Distributed second order methods with variable number of working nodes

Dragana Bajović *† Dušan Jakovetić ‡ Nataša Krejić ‡
Nataša Krklec Jerinkić ‡

September 6, 2017

Abstract

Recently, an idling mechanism has been introduced in the context of distributed *first order* methods for minimization of a sum of nodes' local convex costs over a generic, connected network. With the idling mechanism, each node i, at each iteration k, is active – updates its solution estimate and exchanges messages with its network neighborhood – with probability p_k (p_k increasing to one as k grows large), and it stays idle with probability $1-p_k$, while the activations are independent both across nodes and across iterations. The idling mechanism involves an increasing number of nodes in the algorithm (on average) as the iteration counter k grows, thus avoiding unnecessarily expensive exact updates at the initial iterations while performing beneficial close-to-exact updates near the solution. In this paper, we demonstrate that the idling mechanism can be successfully incorporated in *distributed second order methods* also. Specifically, we apply the idling mechanism to the recently proposed Distributed Quasi Newton method (DQN). We first show theoretically that DQN with idling exhibits very similar theoretical convergence and convergence rates properties as the standard DQN method, thus achieving the same order of convergence rate (R-linear) as the standard DQN, but with significantly cheaper updates. Further, we demonstrate by simulation examples significant communication and computational savings gained through incorporation of the idling mechanism.

Keywords: Distributed optimization, Variable sample schemes, Second order methods, Newton-like methods, Linear convergence, Stochastic convergence.

AMS subject classification.

*Department of Power, Electronics and Communication Engineering, Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia. email: dbajovic@uns.ac.rs.

†Biosense Institute, University of Novi Sad, Ul. Zorana Djindjića 3, 21000 Novi Sad, Serbia.

‡Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia. e-mail: {djakovet@uns.ac.rs, natasak@uns.ac.rs, natasa.krklec@dmi.uns.ac.rs}. Research supported by the Serbian Ministry of Education, Science, and Technological Development, Grant no. 174030

1 Introduction

Context and motivation. The problem of distributed minimization of a sum of nodes' local costs across a (connected) network has received a significant and growing interest in the past decade, e.g., [17, 10, 20, 14]. Such problem arises in various application domains, including wireless sensor networks, e.g., [23], smart grid, e.g., [1], distributed control applications, e.g., [2], etc.

In the recent paper [4], a class of novel distributed *first order methods* has been proposed, motivated by the so-termed "hybrid" methods in [6] for (centralized) minimization of a sum $\sum_{i=1}^{n} f_i(x)$ of convex component functions. The main idea underlying a hybrid method in [6] is that it is designed as a combination of 1) an incremental or stochastic gradient method (or, more generally, an incremental or stochastic Newton-like method) and 2) a full (standard) gradient method (or a standard Newton-like method); it behaves as the stochastic method at the initial algorithm stage, and as the standard method at a later stage. An advantage of the hybrid method is that it potentially inherits some favorable properties of both incremental/stochastic and standard methods, while eliminating their important drawbacks. For example, the hybrid exhibits fast convergence at initial iterations k while having inexpensive updates, just like incremental/stochastic methods. On the other hand, it eliminates the oscillatory behavior of incremental methods around the solution for large k's (because for large k's it behaves as a full/standard method). Hybrid methods calculate a search direction at iteration k based on a subset (sample) of the f_i's, where the sample size is small at the initial iterations (mimicking a stochastic/incremental method), while it approaches the full sample n for large k's (essentially matching a full, standard method).

With distributed first order methods in [4], the sample size at iteration k translates into the number of nodes that participate in the distributed algorithm at k. More precisely, therein we introduce an idling mechanism where each node in the network at iteration k is active with probability p_k and stays idle with probability $1-p_k$, where p_k is increasing to one with k, while the activations are independent both across nodes and through iterations. Reference [4] analyzes convergence rates for a distributed gradient method with the idling mechanism and demonstrates by simulation that idling brings significant communication and computational savings.

Contributions. The purpose of this paper is to demonstrate that the idling mechanism can be incorporated in distributed *second order, i.e., Newton-like* methods also, by both 1) establishing the corresponding convergence rate analytical results, and 2) showing through simulation examples that idling continues to bring significant efficiency improvements. Specifically, we incorporate here the idling mechanism in the Distributed Quasi Newton method (DQN) [3]. (The DQN method has been proposed and analyzed in [3], only for the scenario when all nodes are active at all times.) DQN and its extension PMM-DQN are representative distributed second order methods that exhibit competitive performance with respect to the current distributed second order alternatives, e.g., [14, 22, 15].

Our main results are as follows. Assuming that the f_i's are twice continuously differentiable with bounded Hessians, we show that, as long as p_k converges to one at least as fast as $1 - 1/k^{1+\zeta}$ ($\zeta > 0$ arbitrarily small), the DQN method with idling converges in the mean square sense and almost surely to the same point as the standard DQN method that activates all nodes at all times. Furthermore, when p_k converges to one at a geometric rate, then the DQN algorithm with idling converges to its limit at a R-linear rate in the mean square sense. Finally, simulation examples demonstrate that idling can bring to DQN significant improvements in computational and communication efficiencies.

From the technical side, extending the analysis of either distributed gradient methods with idling [4] or DQN without idling [3] to the scenario considered here is highly nontrivial. With respect to the standard DQN (without idling), here we need to cope with *inexact variants* of the DQN-type second order search directions. With respect to gradient methods with idling, showing boundedness of the sequence of iterates and consequently bounding the "inexactness amounts" of the search directions are considerably more challenging and require a different approach.

Brief literature review. There has been a significant progress in the development of distributed second order methods in past few years. Reference [14] proposes a method based on a penalty-like interpretation [11] of the problem of interest, by applying Taylor expansions of the Hessian of the involved penalty function. In [22], the authors develop a distributed version of the Newton-Raphson algorithm based on consensus and time-scale separation principles. References [15] and [16] propose distributed second order methods based on the alternating direction method of multipliers and the proximal method of multipliers, respectively. References [24, 21] develop distributed second order methods for the problem formulations that are related but different than what we consider in this paper, namely they study network utility maximization type problems. All the above works [14, 22, 3, 15, 16, 24, 21] assume that all nodes are active across all iterations, i.e., they are not concerned with designing nor analyzing stochastic nor asynchronous variants of the methods therein.

Related with our work are also papers that study distributed first and second order methods with either 1) asynchronous updates or 2) random/time varying networks. The former thread of works, e.g., [9, 8, 7], considers asynchronous distributed schemes where, like with the idling-based methods here, a subset of nodes is active at each iteration. However, in contrast with this paper, the subsets with their methods do not eventually increase towards the full set of network nodes. The latter thread of works on distributed consensus and optimization under time varying or random networks, e.g., [17, 19, 13], considers "sparsified" inter-neighbor communications across the network, as we do here. However, differently from [17, 19, 13], our model also allows that the nodes' computation is "sparsified," in the sense that a subset of nodes stays completely idle at an iteration.

Finally, in a companion paper [5], we presented a brief preliminary version of the current paper, wherein a subset of the results here are presented without proofs. Specifically, [5] considers convergence of DQN with idling only when

the activation probabilities geometrically converge to one, while here we also consider the scenarios where the activation probability converges to one sublinearly; we also include here extensions where this parameter stays bounded away from one.

A feature of the standard DQN method (and hence also of DQN with idling) and of related methods [14] is that their convergence point is not the exact global solution $\overline{x}^*$ of the problem of interest, but it is a point in a neighborhood of $\overline{x}^*$. This drawback has been recently successfully eliminated in [16] to make the algorithm convergent exactly to $\overline{x}^*$, and the methodology of [16] is subsequently applied in [3] to derive an exactly-converging extension of DQN. It is certainly of interest to incorporate and analyze idling in the context of such exact second order distributed methods, and this is an interesting future research direction. Nonetheless, the analysis of idling with DQN – carried out in this paper – is a highly nontrivial and relevant first phase towards this goal.

Paper organization. Section 2 describes the model that we assume and gives the necessary preliminaries. Section 3 presents the DQN algorithm with idling, while Section 4 analyzes its convergence and convergence rate. Section 5 considers DQN with idling in the presence of persisting idling, i.e., it considers the extension when p_k does not necessarily converge to one. Section 6 provides numerical examples. Finally, we conclude in Section 7.

2 Model and preliminaries

Subsection 2.1 gives preliminaries and explains the network and optimization models that we assume. Subsection 2.2 briefly reviews the DQN algorithm proposed in [3].

2.1 Optimization–network model

We consider distributed optimization where n nodes in a connected network solve the following unconstrained problem:

$$\min_{x\in\mathbb{R}^p} \; f(x) = \sum_{i=1}^{n} f_i(x). \tag{1}$$

Here, $f_i : \mathbb{R}^p \to \mathbb{R}$ is a convex function known only by node i. We impose the following assumptions on the f_i's.

Assumption A1. Each function $f_i : \mathbb{R}^p \to \mathbb{R}$, $i = 1, \ldots, n$ is twice continuously differentiable, and there exist constants $0 < \mu \leq L < \infty$ such that for every $x \in \mathbb{R}^p$

$$\mu I \preceq \nabla^2 f_i(x) \preceq LI.$$

Here, I denotes the $p \times p$ identity matrix, and $P \preceq Q$ (where P and Q are symmetric matrices) means that $Q - P$ is positive semidefinite. Assumption A1 implies that each f_i is strongly convex with strong convexity parameter μ, and

it also has Lipschitz continuous gradient with Lipschitz constant L, i.e., for all $i = 1, ..., n$, there holds:

$$f_i(y) \geq f_i(x) + \nabla f_i(x)^T(y-x) + \frac{\mu}{2}\|x-y\|^2,\ x, y \in \mathbb{R}^p$$
$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\,\|x-y\|,\ x, y \in \mathbb{R}^p.$$

Here, notation $\|\cdot\|$ stands for the Euclidean norm of its vector argument and the spectral norm of its matrix argument. Under Assumption A1, problem (1) is solvable and has the unique solution $\overline{x}^* \in \mathbb{R}^p$.

Nodes $i = 1, ..., n$ constitute an undirected network $\mathcal{G} = (\mathcal{V}, E)$, where $\mathcal{V}$ is the set of nodes and E is the set of edges. Denote by m the total number of (undirected) edges (the cardinality of E). The presence of edge $\{i, j\} \in E$ means that the nodes i and j can directly exchange messages through a communication link. Further, let O_i be the set of all neighbors of a node i (excluding i), and define also $\bar{O}_i = O_i \bigcup \{i\}$.

Assumption A2. The network $\mathcal{G} = (\mathcal{V}, E)$ is connected, undirected and simple (no self-loops nor multiple links).

We associate with network $\mathcal{G}$ a $n \times n$ weight matrix that has the following properties.

Assumption A3. Matrix $W = W^T \in \mathbb{R}^{n\times n}$ is stochastic, with elements w_{ij} such that

$$w_{ij} > 0 \text{ if } \{i,j\} \in E,\ w_{ij} = 0 \text{ if } \{i,j\} \notin E,\ i \neq j, \text{ and } w_{ii} = 1 - \sum_{j\in O_i} w_{ij};$$

further, there exist constants w_{min} and w_{max} such that for $i = 1, \ldots, n$, there holds:

$$0 < w_{min} \leq w_{ii} \leq w_{max} < 1.$$

Denote by $\lambda_1 \geq \ldots \geq \lambda_n$ the eigenvalues of W. Then, we have that $\lambda_1 = 1$, all the remaining eigenvalues of W are strictly less than one in modulus, and the eigenvector that corresponds to the unit eigenvalue is $e := \frac{1}{\sqrt{n}}(1, \ldots, 1)^T$.

Let $x = (x_1^T, \ldots, x_n^T)^T \in \mathbb{R}^{np}$, with $x_i \in \mathbb{R}^p$, and denote by $\mathbb{Z} \in \mathbb{R}^{np\times np}$ the Kronecker product of W and the identity $I \in \mathbf{R}^{p\times p}$, $\mathbb{Z} = W \otimes I$.[1] We will make use of the following penalty reformulation of (1) [11]:

$$\min_{x\in\mathbb{R}^{np}} \Phi(x) := \alpha \sum_{i=1}^{n} f_i(x_i) + \frac{1}{2} x^T(\mathbb{I} - \mathbb{Z})x. \tag{2}$$

Here, $\alpha > 0$ is a constant that, as we will see ahead, plays the role of a step size with the distributed algorithms that we consider. The rationale behind introducing problem (2) and function Φ is that they enable one to interpret the distributed first order method in [17] to solve (1) as the (ordinary) gradient method applied on Φ, which in turn facilitates the development of second order

[1] Throughout, we shall use "blackboard bold" upper-case letters for matrices of size $(np) \times (np)$ (e.g., $\mathbb{Z}$), and standard upper-case letters for matrices of size $n \times n$ or $p \times p$ (e.g., W).

methods; see, e.g., [14], for details. Denote by $x^* = \left((x_1^*)^T, ..., (x_n^*)^T\right)^T \in \mathbb{R}^{np}$ the solution to (2), where $x_i^* \in \mathbb{R}^p$, for all $i = 1, ..., n$. It can be shown that, for all $i = 1, ..., n$, $\|x_i^* - \overline{x}^*\| = O(\alpha)$, i.e., the distance from the desired solution $\overline{x}^*$ of (1) and x_i^* is of the order of step size α, e.g., [14]. Define also $F : \mathbb{R}^{np} \to \mathbb{R}$, $F(x) = \sum_{i=1}^n f_i(x_i)$.

We study distributed second order algorithms to minimize function Φ (and hence, find a near optimal solution of (1)). Therein, the Hessian of function Φ and its splitting into a diagonal and an off-diagonal part will play an important role. Specifically, first consider the splitting:

$$\begin{aligned} W_d &= diag(W), \quad W_u = W - W_d \\ \mathbb{Z} &= \mathbb{Z}(W) = \mathbb{Z}_d + \mathbb{Z}_u, \text{ where} \\ \mathbb{Z}_d &= W_d \otimes I = diag(\mathbb{Z}), \text{ and } \mathbb{Z}_u = W_u \otimes I. \end{aligned} \tag{3}$$

(Here, $diag(P)$ is the diagonal matrix with the diagonal elements equal to those of matrix P.)

Further, decompose the Hessian of Φ as:

$$\nabla^2 \Phi(x) = \mathbb{A}(x) - \mathbb{G}, \tag{4}$$

with $\mathbb{A} : \mathbb{R}^{np} \to \mathbb{R}^{(np)\times(np)}$ given by:

$$\mathbb{A}(x) = \alpha \nabla^2 F(x) + (1 + \theta)(\mathbb{I} - \mathbb{Z}_d), \tag{5}$$

and

$$\mathbb{G} = \mathbb{G}(\mathbb{Z}, \theta) = \mathbb{Z}_u + \theta(\mathbb{I} - \mathbb{Z}_d), \tag{6}$$

for some $\theta \geq 0$.

We close this subsection with the following result that will be needed in subsequent analysis. For claim (a), see Lemma 3.1 in [18]; for claim (b), see, e.g., Lemma 4.2 in [12].

Lemma 2.1. *Consider a deterministic sequence $\{a_k\}$ converging to zero, with $a_k > 0$, $k = 0, 1, ...$, and let $\nu \in (0, 1)$.*

(a) Then, there holds:

$$\sum_{t=1}^{k} \nu^{k-t} a_{t-1} \to 0 \text{ as } k \to \infty. \tag{7}$$

(b) If, moreover, $\{a_k\}$ converges to zero R-linearly, then the sum in (7) also converges to zero R-linearly.

2.2 Algorithm DQN

We will incorporate the idling mechanism in the algorithm DQN proposed in [3]. The main idea behind DQN is to approximate the Newton direction with respect

to function Φ in (2) in such a way that distributed implementation is possible, while the error in approximating the Newton direction is not large. For completeness, we now briefly review DQN. All nodes are assumed to be synchronized according to a global clock and perform in parallel iterations $k = 0, 1, ...$ The algorithm maintains iterates $x^k = ((x_1^k)^T, ..., (x_n^k)^T)^T \in \mathbb{R}^{np}$, over iterations $k = 0, 1, ...$, where $x_i^k \in \mathbb{R}^p$ plays the role of the solution estimate of node i, $i = 1, ..., n$. DQN is presented in Algorithm 1 below. Therein, $\mathbb{A}_k := \mathbb{A}(x^k)$, where $\mathbb{A}(x)$ is given in (5); also, notation $\|\cdot\|$ stands for the Euclidean norm of its vector argument and spectral norm for its matrix argument.

Algorithm 1: DQN in vector format
Given $x^0 \in \mathbb{R}^{np}, \varepsilon, \rho, \theta, \alpha > 0$. Set $k = 0$.

(1) Chose a diagonal matrix $\mathbb{L}_k \in \mathbb{R}^{np \times np}$ such that

$$\|\mathbb{L}_k\| \leq \rho. \tag{8}$$

(2) Set

$$s^k = -(\mathbb{I} - \mathbb{L}_k \mathbb{G}) \mathbb{A}_k^{-1} \nabla \Phi(x^k). \tag{9}$$

(3) Set

$$x^{k+1} = x^k + \varepsilon s^k, \; k = k + 1. \tag{10}$$

We now briefly comment on the algorithm and the involved parameters. DQN takes the step in the direction s^k scaled with a positive step size $\varepsilon > 0$; direction s^k in (9) is an approximation of the Newton direction

$$s_{\mathrm{N}}^k = -\left[\nabla^2 \Phi(x^k)\right]^{-1} \nabla \Phi(x^k).$$

Unlike Newton direction s_{N}^k, direction s^k admits an efficient distributed implementation. Inequality (8) corresponds to a safeguarding step that is needed to ensure that s^k is a descent direction with respect to function Φ; the nonnegative parameter ρ controls the safeguarding (see [3] for details on how to set ρ for a given problem). The diagonal matrix $\mathbb{L}_k$ controls the off-diagonal part of the Hessian inverse approximation [3]. Various choices for $\mathbb{L}_k$ that are easy to implement and do not induce large extra computational and communication costs have been introduced in [3]. Possible and easy-to-implement choices are $\mathbb{L}_k = 0$ and $\mathbb{L}_k = -\rho\,\mathbb{I}$.

As it is usually the case with second order methods, step size ε should be in general strictly smaller than one to ensure global convergence. However, extensive numerical simulations on quadratic and logistic losses demonstrate that both DQN and DQN with idling converge globally with the full step size $\varepsilon = 1$ and choices $\mathbb{L}_k = 0$, and $\mathbb{L}_k = -\mathbb{I}$.

The remaining algorithm parameters are as follows. Quantity $\theta \geq 0$ controls the splitting (5); reference [3] shows by simulation that it is usually beneficial to adopt a small positive value of θ or $\theta = 0$. Finally, $\alpha > 0$ defines the penalty function Φ in (2) and results in the following tradeoff in the performance of DQN:

a smaller value of α leads to a better asymptotic accuracy of the algorithm, while it also slows down the algorithm's convergence rate. By asymptotic accuracy, we assume here the distance between the point of convergence of DQN x^* (which is actually equal to the solution of (2) – see [3]) and the solution $\overline{x}^*$ of (1). (As noted before, the corresponding distance is $O(\alpha)$ [14].)

In Algorithm 2, we present DQN from the perspective of distributed implementation. Therein, we denote by Λ_i^k and A_i^k, respectively, the $p \times p$ block on the (i,i)-th position of matrices $\mathbb{L}_k$ and $\mathbb{A}_k$. Similarly, G_{ij} is the $p \times p$ block at the (i,j)-th position of $\mathbb{G}$.

Algorithm 2: DQN – distributed implementation
At each node i, require $x_i^0 \in \mathbb{R}^p, \varepsilon, \rho, \theta, \alpha > 0$.

(1) Initialization: Each node i sets $k = 0$ and $x_i^0 \in \mathbb{R}^p$.

(2) Each node i transmits x_i^k to all its neighbors $j \in O_i$ and receives x_j^k from all $j \in O_i$.

(3) Each node i calculates

$$d_i^k = \left(A_i^k\right)^{-1} \left[\alpha \nabla f_i(x_i^k) + \sum_{j \in O_i} w_{ij} \left(x_i^k - x_j^k\right) \right].$$

(4) Each node i transmits d_i^k to all its neighbors $j \in O_i$ and receives d_j^k from all $j \in O_i$.

(5) Each node i chooses a diagonal $p \times p$ matrix Λ_i^k, such that $\|\Lambda_i^k\| \leq \rho$.

(6) Each node i calculates:

$$s_i^k = -d_i^k + \Lambda_i^k \sum_{j \in \bar{O}_i} G_{ij}\, d_j^k.$$

(7) Each node i updates its solution estimate as:

$$x_i^{k+1} = x_i^k + \varepsilon\, s_i^k.$$

(8) Set $k = k + 1$ and go to step 2.

Note that, when $\Lambda_i^k \equiv 0$, for all i, k, steps 4-6 are skipped, and the algorithm involves a single communication round per iteration, i.e., a single transmission of a p-dimensional vector (step 2) by each node; when $\Lambda_i^k \equiv -\rho_k\, I$, $\rho_k \neq 0$, then two communication rounds (steps 2 and 4) per k are involved.

3 Algorithm DQN with idling

Subsection 3.1 explains the idling mechanism, while Subsection 3.2 incorporates this mechanism in the DQN method.

3.1 Idling mechanism

We incorporate in DQN the following idling mechanism. Each node i, at each iteration k, is active with probability p_k, and it is inactive with probability $1-p_k$.[2] Active nodes perform updates of their solution estimates (x_i^k)'s and participate in each communication round of an iteration, while inactive nodes do not perform any computations nor communications, i.e., their solution estimates (x_i^k)'s remain unchanged. Denote by ξ_i^k the Bernoulli random variable that governs the activity of node i at iteration k. Then, we have that probability $P\left(\xi_i^k = 1\right) = 1 - P\left(\xi_i^k = 0\right) = p_k$, for all i. We furthermore assume that ξ_i^k and ξ_j^ℓ are mutually independent over all $i \neq j$ and $k \neq \ell$. Throughout the paper, we impose the following Assumption on sequence $\{p_k\}$.

Assumption A4. Consider the sequence of activation probabilities $\{p_k\}$. We assume that $p_k \geq p_{min}$, for all k, for some $p_{min} > 0$. Further, $\{p_k\}$ is a non-decreasing sequence with $\lim_{k\to\infty} p_k = 1$. Moreover, we assume that

$$0 \leq u_k \leq \frac{\mathcal{C}_{\rm u}}{(k+1)^{1+\zeta}}, \tag{11}$$

where $u_k = 1 - p_k$, $\mathcal{C}_{\rm u}$ is a positive constant and $\zeta > 0$ is arbitrarily small.

Assumption A4 means that, on average, an increasing number of nodes becomes involved in the optimization process; that is, intuitively, in a sense the precision of the optimization process increases with the increase of the iteration counter k. (Extensions to the scenarios when p_k does not necessarily converge to one is provided in Section 5.) We also assume that p_k converges to one sufficiently fast, where sublinear convergence $1 - 1/k^{1+\zeta}$ is sufficient.

For future reference, we also define the diagonal $(np)\times(np)$ (random) matrix $\mathbb{Y}_k = diag(\xi_1^k, \ldots, \xi_n^k) \otimes I$, where I is the $p \times p$ identity matrix. Also, we define the $p \times p$ random matrix $W^k = [w_{ij}^k]$ by $w_{ij}^k = w_{ij}\xi_i^k\xi_j^k$ for $i \neq j$, and $w_{ii}^k = 1 - \sum_{i\neq j} w_{ij}^k$. Further, we let $\mathbb{Z}^k := W^k \otimes I$, and, analogously to (3) and (6), we let:

$$\begin{aligned} W_d^k &= diag(W^k), \;\; W_u^k = W^k - W_d^k & (12)\\ \mathbb{Z}^k &= \mathbb{Z}(W^k) = \mathbb{Z}_d^k + \mathbb{Z}_u^k, \text{ where} \\ \mathbb{Z}_d^k &= W_d^k \otimes I = diag(\mathbb{Z}^k), \text{ and } \mathbb{Z}_u^k = W_u^k \otimes I. \\ \mathbb{G}^k &= \mathbb{G}(\mathbb{Z}^k, \theta) = \mathbb{Z}_u^k + \theta(\mathbb{I} - \mathbb{Z}_d^k). & (13) \end{aligned}$$

Notice that $w_{ii}^k = 1 - \sum_{i\neq j} w_{ij}\xi_i^k\xi_j^k \geq 1 - \sum_{i\neq j} w_{ij} = w_{ii} \geq w_{min}$. Further, recall $\mathbb{A} : \mathbb{R}^{np} \to \mathbb{R}^{(np)\times(np)}$ in (5). Using results from [3], we obtain the

[2] We continue to assume that all nodes are synchronized according to a global iteration counter $k = 0, 1, ...$

following important bounds:[3]

$$\begin{aligned} \|\mathbb{A}^{-1}(x)\| &\leq (\alpha\mu + (1+\theta)(1-w_{max}))^{-1} := \mathcal{C}_{\mathrm{A}},\ x \in \mathbb{R}^{np} && (14)\\ \|\mathbb{G}^k\| &\leq (1+\theta)(1-w_{min}) := \mathcal{C}_{\mathrm{G}},\ k = 0,1,... && (15)\\ \left\|\mathbb{G}^k\mathbb{A}^{-1}(x)\right\| &\leq \frac{(1+\theta)(1-w_{min})}{\alpha\mu + (1+\theta)(1-w_{min})},\ x \in \mathbb{R}^{np},\ k = 0,1,... && (16) \end{aligned}$$

More generally, for $\mathbb{A}(x)$ there holds:

$$(\alpha\mu + (1+\theta)(1-w_{max}))\mathbb{I} \preceq \mathbb{A}(x) \preceq (\alpha L + (1+\theta)(1-w_{min}))\mathbb{I},\ x \in \mathbb{R}^{np}. \quad (17)$$

Also, notice that $\|\mathbb{Y}_k\| \leq 1$, and $\mathbb{Z}^k \preceq \mathbb{I}$, for every k.

3.2 DQN with idling

We now incorporate the idling mechanism in the DQN method. To avoid notational clutter, we continue to denote by $x^k = \left((x_1^k)^T, ..., (x_n^k)^T\right)^T$ the algorithm iterates, $k = 0, 1, ...$, where x_i^k is node i's estimate of the solution to (1) at iteration k. DQN with idling operates as follows. If the activation variable $\xi_i^k = 1$, node i performs an update; else, if $\xi_i^k = 0$, node i stays idle and lets $x_i^{k+1} = x_i^k$. The algorithm is presented in Algorithm 3 below. Therein, $\mathbb{A}_k := \mathbb{A}(x^k)$, and A_i^k is the $p \times p$ block at the (i,i)-th position in $\mathbb{A}_k$, while G_{ij}^k is the $p \times p$ block of $\mathbb{G}^k$ at the (i,j)-th position.

Algorithm 3: DQN with idling – distributed implementation
At each node i, require $x_i^0 \in \mathbb{R}^p, \varepsilon, \rho, \theta, \alpha > 0, \{p_k\}$.

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

(2) Each node i generates ξ_i^k; if $\xi_i^k = 0$, node i is idle, and goes to step 9; else, if $\xi_i^k = 1$, node i is active and goes to step 3; all active nodes do steps 3-8 below in parallel.

(3) (Active) node i transmits x_i^k to all its active neighbors $j \in O_i$ and receives x_j^k from all active $j \in O_i$.

(4) Node i calculates

$$d_i^k = \left(A_i^k\right)^{-1} \left[\frac{\alpha}{p_k} \nabla f_i(x_i^k) + \sum_{j \in O_i} w_{ij}^k \left(x_i^k - x_j^k\right) \right].$$

[3]Throughout subsequent analysis, we shall state several relations (equalities and inequalities) that involve random variables. These relations hold either surely (for every outcome), or in expectation. E.g., relation (15) holds surely. It is clear from notation which of the two cases is in force. Also, auxiliary constants that arise from the analysis will be frequently denoted by the capital calligraphic letter $\mathcal{C}$ with a subscript that indicates a quantity related with the constant in question; e.g., see $\mathcal{C}_{\mathrm{A}}$ in (14).

(5) Node i transmits d_i^k to all its active neighbors $j \in O_i$ and receives d_j^k from all the active $j \in O_i$.

(6) Node i chooses a diagonal $p \times p$ matrix Λ_i^k, such that $\|\Lambda_i^k\| \leq \rho$.

(7) Node i calculates:

$$s_i^k = -d_i^k + \Lambda_i^k \sum_{j \in \bar{O}_i} G_{ij}^k \, d_j^k.$$

(8) Node i updates its solution estimate as:

$$x_i^{k+1} = x_i^k + \varepsilon \, s_i^k.$$

(9) Set $k = k + 1$ and go to step 2.

We make a few remarks on Algorithm 3. First, note that, unlike Algorithm 2, the iterates x_i^k with Algorithm 3 are random variables. (The initial iterates x_i^0, $i = 1, ..., n$, in Algorithm 3 are assumed deterministic.) Next, note that we implicitly assume that all nodes have agreed beforehand on scalar parameters $\varepsilon, \rho, \theta, \alpha$; this can actually be achieved in a distributed way with a low communication and computational overhead (see Subsection 4.2 in [3]). Nodes also agree beforehand on the sequence of activation probabilities $\{p_k\}$. In other words, sequence $\{p_k\}$ is assumed to be available at all nodes. For example, as discussed in more detail in Sections 4 and 6, we can let $p_k = 1 - \sigma^{k+1}$, $k = 0, 1, ...$, where $\sigma \in (0, 1)$ is a scalar parameter known by all nodes. As each node is aware of the global iteration counter k, each node is then able to implement the latter formula for p_k. The nodes' beforehand agreement on σ can be achieved similarly to the agreement on other parameters $\varepsilon, \rho, \theta, \alpha$ [3]. Tuning of parameter σ is discussed in Remark 3 further ahead.

Parameters ϵ, ρ, and α, and the diagonal matrices Λ_i^k play the same role as in DQN. An important difference with respect to standard DQN appears in step 4, where the local *active* node i's gradient contribution is $\frac{\alpha}{p_k} \nabla f_i(x_i^k) = \frac{\alpha}{p_k} \xi_i^k \nabla f_i(x_i^k)$, while with standard DQN this contribution equals $\alpha \nabla f_i(x_i^k)$. Note that the division by p_k for DQN with idling makes the terms of the two algorithms balanced *on average*, because $E[\xi_i^k] = p_k$.

Using notation (12), we represent DQN with idling in Algorithm 4 in a vector format. (Therein, $\mathbb{L}_k = diag\left(\Lambda_1^k, ..., \Lambda_n^k\right)$.)

Algorithm 4: DQN with idling in vector format
Given $x^0 \in \mathbb{R}^{np}, \varepsilon, \rho, \theta, \alpha > 0, \{p_k\}$. Set $k = 0$.

(1) Chose a diagonal matrix $\mathbb{L}_k \in \mathbb{R}^{np \times np}$ such that

$$\|\mathbb{L}_k\| \leq \rho.$$

(2) Set

$$s^k = -(\mathbb{I} - \mathbb{L}_k \mathbb{G}^k) \, \mathbb{A}_k^{-1} \left(\frac{\alpha}{p_k} \mathbb{Y}_k \nabla F(x^k) + (\mathbb{I} - \mathbb{Z}^k) \, x^k \right).$$

(3) Set

$$x^{k+1} = x^k + \varepsilon s^k,\ k = k+1.$$

4 Convergence analysis

In this Section, we carry out convergence and convergence rate analysis of DQN with idling. We have two main results, Theorems 4.4 and 4.5. The former result states that, under Assumptions A1-A4, DQN with idling converges to the solution x^* of (2) in the mean square sense and almost surely. We then show that, when activation probability p_k converges to one at a geometric rate, the mean square convergence towards x^* occurs at a R-linear rate. Therefore, the order of convergence (R-linear rate) of the DQN method is preserved despite the idling. We note that the above result does not explicitly establish computational and communication savings with respect to standard DQN. An explicit quantification of these savings is very challenging even for distributed first order methods [4], and even more so here. However, the theoretical results in the current section are complemented in Section 6 with numerical examples; they demonstrate that communication and computational savings usually occur in practice.

The analysis is organized as follows. In Subsection 4.1, we relate the search direction of DQN with idling and the search direction of DQN, where the former is viewed as an inexact version of the latter. Subsection 4.2 establishes the mean square boundedness of the iterates of DQN with idling and its implications on the "inexactness" of search directions. Finally, Subsection 4.3 makes use of the results in Subsections 4.1 and 4.2 to prove the main results on the convergence and convergence rate of DQN with idling.

4.1 Quantifying inexactness of search directions

We now analyze how "inexact" is the search direction of the DQN with idling with respect to the search direction of the standard DQN. For x^k the iterate of DQN with idling, denote by $\hat{s}^k$ the search direction as with the standard DQN evaluated at x^k, i.e.:

$$\hat{s}^k = -(\mathbb{I} - \mathbb{L}_k\mathbb{G})\mathbb{A}_k^{-1}(\alpha\nabla F(x^k) + (\mathbb{I} - \mathbb{Z})x^k). \tag{18}$$

Then, the search direction s^k of DQN with idling can be viewed as an approximation, i.e., an inexact version, of $\hat{s}^k$. We will show ahead that the error of this approximation is controlled by the activation probability p_k. In order to simplify notation in the analysis, we introduce the following quantities:

$$\begin{aligned}
\mathbb{H}^k &= \mathbb{I} - \mathbb{L}_k\mathbb{G}^k \\
\widehat{\mathbb{H}}^k &= \mathbb{I} - \mathbb{L}_k\mathbb{G} \\
g^k &= \mathbb{A}_k^{-1}(\frac{\alpha}{p_k}\mathbb{Y}_k\nabla F(x^k) + (\mathbb{I} - \mathbb{Z}^k)x^k) \\
\hat{g}^k &= \mathbb{A}_k^{-1}(\alpha\nabla F(x^k) + (\mathbb{I} - \mathbb{Z})x^k).
\end{aligned}$$

Therefore, $\hat{s}^k = -\widehat{\mathbb{H}}^k \hat{g}^k$ and $s^k = -\mathbb{H}^k g^k$. Notice that $\|\mathbb{H}^k\| \leq 1 + \rho \mathcal{C}_{\mathrm{G}} := \mathcal{C}_{\mathrm{H}}$ and that the same is true for $\widehat{\mathbb{H}}^k$. We have the following result on the error in approximating $\hat{s}^k$ with s^k. In the following, we denote by either a_i or $[a]_i$ the i-th $p \times 1$ block of a (np)-dimensional vector a; for example, we write g_i^k for the i-th p-dimensional block of g^k.

In the following Theorem, claims (20) and (21) are from Theorem 3.2 in [3]; claim (19) is a straightforward generalization of (20), mimicking the proof steps of Theorem 3.2 in [3]; hence, proof details are omitted.

Theorem 4.1. *[3] Let assumptions A1-A3 hold. Further, let, $\rho \in [0, \overline{\rho}_{DQN}]$ where*

$$\overline{\rho}_{DQN} = \frac{\alpha\mu + (1+\theta)(1-w_{max})}{(1-w_{min})(1+\theta)} \left(\frac{1}{\alpha L + (1+\theta)(1-w_{min})} - \delta \right),$$

for some constant $\delta \in (0, 1/(\alpha L + (1+\theta)(1-w_{min})))$. Consider random matrices $\mathbb{R}_k := \mathbb{H}^k \mathbb{A}_k^{-1}$ and $\widehat{\mathbb{R}}_k := \widehat{\mathbb{H}}^k \mathbb{A}_k^{-1}$. Then, there holds:

$$\lambda_{min} \left(\frac{\mathbb{R}_k + \mathbb{R}_k^T}{2} \right) \geq \delta \tag{19}$$

$$\lambda_{min} \left(\frac{\widehat{\mathbb{R}}_k + \widehat{\mathbb{R}}_k^T}{2} \right) \geq \delta, \tag{20}$$

where $\lambda_{min}(\cdot)$ denotes the minimal eigenvalue. Moreover, quantity $\hat{s}^k$ in (18) satisfies the following bounds:

$$\nabla^T \Phi(x^k) \hat{s}^k \leq -\delta \|\nabla \Phi(x^k)\|^2 \text{ and } \|\hat{s}^k\| \leq \beta \|\nabla \Phi(x^k)\|, \tag{21}$$

where constant $\beta = \frac{1+\rho(1+\theta)(1-w_{min})}{\alpha\mu + (1+\theta)(1-w_{max})}$.

Moreover, it was shown in [3] that $\delta < \beta$, which we use in the proof of the subsequent result. Furthermore, using the Mean value theorem and Lipschitz continuity of $\nabla\Phi$ we obtain (see the proof of Theorem 3.3 in [3] for instance):

$$\Phi\left(x^k + \varepsilon\, s^k\right) - \Phi(x^*) \leq \Phi(x^k) - \Phi(x^*) + \frac{1}{2}\varepsilon^2 L_\Phi \|s^k\|^2 + \varepsilon \nabla^T \Phi(x^k) s^k, \tag{22}$$

where we recall that x^* is the (unique) solution of (2) and L_Φ is a Lipschitz gradient continuity parameter of Φ, which equals $L_\Phi = \alpha L + 2(1 - w_{min})$. Next, we prove that DQN with idling exhibits a kind of nonmonotone behavior where the "nonmonotonicity" term depends on the difference between the search directions s^k and $\hat{s}^k$.

Theorem 4.2. *Let assumptions A1-A3 hold, $\rho \in [0, \overline{\rho}_{DQN}]$ and $\varepsilon \leq \overline{\varepsilon}_{DQN}$, where*

$$\overline{\varepsilon}_{DQN} = \frac{\delta - q}{2L_\Phi(\beta^2 + q^2)}, \quad q \in (0, \delta). \tag{23}$$

Then

$$\Phi(x^{k+1}) - \Phi(x^*) \leq (\Phi(x^k) - \Phi(x^*))\nu(\varepsilon) + e_k,$$

where $\nu(\varepsilon) \in (0,1)$ is a constant, and $e_k = (\varepsilon^2 L_\Phi + \varepsilon/q)\|s^k - \hat{s}^k\|^2$.

Proof. We start by considering s^k in (22) and using the bounds from Theorem 4.1, i.e.:

$$\begin{aligned}\Phi(x^{k+1})-\Phi(x^*) &\leq \Phi(x^k)-\Phi(x^*)+\frac{1}{2}\varepsilon^2 L_\Phi\|s^k\pm\hat{s}^k\|^2+\varepsilon\nabla^T\Phi(x^k)(s^k\pm\hat{s}^k)\\ &\leq \Phi(x^k)-\Phi(x^*)+\varepsilon^2 L_\Phi\|s^k-\hat{s}^k\|^2+\varepsilon^2 L_\Phi\|\hat{s}^k\|^2+\\ &+ \varepsilon\nabla^T\Phi(x^k)(s^k-\hat{s}^k)+\varepsilon\nabla^T\Phi(x^k)\hat{s}^k\\ &\leq \Phi(x^k)-\Phi(x^*)+\varepsilon^2 L_\Phi\|s^k-\hat{s}^k\|^2+\varepsilon^2 L_\Phi\beta^2\|\nabla\Phi(x^k)\|^2+\\ &+ \varepsilon\|\nabla\Phi(x^k)\|\|(s^k-\hat{s}^k)\|-\varepsilon\delta\|\nabla\Phi(x^k)\|^2 \qquad (24)\end{aligned}$$

We distinguish two cases. First, assume that $\|s^k-\hat{s}^k\|\leq q\|\nabla\Phi(x^k)\|$. In this case, (24) implies that

$$\Phi(x^{k+1})-\Phi(x^*)\leq\Phi(x^k)-\Phi(x^*)+\varphi(\varepsilon)\|\nabla\Phi(x^k)\|^2$$

where $\varphi(\varepsilon)=\varepsilon^2 L_\Phi(\beta^2+q^2)-\varepsilon(\delta-q)$. Notice that φ is convex, $\varphi(0)=0$ and it attains its minimum at $\overline{\varepsilon}_{DQN}$ given in (23) with $\varphi(\overline{\varepsilon}_{DQN})=-((\delta-q)^2)/(4L_\Phi(\beta^2+q^2))$. This also implies that $\varphi(\varepsilon)$ is negative for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$. Moreover, Φ is strongly convex and there holds $\Phi(x^k)-\Phi(x^*)\leq\frac{1}{\mu_\Phi}\|\nabla\Phi(x^k)\|^2$ where $\mu_\Phi=\alpha\mu$ is the strong convexity parameter. Putting all together we obtain

$$\Phi(x^{k+1})-\Phi(x^*)\leq(\Phi(x^k)-\Phi(x^*))\nu(\varepsilon) \qquad (25)$$

where $\nu(\varepsilon)=1+\mu_\Phi\varphi(\varepsilon)$. Notice that $\nu(\varepsilon)<1$ for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$. Moreover,

$$\nu(\varepsilon)\geq 1+\mu_\Phi\varphi(\overline{\varepsilon}_{DQN})=1-\frac{\mu_\Phi(\delta-q)^2}{4L_\Phi(\beta^2+q^2)}.$$

As $\mu_\Phi=\alpha\mu<\alpha L<L_\Phi$ and $(\delta-q)^2<\delta^2<\beta^2<\beta^2+q^2$ we conclude that $\nu(\varepsilon)$ is positive. Therefore, for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$, (25) holds with $\nu(\varepsilon)\in(0,1)$.

Now, assume that $\|s^k-\hat{s}^k\|>q\|\nabla\Phi(x^k)\|$, i.e. $\|\nabla\Phi(x^k)\|<\|s^k-\hat{s}^k\|/q$. Together with (24), this implies

$$\Phi(x^{k+1})-\Phi(x^*)\leq\Phi(x^k)-\Phi(x^*)+\hat{\varphi}(\varepsilon)\|\nabla\Phi(x^k)\|^2+e_k \qquad (26)$$

where $\hat{\varphi}(\varepsilon)=\varepsilon^2 L_\Phi\beta^2-\varepsilon\delta$ and $e_k=(\varepsilon^2 L_\Phi+\varepsilon/q)\|s^k-\hat{s}^k\|^2$. The function $\hat{\varphi}$ has similar characteristics as φ and it retains its minimum at $\varepsilon'=\delta/(2L_\Phi\beta^2)$ with $\hat{\varphi}(\varepsilon')=-\delta^2/(4L_\Phi\beta^2)$. Since $\overline{\varepsilon}_{DQN}\leq\varepsilon'$, $\hat{\varphi}(\varepsilon)<0$ holds for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$. Again, strong convexity of Φ and (26) imply that for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$

$$\Phi(x^{k+1})-\Phi(x^*)\leq(\Phi(x^k)-\Phi(x^*))(1+\mu_\Phi\hat{\varphi}(\varepsilon))+e_k\leq(\Phi(x^k)-\Phi(x^*))\nu(\varepsilon)+e_k$$

where we recall $\mu_\Phi=\alpha\mu$, and where the last inequality comes from the fact that $\hat{\varphi}(\varepsilon)\leq\varphi(\varepsilon)$ for every $\varepsilon>0$.

Finally, taking into account both cases and the fact that e_k is nonnegative, we conclude that the following inequality holds with $\nu(\varepsilon)\in(0,1)$ for all $\varepsilon\in(0,\overline{\varepsilon}_{DQN}]$

$$\Phi(x^{k+1})-\Phi(x^*)\leq(\Phi(x^k)-\Phi(x^*))\nu(\varepsilon)+e_k.$$

□

4.2 Mean square boundedness of the iterates and search directions

We next show that the iterates x^k of DQN with idling are uniformly bounded in the mean square sense. Below, $E(\cdot)$ denotes the expectation operator.

Lemma 4.1. *Let the sequence of random variables $\{x^k\}$ be generated by Algorithm 4, and let assumptions A1-A4 hold. Then, there exist positive constants $\overline{\rho}$ and $\overline{\varepsilon}$ depending on $\alpha, \mu, L, \theta, w_{min}, w_{max}$ and p_{min}, such that, for all $\rho \in [0, \overline{\rho}]$ and $\varepsilon \in (0, \overline{\varepsilon}]$, there holds: $E\left(\|x^k\|^2\right) \leq \mathcal{C}_{\mathrm{x}}$, $k = 0, 1, ...$, for some positive constant $\mathcal{C}_{\mathrm{x}}$.*

Proof. It suffices to prove that $E\left(\Phi(x^k)\right)$ is uniformly bounded for all $k = 0, 1, ...$, since Φ is strongly convex and therefore it holds that $\Phi(x) \geq \Phi(x^*) + \frac{\mu_\Phi}{2}\|x - x^*\|^2$, $x \in \mathbb{R}^{np}$. Further, for the sake of proving boundedness, without loss of generality we can assume that $f_i(x) \geq 0$, for all $x \in \mathbb{R}^p$, for every $i = 1, ..., n$.[4] For $k = 0, 1, ...$, define function $\Phi_k : \mathbb{R}^{np} \to \mathbb{R}$, by

$$\Phi_k(x) = \frac{\alpha}{p_k} F(x) + \frac{1}{2} x^T (\mathbb{I} - \mathbb{Z}) x = \frac{\alpha}{p_k} \sum_{i=1}^{n} f_i(x_i) + \frac{1}{2} \sum_{\{i,j\} \in E, i<j} w_{ij} \|x_i - x_j\|^2.$$

Notice that $\Phi_k(x) = \Phi(x)$, $x \in \mathbb{R}^{np}$, if $p_k = 1$. Also note that, for every $k = 0, 1, ...$, we have:

$$\Phi(x) \leq \Phi_{k+1}(x) \leq \Phi_k(x) \leq \Phi_0(x),$$

since p_k is assumed to be non-decreasing. The core of the proof is to upper bound $\Phi_{k+1}\left(x^{k+1}\right)$ with a quantity involving $\Phi_k\left(x^k\right)$ (see ahead (34)), and after that to "unwind" the resulting recursion.

To start, notice that Φ_k is strongly convex and has Lipschitz continuous gradient for every k. More precisely, for every $k = 0, 1, ...$, and for every $x \in \mathbb{R}^{np}$, we have that

$$\bar{\mu}\mathbb{I} \preceq \nabla^2 \Phi_k(x) \preceq \bar{L}\mathbb{I},$$

where $\bar{\mu} = \alpha\mu$ and $\bar{L} = \alpha L / p_{min} + 1$. Denote by

$$y_k = \nabla \Phi_k(x^k) = \frac{\alpha}{p_k} \nabla F(x^k) + (\mathbb{I} - \mathbb{Z})\, x^k.$$

Further, let us define two more auxiliary maps, as follows. Let $\widehat{\Phi}_k : \mathbb{R}^{np} \times$

[4] Otherwise, since each of the f_i's is lower bounded, we can re-define each f_i as $\widehat{f}_i(x) = f_i(x) + c$, where c is a constant larger than or equal $\max_{i=1,...,n} |\inf_{x \in \mathbb{R}^p} f_i(x)|$, and work with the $\widehat{f}_i$'s throughout the proof.

$\{0,1\}^m \to \mathbb{R}$, and $\widetilde{\Phi}_k : \mathbb{R}^{np} \times \{0,1\}^m \to \mathbb{R}$, be given by:

$$\begin{aligned}
\widehat{\Phi}_k(x;\xi) &= \frac{\alpha}{p_k}\sum_{i=1}^{n}\xi_i f_i(x_i) + \frac{1}{2}\sum_{\{i,j\}\in E, i<j} w_{ij}\xi_i\xi_j\|x_i - x_j\|^2 \\
\widetilde{\Phi}_k(x;\xi) &= \Phi_k(x) - \widehat{\Phi}_k(x) = \frac{\alpha}{p_k}\sum_{i=1}^{n}(1-\xi_i)f_i(x_i) \\
&+ \frac{1}{2}\sum_{\{i,j\}\in E, i<j} w_{ij}(1-\xi_i\xi_j)\|x_i - x_j\|^2,
\end{aligned}$$

where $\{0,1\}^m$ denotes the set of all m-dimensional vectors with the entries from set $\{0,1\}$. Introduce also the short-hand notation

$$\begin{aligned}
\widehat{\Phi}_k(x) &= \widehat{\Phi}_k(x;\xi^k) = \frac{\alpha}{p_k}\sum_{i=1}^{n}\xi_i^k f_i(x_i) + \frac{1}{2}\sum_{\{i,j\}\in E, i<j} w_{ij}\xi_i^k\xi_j^k\|x_i - x_j\|^2 \\
\widetilde{\Phi}_k(x) &= \widetilde{\Phi}_k(x;\xi^k) = \Phi_k(x) - \widehat{\Phi}_k(x) = \frac{\alpha}{p_k}\sum_{i=1}^{n}(1-\xi_i^k)f_i(x_i) \\
&+ \frac{1}{2}\sum_{\{i,j\}\in E, i<j} w_{ij}(1-\xi_i^k\xi_j^k)\|x_i - x_j\|^2,
\end{aligned}$$

where we recall that $\xi^k = (\xi_1^k, ..., \xi_n^k)^T$ is the node activation vector at iteration k. Hence, note that, for any fixed $x \in \mathbb{R}^{np}$, $\widehat{\Phi}_k(x)$ is a random variable, measurable with respect to the σ-algebra generated by ξ^k. On the other hand, for any fixed value that variable ξ^k takes, $x \mapsto \widehat{\Phi}_k(x)$ is a (deterministic) function, mapping $\mathbb{R}^{np}$ to $\mathbb{R}$; analogous observations hold for $\hat{\Phi}_k$ as well. We will be interested in quantities $\widehat{\Phi}_k(x^k)$, $\widetilde{\Phi}_k(x^k)$ $\widehat{\Phi}_k(x^{k+1})$, and $\widetilde{\Phi}_k(x^{k+1})$. Note that they are all random variables, measurable with respect to the σ-algebra generated by $\{\xi^s\}_{s=0,1,...,k}$. We will also work with the gradients of $\widehat{\Phi}_k$ and $\widetilde{\Phi}_k$ with respect to x, evaluated at x^k, that we denote by

$$\widehat{y}_k = \nabla\widehat{\Phi}_k(x^k) = \frac{\alpha}{p_k}\mathbb{Y}_k\nabla F(x^k) + (\mathbb{I} - \mathbb{Z}^k)x^k$$

and

$$\tilde{y}_k = y_k - \widehat{y}_k = \nabla\widetilde{\Phi}_k(x^k).$$

These quantities are also valid random variables, measurable with respect to the σ-algebra generated by $\{\xi^s\}_{s=0,1,...,k}$.

Now, recall that, for each fixed i, ξ_i^k, $k = 0, 1, ...$, are independent identically distributed (i.i.d.) Bernoulli random variables. The same is true for the minimum and the maximum

$$\xi_{min}^k = \min_{i=1,...,n}\xi_i^k, \quad \xi_{max}^k = \max_{i=1,...,n}\xi_i^k.$$

Also, notice that $E(\xi_{min}^k) = p_k^n$ and

$$\sqrt{1-\xi_{min}^k} = 1 - \xi_{min}^k, \quad 1-\xi_i^k \le 1-\xi_{min}^k \quad 1-\xi_i^k\xi_j^k \le 1-\xi_{min}^k.$$

Consider now $\widehat{\Phi}_k$ and $\widehat{y}_k$, regarded as functions of $x \in \mathbb{R}^{np}$. If $\xi^k_{max} = 1$ then $\widehat{\Phi}_k$ is strongly convex (with respect to x) with the same parameters as Φ_k, i.e.,

$$\bar{\mu}\mathbb{I} \preceq \nabla^2 \widehat{\Phi}_k(x) \preceq \bar{L}\mathbb{I}.$$

Now, denote by $\hat{x}^*_k = \hat{x}^*_k(\xi^k)$ the minimizer of $\widehat{\Phi}_k$ with respect to x; using the fact that $\widehat{\Phi}_k$ is nonnegative, and that it has a Lipschitz continuous gradient and is strongly convex, we obtain

$$\|\widehat{y}_k\|^2 \leq \bar{L}^2 \|x^k - \hat{x}^*_k\|^2 \leq \frac{2\bar{L}^2}{\bar{\mu}} (\widehat{\Phi}_k(x^k) - \widehat{\Phi}_k(\hat{x}^*_k)) \leq \frac{2\bar{L}^2}{\bar{\mu}} \widehat{\Phi}_k(x^k).$$

Using the fact that $\widehat{\Phi}_k(x) \leq \Phi_k(x)$, for all x, the previous inequality yields

$$\|\widehat{y}_k\| \leq \bar{L}\sqrt{2/\bar{\mu}} \sqrt{\Phi_k(x^k)}. \tag{27}$$

On the other hand, if $\xi^k_{max} = 0$ then $\mathbb{Y}_k = 0$ and $\mathbb{Z}_k = \mathbb{I}$ which implies $\widehat{y}_k = 0$ and the previous inequality obviously holds.

We perform a similar analysis considering $\widetilde{\Phi}_k$ and $\tilde{y}_k$. First, notice that $\widetilde{\Phi}_k$ is also non-negative. Furthermore, if $\xi^k_{min} = 0$ then $\widetilde{\Phi}_k$ is strongly convex with the same parameters as $\widehat{\Phi}_k$ and the following holds

$$\|\tilde{y}_k\| \leq \bar{L}\sqrt{2/\bar{\mu}} \sqrt{\widetilde{\Phi}_k(x^k)}.$$

Moreover, notice that

$$\widetilde{\Phi}_k(x^k) \leq (1 - \xi^k_{min}) \Phi_k(x^k)$$

and thus

$$\|\tilde{y}_k\| \leq \bar{L}\sqrt{2/\bar{\mu}} \sqrt{(1 - \xi^k_{min})\Phi_k(x^k)} = (1 - \xi^k_{min}) \bar{L}\sqrt{2/\bar{\mu}} \sqrt{\Phi_k(x^k)}. \tag{28}$$

Let us return to Φ_k. This function also satisfies (22), i.e.,

$$\Phi_k\left(x^k + \varepsilon\, s^k\right) \leq \Phi_k(x^k) + \frac{1}{2}\varepsilon^2 \bar{L} \|s^k\|^2 + \varepsilon y_k^T s^k. \tag{29}$$

For the search direction s^k in step 2 of Algorithm 4, and recalling $\mathbb{R}_k = (\mathbb{I} - \mathbb{L}_k \mathbb{G}^k)\, \mathbb{A}_k^{-1}$ from Theorem 4.1, we obtain that

$$s^k = -\mathbb{R}_k \widehat{y}_k.$$

Using the bounds (14) and (15) we conclude that $\|\mathbb{R}_k\| \leq (1 + \rho \mathcal{C}_{\mathrm{G}})\mathcal{C}_{\mathrm{A}} := \mathcal{C}_{\mathrm{R}}$ and therefore

$$\|s^k\| \leq \mathcal{C}_{\mathrm{R}} \|\widehat{y}_k\|. \tag{30}$$

Also, by Theorem 4.1, the following holds for $\varepsilon \leq \overline{\varepsilon}_{DQN}$, and $\rho \leq \overline{\rho}_{DQN}$ (where $\overline{\rho}_{DQN}$ and $\overline{\varepsilon}_{DQN}$ are given in Theorems 4.1 and 4.2, respectively):

$$\widehat{y}_k^T \mathbb{R}_k \widehat{y}_k = \widehat{y}_k^T \left(\frac{\mathbb{R}_k + \mathbb{R}_k^T}{2} \right) \widehat{y}_k \geq \delta \|\widehat{y}_k\|^2. \tag{31}$$

From now on, we assume that $\varepsilon \leq \overline{\varepsilon}_{DQN}$, and $\rho \leq \overline{\rho}_{DQN}$. Substituting (30) and (31) into (29) we obtain

$$\begin{aligned}
\Phi_k(x^{k+1}) &\leq \Phi_k(x^k) + \frac{1}{2}\varepsilon^2 \bar{L} R^2 \|\widehat{y}_k\|^2 - \varepsilon(\widehat{y}_k + \tilde{y}_k)^T \mathbb{R}_k \widehat{y}_k \\
&\leq \Phi_k(x^k) + \frac{1}{2}\varepsilon^2 \bar{L} R^2 \|\widehat{y}_k\|^2 - \varepsilon\delta\|\widehat{y}_k\|^2 + \varepsilon \mathcal{C}_{\mathrm{R}} \|\widehat{y}_k\| \|\tilde{y}_k\| \\
&= \Phi_k(x^k) + (\frac{1}{2}\varepsilon^2 \bar{L} \mathcal{C}_{\mathrm{R}}^2 - \varepsilon\delta)\|\widehat{y}_k\|^2 + \varepsilon \mathcal{C}_{\mathrm{R}} \|\widehat{y}_k\| \|\tilde{y}_k\| \qquad (32)
\end{aligned}$$

Since $(\frac{1}{2}\varepsilon^2 \bar{L} \mathcal{C}_{\mathrm{R}}^2 - \varepsilon\delta) \leq 0$ for $\varepsilon \leq \frac{2\delta}{\bar{L}\mathcal{C}_{\mathrm{R}}^2}$, we conclude that

$$\Phi_k(x^{k+1}) \leq \Phi_k(x^k) + \varepsilon \mathcal{C}_{\mathrm{R}} \|\widehat{y}_k\| \|\tilde{y}_k\| \leq \Phi_k(x^k) + \frac{2\varepsilon \bar{L} \mathcal{C}_{\mathrm{R}}^2}{\bar{\mu}} (1 - \xi_{min}^k) \Phi_k(x^k),$$

for

$$\varepsilon \leq \min\left\{ \frac{2\delta}{\bar{L}\mathcal{C}_{\mathrm{R}}^2}, \overline{\varepsilon}_{DQN} \right\}, \ \rho \leq \overline{\rho}_{DQN}, \qquad (33)$$

where $\overline{\varepsilon}_{DQN}$ and $\overline{\rho}_{DQN}$ are given in Theorems 4.1 and 4.2. Denoting $B = \frac{2\varepsilon \bar{L} \mathcal{C}_{\mathrm{R}}^2}{\bar{\mu}}$, and using the fact that $\Phi_{k+1}(x) \leq \Phi_k(x)$, for all $x \in \mathbb{R}^{np}$, we obtain

$$\Phi_{k+1}(x^{k+1}) \leq (1 + B(1 - \xi_{min}^k)) \Phi_k(x^k). \qquad (34)$$

Applying expectation we obtain

$$E(\Phi_{k+1}(x^{k+1})) \leq (1 + B(1 - p_k^n)) E(\Phi_k(x^k)),$$

where we use independence between ξ_{min}^k and x^k. Furthermore, recall that $u_k = 1 - p_k$ and notice that $1 - p_k^n \leq n u_k$. Moreover, $1 + t \leq e^t$ for $t > 0$ and thus

$$E(\Phi_{k+1}(x^{k+1})) \leq e^{nBu_k} E(\Phi_k(x^k)).$$

Next, by unwinding the recursion, we obtain

$$E(\Phi_k(x^k)) \leq e^{nB\sum_{j=0}^{k-1} u_j} \Phi_0(x^0) := \mathcal{C}_{\Phi,2}.$$

By assumption, $\{u_k\}$ is summable, and since $\Phi(x) \leq \Phi_k(x)$, for all x, we conclude that

$$E(\Phi(x^k)) \leq \mathcal{C}_{\Phi,2}.$$

Finally, since Φ is strongly convex, the desired result holds. □

Notice that an immediate consequence of Lemma 4.1 is that the gradients are uniformly bounded in the mean square sense. Indeed,

$$\begin{aligned}
E\left(\|\nabla F(x^k)\|^2\right) &= E\left(\|\nabla F(x^k) - \nabla F(\tilde{x}^*)\|^2\right) \\
&\leq E\left(L^2 \|x^k - \tilde{x}^*\|^2\right) \\
&\leq 2L^2 (E\left(\|x^k\|^2\right) + \|\tilde{x}^*\|^2) \\
&\leq 2L^2 (\mathcal{C}_{\mathrm{x}} + \|\tilde{x}^*\|^2) := \mathcal{C}_{\mathrm{F}}, \qquad (35)
\end{aligned}$$

where we recall $\mathcal{C}_{\rm x}$ in Lemma 4.1.

Next, we show that the "inexactness" of the search directions of DQN with idling are "controlled" by the activation probabilities p_k's.

Theorem 4.3. *Let assumptions A1-A4 hold, and consider $\overline{\rho}$ and $\overline{\varepsilon}$ as in Lemma 4.1. Then, for all $\rho \in [0, \overline{\rho}]$ and $\varepsilon \in (0, \overline{\varepsilon}]$, the following inequality holds for every k and some positive constant $\mathcal{C}_{\rm s}$:*

$$E(\|s^k - \hat{s}^k\|^2) \leq (1 - p_k)\mathcal{C}_{\rm s}. \tag{36}$$

Proof. We first split the error as follows:

$$\begin{aligned} E\left(\|s^k - \hat{s}^k\|^2\right) &= E\left(\|\mathbb{H}^k g^k - \widehat{\mathbb{H}}^k \hat{g}^k + \widehat{\mathbb{H}}^k g^k - \widehat{\mathbb{H}}^k g^k\|^2\right) \\ &\leq 2\left(E\left(\|\widehat{\mathbb{H}}^k\|^2 \|(g^k - \hat{g}^k)\|^2\right) + E\left(\|(\mathbb{H}^k - \widehat{\mathbb{H}}^k) g^k\|^2\right)\right) \\ &\leq 2\left((\mathcal{C}_{\rm H})^2 E\left(\|(\hat{g}^k - g^k)\|^2\right) + E\left(\|(\mathbb{H}^k - \widehat{\mathbb{H}}^k) g^k\|^2\right)\right). \end{aligned} \tag{37}$$

We will estimate the expectations above separately. Start by observing that:

$$\begin{aligned} \hat{g}_i^k - g_i^k &= -(A_i^k)^{-1}\left(\alpha \nabla f_i(x_i^k) + \sum_{j \in O_i} w_{ij}(x_i^k - x_j^k)\right) + \\ &+ \xi_i^k (A_i^k)^{-1}\left(\frac{\alpha}{p_k}\nabla f_i(x_i^k) + \sum_{j \in O_i} w_{ij}\xi_j^k (x_i^k - x_j^k)\right) \\ &= (\frac{\xi_i^k}{p_k} - 1)\alpha (A_i^k)^{-1}\nabla f_i(x_i^k) + (A_i^k)^{-1}\sum_{j \in O_i} w_{ij}(\xi_i^k \xi_j^k - 1)(x_i^k - x_j^k). \end{aligned}$$

Notice that (14) implies $\|(\frac{\xi_i^k}{p_k} - 1)\alpha (A_i^k)^{-1}\nabla f_i(x_i^k)\|^2 \leq 2(\frac{\xi_i^k}{p_k} - 1)^2 \alpha^2 (\mathcal{C}_{\rm A})^2 \|\nabla f_i(x_i^k)\|^2$, with $\mathcal{C}_{\rm A}$ defined in (14). Also, the assumptions on the w_{ij}'s (Assumption A3) and convexity of the scalar quadratic function $\mathcal{V}(\mu) = \mu^2$ yield:

$$\begin{aligned} \|\sum_{j \in O_i} w_{ij}(\xi_i^k \xi_j^k - 1)(x_i^k - x_j^k)\|^2 &\leq \left(\sum_{j \in O_i} w_{ij}(1 - \xi_i^k \xi_j^k)\|x_i^k - x_j^k\|\right)^2 \\ &\leq \sum_{j \in O_i} w_{ij}(1 - \xi_i^k \xi_j^k)^2 \|x_i^k - x_j^k\|^2. \end{aligned}$$

Therefore,

$$\|\hat{g}_i^k - g_i^k\|^2 \leq 2(\frac{\xi_i^k}{p_k} - 1)^2 \alpha^2 (\mathcal{C}_{\rm A})^2 \|\nabla f_i(x_i^k)\|^2 + 2(\mathcal{C}_{\rm A})^2 \sum_{j \in O_i} w_{ij}(1 - \xi_i^k \xi_j^k)^2 \|x_i^k - x_j^k\|^2.$$

Applying the expectation and using the fact that ξ_i^k's are independent, identically distributed (i.i.d.) across i and across iterations, the fact that

$$E((\frac{\xi_i^k}{p_k} - 1)^2) = (1 - p_k)/p_k \leq (1 - p_k)/p_{min},$$

and

$$E((1-\xi_i^k\xi_j^k)^2) = 1-p_k^2 = (1-p_k)(1+p_k) \le 2(1-p_k)$$

we obtain the following inequality

$$E(\|\hat{g}_i^k - g_i^k\|^2) \le 2(\mathcal{C}_{\rm A})^2(1-p_k)(\frac{\alpha^2}{p_{min}}E(\|\nabla f_i(x_i^k)\|^2) + 2\sum_{j\in O_i} w_{ij}E(\|x_i^k - x_j^k\|^2)).$$

Further, note that $E(\|\hat{g}^k - g^k\|^2) = \sum_{i=1}^n E(\|\hat{g}_i^k - g_i^k\|^2)$. Moreover, as a consequence of Lemma 4.1 we have $\sum_{i=1}^n E(\|\nabla f_i(x_i^k)\|^2) = E(\|\nabla F(x^k)\|^2) \le \mathcal{C}_{\rm F}$, and

$$\begin{aligned}
\sum_{i=1}^n \sum_{j\in O_i} w_{ij}E(\|x_i^k - x_j^k\|^2) &\le \sum_{i=1}^n \sum_{j\in O_i} w_{ij}\, 2\, E(\|x_i^k\|^2 + \|x_j^k\|^2) \\
&= 2(E(\sum_{i=1}^n \|x_i^k\|^2 \sum_{j\in O_i} w_{ij}) + E(\sum_{j\in O_i}\|x_j^k\|^2 \sum_{i=1}^n w_{ij})) \\
&\le 2(E(\|x^k\|^2 + E(\|x^k\|^2)) \\
&\le 4\mathcal{C}_{\rm x}, \qquad (38)
\end{aligned}$$

where $\mathcal{C}_{\rm x}$ is given in Lemma 4.1. Combining the bounds above, we conclude

$$E(\|\hat{g}^k - g^k\|^2) \le (1-p_k)2(\mathcal{C}_{\rm A})^2(\frac{\alpha^2}{p_{min}}\mathcal{C}_{\rm F} + 8\mathcal{C}_{\rm x}) := (1-p_k)\mathcal{C}_{\rm g}. \qquad (39)$$

Now, we will estimate the second expectation term in (37). Again, consider an arbitrary block i of the vector under expectation. We obtain the following.

$$\begin{aligned}
[(\mathbb{H}^k - \widehat{\mathbb{H}}^k)g^k]_i &= [\mathbb{L}_k(\mathbb{G} - \mathbb{G}^k)g^k]_i = [\mathbb{L}_k(\mathbb{Z}_u - \mathbb{Z}_u^k + \theta(\mathbb{Z}_d^k - \mathbb{Z}_d))g^k]_i \\
&= \Lambda_i^k \sum_{j\in O_i}(w_{ij} - w_{ij}^k)g_j^k + \Lambda_i^k\theta(w_{ii}^k - w_{ii})g_i^k \\
&= \Lambda_i^k \sum_{j\in O_i} w_{ij}(1-\xi_i^k\xi_j^k)g_j^k + \Lambda_i^k\theta\sum_{j\in O_i}(w_{ij} - w_{ij}^k)g_i^k \\
&= \Lambda_i^k \sum_{j\in O_i} w_{ij}(1-\xi_i^k\xi_j^k)(g_j^k + \theta g_i^k).
\end{aligned}$$

Moreover, applying the norm and the convexity argument like above we get

$$\begin{aligned}
\|[(\mathbb{H}^k - \widehat{\mathbb{H}}^k)g^k]_i\|^2 &\le \|\Lambda_i^k\|^2\|\sum_{j\in O_i} w_{ij}(1-\xi_i^k\xi_j^k)(g_j^k + \theta g_i^k)\|^2 \\
&\le \rho^2 \sum_{j\in O_i} w_{ij}(1-\xi_i^k\xi_j^k)^2\|g_j^k + \theta g_i^k\|^2 \qquad (40)
\end{aligned}$$

Therefore, $E(\|[(\mathbb{H}^k - \widehat{\mathbb{H}}^k)g^k]_i\|^2) \le \rho^2\sum_{j\in O_i} w_{ij}2(1-p_k)E\|g_j^k + \theta g_i^k\|^2)$ and using the steps similar to the ones in (38) we obtain the inequality

$$E(\|(\mathbb{H}^k - \widehat{\mathbb{H}}^k)g^k\|^2) \le (1-p_k)\rho^2 4(1+\theta^2)E(\|g^k\|^2).$$

Moreover,

$$\begin{aligned} E(\|g^k\|^2) &= E(\|\mathbb{A}_k^{-1}(\frac{\alpha}{p_k}\mathbb{Y}_k\nabla F(x^k)+(\mathbb{I}-\mathbb{Z}^k)x^k)\|^2) \\ &\leq (\mathcal{C}_{\rm A})^2 2\left(\frac{\alpha^2}{p_{min}^2}E(\|\mathbb{Y}_k\|^2)E(\|\nabla F(x^k)\|^2)+E(\|\mathbb{I}-\mathbb{Z}^k\|^2)E(\|x^k\|^2))\right) \\ &\leq (\mathcal{C}_{\rm A})^2 2(\frac{\alpha^2}{p_{min}^2}\mathcal{C}_{\rm F}+\mathcal{C}_{\rm x}) := \mathcal{C}_{\rm g,2} \end{aligned} \tag{41}$$

and thus

$$E(\|(\mathbb{H}^k-\widehat{\mathbb{H}}^k)g^k\|^2) \leq (1-p_k)\rho^2 4(1+\theta^2)\mathcal{C}_{\rm g,2}.$$

Finally, returning to (37), the previous inequality and (39) imply

$$E\left(\|s^k-\hat{s}^k\|^2\right) \leq (1-p_k)2((\mathcal{C}_{\rm H})^2\mathcal{C}_{\rm g}+\rho^2\mathcal{C}_{\rm g,2}) := (1-p_k)\mathcal{C}_{\rm s}.$$

□

4.3 Main results

The next result – first main result – follows from the theorems stated above. Let us define constants $\overline{\rho}_{idl}=\min\{\overline{\rho},\overline{\rho}_{DQN}\}$ and $\overline{\varepsilon}_{idl}=\min\{\overline{\varepsilon},\overline{\varepsilon}_{DQN}\}$ where $\overline{\rho}$ and $\overline{\varepsilon}$ are as in Theorem 4.3 and $\overline{\rho}_{DQN}$ and $\overline{\varepsilon}_{DQN}$ are as in Theorem 4.2.

Theorem 4.4. *Let $\{x^k\}$ be the sequence of random variables generated by Algorithm 4. Further, let assumptions A1-A4 hold. In addition, let $\rho\in[0,\overline{\rho}_{idl}]$ and $\varepsilon\in(0,\overline{\varepsilon}_{idl}]$. Then:*

$$E\left(\Phi(x^{k+1})-\Phi(x^*)\right) \leq E\left(\Phi(x^k)-\Phi(x^*)\right)\nu+\mathcal{C}_\Phi(1-p_k), \tag{42}$$

where $\nu\in(0,1)$ is a constant, and $\mathcal{C}_\Phi$ is a positive constant. Moreover, the iterate sequence $\{x^k\}$ converges to the solution x^* of (2) in the mean square sense and almost surely.

Proof. Claim (42) follows by taking expectation in Theorem 4.3. The remaining two claims follow similarly to the proof of Theorem 2 in [4]. We briefly demonstrate the main arguments for completeness. Namely, unwinding the recursion (42), we obtain for $k=1,2,...$:

$$E\left(\Phi(x^k)-\Phi(x^*)\right) \leq \left(\Phi(x^0)-\Phi(x^*)\right)\nu^k+\mathcal{C}_\Phi\sum_{t=1}^{k}\nu^{k-t}(1-p_{t-1}). \tag{43}$$

Now, we apply Lemma 2.1. From this result and (43), it follows directly that $E\left(\Phi(x^k)-\Phi(x^*)\right)\to 0$ as $k\to\infty$, because it is assumed that $p_k\to 1$. Furthermore, using inequality $\Phi(x^k)-\Phi(x^*)\geq\frac{\mu_\Phi}{2}\|x^k-x^*\|^2$, the mean square convergence of x^k towards x^* follows. It remains to show that $x^k\to x^*$ almost surely, as well. Using condition (11), inequality (43) implies that:

$$E\left(\Phi(x^k)-\Phi(x^*)\right) \leq \left(\Phi(x^0)-\Phi(x^*)\right)\nu^k+\mathcal{C}_\Phi\,\mathcal{C}_{\rm u}\sum_{t=1}^{k}\frac{\nu^{k-t}}{t^{1+\zeta}}. \tag{44}$$

It can be shown that (44) implies (see, e.g., [4]): $E\left(\Phi(x^k)-\Phi(x^*)\right)=O\left(1/k^{1+\zeta}\right)$, which further implies:

$$E\left(\|x^k-x^*\|^2\right)=O\left(1/k^{1+\zeta}\right). \tag{45}$$

Applying the Markov inequality for the random variable $\|x^k-x^*\|^2$, we obtain, for any $\kappa>0$:

$$P\left(\|x^k-x^*\|^2>\kappa\right)\leq\frac{1}{\kappa}E\left(\|x^k-x^*\|^2\right)=O\left(1/k^{1+\zeta}\right). \tag{46}$$

Inequality (46) implies that:

$$\sum_{k=0}^{\infty}P\left(\|x^k-x^*\|^2>\kappa\right)<\infty,$$

and so, by the first Borel-Cantelli lemma, we get $P\left(\|x^k-x^*\|^2>\kappa,\ \text{infinitely often}\right)=0$, which implies that $x^k\to x^*$, almost surely. □

Next, we state and prove our second main result.

Theorem 4.5. *Let $\{x^k\}$ be the sequence of random variables generated by Algorithm 4. Further, let assumptions of Theorem 4.4 hold, and let $p_k=1-\sigma^{k+1}$, with $\sigma\in(0,1)$. Then, $\{x^k\}$ converges to the solution x^* of problem (2) in the mean square sense at an R-linear rate.*

Proof. Denote $z_k=\mathcal{C}_\Phi(1-p_k)$. For the specific choice of p_k we obtain $z_k=\mathcal{C}_\Phi\sigma^{k+1}$ which obviously converges to zero R-linearly. Furthermore, repeatedly applying the relation (42) we obtain

$$E\left(\Phi(x^k)-\Phi(x^*)\right)\leq(\Phi(x^0)-\Phi(x^*))\nu^k+a_k$$

where $a_k=\sum_{j=1}^{k}\nu^{j-1}z_{k-j}$. Moreover, it can be shown (see Lemma 2.1) that a_k also converges to zero R-linearly which implies the R-linear convergence of $E\left(\Phi(x^k)-\Phi(x^*)\right)$. Now, using the strong convexity of Φ (with strong convexity constant $\mu_\Phi=\alpha\mu$), we get: $\|x^k-x^*\|^2\leq(\Phi(x^k)-\Phi(x^*))2/\mu_\Phi$, which in turn implies:

$$E\left(\|x^k-x^*\|^2\right)\leq E\left(\Phi(x^k)-\Phi(x^*)\right)2/\mu_\Phi.$$

The last inequality means that $E\left(\|x^k-x^*\|^2\right)$ also converges to zero R-linearly. □

Theorem 4.5 shows that the DQN method with idling converges at an R-linear rate when $p_k=1-\sigma^{k+1}$. Parameter σ plays an important role in the practical performance of the method. We recommend the tuning $\sigma=1-c\,\alpha\,\mu$, with $c\in[30,80]$, for $\alpha<1/(80L)$. [5] The rationale for the tuning above comes from distributed first order methods with idling [4], where we showed analytically that it is optimal (in an appropriate sense) to set $\sigma^\bullet=(1-\alpha\mu)^2\approx$

[5] Condition $\alpha<1/(80L)$ is not restrictive, as we observed experimentally that usually one needs to take an α smaller than $1/(80L)$ in order to achieve a satisfactory limiting accuracy.

$1-2\alpha\mu$. The value $\sigma^\bullet$ is set to balance 1) the linear convergence factor of the method without idling and 2) the convergence factor of the convergence of p_k to one. As DQN has a better (smaller) convergence factor than the distributed first order method (due to incorporation of the second order information), one has to adjust the rule $1-2\alpha\mu$, replacing 2 with a larger constant c; experimental studies suggest values for c on the order 30-80. In addition, for very small values of $\alpha\mu$, in order to prevent very small p_k's at initial iterations on the one hand and the σ's very close to one on the other hand, we can utilize a "safeguarding" and modify p_k to $p_k = \max\left\{\underline{p}, 1-(\min\{\sigma,\overline{\sigma}\})^{k+1}\right\}$, where $\underline{p}$ can be taken, e.g., as 0.2, and $\overline{\sigma}$ as 0.9999.

5 Extensions: DQN under persisting idling

This section investigates DQN with idling when activation probability p_k does not converge to one asymptotically, i.e., the algorithm is subject to persisting idling. This scenario is of interest when activation probability p_k is not in full control of the algorithm designer (and the networked nodes during execution). For example, in applications like wireless sensor networks, inter-node messages may be lost, due to, e.g., random packet dropouts. In addition, an active node may fail to perform its solution estimate update at a certain iteration, because the actual calculation may take longer than the time slot allocated for one iteration, or simply due to unavailability of sufficient computational resources.

Henceforth, consider the scenario when p_k may not converge to one. In other words, regarding Assumption A4, we only keep the requirement that the sequence $\{p_k\}$ is uniformly bounded from below. We make here an additional assumption that the iterates are bounded in the mean square sense, i.e., $E\left(\|x^k\|^2\right)$ is uniformly bounded from above by a positive constant.[6] Now, consider relation (42); it continues to hold under the assumptions of the Theorem 4.4, i.e., we have

$$E\left(\Phi(x^k)-\Phi(x^*)\right) \leq (\Phi(x^0)-\Phi(x^*))\nu^k + \sum_{j=1}^{k}\nu^{j-1}\mathcal{C}_\Phi(1-p_{k-j}).$$

Therefore, using the previous inequality we obtain

$$E\left(\Phi(x^k)-\Phi(x^*)\right) \leq (\Phi(x^0)-\Phi(x^*))\nu^k + \frac{\mathcal{C}_\Phi(1-p_{min})}{1-\nu}.$$

Using strong convexity of Φ (with strong convexity constant μ_Φ) and letting k go to infinity we obtain

$$\limsup_{k\to\infty} E\left(\|x^k-x^*\|^2\right) \leq \frac{2\mathcal{C}_\Phi(1-p_{min})}{\mu_\Phi(1-\nu)} := \mathcal{E}.$$

[6]The boundedness proof of Lemma 4.1 in Section 4 does not generalize to the setting considered here. Henceforth, the mean square boundedness of the iterates is here stated formally as an assumption. Our conjecture is that the mean square boundedness of the iterates holds under the setting in Section 5 as well.

Therefore, the proposed algorithm converges (in the mean square sense) to a neighborhood of the solution x^* of (2). Hence, an additional limiting error (in addition to the error due to the difference between the solutions of (1) and (2)) is introduced with respect to the case $p_k \to 1$. In order to analyze further quantity $\mathcal{E}$, we unfold $\mathcal{C}_\Phi$ to get

$$\mathcal{E} = (1 - p_{min})h(\varepsilon)l(\rho),$$

where

$$h(\varepsilon) = \frac{\varepsilon^2 L_\Phi + \varepsilon/q}{1 - \nu(\varepsilon)} \quad \text{and} \quad l(\rho) = \frac{4}{\mu_\Phi}((1 + \rho\mathcal{C}_{\mathrm{G}})^2\mathcal{C}_{\mathrm{g}} + \rho^2\mathcal{C}_{\mathrm{g},2});$$

and $\nu(\varepsilon)$ is as in the proof of Theorem 4.2. (Recall $\mathcal{C}_{\mathrm{G}}$, $\mathcal{C}_{\mathrm{g}}$, and $\mathcal{C}_{\mathrm{g},2}$ in (15), (39), and (41), respectively.) It can be shown that $h(\varepsilon)$ is an increasing function of ε so taking smaller step size ε brings us closer to the solution. However, the convergence factor $\nu(\varepsilon)$ is also increasing with ε; thus, there is a tradeoff between the precision and the convergence rate. Furthermore, considering $l(\rho)$, we can see that it is also an increasing function. However, as it is expected, $l(0)$ is strictly positive. In other words, the error remains positive when the safeguarding parameter $\rho = 0$. Finally, the size of the error is proportional to $(1-p_{min})$ – the closer p_{min} to one, the smaller the error. Simulation examples in Section 6 demonstrate that the error is only moderately increased (with respect to the case $p_k \to 1$), even in the presence of very strong persisting idling.

6 Numerical results

This section demonstrates by simulation significant computational and communication savings incurred through the idling mechanism within DQN. It also shows that persisting idling (p_k not converging to one) induces only a moderate additional limiting error, i.e., the method continues to converge to a solution neighborhood even under persisting idling.

We consider the problem with strongly convex local quadratic costs; that is, for each $i = 1, ..., n$, we let $f_i : \mathbb{R}^p \to \mathbb{R}$, $f_i(x) = \frac{1}{2}(x - b_i)^T A_i(x - b_i)$, $p = 10$, where $b_i \in \mathbb{R}^p$ and $A_i \in \mathbb{R}^{p\times p}$ is a symmetric positive definite matrix. The data pairs A_i, b_i are generated at random, independently across nodes, as follows. Each b_i's entry is generated mutually independently from the uniform distribution on $[1, 31]$. Each B_i is generated as $B_i = Q_i\, D_i\, Q_i^T$; here, Q_i is the matrix of orthonormal eigenvectors of $\frac{1}{2}(\widehat{B}_i + \widehat{B}_i^T)$, and $\widehat{B}_i$ is a matrix with independent, identically distributed (i.i.d.) standard Gaussian entries; and D_i is a diagonal matrix with the diagonal entries drawn in an i.i.d. fashion from the uniform distribution on $[1, 31]$.

The network is a $n = 100$-node instance of the random geometric graph model with the communication radius $r = \sqrt{\frac{\ln(n)}{n}}$, and it is connected. The weight matrix W is set as follows: for $\{i, j\} \in E$, $i \neq j$, $w_{ij} = \frac{1}{2(1+\max\{d_i, d_j\})}$,

where d_i is the node i's degree; for $\{i,j\} \notin E$, $i \neq j$, $w_{ij} = 0$; and $w_{ii} = 1 - \sum_{j \neq i} w_{ij}$, for all $i = 1, ..., n$.

We compare the standard DQN method and the DQN method with incorporated idling mechanism. Specifically, we study how the relative error (averaged across nodes):

$$\frac{1}{n} \sum_{i=1}^{n} \frac{\|x_i - \overline{x}^*\|}{\|\overline{x}^*\|}, \ \overline{x}^* \neq 0,$$

evolves with the elapsed total number of activations per node. Note that the number of activations relates directly to both the communication and computational costs of the algorithm.

The parameters for both algorithms are set in the same way, and the only difference is in the activation schedule. For the method with idling, we set $p_k = 1 - \sigma^{k+1}$, $k = 0, 1, ...$ We set $\sigma = 1 - c\,\alpha\,\mu$, with $c = 40$. (Clearly, for the method without idling, $p_k \equiv 1$, for all k.) The remaining algorithm parameters are as follows. We set $\alpha = 1/(100\,L)$, where L is the Lipschitz constant of the gradients of the f_i's that we take as $\max_{i=1,...,n} \|A_i\|$. Further, we let $\theta = 0$, and $\epsilon = 1$ (full step size). We consider two choices for Λ_i^k in step 6 of Algorithm 3: $\Lambda_i^k = 0$, for all i, k; and $\Lambda_i^k = -I$, for all i, k. (We apply no safeguarding on the above choices of Λ_i^k, i.e., we let $\rho = 1$.) Note that the former choice corresponds to the algorithms with a single communication round per iteration k, while the latter corresponds to the algorithms with two communication rounds per k.

Figure 1 (a) plots the relative error versus total cost (equal to total number of activations per node up to the current iteration) for one sample path realization, for $\Lambda_i^k = 0$. We can see that incorporating the idling mechanism significantly improves the efficiency of the algorithm: for the method to achieve the limiting accuracy of approximately 0.025, the method without idling takes about 410 activations per node, while the method with idling takes about 310 activations. Hence, the idling mechanism reduces total cost by approximately 24%. Figure 1 (b) repeats the plots for $\Lambda_i^k = -I$, still showing clear gains of idling, though smaller than with $\Lambda_i^k = 0$.

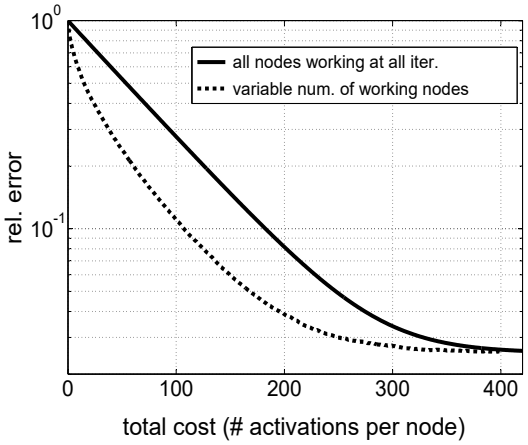
To account for randomness of the DQN method with idling that arises due to the random nodes' activation schedule, we include histograms of the total cost needed to achieve a fixed level of relative error. Specifically, in Figure 2 we plot histograms of the total cost (corresponding to 50 generated sample paths – 50 different realizations of the ξ_i^k's along iterations) needed to reach the relative error equal 0.04; Figure 2 (a) corresponds to $\Lambda_i^k = 0$, while Figure 2 (b) corresponds to $\Lambda_i^k = -I$. The Figures also indicate with arrows the total cost needed by standard DQN to achieve the same accuracy. The results confirm the gains of idling. Also, the variability of total cost across different sample paths is small relative to the gain with respect to standard DQN.

Figures 3 and 4 investigate the scenarios when the activation probability p_k may not asymptotically converge to one. The network is a (connected) random geometric graph instance with $n = 40$ and $r = \sqrt{\frac{\ln(n)}{n}}$; step size $\alpha = 1/(200L)$; the remaining system and algorithmic parameters are the same as with the pre-

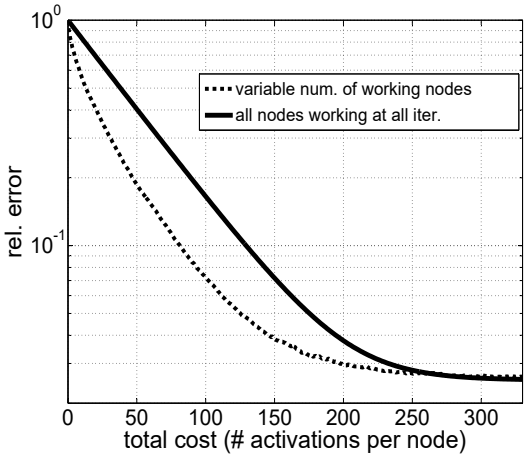
vious simulation example. We consider the following choices for p_k: 1) $p_k \equiv 1$ (standard DQN); 2) $p_k = 1 - \sigma^{k+1}$; 3) $p_k = p_{\max}\left(1 - \sigma^{k+1}\right)$; and 4) $p_k = p_{\max}$, for all k. With the third and fourth choices, the presence of $p_{\max}$ models external effects on the p_k (out of control of the networked nodes), e.g., due to link failures and unavailability of computing resources at certain iterations; it is varied within the set $\{0.5, 0.7, 0.9\}$. Figure 3 (a) compares the methods for one sample path realization with the four choices of p_k above, with $p_{\max} = 0.7$, for $\Lambda_i^k = 0$. Figure 3 (b) compares for the same experiment the standard DQN ($p_k \equiv 1$), $p_k = 1 - \sigma^{k+1}$, and $p_k = p_{\max}\left(1 - \sigma^{k+1}\right)$, with $p_{\max} \in \{0.5, 0.7, 0.9\}$. Several important observations stand out from the experiments. First, we can see that the limiting error increases when p_k does not converge to one with respect to the case when it converges to one. However, this increase (deterioration) is moderate, and the algorithm still manages to converge to a good solution neighborhood despite the persisting idling. In particular, from Figure 3 (b), we can see that the limiting relative error increases from about 10^{-2} (with $p_k \to 1$) to about $3.5 \cdot 10^{-2}$ with $p_{\max} = 0.5$–a case with a strong persisting idling. This corroborates that DQN with idling is an effective method even when activation probability p_k is not in full control of the algorithm designer. Second, the limiting error decreases when $p_{\max}$ increases, as it is expected – see Figure 3 (b). Finally, from Figure 3 (a), we can see that the method with $p_k = p_{\max}\left(1 - \sigma^{k+1}\right)$ performs significantly better than the method with $p_k = p_{\max}$, for all k. In particular, the methods have the same limiting error (approximately $2 \cdot 10^{-2}$), while the former approaches this error much faster. This confirms that the proposed judicious design of increasing p_k's, as opposed to just keeping them constant, significantly improves the algorithm performance. Figure 4 repeats the same experiment for $\Lambda_i^k = -I$. We can see that the analogous conclusions can be drawn.

7 Conclusion

We incorporated an idling mechanism, recently proposed in the context of distributed first order methods [4], into distributed second order methods. Specifically, we study the DQN algorithm [3] with idling. The idling mechanism involves an increasing number of nodes in the optimization process as the iteration counter k grows, so that, on average, $n\,p_k$ nodes (out of n nodes in total) participate at iteration k, where p_k is the per-node activation probability. We showed that, as long as p_k converges to one at least as fast as $1/k^{1+\zeta}$, $\zeta > 0$ arbitrarily small, the DQN algorithm with idling converges in the mean square sense and almost surely to the same point as the standard DQN method that activates all nodes at all iterations. Furthermore, when p_k grows to one at a geometric rate, DQN with idling converges at a R-linear rate in the mean square sense. Therefore, DQN with idling achieves the same order of convergence (R-linear) as standard DQN, but with significantly cheaper iterations. Simulation examples corroborate communication and computational savings incurred by incorporating the idling mechanism.

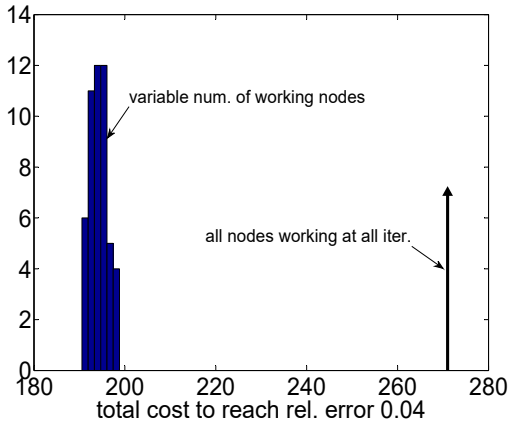


(a)

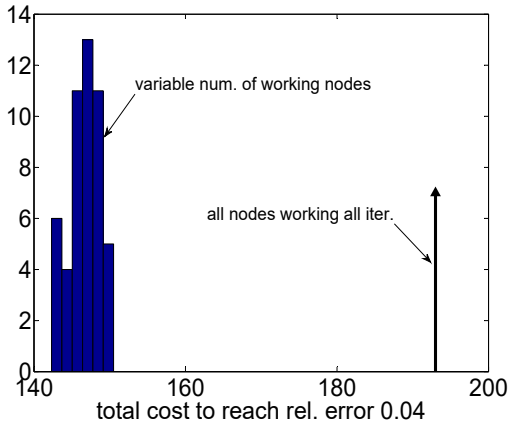


(b)

Figure 1: Relative error versus total cost (number of activations per node) for quadratic costs and $n = 100$-node network; Figure (a): $\Lambda_i^k = 0$; Figure (b): $\Lambda_i^k = -I$. The solid lines correspond to all nodes working at all iterations; the dashed lines correspond to the method with idling (increasing number of working nodes).

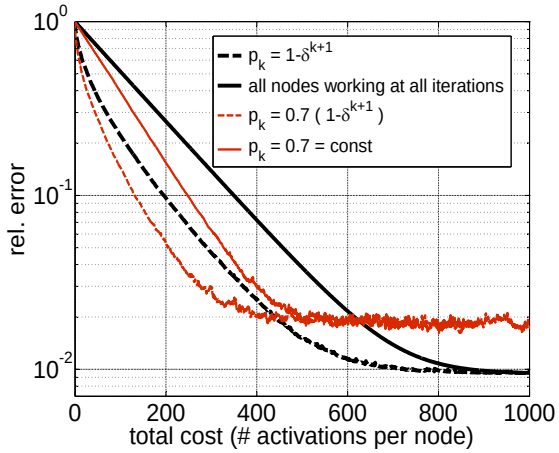


(a)

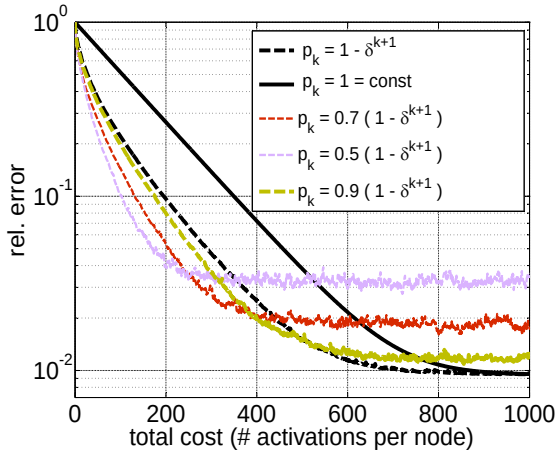


(b)

Figure 2: Total cost (number of activations per node) to reach relative error 0.04 for quadratic costs and $n = 100$-node network; Figure (a): $\Lambda_i^k = 0$; Figure (b): $\Lambda_i^k = -I$. The histograms corresponds to the DQN algorithm with idling; the arrow indicates the total cost needed by standard DQN.

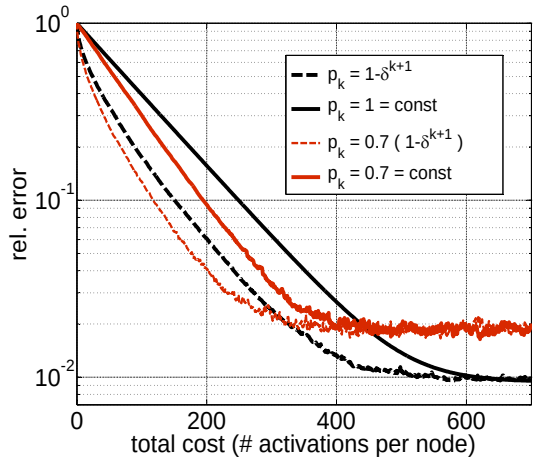


(a)

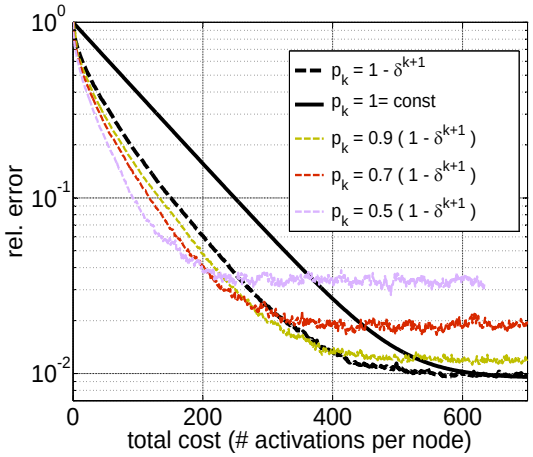


(b)

Figure 3: Relative error versus total cost (number of activations per node) for strongly convex quadratic costs, $n = 40$-node network, and $\Lambda_i^k = 0$. The Figures compare the following scenarios: 1) $p_k \equiv 1$ (standard DQN); 2) $p_k = 1 - \sigma^{k+1}$; 3) $p_k = p_{\max}\left(1 - \sigma^{k+1}\right)$; and 4) $p_k = p_{\max}$, for all k.



(a)



(b)

Figure 4: Relative error versus total cost (number of activations per node) for strongly convex quadratic costs, $n = 40$-node network, and $\Lambda_i^k = -I$. The Figures compare the following scenarios: 1) $p_k \equiv 1$ (standard DQN); 2) $p_k = 1 - \sigma^{k+1}$; 3) $p_k = p_{\max}\left(1 - \sigma^{k+1}\right)$; and 4) $p_k = p_{\max}$, for all k.

References

[1] Hug, G., Kar, S., Wu, C., Consensus + Innovations Approach for Distributed Multiagent Coordination in a Microgrid, IEEE Trans. Smart Grid, vol. 6(4), pp. 1893-1903, 2015.

[2] Bullo, F., Cortes, J., Martnez, S., Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms, Princeton University Press, 2009.

[3] Bajović, D. Jakovetić, D., Krejić, N., Krklec Jerinkić, N., Newton-like method with diagonal correction for distributed optimization, to appear in SIAM J. Opt., 2017, available at http://arxiv.org/abs/1509.01703

[4] Bajović, D. Jakovetić, D., Krejić, N., Krklec Jerinkić, N., Distributed Gradient Methods with Variable Number of Working Nodes, IEEE Trans. Signal Processing, vol. 64, no. 15, pp. 4080-4095, 2016

[5] Bajović, D. Jakovetić, D., Krejić, N., Krklec Jerinkić, N., Distributed first and second order methods with variable number of working nodes, to appear in proc. IEEE Global Conference on Signal and Information Processing, Greater Washington DC, VA, USA, Dec. 2016

[6] Friedlander, M. P. , Schmidt, M., Hybrid deterministic-stochastic methods for data fitting, *SIAM Journal on Scientific Computing* 34 (2012), 1380-1405.

[7] Eisen, M., Mokhtari, A., Ribeiro, A., Decentralized quasi-newton methods, 2016, preprint arXiv:1605.00933.

[8] Liu, J., Wright, S., Asynchronous stochastic coordinate descent: Parallelism and convergence properties, SIAM J. Opt., vol. 25, no. 1, pp. 35117376, 2015.

[9] Chang, T-H., Wei-Cheng, L., Hong, M., Wang, X., Distributed ADMM for large-scale optimization part II: Linear convergence analysis and numerical performance, IEEE Trans. Sig. Proc., vol. 64, no. 12, pp. 3131173144, 2016.

[10] Cattivelli, F., Sayed, A. H., Diffusion LMS strategies for distributed estimation, *IEEE Transactions on Signal Processing*, vol. 58, no. 3, (2010) pp. 1035–1048.

[11] Jakovetić, D., Moura, J. M. F., Xavier, J., Distributed Nesterov-like gradient algorithms, in *CDC'12, 51st IEEE Conference on Decision and Control*, Maui, Hawaii, December 2012, pp. 5459–5464.

[12] Krejić, N., Krklec Jerinkić, N., Nonmonotone line search methods with variable sample size, *Numerical Algorithms* 68 (2015), pp. 711-739.

[13] Lobel, I., Ozdaglar, A., Feijer, D., Distributed Multi-agent Optimization with State-Dependent Communication, *Mathematical Programming,* vol. 129, no. 2, (2014) pp. 255-284.

[14] Mokhtari, A., Ling, Q., Ribeiro, A., Network Newton Distributed Optimization Methods, IEEE Trans. Signal Processing, vol. 65, no. 1, pp. 146-161, 2017

[15] Mokhtari, A., Shi, W., Ling, Q., Ribeiro, A., DQM: Decentralized Quadratically Approximated Alternating Direction Method of Multipliers, IEEE Trans. Signal Processing, vol. 64, no. 19, pp. 5158-5173, 2016

[16] Mokhtari, A., Shi, W., Ling, Q., Ribeiro, A., A Decentralized Second Order Method with Exact Linear Convergence Rate for Consensus Optimization, IEEE Trans. Signal and Information Processing over Networks, vol. 2, no. 4. pp. 507-522, 2016

[17] Nedić, A., Ozdaglar, A., Distributed subgradient methods for multi-agent optimization, *IEEE Transactions on Automatic Control*, vol. 54, no. 1, (2009) pp. 48–61.

[18] Ram, S., Nedic, A., Veeravalli, V., Distributed stochastic subgradient projection algorithms for convex optimization, J. Optim. Theory Appl., vol. 147, no. 3, pp. 516545, 2011.

[19] Ram, S. S. , Nedić, A., Veeravalli, V., Asynchronous gossip algorithms for stochastic optimization, in *CDC '09, 48th IEEE International Conference on Decision and Control*, Shanghai, China, December 2009, pp. 3581 – 3586.

[20] Shi, W., Ling, Q., Wu, G., Yin, W., EXTRA: an Exact First-Order Algorithm for Decentralized Consensus Optimization, SIAM Journal on Optimization No. 25 vol. 2, (2015) pp. 944-966.

[21] Wei, E., Ozdaglar, A., Jadbabaie, A., A distributed Newton method for network utility maximizationI: algorithm, *IEEE Transactions on Automatic Control,* vol. 58, no. 9, (2013) pp. 2162- 2175.

[22] D. Varagnolo, F. Zanella, A. Cenedese, G. Pillonetto, and L. Schenato, Newton-Raphson Consensus for Distributed Convex Optimization, IEEE Trans. Aut. Contr., vol. 61, no. 4, 2016.

[23] Xiao, L., Boyd, S., Lall, S., A scheme for robust distributed sensor fusion based on average consensus, in *IPSN '05, Information Processing in Sensor Networks*, Los Angeles, California, 2005, pp. 63–70.

[24] Zargham, M., Ribeiro, A., Jadbabaie, A., Accelerated dual descent for constrained convex network flow optimization, Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, pp. 1037-1042, 2013.