Distributed second order methods with increasing number of working nodes

Nataša Krklec Jerinkić, Dušan Jakovetić, Nataša Krejić, Dragana Bajović

***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, and it stays idle with probability $1-p_k$, while the activations are independent both across nodes and across iterations. 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 show that, when p_k grows to one across iterations geometrically, 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.**

***Index Terms*—Distributed optimization, Variable sample schemes, Second order methods, Newton-like methods, Linear convergence.**

I. Introduction

Context and motivation. Distributed optimization has received a significant and growing interest in the past decade, e.g., [1], [2], [3], [4], [5], [6], due to many emerging applications in various domains, including wireless sensor networks, e.g., [7], smart grid, e.g., [8], distributed control applications, e.g., [9], etc. Various formulations of distributed optimization problems have been considered, including consensus optimization, e.g., [1], [3], utility maximization and network flow problems, e.g., [5], [10], and learning personalized models, e.g., [11], [12]. In this paper, we focus on the latter class of problems, where each node in a connected network has a local private cost function, and the goal is to minimize the sum of nodes' local costs f_i's while keeping the nodes' local solutions close to each other. This class of problems is also of direct interest for consensus optimization, as the problem of learning personalized models can be considered a penalty-like reformulation of the consensus optimization problem, e.g., [4], [13], [14].

In a recent work [15], we proposed a "hybrid" distributed first order method, motivated by "hybrid" methods from centralized optimization [16], wherein the tradeoff between the computational cost of an iteration and the search direction accuracy is governed by an idling mechanism. More precisely, 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.

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. Specifically, we incorporate here the idling mechanism in the Distributed Quasi Newton method (DQN) [14]. (The DQN method has been proposed and analyzed in [14], only for the scenario when all nodes are active at all times.) Our main results are as follows. We first carry out a theoretical analysis of the idling-DQN assuming that the f_i's are strongly convex and 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 the solution at a R-linear rate in the mean square sense.

Brief literature review. There has been a significant progress in the development of distributed second order methods in past few years. References [4], [17], [14], [18], [19], [10], [5] propose and analyze various distributed second order methods. Each of these works assumes that all nodes are active across all iterations, i.e., they are not concerned with designing nor analyzing methods with randomized nodes' activations. Next, there have been several works that study distributed first and second order methods with randomized nodes' or links' activations, e.g., [1], [20], [21], [22], [23], [2], [24], [25], [26]. It is also worth noting that randomization and "sparsification" of activations has been considered in different contexts as well; see, e.g., [27], for a graph filtering perspective.

The authors of [28] propose an asynchronous version of the (second order) Network Newton method in [4], wherein a randomly selected node becomes active at a time and performs a Network Newton-type second order update. The main difference between the algorithm in [28] and the DQN with idling lies in the very different nodes' activation schedule. With the algorithm in [28], only one node is active at each algorithm update. In contrast, in the DQN with idling, a time-varying, random subset of nodes, with a variable size, is active

Part of results of this paper was presented at the IEEE Global Conference on Signal and Information Processing (GlobalSIP), Washington, DC, VA, USA, Dec. 2016. The work of first three authors is supported by the Serbian Ministry of Education, Science, and Technological Development, Grant no. 174030. The first three authors are with the Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia. The fourth author is with the Department of Power, Electronics and Communication Engineering, Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia. Authors' emails: natasa.krklec@dmi.uns.ac.rs, djakovet@uns.ac.rs, natasak@uns.ac.rs, dbajovic@uns.ac.rs.

at a time. The paper [29], see also [30], [31], proposes and analyzes an asynchronous distributed quasi-Newton method that is based on an asynchronous implementation of the Broyden-Fletcher-Goldfarb-Shanno (BFGS) matrix update. Perhaps the closest to our randomized activation model are the models studied in, e.g., [22], [23], [24], [29]. However, they are primarily concerned with establishing convergence conditions under various asynchrony effects that are not controlled by the networked nodes. In contrast, our aim here is to demonstrate that a carefully designed "sparsification" of the workload across network – as inspired by work [16] from centralized optimization – can yield significant savings in communication and computation. Finally, in a companion paper [32], we presented a brief preliminary version of the current paper, wherein a subset of the results here are presented without proofs. Specifically, [32] 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 sub-linearly; we also include here extensions where the activation probability stays bounded away from one or is kept constant (and less than one) across iterations.

Paper organization. Section II describes the model that we assume and gives the necessary preliminaries. Section III presents the DQN algorithm with idling, while Section IV analyzes its convergence and convergence rate. Section V 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, and it provides numerical examples. Finally, we conclude in Section VI. This note focuses on key technical novelties; more complete proofs can be found in [33].

Notation. We denote by: $\mathbb{R}$ the set of real numbers; $\mathbb{R}^p$ the p-dimensional real coordinate space; a_{ij} the entry in the i-th row and j-th column of a matrix A; A^T the transpose of a matrix A; $\otimes$ the Kronecker product of matrices; I, 0, and e, respectively, the identity matrix, the zero matrix, and the column vector with unit entries; $A \succ 0\,(A \succeq 0)$ means that the symmetric matrix A is positive definite (respectively, positive semi-definite); $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument; $\lambda_i(\cdot)$ the i-th largest eigenvalue, $diag\,(A)$ the diagonal matrix with the diagonal entries equal to those of matrix A; $diag\,(A_1, ..., A_n)$ the $n^2 \times n^2$ diagonal matrix whose (i,i)-th $n \times n$ block equals A_i, $i = 1, ..., n$; $\nabla h(w)$ and $\nabla^2 h(w)$, respectively, the gradient and Hessian evaluated at w of a function $h : \mathbb{R}^p \to \mathbb{R}$, $p \geq 1$; $P(\mathcal{A})$ and $E[u]$ the probability of an event $\mathcal{A}$ and expectation of a random variable u, respectively.

II. Model and preliminaries

We consider distributed optimization where n nodes, $i = 1, ..., n$, constitute an undirected network $\mathcal{G} = (\mathcal{V}, E)$, with $\mathcal{V}$ being the set of nodes and E the set of undirected edges $\{i, j\}$. The nodes collaboratively solve the following problem:

$$\min \Phi(x) := \alpha \sum_{i=1}^{n} f_i(x_i) + \frac{1}{2} \sum_{i<j,\, \{i,j\} \in E} w_{ij} \|x_i - x_j\|^2. \quad (1)$$

Here, $f_i : \mathbb{R}^p \to \mathbb{R}$, $i = 1, ..., n$, is a convex cost function known only by node i, and the optimization variable is $x = ((x_1)^T, ..., (x_n)^T)^T \in \mathbb{R}^{np}$, with $x_i \in \mathbb{R}^p$ being its i-th sub-block. Further, α and the w_{ij}'s are positive parameters detailed ahead. There are several motivations for model (1), including learning personalized models, spatial field estimation, animal flocking, etc. [34], and it has been extensively studied, e.g., [11], [12], [34]. For example, with learning personalized models [11], each node aims to find a model that minimizes both 1) the mismatch with its neighbors models (encoded through the second summand in (1)) and 2) its personal loss f_i (encoded through the first summand in (1)). Here, parameter α trades off between the two described criteria, while parameter w_{ij} weighs the significance of similarity between node i and node j's models. In addition, formulation (1) is often used as a tractable relaxation of (nonlinear) consensus optimization problems, e.g., [4], [14].

We make the following assumptions.

Assumption A1. Each function $f_i : \mathbb{R}^p \to \mathbb{R}$, $i = 1, \ldots, n$ is convex, twice continuously differentiable, and there exist constants $0 < \mu \leq L < \infty$ such that $\mu I \preceq \nabla^2 f_i(x) \preceq LI$ for every $x \in \mathbb{R}^p$.

Under Assumption A1, problem (1) is solvable and has the unique solution $x^* \in \mathbb{R}^p$. Assumption A1 also implies that each function f_i is strongly convex with strong convexity parameter μ, and it also has Lipschitz continuous gradient with Lipschitz constant L.

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

In our setting, 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 $\bar{O}_i = O_i \bigcup \{i\}$. Collect all the weights w_{ij} in (1) in a $n \times n$ matrix W, such that: 1) $w_{ij} = 0$ if $\{i, j\} \notin E$, $i \neq j$; and 2) $w_{ii} = 1 - \sum_{j \neq i} w_{ij}$. Quantity w_{ii} is (without loss of generality – in view of the degree of freedom α in (1)) assumed to be strictly positive, for all $i = 1, ..., n$. Note that matrix W is, by construction, row-stochastic. We also assume it is symmetric and has the diagonal elements bounded away from zero and one; that is, there exist constants w_{min} and w_{max} such that for $i = 1, \ldots, n$, there holds $0 < w_{min} \leq w_{ii} \leq w_{max} < 1$.

Remark. Under the assumed setting, function Φ has Lipschitz continuous gradient with a Lipschitz constant that can be taken as $L_\Phi = \alpha L + 2(1 - w_{min})$. Moreover, function $\Phi(x)$ is strongly convex with a strong convexity modulus that can be taken as $\mu_\Phi = \alpha\, \mu > 0$.

Denote by $\lambda_1 > \lambda_2 \geq \ldots \geq \lambda_n$ 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 $e := \frac{1}{\sqrt{n}}(1, \ldots, 1)^T$. For future reference, we introduce the following notation. We 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] Define also

[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).

$F : \mathbb{R}^{np} \to \mathbb{R}$, $F(x) = \sum_{i=1}^{n} f_i(x_i)$. Then, (1) can be written as $\min\, \alpha\, F(x) + \frac{1}{2} x^T(\mathbb{I} - \mathbb{Z})x$ s.t. $x \in \mathbb{R}^{np}$.

We study distributed second order algorithms to solve this problem, or equivalently, (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:

$$W_d \;\; = \;\; diag(W), \;\; W_u = W - W_d \qquad (2)$$

and the corresponding $\mathbb{Z} = \mathbb{Z}(W) = \mathbb{Z}_d + \mathbb{Z}_u$, where $\mathbb{Z}_d = W_d \otimes I = diag(\mathbb{Z})$, and $\mathbb{Z}_u = W_u \otimes I$. Further, decompose the Hessian of Φ as: $\nabla^2 \Phi(x) = \mathbb{A}(x) - \mathbb{G}$, with $\mathbb{A} : \mathbb{R}^{np} \to \mathbb{R}^{(np)\times(np)}$, $\mathbb{A}(x) = \alpha \nabla^2 F(x) + (1+\theta)(\mathbb{I} - \mathbb{Z}_d)$ and $\mathbb{G}$ given by:

$$\mathbb{G} \;\; = \;\; \mathbb{G}(\mathbb{Z}, \theta) = \mathbb{Z}_u + \theta(\mathbb{I} - \mathbb{Z}_d), \qquad (3)$$

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 [35]; for claim (b), see, e.g., Lemma 4.2 in [36].

Lemma II.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 $\sum_{t=1}^{k} \nu^{k-t} a_{t-1} \to 0$ as $k \to \infty$. (b) If, moreover, $\{a_k\}$ converges to zero R-linearly, then the sum in part (a) converges to zero R-linearly.*

III. Algorithm DQN with idling

In this section, we present the proposed algorithm to solve (1) – the DQN with idling, that builds upon the distributed second order method DQN [14]. Subsection III-A first describes the idling mechanism, while Subsection III-B incorporates this mechanism in the DQN method.

A. 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 (to be specified soon) and participate in each communication round of an iteration, while inactive nodes do not perform any computations nor communications, i.e., their solution estimates 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 the ξ_i^k's are independent both across nodes and across iterations.

Throughout the paper, we impose the following Assumption on sequence $\{p_k\}$.

Assumption A3. 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 $0 \leq u_k \leq \frac{\mathcal{C}_{\mathrm{u}}}{(k+1)^{1+\zeta}}$, where $u_k = 1 - p_k$, $\mathcal{C}_{\mathrm{u}}$ is a positive constant and $\zeta > 0$ is arbitrarily small.

Assumption A3 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 V.) We also assume that p_k converges to one sufficiently fast, where the 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 (2) and (3):

$$W_d^k \;\; = \;\; diag(W^k), \;\; W_u^k = W^k - W_d^k, \qquad (4)$$

where $\mathbb{Z}^k = \mathbb{Z}(W^k) = \mathbb{Z}_d^k + \mathbb{Z}_u^k$, $\mathbb{Z}_d^k = W_d^k \otimes I = diag(\mathbb{Z}^k)$, and $\mathbb{Z}_u^k = W_u^k \otimes I$, and

$$\mathbb{G}^k \;\; = \;\; \mathbb{G}(\mathbb{Z}^k, \theta) = \mathbb{Z}_u^k + \theta(\mathbb{I} - \mathbb{Z}_d^k), \qquad (5)$$

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, using the results from [14] (see Lemma 3.1), for the Hessian splitting matrices we obtain the following important bounds: [3]

$$\|\mathbb{A}^{-1}(x)\| \;\; \leq \;\; \mathcal{C}_{\mathrm{A}}, \; x \in \mathbb{R}^{np} \qquad (6)$$

$$\|\mathbb{G}^k\| \;\; \leq \;\; \mathcal{C}_{\mathrm{G}}, \; k = 0, 1, ... \qquad (7)$$

where $\mathcal{C}_{\mathrm{A}} = (\alpha\mu + (1+\theta)(1 - w_{max}))^{-1}$ and $\mathcal{C}_{\mathrm{G}} = (1 + \theta)(1 - w_{min})$. Also, notice that $\|\mathbb{Y}_k\| \leq 1$, and $\mathbb{Z}^k \preceq \mathbb{I}$, for every k.

B. DQN with idling

We now incorporate the idling mechanism in the DQN method. 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 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 1 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 1: 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 $j \in O_i$.

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

[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 (7) 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 (6).

```
4. Node $i$ calculates
       $d_i^k = \left(A_i^k\right)^{-1} \left[ \frac{\alpha}{p_k} \nabla f_i(x_i^k) + \sum_{j \in O_i} w_{ij}^k \left(x_i^k - x_j^k\right) \right].$
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 $\Lambda_i^k$, 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 1. First, note that the iterates x_i^k generated by Algorithm 1 are random variables. (The initial iterates x_i^0, $i = 1, ..., n$, in Algorithm 1 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 [14]). 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 [33], 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 $\varepsilon, \rho, \theta, \alpha$ [14].

Parameters ϵ, ρ, and α, and the diagonal matrices Λ_i^k play the same role as in DQN [14]. An important difference with respect to standard DQN appears in step 4, where the local node i's gradient contribution is $\frac{\alpha}{p_k} \xi_i^k \nabla f_i(x_i^k)$, 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 $E[\xi_i^k] = p_k$.

We next provide practical considerations on how the idling mechanism and Algorithm 1 can be realized. First, suppose that there exists a dedicated reliable bi-directional communication link between any two neighboring nodes. Next, consider a link between nodes i and j at iteration k. If $\xi_i^k = 1$, node i performs communication and computation; in particular, it turns on both its transmitting and receiving antennas. On the other hand, if $\xi_i^k = 0$, node i switches off both its transmitting and receiving antennas. Suppose that $\xi_i^k = 1$, and consider two scenarios: 1) $\xi_j^k = 0$; and 2) $\xi_j^k = 1$. In the former case, node i listens the dedicated channel to node j. As node j does not transmit, node i verifies that it does not receive the respective message from node j (e.g., within a prescribed time window), and hence it does not incorporate node j's estimate in the update rule. On the other hand, as $\xi_j^k = 0$, node j does not include the estimate by node i, by the algorithm construction. In fact, node j stays idle. Next, consider the case $\xi_j^k = 1$. In this case, node i receives the message by node j (reliable communication assumed), and thus it incorporates node j's estimate in its update. Symmetrically, node j listens the channel from node i to node j, receives the respective message, and includes node i's estimate in its update. A similar mechanism works if the links are unreliable but still symmetric, in the sense that if the link from i to j is online, is strong enough to support communication, then so is the link from j to i. However, if the physical links can fail in an asymmetric fashion, then Algorithm 1 cannot be implemented in its direct form. The algorithm and the corresponding analysis have to be changed in such scenario. This lies outside the scope of this paper, but it corresponds to an interesting future research direction.

We also comment on the relation between the proposed method and the asynchronous NN in [28]. When the control parameter p_k is available to the algorithm designer for tuning, our simulation experience suggests that the proposed idling-DQN method usually shows an improved convergence speed with respect to the asynchronous NN (See Section V). With respect to the tuning parameters, besides the sequence of activation probabilities $\{p_k\}$, when compared with the asynchronous NN, the idling-DQN has two additional parameters, θ and ρ. The remaining tuning parameters of the idling-DQN, namely ϵ and α, have a very similar role, and a very similar tuning process, as with the asynchronous NN. We refer to [14] and [33] for the tuning recommendations for all the parameters of the idling-DQN.

Using notation (4)–(5), we represent DQN with idling in Algorithm 2 in a vector format. (Therein, $\mathbb{L}_k = diag\left(\Lambda_1^k, ..., \Lambda_n^k\right)$.)

Algorithm 2: 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 $\mathbb{L}_k \in \mathbb{R}^{np \times np}$ with $\|\mathbb{L}_k\| \leq \rho$.
  2. $s^k = -(\mathbb{I} - \mathbb{L}_k \mathbb{G}^k) \, \mathbb{A}_k^{-1} \left( \frac{\alpha}{p_k} \mathbb{Y}_k \nabla F(x^k) + (\mathbb{I} - \mathbb{Z}^k) \, x^k \right).$
  3. $x^{k+1} = x^k + \varepsilon s^k, \; k = k + 1.$
```

IV. Convergence analysis

Subsection IV-A states our main convergence and convergence rate results on DQN with idling, while Subsection IV-B presents main arguments behind the proofs of the results.

A. Statement of main result

The next theorem summarizes our main results on convergence and convergence rate of DQN with idling.

Theorem IV.1 *Let $\{x^k\}$ be the sequence of random variables generated by Algorithm 1, and let assumptions A1-A3 hold. Then, there exist positive constants $\overline{\rho}$ and $\overline{\varepsilon}$, such that, for any $\rho \in [0, \overline{\rho}]$ and for any $\varepsilon \in (0, \overline{\varepsilon}]$, the sequence of iterates $\{x^k\}$ converges to the solution x^* of* (1) *in the mean square sense and almost surely. Moreover, if $p_k = 1 - \sigma^{k+1}$, with $\sigma \in (0, 1)$, $\{x^k\}$ converges to the solution x^* of problem (1) in the mean square sense at an R-linear rate.*

The constant $\overline{\rho}$ is given in Lemma IV.1, and $\overline{\varepsilon}$ is given in the proof of Lemma IV.2. Theorem IV.1 demonstrates that, under the stated conditions, the order of convergence (R-linear rate) of the DQN method is preserved despite the idling.

From the technical side, the analysis here brings several novelties with respect to [15]. A key novelty here lies in the introduction of time-varying surrogate functions Φ_k's (see ahead the proof of Lemma IV.2). This corresponds to establishing the boundedness of the iterates of DQN with idling. With respect to [15], the analysis presented here is very different as the problem considered in [15] is constrained within a compact constraint set, and therefore the boundedness of the iterative sequence is given by the assumed setting. In contrast, boundedness is very challenging to establish here, due to the *random, inexact, second order* search directions utilized. Another important difference with respect to [15] is in establishing and quantifying decrease in the objective function (see Lemma IV.1), as the search directions here are of the second order. Hence, establishing decrease in the objective function is more challenging than with the first order directions in [15].

B. Proof of the main result

We now provide the proof of Theorem IV.1. The analysis is organized as follows. We first 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 (Lemma IV.1). Next, we establish the mean square boundedness of the iterates of DQN with idling (Lemma IV.2) and show the implications of this result on the "inexactness" of search directions (Lemma IV.3). Finally, we make use of these results to prove Theorem IV.1.

Quantifying inexactness of search directions. 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.:

$$\hat{s}^k = -(\mathbb{I} - \mathbb{L}_k\mathbb{G})\mathbb{A}_k^{-1}(\alpha\nabla F(x^k) + (\mathbb{I} - \mathbb{Z})x^k). \tag{8}$$

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. The following result is a straightforward generalization of Theorem 3.2 in [14].

Lemma IV.1 *Let assumptions A1-A3 hold. Further, let,* $\rho \in [0, \overline{\rho}]$ *where*

$$\begin{aligned}
\overline{\rho} = &\frac{\alpha\mu + (1+\theta)(1-w_{max})}{(1-w_{min})(1+\theta)} \\
&\times \left(\frac{1}{\alpha L + (1+\theta)(1-w_{min})} - \delta\right),
\end{aligned}$$

for some constant $\delta \in (0, 1/(\alpha L + (1+\theta)(1-w_{min})))$. *Further, let* $\varepsilon \leq \overline{\varepsilon}_{DQN}$, *where*

$$\overline{\varepsilon}_{DQN} = \frac{\delta - q}{2L_\Phi(\beta^2 + q^2)}, \tag{9}$$

for some constant $q \in (0, \delta)$. *Then, there holds*

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

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

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.

Lemma IV.2 *Let the sequence of random variables* $\{x^k\}$ *be generated by Algorithm 1, and let assumptions A1-A3 hold. Then, for all* $\rho \in [0, \overline{\rho}]$ *and for all* $\varepsilon \in (0, \overline{\varepsilon}]$, *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.

Setting up the 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] The proof is based on introducing certain iteration-varying auxiliary functions, and it is divided in 3 steps: 1) introducing auxiliary functions and establishing their properties; 2) bounding the auxiliary functions' gradients; and 3) establishing (approximate) descend on the auxiliary functions.

Step 1: Auxiliary functions and their properties. For $k = 0, 1, ...,$ define function $\Phi_k : \mathbb{R}^{np} \to \mathbb{R}$, by

$$\Phi_k(x) = \frac{\alpha}{p_k}F(x) + \frac{1}{2}x^T(\mathbb{I} - \mathbb{Z})x.$$

Notice that $\Phi_k(x) = \Phi(x)$, $x \in \mathbb{R}^{np}$, if $p_k = 1$. Also notice 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 $\Phi_k\left(x^k\right)$, and after that to "unwind" the resulting recursion.

To start, notice that, for every $x \in \mathbb{R}^{np}$, we have that

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

[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.

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

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

Introduce

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

where we recall that $\xi^k = (\xi_1^k, ..., \xi_n^k)^T$ is the node activation vector at iteration k. We will be interested in quantities $\widehat{\Phi}_k(x^k)$, $\widetilde{\Phi}_k(x^k)$ $\widehat{\Phi}_k(x^{k+1})$, and $\widetilde{\Phi}_k(x^{k+1})$. Notice that they are all random variables, measurable with respect to the σ-algebra generated by $\{\xi^s\}_{s=0,1,\ldots,k}$. 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 random variables, measurable with respect to the σ-algebra generated by $\{\xi^s\}_{s=0,1,\ldots,k}$.

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

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

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

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

Step 2: Bounding gradients of auxiliary functions. 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 Lipschitz continuous gradient and is strongly convex, we obtain

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

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

$$\|\widehat{y}_k\| \leq \bar{L}\sqrt{2/\bar{\mu}} \sqrt{\Phi_k(x^k)}. \tag{10}$$

On the other hand, if $\xi_{max}^k = 0$ then $\mathbb{Y}_k = 0$ and $\mathbb{Z}_k = \mathbb{I}$ which implies $\widehat{y}_k = 0$ and the previous inequality obviously holds. An analogous analysis considering $\widetilde{\Phi}_k$ and $\tilde{y}_k$ shows the following:

$$\|\tilde{y}_k\| \leq (1 - \xi_{min}^k) \bar{L}\sqrt{2/\bar{\mu}} \sqrt{\Phi_k(x^k)}. \tag{11}$$

Step 3: Establishing (approximate) descend on auxiliary functions. Recall the search direction s^k in step 2 of Algorithm 1, and define $\mathbb{R}_k = (\mathbb{I} - \mathbb{L}_k \mathbb{G}^k)\, \mathbb{A}_k^{-1}$. Then, we have $s^k = -\mathbb{R}_k \widehat{y}_k$. Using the bounds (6) and (7), we conclude that $\|\mathbb{R}_k\| \leq (1 + \rho \mathcal{C}_{\mathrm{G}})\mathcal{C}_{\mathrm{A}} := \mathcal{C}_{\mathrm{R}}$ and therefore

$$\|s^k\| \leq \mathcal{C}_{\mathrm{R}} \|\widehat{y}_k\|. \tag{12}$$

Now, consider Φ_k. It can be shown that this function satisfies the following relation:

$$\Phi_k\left(x^k + \varepsilon\, s^k\right) \leq \Phi_k(x^k) + \frac{1}{2}\varepsilon^2 \bar{L} \|s^k\|^2 + \varepsilon y_k^T s^{\text{'}}. \tag{13}$$

and using the fact that $\tilde{y}_k = y_k - \widehat{y}_k$ we conclude that

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

for

$$\varepsilon \leq \overline{\varepsilon} := \min\left\{ \frac{2\delta}{\bar{L}\mathcal{C}_{\mathrm{R}}^2}, \overline{\varepsilon}_{DQN} \right\},\ \rho \leq \overline{\rho}, \tag{14}$$

where $\overline{\varepsilon}_{DQN}$ and $\overline{\rho}$ are given in Lemma IV.1 and δ is assumed to be small enough, i.e., $\delta \in (0, 1/(\alpha L + (1+\theta)(1 - w_{min})))$ (see the proof of Lemma IV.1 in [33] for details). Denoting $B = \left(2\varepsilon \bar{L} \mathcal{C}_{\mathrm{R}}^2\right)/\bar{\mu}$, and using the fact that $\Phi_{k+1}(x) \leq \Phi_k(x)$, for all $x \in \mathbb{R}^{np}$, we obtain

$$\Phi_{k+1}(x^{k+1}) \leq (1 + B(1 - \xi_{min}^k))\Phi_k(x^k). \tag{15}$$

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 the fact that ξ_{min}^k and x^k are mutually independent. 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 $E(\Phi(x^k)) \leq \mathcal{C}_{\Phi,2}$. Since Φ is strongly convex, the desired result holds. □

Notice that an immediate consequence of Lemma IV.1 is that the gradients are uniformly bounded in the mean square sense. Indeed,

$$\begin{aligned}
E\left(\|\nabla F(x^k)\|^2\right) &= E\left(\|\nabla F(x^k) - \nabla F(\tilde{x}^*)\|^2\right) \\
&\leq E\left(L^2 \|x^k - \tilde{x}^*\|^2\right) \\
&\leq 2L^2(\mathcal{C}_{\mathrm{x}} + \|\tilde{x}^*\|^2) := \mathcal{C}_{\mathrm{F}},
\end{aligned} \tag{16}$$

where $\tilde{x}^* \in \mathbb{R}^{np}$ is the minimizer of F, and where we recall $\mathcal{C}_{\mathrm{x}}$ in Lemma IV.2. Next, we show that the “inexactness” of the search directions of DQN with idling are “controlled” by the activation probabilities p_k’s. For the proof, see [33], Theorem IV.3.

Lemma IV.3 *Let assumptions A1-A3 hold, and consider $\overline{\rho}$ and $\overline{\varepsilon}$ as in Lemma IV.2. 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 $\mathcal{C}_{\rm s}$: $E(\|s^k - \hat{s}^k\|^2) \leq (1-p_k)\mathcal{C}_{\rm s}$.*

Proof. We first split the error as follows: $E\left(\|s^k - \hat{s}^k\|^2\right) \leq 2\left((\mathcal{C}_{\rm H})^2 E\left(\|(\hat{g}^k - g^k)\|^2\right) + E\left(\|(\mathbb{H}^k - \widehat{\mathbb{H}}^k)g^k\|^2\right)\right).$, where $\mathcal{C}_{\rm H}$ is a positive constant. Then, going through the blocks, using the fact that ξ_i^k are i.i.d., and using Lemma IV.2 and its consequence (16), it can be shown (see [33], Theorem IV.3.) that both error terms on the right hand side are $\mathcal{O}(1-p_k)$. □

We are now ready to prove the main result, Theorem IV.1.

Proof. The proof follows similarly to the proof of Theorem 2 in [15]. Namely, unwinding the recursion in Lemma IV.1 and taking expectation, we obtain for $k = 1, 2, ...$:

$$E(\Phi(x^k) - \Phi(x^*)) \leq \left(\Phi(x^0) - \Phi(x^*)\right)\nu^k + \mathcal{C}_\Phi \sum_{t=1}^{k} \nu^{k-t}(1-p_{t-1}). \tag{17}$$

Now, we apply part (a) of Lemma II.1. From this result and (17), 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. This is accomplished by using the Borel-Cantelli lemma-based argument similar to Subsection IV-B in [37]. Finally, the R-linear mean square convergence rate follows by applying part (b) of Lemma II.1 on (17). □

V. Extensions and numerical results

DQN under persisting idling. This section investigates DQN with idling when activation probability p_k does not converge to one asymptotically. 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). Henceforth, regarding Assumption A3, 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. Then, the bound in (22) continues to hold, and we have:

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

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

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

Therefore, the proposed algorithm converges (in the mean square sense) to a neighborhood of the solution x^* of (1). Hence, a limiting error is introduced with respect to the case $p_k \to 1$. We can see that the size of the error is proportional to $(1-p_{min})$ – the closer p_{min} to one, the smaller the error. However, numerical simulations suggest 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; see [33], Figures 3-5.

Simulation example. We compare the proposed DQN method with idling with other existing methods that utilize randomized activations of nodes, namely the methods in [28], referred to as asynchronous network Newton, and [15]. The comparison is performed on a $n = 30$-node (connected) random geometric graph network with 91 links and randomly generated strongly convex quadratic f_i's. More precisely, for each $i = 1, ..., n$, we let $f_i : \mathbb{R}^p \to \mathbb{R}$, $f_i(x) = \frac{1}{2}(x - b_i)^T B_i (x - b_i)$, $p = 5$, where $b_i \in \mathbb{R}^p$ and $B_i \in \mathbb{R}^{p \times p}$ is a symmetric positive definite matrix. The data pairs B_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 $[1, 31]$. With the proposed idling-DQN, the activation probability is set to $p_k = 1 - \sigma^{k+1}$, with $\sigma = 1 - c\,\alpha\,\mu$, $c = 80$. With the algorithm in [15], we let the activation probability $p_k = 1 - ((1-\alpha\,\mu)^2)^{k+1}$. The weight matrix W is 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 set $\alpha = 1/(100L)$. Figure 1 plots the relative error $\|x^k - x^*\|/\|x^*\|$ (where we recall that x^* is the solution to (1)) versus total number of activations for the three methods. We can see that the proposed method performs better than the other methods on the considered example.

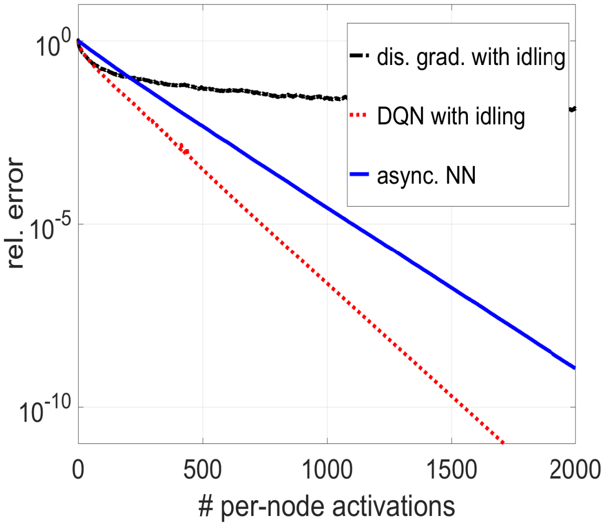

Fig. 1. Relative error versus total cost (number of activations per node) for strongly convex quadratic costs and $n = 30$-node network. The red, dotted line corresponds to the proposed DQN with idling; blue, solid line to [28]; and black, dash-dot line to [15].

VI. Conclusion

We incorporated an idling mechanism, recently proposed in the context of distributed first order methods [15], into distributed second order methods. Specifically, we study the DQN algorithm [14] with idling. We showed that, as long as p_k converges to one at least as fast as $1-1/k^{1+\zeta}$, where $\zeta > 0$ is 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. An interesting future research direction is to extend the DQN with idling algorithm in the direction that, instead of utilizing a pre-defined increasing sequence of activation probabilities, the activation law is controlled locally to trade-off communication and computation.

References

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

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

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

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

[5] E. Wei, A. Ozdaglar, A. Jadbabaie, "A distributed Newton method for network utility maximization–I: Algorithm," *IEEE Transactions on Automatic Control,* vol. 58, no. 9, pp. 2162–2175, 2013.

[6] G. Zhang, R. Heusdens, "Distributed Optimization Using the Primal-Dual Method of Multipliers," *IEEE Trans. Sig. Info. Proc. Net.*, vol. 4, no. 1, March 2018.

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

[8] G. Hug, S. Kar, C. Wu, "Consensus + Innovations Approach for Distributed Multiagent Coordination in a Microgrid," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1893-1903, 2015.

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

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

[11] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," 20th Int. Conf. Artif. Intell. Statist., pp. 509-517, 2017.

[12] I. Almeida, J. Xavier, "DJAM: Distributed Jacobi Asynchronous Method for Learning Personal Models," *IEEE Sig. Proc. Let.*, vol. 25, no. 9, pp. 1389–1392, 2018.

[13] K. Yuan, Q. Ling, and W. Yin, "On the Convergence of Decentralized Gradient Descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835-1854, 2016.

[14] D. Bajović, D. Jakovetić, N. Krejić, N. Krklec Jerinkić, "Newton-like method with diagonal correction for distributed optimization," *SIAM J. Opt.*, vol. 27, no. 2, pp. 1171-1203. 2017.

[15] D. Jakovetic, D. Bajovic, N. Krejic, N. Krklec Jerinkic, "Distributed Gradient Methods with Variable Number of Working Nodes," *IEEE Trans. Signal Processing*, vol. 64, no. 15, pp. 4080-4095, 2016.

[16] M. P. Friedlander, M. Schmidt, "Hybrid deterministic-stochastic methods for data fitting," *SIAM Journal on Scientific Computing*, vol. 34, pp. 1380-1405, 2012.

[17] 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.

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

[19] A. Mokhtari, W. Shi, Q. Ling, A. Ribeiro, "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.

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

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

[22] X. Zhao, A. H. Sayed, "Asynchronous Adaptation and Learning Over Networks–Part I: Modeling and Stability Analysis," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, Feb. 2015.

[23] X. Zhao, A. H. Sayed, "Asynchronous Adaptation and Learning Over Networks–Part II: Performance Analysis," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, Feb. 2015.

[24] T. Wu, K. Yuan, Q. Ling, W. Yin, A. H, Sayed, "Decentralized Consensus Optimization with Asynchrony and Delays," *IEEE Transactions on Signal and Information Processing over Networks*, to appear, 2017, DOI: 10.1109/TSIPN.2017.2695121.

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

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

[27] E. Isufi, A. Loukas, A. Simonetto, G. Leus, "Filtering Random Graph Processes Over Random Time-Varying Graphs," *IEEE Trans. Sig. Proc.*, vol. 65, no. 16, pp. 4406-4421, 2017.

[28] F. Mansoori, E. Wei, "Superlinearly Convergent Asynchronous Distributed Network Newton Method," 2017, available at https://arxiv.org/abs/1705.03952.

[29] M. Eisen, A. Mokhtari, A. Ribeiro, "Decentralized quasi-Newton methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2613–2628, May 2017.

[30] M. Eisen, A. Mokhtari, A. Ribeiro, "A Decentralized Quasi-Newton Method for Dual Formulations of Consensus Optimization," *IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, Dec. 2016.

[31] M. Eisen, A. Mokhtari, A. Ribeiro, "An asynchronous quasi-Newton method for consensus optimization," *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Washington, DC, VA, USA, Dec. 2016.

[32] D. Bajović, D. Jakovetić, N. Krejić, N. Krklec Jerinkić, "Distributed first and second order methods with increasing number of working nodes," *IEEE Global Conference on Signal and Information Processing*, Washington DC, VA, USA, Dec. 2016.

[33] N. Krklec Jerinkic, D. Jakovetic, N. Krejic, D, Bajovic, "Distributed second order methods with increasing number of working nodes," available at: https://arxiv.org/abs/1709.01307.

[34] C. Eksin, A. Ribeiro, "Distributed network optimization with heuristic rational agents," *IEEE Trans. Signal Processing*, vol. 60, no. 10, pp. 5396-5411, 2012.

[35] S. Ram, A. Nedic, V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516-545, 2011.

[36] N. Krejić, N. Krklec Jerinkić, "Nonmonotone line search methods with variable sample size," *Numerical Algorithms*, vol. 68, pp. 711–739, 2015.

[37] A. Tahbaz-Salehi, A. Jadbabaie, "On consensus over random networks," 44th Annual Allerton Conference on Communication, Control, and Computing, 2006.