\mathbf{x}}(t):=\frac{1}{S}\sum_{i=1}^{S}\mathbf{x}_i(t)$, and moreover $\overline{\mathbf{x}}(t)$ evolves with good approximation following the ordinary differential equation

$$\dot{\overline{\mathbf{x}}}(t)=-\left[\nabla^2\overline{f}\left(\overline{\mathbf{x}}(t)\right)\right]^{-1}\nabla\overline{f}\left(\overline{\mathbf{x}}(t)\right) \tag{13}$$

corresponding to a continuous Newton-Raphson algorithm[2] that we will prove to be always convergent to the global optimum $\mathbf{x}^*$. These observations are formally stated in the following (proof in Appendix):

Proposition 2. Consider Alg. 2, equivalent to system (11) with initial conditions $V(0)=Y(0)=\mathbf{0}$ and $\mathbf{W}(0)=\mathbf{Z}(0)=\mathbf{0}$. If Assumption 1 holds true, then there exists an $\overline{\varepsilon}\in\mathbb{R}_+$ s.t. if $\varepsilon<\overline{\varepsilon}$ then Alg. 2 distributedly and asymptotically computes the global optimum $\mathbf{x}^*$, i.e. $\lim_{k\to+\infty}\mathbf{x}_i(k)=\mathbf{x}^*$ for all i.

V. Distributed Multidimensional Jacobi

Implementation of Alg. 2 requires agents to exchange information on about $O\left(N^2\right)$ scalars. This could be prohibitive in multidimensional scenarios with serious communication bandwidth constraints and large N. In these cases, to minimize the amount of information to be exchanged it is meaningful to let $H_i(k)$ be not the whole Hessian matrix $\nabla^2 f_i\left(\mathbf{x}_i(k)\right)$, but only its diagonal. The corresponding algorithm, that we call *Jacobi* due to the underlying diagonalization process, is offered in Alg. 3. We notice that this diagonalization process has already been used in literature, e.g., see [24], [25], even if in conjunction with different communication structures.

Algorithm 3 Distributed Jacobi

Execute Alg. 1 with definitions

$$\begin{aligned} \mathbf{g}_i(k) &:= H_i(k)\,\mathbf{x}_i(k)-\nabla f_i\left(\mathbf{x}_i(k)\right)\in\mathbb{R}^N \\ H_i(k) &:= \begin{bmatrix} \left.\frac{\partial^2 f_i}{\partial x_1^2}\right|_{\mathbf{x}_i(k)} & & 0 \\ & \ddots & \\ 0 & & \left.\frac{\partial^2 f_i}{\partial x_N^2}\right|_{\mathbf{x}_i(k)} \end{bmatrix}\in\mathbb{R}^{N\times N}. \end{aligned}$$

Possible interpretations of the proposed approximation are:

- agents perform modified second-order Taylor approximations of the local functions;
- agents choose a steepest descent direction in a simplified norm;
- ellipsoids corresponding to the various Hessians $\nabla^2 f_i$ are approximated with ellipsoids having axes that are parallel with the current coordinate system.

It is easy to show that this approximated strategy is invariant over affine transformations $T:\mathbb{R}^{N\times N}\mapsto\mathbb{R}^{N\times N}$, T invertible and s.t. $f_{\text{new}}(\mathbf{x})=f(T\mathbf{x})$, as classical Newton-Raphson algorithms are [26, Sec. 9.5]. It is moreover possible to prove that also Alg. 3 ensures the convergence to the global optimum, i.e. to prove the following (proof in Appendix):

Proposition 3. If Assumption 1 holds true, then there exists an $\overline{\varepsilon}'\in\mathbb{R}_+$ s.t. if $\varepsilon<\overline{\varepsilon}'$ then Alg. 3 distributedly and asymptotically computes the global optimum $\mathbf{x}^*$, i.e. $\lim_{k\to+\infty}\mathbf{x}_i(k)=\mathbf{x}^*$ for all i.

Analytical characterization of the convergence speed of Alg. 2 and Alg. 3 is left as a future work.

VI. Distributed Multidimensional Gradient Descent

We notice now that the distributed Jacobi relieves the computational requirements of the distributed Newton-Raphson, since the inversion of $H_i\left(\mathbf{x}_i(k)\right)$ corresponds to the inversion of N scalars, but nonetheless agents still have to compute the local second derivatives $\left.\frac{\partial^2 f_i}{\partial x_n^2}\right|_{\mathbf{x}_i(k)}$. If this task is still too consuming, e.g. in cases where nodes have severe computational constraints, it is possible to redefine $H_i(k)$ in Alg. 1 in a way that it reduces to a gradient-descent procedure, as did in the following algorithm.

Algorithm 4 Distributed gradient-descent

Execute Alg. 1 with definitions

$$\begin{aligned} \mathbf{g}_i(k) &:= \mathbf{x}_i(k)-\nabla f_i\left(\mathbf{x}_i(k)\right)\in\mathbb{R}^N \\ H_i(k) &:= I_N\in\mathbb{R}^{N\times N}. \end{aligned}$$

[2] Asymptotic properties of the scalar and continuous time Newton-Raphson method can be found e.g. in [22], [23].

VII. Discussion on the previous algorithms

The costs associated to the previously proposed strategies are summarized in Tab. I.

Algorithm	2	3	4
Computational Cost	$O\left(N^3\right)$	$O\left(N\right)$	$O\left(N\right)$
Communication Cost	$O\left(N^2\right)$	$O\left(N\right)$	$O\left(N\right)$
Memory Cost	$O\left(N^2\right)$	$O\left(N\right)$	$O\left(N\right)$

TABLE I

Computational, communication and memory costs of algorithms 2, 3, and 4 per single unit and single step (lines 6 to 13 of algorithm 1).

We notice that the approximation of the Hessian by neglecting the off-diagonal terms has been already proposed in centralized approaches, e.g. [27]. Intuitively, the effect of this diagonal approximation is the following: the full Newton method perform both scaling and rotation of the steepest descent step. The diagonal modified Newton method only scales the descent step in each direction, thus the more the directions of the maximal and minimal curvatures are aligned with the axes, the more the approximated method captures the curvature information and performs better.

A final remark is that the analytic Hessian can be approximated in several ways, but in general it is necessary to consider only approximations that maintain symmetry and positive definiteness. In cases where this definiteness is lacking, or matrices are bad conditioned, modifications are usually performed e.g. through Cholesky factorizations [28].

VIII. Numerical examples

We consider a ring communication graph, where agents can communicate only to their left and right neighbors, and thus the symmetric circulant communication matrix

$$P = \begin{bmatrix} 0.5 & 0.25 & & & 0.25 \\ 0.25 & 0.5 & 0.25 & & \\ & \ddots & \ddots & \ddots & \\ & & 0.25 & 0.5 & 0.25 \\ 0.25 & & & 0.25 & 0.5 \end{bmatrix}. \tag{14}$$

We consider $S = 15$, $N = 2$, and local objective functions randomly generated as

$$f_i\left(\mathbf{x}\right) = \exp\left(\left(\mathbf{x} - \mathbf{b}_i\right)^T A_i \left(\mathbf{x} - \mathbf{b}_i\right)\right), \quad i = 1, \ldots, S$$

where $\mathbf{b}_i \sim \left[\mathcal{U}\left[-5, 5\right], \mathcal{U}\left[-5, 5\right]\right]^T$, $A_i = D_i D_i^T > 0$, and

$$D_i := \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{15}$$

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

$$\mathrm{S}_1: \quad \begin{cases} d_{11} & = d_{22} \sim \mathcal{U}\left[-0.08, 0.08\right] \mathcal{R}\left[-1, 1\right] \\ d_{12} & \sim \mathcal{U}\left[-0.08, 0.08\right] \mathcal{R}\left[-0.25, 0.5\right] \\ d_{21} & \sim \mathcal{U}\left[-0.08, 0.08\right] \mathcal{R}\left[-0.5, 0.25\right] \end{cases} \tag{16}$$

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

$$\mathcal{R}\left[c, d\right] := \begin{cases} c & \text{with probability} \quad 1/2 \\ d & \text{with probability} \quad 1/2 \end{cases}$$

i.e. the axes of each contour plot are randomly oriented in the 2-D plane.

$$\mathrm{S}_2: \quad \begin{cases} d_{11} & \sim \mathcal{U}\left[-0.08, 0.08\right] \\ d_{12} & = d_{21} = 0 \\ d_{22} & = 2\, d_{11} \end{cases} \tag{17}$$

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

$$\mathrm{S}_3: \quad \begin{cases} d_{11} & \sim \mathcal{U}\left[-0.08, 0.08\right] \\ d_{12} & = d_{21} = -0.01 \\ d_{22} & \sim \mathcal{R}\left[0.9, 1.1\right]\, d_{11} \end{cases} \tag{18}$$

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

The contour plots of the global cost functions $\bar{f}$'s generated using (16), (17) and (18), and the evolution of the local states $\mathbf{x}_i$ for the three algorithms are shown in Fig. 1.

We notice that Alg. 2 and Alg. 3 have qualitatively the same behavior for the scenarios (16) and (17). This is because the approximation introduced in Alg. 3 is actually a good approximation of the analytical Hessians $\nabla^2 f_i\left(\mathbf{x}_i(k)\right)$. Conversely, Alg. 4 presents a remarkably slower convergence rate. Since the computational time of Alg. 3 and 4 are comparable, Alg. 3 seems to represent the best choice among all the presented solutions.

IX. Conclusions and Future works

Starting from [16], we offered a multidimensional distributed convex optimization algorithm that behaves approximatively as a Newton-Raphson procedure. We then proposed 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 under the assumptions of dealing with smooth convex functions, and numerical simulations to compare the performances of the proposed algorithms.

Currently there are many open future research directions. A first branch is about the analytical characterization of the speeds of convergence of the proposed strategies, while an other one is about the application of quasi-Newton methods to avoid the computation of the Hessians and the use of trust region methods. Finally, an important future extension is to allow the strategy to be implemented in asynchronous communication frameworks.

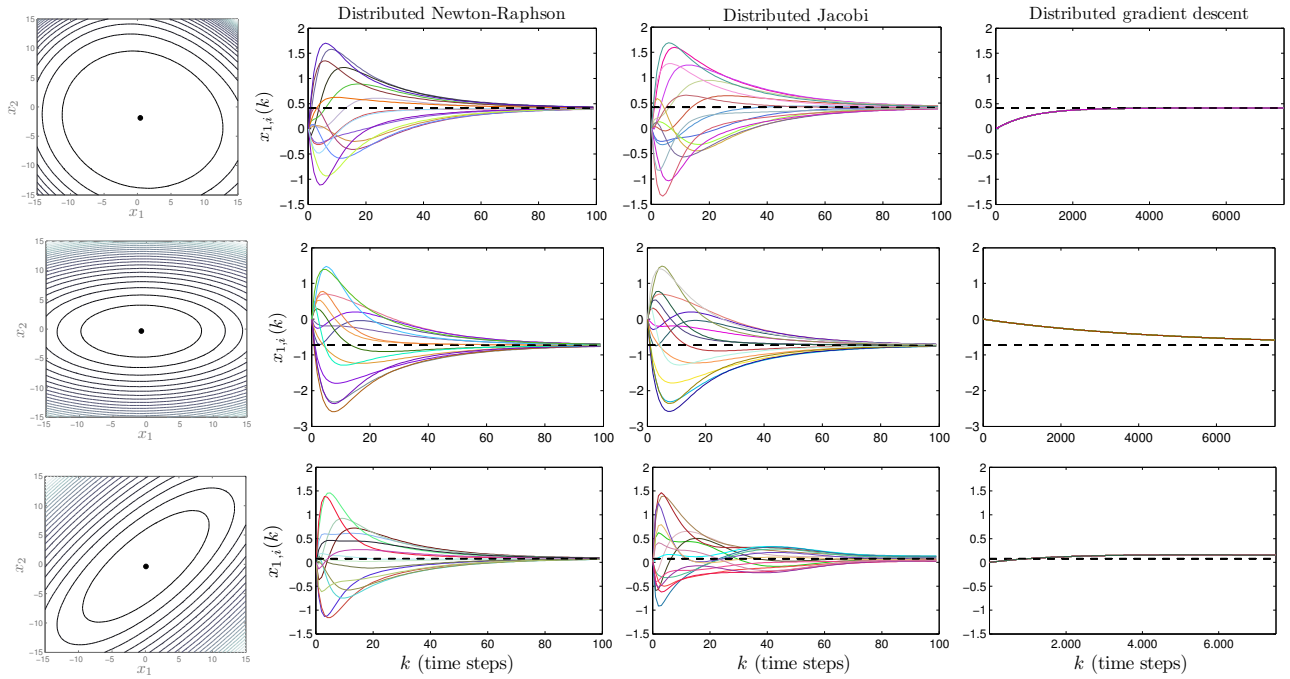

Fig. 1. First column on the left, contours plot of global function $\bar{f}$'s for scenarios S_1, S_2, S_3, respectively (from top to bottom). Black dots indicate the positions of the global minima $\mathbf{x}^*$. Second, third and fourth columns, temporal evolutions of the first components of the local states $\mathbf{x}_1$, for the case $\varepsilon = 0.25$ and $N = 15$. In particular: second column, distributed Newton-Raphson (Alg. 2). Third column, distributed Jacobi (Alg. 3). Fourth column, distributed gradient descent (Alg. 4). First row, scenario S_1. Second row, scenario S_2. Third row, scenario S_3. 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 gradient descent algorithm (fourth column).

REFERENCES

[1] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[2] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*. Belmont, Massachusetts: Athena Scientific, 1998.

[3] K. C. Kiwiel, "Convergence of approximate and incremental subgradient methods for convex optimization," *SIAM J. on Optim.*, vol. 14, no. 3, pp. 807 – 840, 2004.

[4] B. Johansson, "On distributed optimization in networked systems," Ph.D. dissertation, KTH Electrical Engineering, 2008.

[5] A. Nedić and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM J. on Optim.*, vol. 12, no. 1, pp. 109 – 138, 2001.

[6] D. Blatt, A. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM J. on Optim.*, vol. 18, no. 1, pp. 29 – 51, 2007.

[7] S. S. Ram, A. Nedić, and V. Veeravalli, "Incremental stochastic subgradient algorithms for convex optimzation," *SIAM J. on Optim.*, vol. 20, no. 2, pp. 691 – 717, 2009.

[8] A. Nedić, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE TAC*, vol. 55, no. 4, pp. 922 – 938, 2010.

[9] L. Xiao, M. Johansson, and S. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Trans. on Comm.*, vol. 52, no. 7, pp. 1136 – 1144, 2004.

[10] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links - part I: Distributed estimation of deterministic signals," *IEEE Trans. on Sig. Proc.*, vol. 56, pp. 350 – 364, 2008.

[11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," Stanford Statistics Dept., Tech. Rep., 2010.

[12] C. Fischione, "F-Lipschitz optimization with Wireless Sensor Networks applications," *IEEE TAC*, vol. to appear, pp. –, 2011.

[13] C. Fischione and U. Jönsson, "Fast-Lipschitz optimization with Wireless Sensor Networks applications," in *IPSN*, 2011.

[14] J. Van Ast, R. Babška, and B. D. Schutter, "Particle swarms in optimization and control," in *IFAC World Congress*, 2008.

[15] E. Alba and J. M. Troya, "A survey of parallel distributed genetic algorithms," *Complexity*, vol. 4, no. 4, pp. 31 – 52, 1999.

[16] F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato, "Newton-Raphson consensus for distributed convex optimization," in *IEEE Conference on Decision and Control*, 2011.

[17] F. Garin and L. Schenato, *Networked Control Systems*. Springer, 2011, ch. A survey on distributed estimation and control applications using linear consensus algorithms, pp. 75–107.

[18] P. Kokotović, H. K. Khalil, and J. O'Reilly, *Singular Perturbation Methods in Control: Analysis and Design*, ser. Classics in applied mathematics. SIAM, 1999, no. 25.

[19] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2001.

[20] Y. C. Ho, L. Servi, and R. Suri, "A class of center-free resource allocation algorithms," *Large Scale Systems*, vol. 1, pp. 51 – 62, 1980.

[21] L. Xiao and S. Boyd, "Optimal scaling of a gradient method for distributed resource allocation," *J. Opt. Theory and Applications*, vol. 129, no. 3, pp. 469 – 488, 2006.

[22] K. Tanabe, "Global analysis of continuous analogues of the Levenberg-Marquardt and Newton-Raphson methods for solving nonlinear equations," *Inst. of Stat. Math.*, vol. 37, pp. 189–203, 1985.

[23] R. Hauser and J. Nedić, "The continuous Newton-Raphson method can look ahead," *SIAM J. on Opt.*, vol. 15, pp. 915 – 925, 2005.

[24] S. Athuraliya and S. H. Low, "Optimization flow control with newton-like algorithm," *Telecommunication Systems*, vol. 15, no. 3-4, pp. 345–358, 2000.

[25] M. Zargham, A. Ribeiro, A. Ozdaglar, and A. Jadbabaie, "Accelerated dual descent for network optimization," in *ACC*, 2011.

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

[27] S. Becker and Y. L. Cun, "Improving the convergence of back-propagation learning with second order models," University of Toronto, Tech. Rep. CRG-TR-88-5, September 1988.

[28] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996, sec. 4.2.