Network Newton–Part II: Convergence Rate and Implementation

Aryan Mokhtari, Qing Ling and Alejandro Ribeiro

Abstract—**The use of network Newton methods for the decentralized optimization of a sum cost distributed through agents of a network is considered. Network Newton methods reinterpret distributed gradient descent as a penalty method, observe that the corresponding Hessian is sparse, and approximate the Newton step by truncating a Taylor expansion of the inverse Hessian. Truncating the series at K terms yields the NN-K that requires aggregating information from K hops away. Network Newton is introduced and shown to converge to the solution of the penalized objective function at a rate that is at least linear in a companion paper [3]. The contributions of this work are: (i) To complement the convergence analysis by studying the methods' rate of convergence. (ii) To introduce adaptive formulations that converge to the optimal argument of the original objective. (iii) To perform numerical evaluations of NN-K methods. The convergence analysis relates the behavior of NN-K with the behavior of (regular) Newton's method and shows that the method goes through a quadratic convergence phase in a specific interval. The length of this quadratic phase grows with K and can be made arbitrarily large. The numerical experiments corroborate reductions in the number of iterations and the communication cost that are necessary to achieve convergence relative to distributed gradient descent.**

Index Terms—**Multi-agent network, distributed optimization, Newton's method.**

I. Introduction

In decentralized optimization problems a group of agents is tasked with minimizing a sum cost when each of them has access to a specific summand. They do so by working through subsequent rounds of local processing and variable exchanges with adjacent peers. This architecture arises naturally in decentralized control [4]–[6] as well as in wireless [7], [8] and sensor networks [9]–[11]. In these problems agents have access to local information but want to achieve a common goal, administer a shared resource, or estimate the state of a global environment. Decentralized optimization is also relevant to large scale machine learning [12]–[14], where problems are not inherently distributed but are divvied up to process big datasets.

Irrespectively of the specific application, various methods have been developed for decentralized optimization. These include distributed gradient descent (DGD) [15]–[18] as well as distributed implementations of the alternating direction method of multipliers [9], [19]–[21] and dual averaging [22], [23]. At the core of all of these methods lies a gradient descent iteration that endows them with their convergence properties, but also results in large convergence times for problems with poor conditioning. In a companion paper, we introduced the network Newton family of decentralized optimization methods that incorporates second order information into DGD iterations to accelerate convergence [3]. Methods in the network Newton family are derived by introducing a penalty formulation of distributed optimization objectives (Section II) for which the resulting Hessians have the same sparsity pattern of the underlying network (Section II-A). The Hessian inverse that is necessary to compute Newton steps is then expressed as a Taylor series expansion that we truncate at K terms to obtain the Kth member of the network Newton family – which we abbreviate as NN-K. These truncations can be computed in a distributed manner by aggregating information from, at most, K hops away.

The network Newton methods have been proven to converge to the optimal solution of the penalized objective at a rate that is at least linear [3]. The main goal of this paper is to complete the convergence analysis of NN-K by studying its rate of convergence (Section III). We show that for all iterations except the first few, a weighted gradient norm associated with NN-K iterates follows a decreasing path akin to the path that would be followed by regular Newton iterates (Lemma 2). The only difference between these residual paths is that the NN-K path contains a term that captures the error of the Hessian inverse approximation. Leveraging this similarity, it is possible to show that the rate of convergence is quadratic in a specific interval whose length depends on the order K of the selected network Newton method (Theorem 2). Existence of this quadratic convergence phase explains why NN-K methods converge faster than DGD – as we indeed observe in numerical analyses. It is also worth remarking that the error in the Hessian inverse approximation can be made arbitrarily small by increasing the method's order K and, as a consequence, the quadratic phase can be made arbitrarily large.

Given that NN-K solves the minimization of a penalized objective, it converges to a point that is close to the optimum. To achieve exact convergence we introduce an adaptive version of NN-K – which we term ANN-K – that uses a sequence of increasing penalty coefficients to achieve exact convergence to the optimal solution (Section IV). We wrap up the paper with numerical analyses. We first demonstrate the advantages of NN-K relative to DGD for the minimization of a family of quadratic objective functions with varying condition number and network connectivity (Section V). As expected, NN-K methods reduce convergence times by substantive factors when the objective functions are not well conditioned. Advantages

Work supported by NSF CAREER CCF-0952867 and ONR N00014-12-1-0997. A. Mokhtari and A. Ribeiro are with the Department of Electrical and Systems Engineering, University of Pennsylvania, 200 South 33rd Street, Philadelphia, PA 19104. Q. Ling is with the Department of Automation, University of Science and Technology of China, Hefei, Anhui 230026, China. Email at: {aryanm, aribeiro}@seas.upenn.edu, qingling@mail.ustc.edu.cn. Part of the results in this paper appeared in [1] and [2]. This paper expands the results and presents convergence proofs that are referenced in [1] and [2].

in terms of communication cost are less marked because NN-K aggregates information from K-hop neighborhoods but still substantial. Network Newton is also applied to solve a logistic regression problem. The results reinforce the conclusions reached for the quadratic objective problem (Section V-B). Numerical analyses also show that network Newton methods with $K = 1$ and $K = 2$ tend to work best when measured in terms of overall communication cost. Numerical experiments for ANN-K illustrate the tradeoffs that appear in the selection of the initial penalty coefficient and its rate of change (Section V-A). The paper closes with concluding remarks (Section VI).

Notation. Vectors are written as $\mathbf{x} \in \mathbb{R}^n$ and matrices as $\mathbf{A} \in \mathbb{R}^{n\times n}$. Given n vectors $\mathbf{x}_i$, the vector $\mathbf{y} = [\mathbf{x}_1; \dots; \mathbf{x}_n]$ represents a stacking of the elements of each individual $\mathbf{x}_i$. The null space of matrix $\mathbf{A}$ is denoted by null$(\mathbf{A})$ and the span of a vector by span$(\mathbf{x})$. The i-th eigenvalue of matrix $\mathbf{A}$ is denoted by $\mu_i(\mathbf{A})$. For matrices $\mathbf{A}$ and $\mathbf{B}$ their Kronecker product is denoted as $\mathbf{A} \otimes \mathbf{B}$. The gradient of a function $f(\mathbf{x})$ is denoted as $\nabla f(\mathbf{x})$ and the Hessian matrix is denoted by $\nabla^2 f(\mathbf{x})$.

II. Algorithm definition

We consider a connected and symmetric network with n agents generically indexed by $i = 1, \dots, n$. The network is specified by the n neighborhood sets $\mathcal{N}_i$, each of which is defined as the group of nodes that are connected to i. Nodes have access to strongly convex local objective functions $f_i(\mathbf{x})$, but cooperate to minimize the global cost $f(\mathbf{x}) := \sum_{i=1}^n \, f_i(\mathbf{x})$,

$$\mathbf{x}^* \; := \; \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) \; = \; \operatorname*{argmin}_{\mathbf{x}} \sum_{i=1}^n f_i(\mathbf{x}). \tag{1}$$

To rewrite the global problem in a form that is suitable for distributed implementation we define local variables $\mathbf{x}_i \in \mathbb{R}^p$ and rewrite the cost to be minimized as $\sum_{i=1}^n \, f_i(\mathbf{x}_i)$. For a problem formulation equivalent to (1), we have to further add the restriction that local variables $\mathbf{x}_i$ be the same as neighboring variables $\mathbf{x}_j$ with $j \in \mathcal{N}_i$,

$$\begin{aligned} \{\mathbf{x}_i^*\}_{i=1}^n \; &:= \; \operatorname*{argmin}_{\mathbf{x}} \; \sum_{i=1}^n \; f_i(\mathbf{x}_i), \\ &\text{s.t. } \mathbf{x}_i = \mathbf{x}_j, \quad \text{for all } i, j \in \mathcal{N}_i. \end{aligned} \tag{2}$$

The optimization problems in (2) and (1) are equivalent in the sense that $\mathbf{x}_i^* = \mathbf{x}^*$ for all i. This has to be true because the feasible set of (2) is restricted to configurations in which all variables $\mathbf{x}_i$ are equal given that the network is connected.

The constraints $\mathbf{x}_i = \mathbf{x}_j$ imposed for all i and all $j \in \mathcal{N}_i$ are a way of making all local variables equal but there are other alternatives. The one that is germane to this paper consists of introducing weights w_{ij} that we group in the matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$. The weights w_{ij} are chosen so that the matrix $\mathbf{W}$ is symmetric, row stochastic, and such that the null space of $\mathbf{I} - \mathbf{W}$ is the span of the all one vector $\mathbf{1}$

$$\mathbf{W}^T = \mathbf{W}, \quad \mathbf{W}\mathbf{1} = \mathbf{1}, \quad \text{null}(\mathbf{I} - \mathbf{W}) = \text{span}(\mathbf{1}). \tag{3}$$

We further define the extended weight matrix

$$\mathbf{Z} := \mathbf{W} \otimes \mathbf{I} \in \mathbb{R}^{np\times np} \tag{4}$$

as the Kronecker product of the weight matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$ and the identity matrix $\mathbf{I} \in \mathbb{R}^{p\times p}$ as well as the vector $\mathbf{y} := [\mathbf{x}_1; \dots; \mathbf{x}_n]$ as the concatenation of the local vectors $\mathbf{x}_i$. It follows that the equality constraint $\mathbf{Z} = \mathbf{W} \otimes \mathbf{I}$ can be satisfied if and only if all the local variables are equal, i.e., if and only if $\mathbf{x}_1 = \dots = \mathbf{x}_n$. Indeed, since the null space of $\mathbf{I} - \mathbf{W}$ is null$(\mathbf{I} - \mathbf{W})$ = span$(\mathbf{1})$ as per the last condition in (3), the null space of $\mathbf{I} - \mathbf{Z}$ must be null$(\mathbf{I} - \mathbf{Z})$ = span$(\mathbf{1} \otimes \mathbf{I})$. Thus, vectors $\mathbf{y} := [\mathbf{x}_1; \dots; \mathbf{x}_n]$ in the null space of $\mathbf{I} - \mathbf{Z}$, which, by definition, are the only ones that satisfy the equality $(\mathbf{I} - \mathbf{Z})\mathbf{y} = \mathbf{0}$ are multiples of $\mathbf{1} \otimes \mathbf{I}$ and therefore satisfy $\mathbf{x}_1 = \dots = \mathbf{x}_n$.

Further observe that the matrix $\mathbf{Z}$, being stochastic and symmetric, is positive semidefinite. Consequently, the square root matrix $(\mathbf{I} - \mathbf{Z})^{1/2}$ exists and has the same null space of $\mathbf{I} - \mathbf{Z}$. It then follows that $(\mathbf{I} - \mathbf{Z})^{1/2}\mathbf{y} = \mathbf{0}$ if and only if the components of $\mathbf{y}$ satisfy $\mathbf{x}_1 = \dots = \mathbf{x}_n$. In turn, this implies that the optimization problem in (2) is equivalent to

$$\begin{aligned} \tilde{\mathbf{y}}^* \; &:= \; \operatorname*{argmin}_{\mathbf{x}} \; \sum_{i=1}^n \; f_i(\mathbf{x}_i), \\ &\text{s.t.} \quad (\mathbf{I} - \mathbf{Z})^{1/2}\mathbf{y} = \mathbf{0}. \end{aligned} \tag{5}$$

Here, we solve (5) using a penalized version of the objective function. To do so we consider a given penalty coefficient $1/\alpha$ and the squared norm penalty function $(1/2)\|(\mathbf{I} - \mathbf{Z})^{1/2}\mathbf{y}\|^2 = (1/2)\mathbf{y}^T(\mathbf{I} - \mathbf{Z})\mathbf{y}$ associated with the constraint $(\mathbf{I} - \mathbf{Z})^{1/2}\mathbf{y} = \mathbf{0}$. With penalty function and coefficient so defined, we can now introduce the penalized objective $F(\mathbf{y}) := (1/2)\mathbf{y}^T(\mathbf{I} - \mathbf{Z})\mathbf{y} + \alpha\sum_{i=1}^n f_i(\mathbf{x}_i)$ and the penalized optimization problem

$$\begin{aligned} \mathbf{y}^* &:= \operatorname{argmin} \; F(\mathbf{y}) \\ &:= \operatorname{argmin} \frac{1}{2} \; \mathbf{y}^T(\mathbf{I} - \mathbf{Z}) \; \mathbf{y} + \alpha \sum_{i=1}^n f_i(\mathbf{x}_i). \end{aligned} \tag{6}$$

As the penalty coefficient $1/\alpha$ grows, or, equivalently, as α vanishes, the optimal argument $\mathbf{y}^*$ of the penalized problem (6) converges towards the optimal argument $\tilde{\mathbf{y}}^*$ of (2) and (5). In that sense, (6) is a reasonable proxy for (2), (5), and the equivalent original formulation in (1).

The property that makes the penalized problem in (6) amenable to distributed implementation is that its gradients can be computed by exchanging information between neighboring nodes. This property is the basis for the development of the DGD method of [15] and the NN method of [3]. In the following section we study the idea of using Newton's method for solving (6).

A. Newton's method and Hessian splitting

We proceed to minimize the penalized objective function $F(\mathbf{y})$ in (6) using Newton's method. The Newton update with stepsize ϵ for function $F(\mathbf{y})$ can be written as

$$\mathbf{y}_{t+1} = \mathbf{y}_t - \epsilon \, \nabla^2 F(\mathbf{y}_t)^{-1} \nabla F(\mathbf{y}_t), \tag{7}$$

where $\nabla^2 F(\mathbf{y}_t)$ and $\nabla F(\mathbf{y}_t)$ are the Hessian and gradient of function F evaluated at point $\mathbf{y}_t$, respectively.

To compute the gradient $\nabla F(\mathbf{y}_t)$ we introduce the vector $\mathbf{h}(\mathbf{y}) := [\nabla f_1(\mathbf{x}_1); \dots; \nabla f_n(\mathbf{x}_n)]$ that concatenates the local gradients $\nabla f_i(\mathbf{x}_i)$. Given the definition of the penalized function $F(\mathbf{y})$ in (6) it follows that the gradient of $F(\mathbf{y})$ at $\mathbf{y} = \mathbf{y}_t$ is

$$\mathbf{g}_t \ := \ \nabla F(\mathbf{y}_t) \ = \ (\mathbf{I} - \mathbf{Z})\mathbf{y}_t + \alpha \mathbf{h}(\mathbf{y}_t). \qquad (8)$$

The computation of the gradient $\mathbf{g}_t$ can be distributed through the network because $\mathbf{Z}$ has the sparsity pattern of the graph. Specifically, define the local gradient component $\mathbf{g}_{i,t}$ as the ith element of the gradient $\mathbf{g}_t = [\mathbf{g}_{i,t}; \ldots; \mathbf{g}_{i,t}]$ and recall that $\mathbf{x}_{i,t}$ and $\mathbf{x}_{i,t+1}$ are the ith components of the vector $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$. The local gradient component at node i is given by

$$\mathbf{g}_{i,t} = (1 - w_{ii})\mathbf{x}_{i,t} - \sum_{j \in \mathcal{N}_i} w_{ij}\mathbf{x}_{j,t} + \alpha \nabla f_i(\mathbf{x}_{i,t}). \qquad (9)$$

Using (9), node i can compute its local gradient using its local iterate $\mathbf{x}_{i,t}$, the gradient of the local function $\nabla f_i(\mathbf{x}_{i,t})$ and the $\mathbf{x}_{j,t}$ iterates of its neighbors $j \in \mathcal{N}_i$.

To implement Newton's method as defined in (7) we also need to compute the Hessian $\mathbf{H}_t := \nabla^2 F(\mathbf{y}_t)$ of the penalized objective. Start by differentiating twice the objective function F in (6) in order to write the Hessian $\mathbf{H}_t$ as

$$\mathbf{H}_t := \nabla^2 F(\mathbf{y}_t) = \mathbf{I} - \mathbf{Z} + \alpha \mathbf{G}_t, \qquad (10)$$

where the matrix $\mathbf{G}_t \in \mathbb{R}^{np \times np}$ is a block diagonal matrix formed by blocks $\mathbf{G}_{ii,t} \in \mathbb{R}^{p \times p}$ containing the Hessian of the ith local function,

$$\mathbf{G}_{ii,t} = \nabla^2 f_i(\mathbf{x}_{i,t}). \qquad (11)$$

It follows from (10) and (11) that the Hessian $\mathbf{H}_t$ is block sparse with blocks $\mathbf{H}_{ij,t} \in \mathbb{R}^{p \times p}$ having the sparsity pattern of $\mathbf{Z}$, which is the sparsity pattern of the graph. The diagonal blocks are of the form $\mathbf{H}_{ii,t} = (1 - w_{ii})\mathbf{I} + \alpha \nabla^2 f_i(\mathbf{x}_{i,t})$ and the off diagonal blocks are not null only when $j \in \mathcal{N}_i$ in which case $\mathbf{H}_{ij,t} = w_{ij}\mathbf{I}$.

Recall that for the Newton update in (7), the Hessian inverse $\nabla^2 F(\mathbf{y}_t)^{-1} = \mathbf{H}_t^{-1}$ evaluated at $\mathbf{y} = \mathbf{y}_t$ is required not the Hessian $\mathbf{H}_t$. While the Hessian $\mathbf{H}_t$ is sparse, the inverse $\mathbf{H}_t^{-1}$ is not necessarily sparse. Therefore, the Hessian inverse is not necessarily computable in a decentralized setting. To overcome this problem we split the diagonal and off diagonal blocks of $\mathbf{H}_t$ and rely on the Taylor's expansion of the inverse $\mathbf{H}_t^{-1}$. To be precise, write $\mathbf{H}_t = \mathbf{D}_t - \mathbf{B}$ where the matrix $\mathbf{D}_t$ is defined as

$$\mathbf{D}_t := \alpha \mathbf{G}_t + 2\ (\mathbf{I} - \text{diag}(\mathbf{Z})) := \alpha \mathbf{G}_t + 2\ (\mathbf{I} - \mathbf{Z}_d). \qquad (12)$$

In the second equality we defined $\mathbf{Z}_d := \text{diag}(\mathbf{Z})$ for future reference. Observe that the matrix $\mathbf{I} - \mathbf{Z}_d$ is positive definite because in a connected network the local weights are $w_{ii} < 1$. The block diagonal matrix $\mathbf{G}_t$ is also positive definite because the local functions are assumed strongly convex. it follows from these two observations that the matrix $\mathbf{D}_t$ is block diagonal and positive definite. Further note that the ith diagonal block $\mathbf{D}_{ii,t} \in \mathbb{R}^p$ of $\mathbf{D}_t$ can be computed and stored by node i as $\mathbf{D}_{ii,t} = \alpha \nabla^2 f_i(\mathbf{x}_{i,t}) + 2(1 - w_{ii})\mathbf{I}$ using local information only. To have $\mathbf{H}_t = \mathbf{D}_t - \mathbf{B}$ we must define $\mathbf{B} := \mathbf{D}_t - \mathbf{H}_t$. Considering the definitions of $\mathbf{H}_t$ and $\mathbf{D}_t$ in (10) and (12), respectively, it follows that

$$\mathbf{B} = \mathbf{I} - 2\mathbf{Z}_d + \mathbf{Z}. \qquad (13)$$

Observe that $\mathbf{B}$ is independent of time and only depends on the weight matrix $\mathbf{Z}$. As in the case of the Hessian $\mathbf{H}_t$, the matrix $\mathbf{B}$ is block sparse with blocks $\mathbf{B}_{ij} \in \mathbb{R}^{p \times p}$ having the sparsity pattern of $\mathbf{Z}$, which is the sparsity pattern of the graph. Since $\mathbf{B}$ is block sparse, node i can compute the diagonal block $\mathbf{B}_{ii} = (1 - w_{ii})\mathbf{I}$ and the off diagonal blocks $\mathbf{B}_{ij} = w_{ij}\mathbf{I}$ using local information about its own weights.

Proceed now to factor $\mathbf{D}_t^{1/2}$ from both sides of the splitting relationship to write $\mathbf{H}_t = \mathbf{D}_t^{1/2}(\mathbf{I} - \mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2})\mathbf{D}_t^{1/2}$. This decomposition implies that the Hessian inverse $\mathbf{H}_t^{-1}$ can be computed from the Taylor series expansion $(\mathbf{I} - \mathbf{X})^{-1} = \sum_{j=0}^{\infty} \mathbf{X}^j$ with $\mathbf{X} = \mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}$. Therefore, we can write

$$\mathbf{H}_t^{-1} = \mathbf{D}_t^{-1/2} \sum_{k=0}^{\infty} \left(\mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}\right)^k \ \mathbf{D}_t^{-1/2}. \qquad (14)$$

The sum in (14) converges if the absolute value of all the eigenvalues of the matrix $\mathbf{D}^{-1/2}\mathbf{B}\mathbf{D}^{-1/2}$ are strictly less than 1 – we prove that this is true in Proposition 1. Truncations of this convergent series are utilized to define the family of Network Newton methods in the following section.

B. Network Newton

Network Newton is defined as a family of algorithms that rely on truncations of the series in (14). The Kth member of this family, NN-K, considers the first $K+1$ terms of the series to define the approximate Hessian inverse

$$\hat{\mathbf{H}}_t^{(K)^{-1}} := \mathbf{D}_t^{-1/2} \ \sum_{k=0}^{K} \left(\mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}\right)^k \ \mathbf{D}_t^{-1/2}. \qquad (15)$$

NN-K uses the approximate Hessian $\hat{\mathbf{H}}_t^{(K)^{-1}}$ as a curvature correction matrix that is used in lieu of the exact Hessian inverse $\mathbf{H}^{-1}$ to estimate the Newton step. I.e., instead of descending along the Newton step $\mathbf{d}_t := -\mathbf{H}_t^{-1}\mathbf{g}_t$ we descend along the NN-K step $\mathbf{d}_t^{(K)} := -\hat{\mathbf{H}}_t^{(K)^{-1}}\mathbf{g}_t$, which we intend as an approximation of $\mathbf{d}_t$. Using the explicit expression for $\hat{\mathbf{H}}_t^{(K)^{-1}}$ in (15) we write the NN-K step as

$$\mathbf{d}_t^{(K)} = -\ \mathbf{D}_t^{-1/2} \ \sum_{k=0}^{K} \left(\mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}\right)^k \ \mathbf{D}_t^{-1/2} \ \mathbf{g}_t, \qquad (16)$$

where, we recall, the vector $\mathbf{g}_t$ is the gradient of the objective function $F(\mathbf{y})$ defined in (8). The NN-K update formula can then be written as

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \epsilon \ \mathbf{d}_t^{(K)}. \qquad (17)$$

The algorithm defined by recursive application of (17) can be implemented in a distributed manner. Specifically, define the components $\mathbf{d}_{i,t}^{(K)} \in \mathbb{R}^p$ of the NN-K step $\mathbf{d}_t^{(K)} = [\mathbf{d}_{1,t}^{(K)}; \ldots; \mathbf{d}_{n,t}^{(K)}]$ and rewrite (17) componentwise as

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \epsilon \ \mathbf{d}_{i,t}^{(K)}. \qquad (18)$$

To determine the step components $\mathbf{d}_{i,t}^{(K)}$ in (18) observe that considering the definition of the NN-K descent direction in (16), Network Newton descent directions can be computed by the recursive expression

$$\mathbf{d}_t^{(k+1)} = \mathbf{D}_t^{-1}\mathbf{B}\mathbf{d}_t^{(k)} - \mathbf{D}_t^{-1}\mathbf{g}_t = \mathbf{D}_t^{-1}\left(\mathbf{B}\mathbf{d}_t^{(k)} - \mathbf{g}_t\right). \quad (19)$$

If we expand the product $\mathbf{B}\mathbf{d}_t^{(k)}$ as a sum and utilize the fact that the blocks of the matrix $\mathbf{B}$ have the sparsity pattern of the graph, we can separate (19) into the following componentwise recursions

$$\mathbf{d}_{i,t}^{(k+1)} = \mathbf{D}_{ii,t}^{-1}\left[\sum_{j\in\mathcal{N}_i, j=i} \mathbf{B}_{ij}\mathbf{d}_{j,t}^{(k)} - \mathbf{g}_{i,t}\right]. \quad (20)$$

That the matrix $\mathbf{B}$ is block-sparse permits writing the sum in (20) as a sum over neighbors, instead of a sum across all nodes.

In (20), the matrix blocks $\mathbf{D}_{ii,t} = \alpha\nabla^2 f_i(\mathbf{x}_{i,t}) + 2(1-w_{ii})\mathbf{I}$, $\mathbf{B}_{ii} = (1-w_{ii})\mathbf{I}$, and $\mathbf{B}_{ij} = w_{ij}\mathbf{I}$ are evaluated and stored at node i. The gradient component $\mathbf{g}_{i,t}$ is also stored and computed at i upon being communicated the values of neighboring iterates $\mathbf{x}_{j,t}$ [cf. (9)]. Thus, if the NN-k step components $\mathbf{d}_{j,t}^{(k)}$ are available at neighboring nodes j, node i can determine the NN-$(k+1)$ step component $\mathbf{d}_{i,t}^{(k+1)}$ upon being communicated that information. We use this property to embed an iterative computation of the NN-K step inside the NN-K recursion in (18). For each iteration index t, we compute the local component of the NN-0 step $\mathbf{d}_{i,t}^{(0)} = -\mathbf{D}_{ii,t}^{-1}\mathbf{g}_{i,t}$. Upon exchanging this information with neighbors we use (20) to determine the NN-1 step components $\mathbf{d}_{i,t}^{(1)}$. These can be exchanged and plugged in (20) to compute $\mathbf{d}_{i,t}^{(2)}$. Repeating this procedure K times, nodes end up having determined their NN-K step component $\mathbf{d}_{i,t}^{(K)}$. They use this step to update $\mathbf{x}_{i,t}$ according to (18) and move to the next iteration. We analyze the convergence rate of the resulting algorithm in Section III and develop a numerical analysis in Section V.

Remark 1 By trying to approximate the Newton step, NN-K ends up reducing the number of iterations required for convergence. Furthermore, the larger K is, the closer that the NN-K step gets to the Newton step, and the faster NN-K converges. We will justify these assertions both, analytically in Section III, and numerically in Section V. It is important to observe, however, that reducing the number of iterations reduces the computational cost but not necessarily the communication cost. In DGD, each node i shares its vector $\mathbf{x}_{i,t} \in \mathbb{R}^p$ with each of its neighbors $j \in \mathcal{N}_i$. In NN-K, node i exchanges not only the vector $\mathbf{x}_{i,t} \in \mathbb{R}^p$ with its neighboring nodes, but it also communicates iteratively the local components of the descent directions $\{\mathbf{d}_{i,t}^{(k)}\}_{k=0}^{K-1} \in \mathbb{R}^p$ so as to compute the descent direction $\mathbf{d}_{i,t}^{(K)}$. Therefore, at each iteration, node i sends $|\mathcal{N}_i|$ vectors of size p to the neighboring nodes in DGD, while in NN-K it sends $(K+1)|\mathcal{N}_i|$ vectors of the same size. Unless the original problem is well conditioned, NN-K also reduces total communication cost until convergence, even though the cost of each individual iteration is larger. However, the use of large K is unwarranted because the added benefit of better approximating the Newton step does not compensate the increase in communication cost.

III. Convergence Rate

Linear convergence of the sub optimality sequence $F(\mathbf{y}_t) - F(\mathbf{y}^*)$ associated with NN-K iterates $\mathbf{y}_t$ has been proven in [3]. We improve this result by showing that regardless of the choice of K, the rate of convergence is quadratic in a specific interval. To prove this result we utilize some of the results in [3] which we repeat here for completeness. We start by restating assumptions that are necessary for the convergence analysis.

Assumption 1 *There exists constants $0 \leq \delta \leq \Delta < 1$ that lower and upper bound the diagonal weights for all i,*

$$0 \leq \delta \leq w_{ii} \leq \Delta < 1, \qquad i = 1, \ldots, n. \quad (21)$$

Assumption 2 *The local objective functions $f_i(\mathbf{x})$ are twice differentiable and the eigenvalues of the local objective function Hessians are bounded with positive constants $0 < m \leq M < \infty$, i.e.*

$$m\mathbf{I} \preceq \nabla^2 f_i(\mathbf{x}) \preceq M\mathbf{I}. \quad (22)$$

Assumption 3 *The local objective function Hessians $\nabla^2 f_i(\mathbf{x})$ are Lipschitz continuous with respect to the Euclidian norm with parameter L. I.e., for all $\mathbf{x}, \hat{\mathbf{x}} \in \mathbb{R}^p$, it holds*

$$\left\|\nabla^2 f_i(\mathbf{x}) - \nabla^2 f_i(\hat{\mathbf{x}})\right\| \ \leq\ L\ \|\mathbf{x} - \hat{\mathbf{x}}\|. \quad (23)$$

The upper bound $\Delta < 1$ on the local weights w_{ii} in Assumption 1 exits for connected networks. The non-negative lower bound δ on the local weights w_{ii} is more a definition than a constraint since we may have $\delta = 0$. Strong convexity of the local objective functions f_i enforces the existence of a lower bound m for the eigenvalues of the local Hessian $\nabla^2 f_i$ as in (22). The upper bound M for the eigenvalues of local objective function Hessians $\nabla^2 f_i(\mathbf{x})$ in Assumption 2 is equivalent to the assumption that local gradients $\nabla f_i(\mathbf{x})$ are Lipschitz continuous with parameter M. Assumption 3 states that the local objective function Hessians are Lipschitz continuous with parameter L. A particular consequence of this assumption is that the penalized objective function Hessian $\mathbf{H}(\mathbf{y}) := \nabla^2 F(\mathbf{y})$ is also Lipschitz continuous with parameter αL – see Lemma 1 of [3]. I.e. for all $\mathbf{y}, \hat{\mathbf{y}} \in \mathbb{R}^{np}$ it holds,

$$\|\mathbf{H}(\mathbf{y}) - \mathbf{H}(\hat{\mathbf{y}})\| \leq \alpha L \|\mathbf{y} - \hat{\mathbf{y}}\|, \quad (24)$$

Recall that the block diagonal matrix $\mathbf{D}_t$, being the sum of positive definite $\alpha\mathbf{G}_t$ and $2(\mathbf{I} - \mathbf{Z}_d)$, is positive definite and, therefore, invertible. Further recall that the matrix $\mathbf{B}$, being symmetric and doubly stochastic, has eigenvalues that lie between 0 and 1 and is therefore positive semidefinite. These facts can be used to prove that the eigenvalues of the matrix $\mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}$ must be nonnegative and strictly smaller than 1 as we state next [3, Proposition 2].

Proposition 1 *Consider the definitions of matrices $\mathbf{D}_t$ in (12) and $\mathbf{B}$ in (13). If Assumptions 1 and 2 hold true, the matrix $\mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2}$ is positive semidefinite and the eigenvalues are bounded above by a constant $\rho < 1$*

$$\mathbf{0} \ \preceq\ \mathbf{D}_t^{-1/2}\mathbf{B}\mathbf{D}_t^{-1/2} \ \preceq\ \rho\mathbf{I}, \quad (25)$$

where $\rho := 2(1-\delta)/(2(1-\delta) + \alpha m)$.

The result in Proposition 1 makes the expansion in (14) valid and is used in subsequent proofs. These proofs also rely on guarantees that the eigenvalues of the approximate Hessian inverse $\hat{\mathbf{H}}_t^{(K)^{-1}}$ are positive and finite for all choices of K and for all steps t. We state these guarantees next [3, Lemma 2].

Lemma 1 *Consider the NN-K method as defined by* (16)-(17) *with the gradient* $\mathbf{g}_t$ *as defined in* (8) *and the matrices* $\mathbf{B}$ *and* $\mathbf{D}_t$ *defined as in* (11)-(13)*. If Assumptions 1 and 2 hold true, the eigenvalues of approximate Hessian inverse* $\hat{\mathbf{H}}_t^{(K)^{-1}}$ *are bounded as*

$$\lambda\mathbf{I} \preceq \hat{\mathbf{H}}_t^{(K)^{-1}} \preceq \Lambda\mathbf{I}, \tag{26}$$

where constants λ *and* Λ *are defined as*

$$\lambda := \frac{1}{2(1-\delta)+\alpha M} \quad \text{and} \quad \Lambda := \frac{1-\rho^{K+1}}{(1-\rho)(2(1-\Delta)+\alpha m)}. \tag{27}$$

The lower bound $\lambda > 0$ for the eigenvalues of the approximate Hessian inverse $\hat{\mathbf{H}}_t^{(K)^{-1}}$ guarantees decrement in each network Newton iteration. The upper bound $\Lambda < \infty$ ensures that the norm of the network Newton step $\|\hat{\mathbf{H}}_t^{(K)^{-1}}\mathbf{g}_t\|$ is bounded by a factor proportional to the gradient norm $\|\mathbf{g}_t\|$. Both of these results are necessary to show that the network Newton direction $\hat{\mathbf{H}}_t^{(K)^{-1}}\mathbf{g}_t$ is a descent direction. This is claimed to be true in the following theorem [3, Theorem 1].

Theorem 1 *Consider the objective function* $F(\mathbf{y})$ *as introduced in* (6) *and the NN-K method as defined by* (16)-(17) *with the gradient* $\mathbf{g}_t$ *as defined in* (8) *and the matrices* $\mathbf{B}$ *and* $\mathbf{D}_t$ *defined as in* (11)-(13)*. If the stepsize* ϵ *is chosen as*

$$\epsilon = \min\left\{1\ ,\ \left[\frac{3m\lambda^{\frac{5}{2}}}{L\Lambda^3(F(\mathbf{y}_0)-F(\mathbf{y}^*))^{\frac{1}{2}}}\right]^{\frac{1}{2}}\right\} \tag{28}$$

and Assumptions 1, 2, and 3 hold true, the sequence $F(\mathbf{y}_t)$ *converges to the optimal argument* $F(\mathbf{y}^*)$ *at least linearly with constant* $1-\zeta$*. I.e.,*

$$F(\mathbf{y}_t)-F(\mathbf{y}^*) \le (1-\zeta)^t(F(\mathbf{y}_0)-F(\mathbf{y}^*)), \tag{29}$$

where the constant $0<\zeta<1$ *is explicitly given by*

$$\zeta := (2-\epsilon)\epsilon\alpha m\lambda - \frac{\alpha\epsilon^3 L\Lambda^3(F(\mathbf{y}_0)-F(\mathbf{y}^*))^{\frac{1}{2}}}{6\lambda^{\frac{3}{2}}}. \tag{30}$$

Theorem 1 establishes linear convergence of the sequence of penalized objective functions $F(\mathbf{y}_t)$ generated by NN-K to the optimal objective function $F(\mathbf{y}^*)$ – which implies convergence of $\mathbf{y}_t$ to the optimal argument $\mathbf{y}^*$. This result is identical to the convergence behavior of DGD as shown in, e.g., [17]. We expect to observe faster convergence for NN-K relative to DGD, since NN-K uses an approximation of the curvature of the penalized objective function F. In the following section we show that this expectation is fulfilled and that NN-K has a quadratic convergence phase regardless of the choice of K.

A. Quadratic convergence phase

To characterize convergence rate of NN-K, we first study the difference between this algorithm and (exact) Newton's method. In particular, the following lemma shows that the convergence of the norm of the weighted gradient $\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$ in NN-K is akin to the convergence of Newton's method with constant stepsize. The difference is the appearance of a term associated with the error of the Hessian inverse approximation as we formally state next.

Lemma 2 *Consider the NN-K method as defined by* (16)-(17) *with the gradient* $\mathbf{g}_t$ *as defined in* (8) *and the matrices* $\mathbf{B}$ *and* $\mathbf{D}_t$ *defined as in* (11)-(13)*. If Assumptions 1, 2, and 3 hold true, the sequence of weighted gradients* $\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}$ *satisfies*

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| \le \\ &\left(1-\epsilon+\epsilon\rho^{K+1}\right)\left[1+\Gamma_1(1-\zeta)^{\frac{(t-1)}{4}}\right]\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \\ &+\epsilon^2\Gamma_2\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2,\end{aligned} \tag{31}$$

where the constants Γ_1 *and* Γ_2 *are defined as*

$$\begin{aligned}\Gamma_1 &:= \frac{(\alpha\epsilon L\Lambda)^{1/2}(F(\mathbf{y}_0)-F(\mathbf{y}^*))^{1/4}}{\lambda^{3/4}(2(1-\Delta)+\alpha m)}, \\ \Gamma_2 &:= \frac{\alpha L\Lambda^2}{2\lambda(2(1-\Delta)+\alpha m)^{1/2}}.\end{aligned} \tag{32}$$

Proof: See Appendix A. ■

As per Lemma 2 the weighted gradient norm $\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\|$ is upper bounded by terms that are linear and quadratic on the weighted norm $\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$ associated with the previous iterate. This is akin to the gradient norm decrease of Newton's method with constant stepsize. To make this connection clearer, further note that for all except the first few iterations the term $\Gamma_1(1-\zeta)^{(t-1)/4} \approx 0$ is close to 0 and the relation in (31) can be simplified to

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\lesssim \left(1-\epsilon+\epsilon\rho^{K+1}\right)\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \\ &\quad+\epsilon^2\Gamma_2\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2.\end{aligned} \tag{33}$$

In (33), the coefficient in the linear term is reduced to $(1-\epsilon+\epsilon\rho^{K+1})$ and the coefficient in the quadratic term stays at $\epsilon^2\Gamma_2$. If, for discussion purposes, we set $\epsilon = 1$ as in Newton's quadratic phase, we see that the upper bound in (33) is further reduced to

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| \lesssim \rho^{K+1}\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\| + \Gamma_2\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|^2 \tag{34}$$

We do not obtain quadratic convergence as in Newton's method because of the term $\rho^{K+1}\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$. However, since the constant ρ (cf. Proposition 1) is smaller than 1 the term ρ^{K+1} can be made arbitrarily small by increasing the approximation order K. Equivalently, this means that by selecting K to be large enough, we can make the quadratic term in (34) dominant and observe a quadratic convergence phase. The boundaries of this quadratic convergence phase are formally determined in the following Theorem.

Theorem 2 *Consider the NN-K method as defined by* (16)-(17) *with the gradient* $\mathbf{g}_t$ *as defined in* (8) *and the matrices* $\mathbf{B}$ *and* $\mathbf{D}_t$ *defined as in* (11)-(13)*. Define the sequence* $\eta_t :=$

$[(1-\epsilon+\epsilon\rho^{K+1})(1+\Gamma_1(1-\zeta)^{(t-1)/4})]$ *and the time* t_0 *as the first time at which sequence* η_t *is smaller than 1, i.e.* $t_0 := \operatorname{argmin}_t\{t \mid \eta_t < 1\}$. *If Assumptions 1, 2, and 3 hold true, for all* $t \geq t_0$ *when the sequence* $\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$ *satisfies*

$$\frac{\sqrt{\eta_t}(1-\sqrt{\eta_t})}{\epsilon^2\Gamma_2} \leq \left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| < \frac{1-\sqrt{\eta_t}}{\epsilon^2\Gamma_2}\ , \tag{35}$$

the rate of convergence is quadratic in the sense that

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| \leq \frac{\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2. \tag{36}$$

Proof : Considering the definition of η_t we can rewrite the result of Lemma 2 as

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| \leq \eta_t\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| + \epsilon^2\Gamma_2\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2. \tag{37}$$

we use this expression to prove that the inequality in (36) holds true. To do so rearrange terms in the first inequality in (35) and write

$$\sqrt{\eta_t} \ \leq\ \frac{\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|. \tag{38}$$

Multiplying both sides of (38) by $\sqrt{\eta_t}\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$ yields

$$\eta_t\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \leq \frac{\sqrt{\eta_t}\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2. \tag{39}$$

Substituting $\eta_t\|\mathbf{D}_{t-1}^{-\frac{1}{2}}\mathbf{g}_t\|$ in (37) for its upper bound in (39) implies that

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\leq \frac{\sqrt{\eta_t}\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2 + \epsilon^2\Gamma_2\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2 \\ &= \frac{\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2. \end{aligned} \tag{40}$$

To verify quadratic convergence, it is necessary to prove that the sequence $\|\mathbf{D}_{i-1}^{-1/2}\mathbf{g}_i\|$ of weighted gradient norms is decreasing. For this to be true we must have

$$\frac{\epsilon^2\Gamma_2}{1-\sqrt{\eta_t}}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| < 1. \tag{41}$$

But (41) is true because we are looking at a range of gradients that satisfy the second inequality in (35). ■

As per Theorem 1 $\mathbf{y}_t$ is converging to $\mathbf{y}^*$ at a rate that is at least linear. Thus, the gradients $\mathbf{g}_t$ will be such that at some point in time they satisfy the rightmost inequality in (35). At that point in time, progress towards $\mathbf{y}^*$ proceeds at a quadratic rate as indicated by (36). This quadratic rate of progress is maintained until the leftmost inequality in (35) is satisfied, at which point the linear term in (31) dominates and the convergence rate goes back to linear. We emphasize that the quadratic convergence region is nonempty because we have $\sqrt{\eta_t} < 1$ for all $t \geq t_0$. Furthermore, making $\epsilon = 1$ and K sufficiently large it is possible to reduce η_t arbitrarily and make the quadratic convergence region last longer. In practice, this calls for making K large enough so that $\sqrt{\eta_t}$ is close to the desired gradient norm accuracy.

Remark 2 Making ρ^{K+1} small reduces the factor in front of the linear term in (34) and makes the quadratic phase longer. This factor, as it follows from the definition in Proposition 1, is $\rho^{K+1} = [2(1-\delta)/(2(1-\delta)+\alpha m)]^{K+1}$. Thus, other than increasing K, we can make ρ small by increasing the product αm. That implies making the inverse penalty coefficient α large relative to the smallest Hessian eigenvalue of the local functions f_i [cf. (22)]. This is not possible if we want to keep the solution $\mathbf{y}^*$ of (6) close to the solution of $\tilde{\mathbf{y}}^*$ of (5). This calls for the use of adaptive rules to decrease the inverse penalty coefficient α as we elaborate in Section IV. Further observe that ρ is independent of the condition number M/m of the local objectives. Making ρ small is an algorithmic choice – controlled by the selection of α and K –, and not a property of the function being minimized.

Algorithm 1 Network Newton-K method at node i

```
function $\mathbf{x}_i$ = NN-$K(\alpha, \mathbf{x}_i, \text{tol})$
repeat
    $\mathbf{B}$ matrix blocks: $\mathbf{B}_{ii} = (1-w_{ii})\mathbf{I}$ and $\mathbf{B}_{ij} = w_{ij}\mathbf{I}$
    $\mathbf{D}$ matrix block: $\mathbf{D}_{ii} = \alpha\nabla^2 f_i(\mathbf{x}_i) + 2(1-w_{ii})\mathbf{I}$
    Exchange iterates $\mathbf{x}_i$ with neighbors $j \in \mathcal{N}_i$.
    Gradient: $\mathbf{g}_i = (1-w_{ii})\mathbf{x}_i - \sum_{j\in\mathcal{N}_i} w_{ij}\mathbf{x}_j + \alpha\nabla f_i(\mathbf{x}_i)$.
    Compute NN-0 descent direction $\mathbf{d}_i^{(0)} = -\mathbf{D}_{ii}^{-1}\mathbf{g}_i$
    for $k = 0, \ldots, K-1$ do
        Exchange elements $\mathbf{d}_i^{(k)}$ of the NN-$k$ step with neighbors
        NN-$(k+1)$ step: $\mathbf{d}_i^{(k+1)} = \mathbf{D}_{ii}^{-1}\Big[\sum_{j\in\mathcal{N}_i, j=i}\mathbf{B}_{ij}\mathbf{d}_j^{(k)} - \mathbf{g}_i\Big]$.
    end for
    Update local iterate: $\mathbf{x}_i = \mathbf{x}_i + \epsilon\ \mathbf{d}_i^{(K)}$.
until $\|\mathbf{g}_i\| < \text{tol}$
```

Remark 3 For a quadratic objective function F, the Lipschitz constant for the Hessian is $L = 0$. Then, the optimal choice of stepsize for NN-K is $\epsilon = 1$ as a result of stepsize rule in (28). Moreover, the constants for the linear and quadratic terms in (31) are $\Gamma_1 = \Gamma_2 = 0$ as it follows from their definitions in (32). For quadratic functions we also have that the Hessian of the objective function $\mathbf{H}_t = \mathbf{H}$ and the block diagonal matrix $\mathbf{D}_t = \mathbf{D}$ are time invariant, which implies that we can rewrite (31) as

$$\|\mathbf{D}^{-1/2}\mathbf{g}_{t+1}\| \leq \rho^{K+1}\|\mathbf{D}^{-1/2}\mathbf{g}_t\|. \tag{42}$$

We know that when applying Newton's method to quadratic functions we converge in a single step. This property follows from (42) because Newton's method is equivalent to NN-K as $K \to \infty$. The expression in (42) states that NN-K converges linearly with a constant decrease factor of ρ^{K+1} per iteration. This factor is independent of the condition number of the quadratic function; see Remark 2. This in contrast with first order methods like DGD that converge with a linear rate that depends on the condition number of the objective.

IV. Implementation issues

As mentioned in Section II, NN-K does not solve (1) or its equivalent (5), but the penalty version introduced in (6). The optimal solutions of the optimization problems in (5) and (6) are different and the gap between them is of order $O(\alpha)$, [17]. This observation implies that by setting a decreasing policy

Algorithm 2 Adaptive Network Newton-K method at node i

```
Require: Iterate $\mathbf{x}_i$. Initial parameter $\alpha$. Flags $s_{ij} = 0$. Factor $\eta < 1$.
1: for $t = 0, 1, 2, \ldots$ do
2:     Call NN-$K$ function: $\mathbf{x}_i$ = NN-$K(\alpha, \mathbf{x}_i, \text{tol})$
3:     Set $s_{ii} = 1$ and broadcast it to all nodes.
4:     Set $s_{ij} = 1$ for all nodes $j$ that sent the signal $s_{jj} = 1$.
5:     if $s_{ij} = 1$ for all $j = 1, \ldots, n$ then
6:         Update penalty parameter $\alpha = \eta\alpha$.
7:         Set $s_{ij} = 0$ for all $j = 1, \ldots, n$.
8:     end if
9: end for
```

for α, or equivalently, an increasing policy for the penalty coefficient $1/\alpha$, the solution of (5) approaches the minimizer of (6), i.e. $\tilde{\mathbf{y}}^* \to \mathbf{y}^*$ for $\alpha \to 0$.

There are various possible alternatives to reduce α. Given the penalty method interpretation in Section II it is more natural to consider fixed penalty parameters α that are decreased after detecting convergence to the optimum argument of the function $F(\mathbf{y})$ [cf. (6)]. This latter idea is summarized under the name of Adaptive Network Newton-K (ANN-K) in Algorithm 2 where α is reduced by a given factor $\eta < 1$.

Specifically, ANN-K relies on Algorithm 1, which receives an initial iterate $\mathbf{x}_i$, a penalty parameter α, and a given tolerance tol (Step 1) and runs the local NN-K iterations in (18) and (20) for node i until the local gradient norm $\|\mathbf{g}_i\|$ becomes smaller than tol (Step 13). The descent iteration in (18) is implemented in Step 12. Implementation of this descent requires access to the NN-K descent direction $\mathbf{d}_{i,t}^{(K)}$ which is computed by the loop in steps 7-11. Step 7 initializes the loop by computing the NN-0 step $\mathbf{d}_{i,t}^{(0)} = -\mathbf{D}_{ii,t}^{-1}\mathbf{g}_{i,t}$. The core of the loop is in Step 10 which corresponds to the recursion in (20). Step 8 stands for the variable exchange that is necessary to implement Step 7. After K iterations through this loop, the NN-K descent direction $\mathbf{d}_{i,t}^{(K)}$ is computed and can be used in Step 12. Both, steps 7 and 10, require access to the local gradient component $\mathbf{g}_{i,t}$. This is evaluated in Step 6 after receiving the prerequisite information from neighbors in Step 5. Steps 3 and 4 compute the blocks $\mathbf{B}_{ii,t}$, $\mathbf{B}_{ij,t}$, and $\mathbf{D}_{ii,t}$ that are also necessary in steps 7 and 10. This process is repeated until $\|\mathbf{g}_i\| <$ tol (Step 13). Notice however, that if Algorithm 1 is called with a variable $\mathbf{x}_i$ with $\|\mathbf{g}_i\| <$ tol we still run at least one iteration of NN-K.

ANN-K calls Algorithm 1 in Step 2 of Algorithm 2. The factor α is subsequently reduced by the factor $\eta < 1$ as indicated in Step 6 of Algorithm 2 that implements the replacement $\alpha = \eta\alpha$. The rest of Algorithm 2 is designed to handle the fact that a small local gradient norm does not necessarily imply a small global gradient norm. To handle this possible mismatch, flag variables s_{ij} are introduced at node i to signal the fact that node j has reached a local gradient $\mathbf{g}_j$ with norm $\|\mathbf{g}_j\| \leq$ tol. Whenever node i completes a run of Algorithm 1 it broadcasts the signal s_{ii} to all other nodes (Step 3) and updates the variables s_{ij} to $s_{ij} = 1$ for all the nodes that sent the signals $s_{jj} = 1$ while Algorithm 1 was executing (Step 4). If all the variables $s_{ij} = 1$ (Step 5) it must be that this is true for all nodes and it is thus safe to modify α (Step 6). The flag variables are reset to $s_{ij} = 0$ and Algorithm 1 is called with the reduced α.

As is typical of penalty methods there are tradeoffs on the selection of the initial value of α and the decrease factor η. Small values of the initial penalty parameter and α and factor η results in sequence of approximate problems having solutions $\tilde{\mathbf{y}}^*$ that are closer to the actual solution $\mathbf{y}^*$. However, problems with small α may take a large number of iterations to converge if initialized far from the optimum value because the constant ρ approaches 1 when α is small – as we discussed in Remark 2. It is therefore better to initialize Algorithm 2 with values of α that are not too small and to decrease α by a factor η that is not too aggressive. We discuss these tradeoffs in the numerical examples of Section V-A.

V. Numerical analysis

We compare the performance of DGD and different versions of network Newton in the minimization of a distributed quadratic objective. The comparison is done in terms of both, number of iterations and number of information exchanges. For each agent i we consider a positive definite diagonal matrix $\mathbf{A}_i \in \mathbb{S}_p^{++}$ and a vector $\mathbf{b}_i \in \mathbb{R}^p$ to define the local objective function $f_i(\mathbf{x}) := (1/2)\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$. Therefore, the global cost function $f(\mathbf{x})$ is written as

$$f(\mathbf{x}) := \sum_{i=1}^{n} \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}\ . \tag{43}$$

The difficulty of solving (43) is given by the condition number of the matrices $\mathbf{A}_i$. To adjust condition numbers we generate diagonal matrices $\mathbf{A}_i$ with random diagonal elements a_{ii}. The first $p/2$ diagonal elements a_{ii} are drawn uniformly at random from the discrete set $\{1, 10^{-1}, \ldots, 10^{-\xi}\}$ and the next $p/2$ are uniformly and randomly chosen from the set $\{1, 10^1, \ldots, 10^{\xi}\}$. This choice of coefficients yields local matrices $\mathbf{A}_i$ with eigenvalues in the interval $[10^{-\xi}, 10^{\xi}]$ and global matrices $\sum_{i=1}^n \mathbf{A}_i$ with eigenvalues in the interval $[n10^{-\xi}, n10^{\xi}]$. The linear terms $\mathbf{b}_i^T\mathbf{x}$ are added so that the different local functions have different minima. The vectors $\mathbf{b}_i$ are chosen uniformly at random from the box $[0, 1]^p$.

For the quadratic objective in (43) we can compute the optimal argument $\mathbf{x}^*$ in closed form. We then evaluate convergence through the relative error that we define as the average normalized squared distance between local vectors $\mathbf{x}_i$ and the optimal decision vector $\mathbf{x}^*$,

$$e_t := \frac{1}{n}\sum_{i=1}^{n} \frac{\|\mathbf{x}_{i,t} - \mathbf{x}^*\|^2}{\|\mathbf{x}^*\|^2}. \tag{44}$$

The network connecting the nodes is a d-regular cycle where each node is connected to exactly d neighbors and d is assumed even. The graph is generated by creating a cycle and then connecting each node with the $d/2$ nodes that are closest in each direction. The diagonal weights in the matrix $\mathbf{W}$ are set to $w_{ii} = 1/2 + 1/2(d + 1)$ and the off diagonal weights to $w_{ij} = 1/2(d + 1)$ when $j \in \mathcal{N}_i$.

In the subsequent experiments we set the network size to $n = 100$, the dimension of the decision vectors to $p = 4$, the condition number parameter to $\xi = 2$, the penalty coefficient

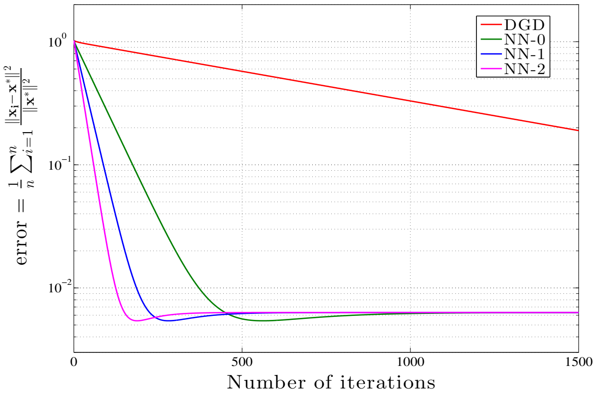

Fig. 1: Convergence of DGD, NN-0, NN-1, and NN-2 in terms of number of iterations. The network Newton methods converges faster than DGD. Furthermore, the larger K is, the faster NN-K converges.

inverse to $\alpha = 10^{-2}$, and the network degree to $d = 4$. The network Newton step size is set to $\epsilon = 1$, which is always possible when we have quadratic objectives [cf. Remark 3]. Figure 1 illustrates a sample convergence path for DGD, NN-0, NN-1, and NN-2 by measuring the relative error e_t in (44) with respect to the number of iterations t. As expected for a problem that doesn't have a small condition number – in this particular instantiation of the function in (43) the condition number is 95.2 – different versions of network Newton are much faster than DGD. E.g., after $t = 1.5 \times 10^3$ iterations the error associated which DGD iterates is $e_t \approx 1.9 \times 10^{-1}$. Comparable or better accuracy $e_t < 1.9 \times 10^{-1}$ is achieved in $t = 132$, $t = 63$, and $t = 43$ iterations for NN-0, NN-1, and NN-2, respectively.

Further recall that α controls the difference between the actual optimal argument $\tilde{\mathbf{y}}^* = [\mathbf{x}^*; \dots; \mathbf{x}^*]$ [cf. (5)] and the argument $\mathbf{y}^*$ [cf. (6)] to which DGD and network Newton converge. Since we have $\alpha = 10^{-2}$ and the difference between these two vectors is of order $O(\alpha)$, we expect the error in (44) to settle at $e_t \approx 10^{-2}$. The error actually settles at $e_t \approx 6.3 \times 10^{-3}$ and it takes all three versions of network Newton less than $t = 400$ iterations to do so. It takes DGD more than $t = 10^4$ iterations to reach this value. This relative performance difference decreases if the problem has better conditioning but can be made arbitrarily large by increasing the condition number of the matrix $\sum_{i=1}^{n} \mathbf{A}_i$. The number of iterations required for convergence can be further decreased by considering higher order approximations in (16). The advantages would be misleading because they come at the cost of increasing the number of communications required to approximate the Newton step.

To study this latter effect we consider the relative performance of DGD and different versions of network Newton in terms of the number of local information exchanges. As pointed out in Remark 1, each iteration in NN-K requires a total of $K + 1$ information exchanges with each neighbor, as opposed to the single variable exchange required by DGD. After t iterations the number of variable exchanges between each pair of neighbors is t for DGD and $(K + 1)t$ for NN-K. Thus, we can translate Figure 1 into a path in terms of number of communications by scaling the time axis by $(K+1)$. The result of this scaling is shown in Figure 2. The different versions of network Newton retain a significant, albeit smaller, advantage with respect to DGD. Error $e_t < 10^{-2}$ is achieved by NN-0, NN-1, and NN-2 after $(K + 1)t = 3.7 \times 10^2$, $(K + 1)t = 3.1 \times 10^2$, and $(K + 1)t = 3.4 \times 10^2$ variable exchanges, respectively. When measured in this metric it is no longer true that increasing K results in faster convergence. For this particular problem instance it is actually NN-1 that converges fastest in terms of number of communication exchanges.

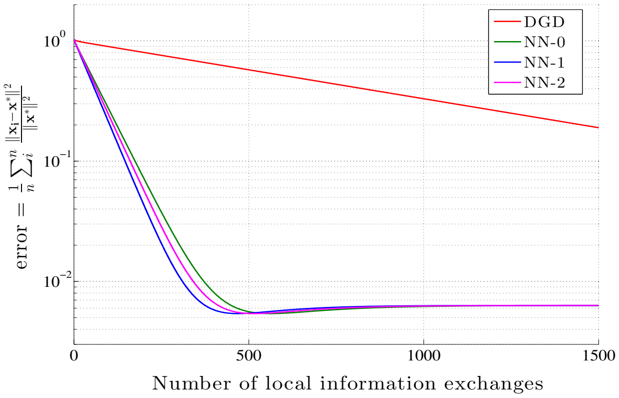

Fig. 2: Convergence of DGD, NN-0, NN-1, and NN-2 in terms of number of communication exchanges. The NN-K methods retain the advantage over DGD but increasing K may not result in faster convergence. For this particular instance it is actually NN-1 that converges fastest in terms of number of communication exchanges.

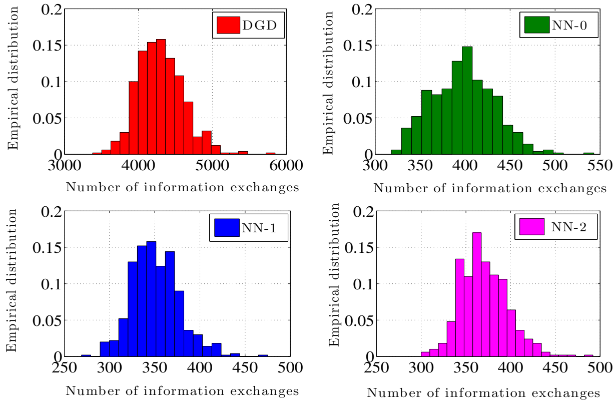

Fig. 3: Histograms of the number of information exchanges required to achieving accuracy $e_t < 10^{-2}$. The qualitative observations made in figures 1 and 2 hold over a range of random problem realizations.

For a more comprehensive evaluation we consider 10^3 different random realizations of (43) where we also randomize the degree d of the d-regular graph that we choose from the even numbers in the set $[2, 10]$. The remaining parameters are the same used to generate figures 1 and 2. For each joint random realization of network and objective we run DGD, NN-0, NN-1, and NN-2, until achieving error $e_t < 10^{-2}$ and record the number of communication exchanges that have elapsed – which amount to simply t for DGD and $(K+1)t$ for NN. The resulting histograms are shown in Figure 3. The mean times required to reduce the error to $e_t < 10^{-2}$ are 4.3×10^3 for

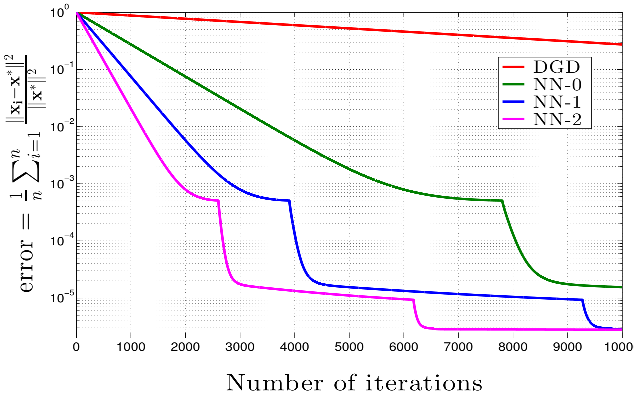

Fig. 4: Convergence of adaptive DGD, NN-0, NN-1, and NN-2 for $\alpha_0 = 10^{-2}$. network Newton methods require less iterations than DGD.

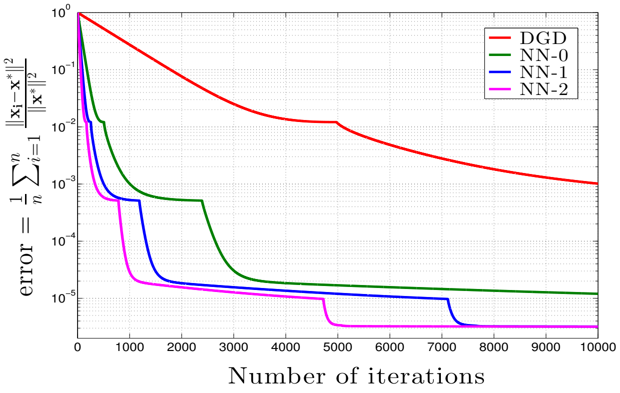

Fig. 5: Convergence of Adaptive DGD, NN-0, NN-1, and NN-2 for $\alpha_0 = 10^{-1}$. ANN methods require less iterations than DGD and convergence of all algorithms are faster relative to the case that $\alpha_0 = 10^{-2}$.

DGD and 4.0×10^2, 3.5×10^2, and 3.7×10^2 for NN-0, NN-1, and NN-2. As in the particular case shown in figures 1 and 2, NN-1 performs best in terms of communication exchanges. Observe, however, that the number of communication exchanges required by NN-2 is not much larger and that NN-2 requires less computational effort than NN-1 because the number of iterations t is smaller.

A. Adaptive Network Newton

Given that DGD and network Newton are penalty methods it is of interest to consider their behavior when the inverse penalty coefficient α is decreased recursively. The adaptation of α for NN-K is discussed in Section IV where it is termed adaptive (A)NN-K. The same adaptation strategy is considered here for DGD. The parameter α is kept constant until the local gradient components $\mathbf{g}_{i,t}$ become smaller than a given tolerance tol, i.e., until $\|\mathbf{g}_{i,t}\| \leq$ tol for all i. When this tolerance is achieved, the parameter α is scaled by a factor $\eta < 1$, i.e., α is decreased from its current value to $\eta\alpha$. This requires the use of a signaling method like the one summarized in Algorithm 2 for ANN-K.

We consider the objective in (43) and nodes connected by a d-regular cycle. We use the same parameters used to generate figures 1 and 2. The adaptive gradient tolerance is set to tol $= 10^{-3}$ and the scaling parameter to $\eta = 0.1$. We consider two different scenarios where the initial penalty parameters are $\alpha = \alpha_0 = 10^{-1}$ and $\alpha = \alpha_0 = 10^{-2}$. The respective error trajectories e_t with respect to the number of iterations are shown in figures 4 – where $\alpha_0 = 10^{-2}$ – and 5 – where $\alpha_0 = 10^{-1}$. In each figure we show e_t for adaptive DGD, ANN-0, ANN-1, and ANN-2. Both figures show that the ANN methods outperform adaptive DGD and that larger K reduces the number of iterations that it takes ANN-K to achieve a target error. These results are consistent with the findings summarized in figures 1-3.

More interesting conclusions follow from a comparison across figures 4 and 5. We can see that it is better to start with the (larger) value $\alpha = 10^{-1}$ even if the method initially converges to a point farther from the actual optimum. This happens because increasing α decreases the constant $\rho = 2(1-\delta)/(2(1-\delta)+\alpha m)$.

B. Logistic regression

For a non-quadratic test we consider the application of network Newton for solving a logistic regression problem. In this problem we are given q training samples that we distribute across n distinct servers. Denote as q_i the number of samples that are assigned to server i. Each of the training samples at node i contains a feature vector $\mathbf{u}_{il} \in \mathbb{R}^p$ and a class $v_{il} \in \{-1, 1\}$. The goal is to predict the probability $\mathrm{P}\left(v = 1 \mid \mathbf{u}\right)$ of having label $v = 1$ when given a feature vector $\mathbf{u}$ whose class is not known. The logistic regression model assumes that this probability can be computed as $\mathrm{P}\left(v = 1 \mid \mathbf{u}\right) = 1/(1+\exp(-\mathbf{u}^T\mathbf{x}))$ for a linear classifier $\mathbf{x}$ that is computed based on the training samples. It follows from this model that the regularized maximum log likelihood estimate of the classifier $\mathbf{x}$ given the training samples $(\mathbf{u}_{il}, v_{il})$ for $l = 1, \ldots, q_i$ and $i = 1, \ldots, n$ is given by

$$\begin{aligned} \mathbf{x}^* &= \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) \qquad (45)\\ &:= \operatorname*{argmin}_{\mathbf{x}} \frac{\lambda}{2}\|\mathbf{x}\|^2 + \sum_{i=1}^{n}\sum_{l=1}^{q_i} \log\left[1 + \exp(-v_{il}\mathbf{u}_{il}^T\mathbf{x})\right], \end{aligned}$$

where we defined the function $f(\mathbf{x})$ for future reference. The regularization term $(\lambda/2)\|\mathbf{x}\|^2$ is added to reduce overfitting to the training set.

The optimization problem in (45) can be written in the form of the optimization problem in (1). To do so simply define the local objective functions f_i as

$$f_i(\mathbf{x}) = \frac{\lambda}{2n}\|\mathbf{x}\|^2 + \sum_{l=1}^{q_i} \log\left[1 + \exp(-v_{il}\mathbf{u}_{il}^T\mathbf{x})\right], \qquad (46)$$

and observe that given this definition we can write the objective in (45) as $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$. We can then solve (45) in a distributed manner using DGD and NN-K methods.

In our experiments we use a synthetic dataset where each component of the feature vector $\mathbf{u}_{il}$ with label $v_{il} = 1$ is generated from a normal distribution with mean μ and standard deviation σ_+, while sample points with label $v_{il} = -1$ are generated with mean $-\mu$ and standard deviation σ_-. The

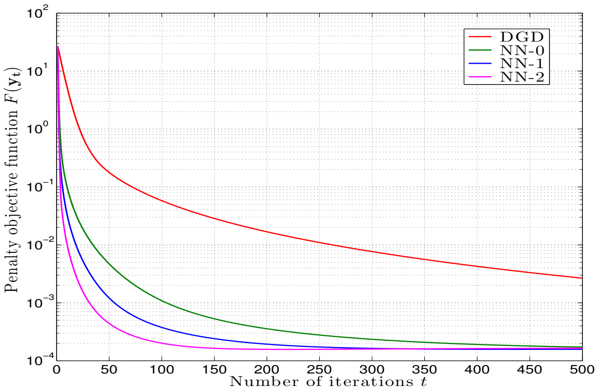

Fig. 6: Convergence of DGD, NN-0, NN-1, and NN-2. network Newton methods for a linearly separable logistic regression.

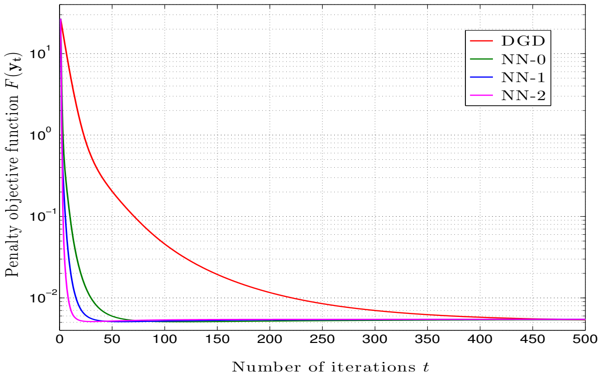

Fig. 7: Convergence of DGD, NN-0, NN-1, and NN-2. network Newton methods for a non-linearly separable logistic regression.

network is a d-regular cycle. The diagonal weights in the matrix $\mathbf{W}$ are set to $w_{ii} = 1/2 + 1/2(d+1)$ and the off diagonal weights to $w_{ij} = 1/2(d+1)$ when $j \in \mathcal{N}_i$. We set the feature vector dimension to $p = 10$, the number of training samples per node at $q_i = 50$, and the regularization parameter to $\lambda = 10^{-4}$. The number of nodes is $n = 100$ and the degree of the d-regular cycle is $d = 4$.

We consider first a scenario in which the dataset is linearly separable. To generate a linearly separable dataset the mean is set to $\mu = 3$ and the standard deviations to $\sigma_+ = \sigma_- = 1$. Figure 6 illustrates the convergence path of the objective function $F(\mathbf{y})$ [cf. (6)] when the penalty parameter is $\alpha = 10^{-2}$ and the network Newton step size is $\epsilon = 1$. The reduction in the number of iterations required to achieve convergence is a little more marked than in the quadratic example considered in figures 1-3. The objective function values $F(\mathbf{y}_t)$ for NN-0, NN-1 and NN-2 after $t = 500$ iterations are below 1.6×10^{-4}, while for DGD the objective function value after the same number of iterations have passed is $F(\mathbf{y}_t) = 2.6 \times 10^{-3}$. Conversely, achieving accuracy $F(\mathbf{y}_t) = 2.6 \times 10^{-3}$ for NN-0, NN-1, and NN-2 requires 68, 33, and 19 iterations, respectively, while DGD requires 500 iterations. Observe that for this example NN-2 performs better than NN-1 and NN-0 not only in the number of iterations but also in the number of variable exchanges required to achieve a target accuracy.

We also consider a case in which the dataset is *not* linearly separable. To generate this dataset we set the mean to $\mu = 2$ and the standard deviations to $\sigma_+ = \sigma_- = 2$. The penalty parameter is set to $\alpha = 10^{-2}$ and the network Newton step size to $\epsilon = 1$. The resulting objective trajectories $F(\mathbf{y}_t)$ of DGD, NN-0, NN-1, and NN-2 are shown in Figure 7. The advantages of the network Newton methods relative to DGD are less pronounced but still significant. In this case we also observe that NN-2 performs best in terms of number of iterations and number of communication exchanges.

VI. Conclusions

Network Newton is a decentralized approximation of Newton's method for solving decentralized optimization problems. This paper studied convergence properties and implementation details of this method. Network Newton approximates the Newton direction by truncating a Taylor series expansion of the exact Newton step. This procedure produces a class of algorithms identified by K, which is the number of Taylor series terms that network Newton uses for approximating the Newton step. The algorithm is called NN-K when we keep K terms of the Newton step Taylor series. Linear convergence of NN-K is established in a companion paper [3]. Here, we completed the convergence analysis of NN-K by showing that the sequence of iterates generated by NN-K has a quadratic convergence rate in a specific interval. A quadratic phase exists for all choices of K, but this phase can be made arbitrarily large by increasing K. The analysis presented here also shows that for the particular case of quadratic objective functions, the convergence rate of NN-K is independent of the condition number of the objective function. This is in contrast to distributed gradient descent methods that require more iterations for problems with larger condition number. Numerical analyses compared the performances of distributed gradient descent and NN-K with different choices of K for minimizing quadratic objectives with large condition numbers as well as the log likelihood objective of a logistic regression problem. In either case we observe that all NN-K methods work faster than distributed gradient descent in terms of number of iterations and number of communications required to achieve convergence. Overall, the theoretical and numerical analyses in this paper prove that NN-K achieves the design goal of accelerating the convergence of distributed gradient descent methods.

We further analyzzed a tradeoff on the selection of a penalty parameter that controls both, the accuracy of the optimal objective computed by network Newton methods and the rate of convergence. We proposed an adaptive version of network Newton (ANN) that achieves exact convergence by executing network Newton with an increasing sequence of penalty coefficients. Numerical analyses of ANN show that it is best to initialize penalty coefficients at moderate values and decrease them through moderate factors.

Appendix A
Proof of Lemma 2

To prove the result in Lemma 2, we first use the Fundamental theorem of Calculus and the Lipschitz continuity of the Hessians $\mathbf{H}_t := \mathbf{H}(\mathbf{y}_t) = \nabla^2 F(\mathbf{y}_t)$ to prove the following Lemma.

Lemma 3 *Consider the NN-K method as defined in* (12)-(17). *If Assumption 3 holds true, then*

$$\|\mathbf{g}_{t+1} - \mathbf{g}_t - \mathbf{H}_t\ (\mathbf{y}_{t+1} - \mathbf{y}_t)\| \leq \frac{\alpha L}{2}\|\mathbf{y}_{t+1} - \mathbf{y}_t\|^2. \quad (47)$$

Proof: Considering the definitions of the objective function gradient $\mathbf{g}_t = \mathbf{g}(\mathbf{y}_t) = \nabla F(\mathbf{y}_t)$ and Hessian $\mathbf{H}_t = \mathbf{H}(\mathbf{y}_t) = \nabla^2 F(\mathbf{y}_t)$, the Fundamental Theorem of calculus implies that

$$\mathbf{g}_{t+1} = \mathbf{g}_t + \int_0^1 \mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t))\ (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega. \quad (48)$$

Adding and subtracting the integral $\int_0^1 \mathbf{H}(\mathbf{y}_t)\ (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega$ to the right hand side of (48) yields

$$\begin{aligned}\mathbf{g}_{t+1} = \mathbf{g}_t &+ \int_0^1 \mathbf{H}(\mathbf{y}_t)\ (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega \quad (49)\\ &+ \int_0^1 [\mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t)) - \mathbf{H}(\mathbf{y}_t)]\, (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega.\end{aligned}$$

Note that $\mathbf{H}(\mathbf{y}_t)\ (\mathbf{y}_{t+1} - \mathbf{y}_t)$ is not a function of the variable ω and the first integral in (49) can be simplified as $\int_0^1 \mathbf{H}(\mathbf{y}_t)\ (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega = \mathbf{H}(\mathbf{y}_t)\ (\mathbf{y}_{t+1} - \mathbf{y}_t)$. By considering this simplification and regrouping the terms in (49) we obtain

$$\begin{aligned}&\mathbf{g}_{t+1} - \mathbf{g}_t - \mathbf{H}(\mathbf{y}_t)(\mathbf{y}_{t+1} - \mathbf{y}_t) = \quad (50)\\ &\qquad \int_0^1 [\mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t)) - \mathbf{H}(\mathbf{y}_t)]\, (\mathbf{y}_{t+1} - \mathbf{y}_t) d\omega.\end{aligned}$$

We proceed by computing the norm of both sides of (50). Observing that for any vector $\mathbf{a}$ the inequality $\int \|\mathbf{a}\| d\omega \leq \| \int \mathbf{a}\, d\omega\|$ holds and considering that the product of norms is greater than the norm of the respective product, it follows that

$$\begin{aligned}&\|\mathbf{g}_{t+1} - \mathbf{g}_t - \mathbf{H}(\mathbf{y}_t)(\mathbf{y}_{t+1} - \mathbf{y}_t)\| \leq \quad (51)\\ &\qquad \int_0^1 \|\mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t)) - \mathbf{H}(\mathbf{y}_t)\|\, \|\mathbf{y}_{t+1} - \mathbf{y}_t\| d\omega.\end{aligned}$$

Based on (24), the Hessians $\mathbf{H}(\mathbf{y}_t)$ are Lipschitz continuous with parameter αL. Therefore, we can write

$$\|\mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t)) - \mathbf{H}(\mathbf{y}_t)\| \leq \alpha L \omega \|\mathbf{y}_{t+1} - \mathbf{y}_t\|. \quad (52)$$

Substituting the upper bound in (52) for the term $\|\mathbf{H}(\mathbf{y}_t + \omega(\mathbf{y}_{t+1} - \mathbf{y}_t)) - \mathbf{H}(\mathbf{y}_t)\|$ in (51) leads to

$$\begin{aligned}\|\mathbf{g}_{t+1} - \mathbf{g}_t &- \mathbf{H}(\mathbf{y}_t)(\mathbf{y}_{t+1} - \mathbf{y}_t)\|\\ &\leq \int_0^1 \alpha L \omega \|\mathbf{y}_{t+1} - \mathbf{y}_t\|^2 d\omega\\ &= \frac{\alpha L}{2}\|\mathbf{y}_{t+1} - \mathbf{y}_t\|^2, \quad (53)\end{aligned}$$

where the equality in (53) is valid since $\int_0^1 \omega\ d\omega = 1/2$. The inequality in (53) yields the result in (47) considering the notation $\mathbf{H}_t = \mathbf{H}(\mathbf{y}_t)$. ■

Proof of Lemma 2: In this proof to simplify the notation we use $\hat{\mathbf{H}}_t^{-1}$ to indicate the approximate Hessian inverse $\hat{\mathbf{H}}_t^{(K)^{-1}}$. Recall the result in (47). Considering the update formula for NN-K in (16), the term $\mathbf{y}_{t+1} - \mathbf{y}_t$ can be substituted by $-\epsilon \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t$. Making this substitution into (47) implies that

$$\left\| \mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\| \leq \frac{\epsilon^2 \alpha L}{2} \left\| \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\|^2. \quad (54)$$

The definition of matrix norm implies that the norm of product $\mathbf{D}_t^{-1/2}(\mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t)$ is bounded above as

$$\begin{aligned}&\left\| \mathbf{D}_t^{-1/2} \left[\mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right] \right\| \leq\\ &\qquad \left\| \mathbf{D}_t^{-1/2} \right\| \left\| \mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\|. \quad (55)\end{aligned}$$

Substituting $\|\mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t\|$ in the right hand side of (55) by the upper bound in (54) leads to

$$\begin{aligned}&\left\| \mathbf{D}_t^{-1/2} \left[\mathbf{g}_{t+1} - \mathbf{g}_t + \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right] \right\| \leq\\ &\qquad \frac{\epsilon^2 \alpha L}{2} \left\| \mathbf{D}_t^{-1/2} \right\| \left\| \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\|^2. \quad (56)\end{aligned}$$

Observe that the triangle inequality states that for any vectors $\mathbf{a}$ and $\mathbf{b}$, and a positive constant C, if the relation $\|\mathbf{a} - \mathbf{b}\| \leq C$ holds true, then $\|\mathbf{a}\| \leq \|\mathbf{b}\| + C$. By setting $\mathbf{a} := \mathbf{D}_t^{-1/2} \mathbf{g}_{t+1}$, $\mathbf{b} := \mathbf{D}_t^{-1/2}(\mathbf{g}_t - \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1})$ and $C := (\epsilon^2 \alpha L/2)\|\mathbf{D}_t^{-1/2}\| \|\hat{\mathbf{H}}_t^{-1} \mathbf{g}_t\|^2$ and considering the relation in (56) we obtain that the inequality $\|\mathbf{a} - \mathbf{b}\| \leq C$ is satisfied. Therefore, $\|\mathbf{a}\| \leq \|\mathbf{b}\| + C$ holds true which is equivalent to

$$\begin{aligned}\left\| \mathbf{D}_t^{-1/2} \mathbf{g}_{t+1} \right\| &\leq \left\| \mathbf{D}_t^{-1/2} \left[\mathbf{g}_t - \epsilon \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right] \right\|\\ &\quad + \frac{\epsilon^2 \alpha L}{2} \left\| \mathbf{D}_t^{-1/2} \right\| \left\| \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\|^2. \quad (57)\end{aligned}$$

By rewriting the term $\mathbf{D}_t^{-1/2} \mathbf{g}_t$ as the sum $(1-\epsilon)(\mathbf{D}_t^{-1/2} \mathbf{g}_t) + \epsilon(\mathbf{D}_t^{-1/2} \mathbf{g}_t)$ and using the triangle inequality we can update the right hand side of (57) as

$$\begin{aligned}\left\| \mathbf{D}_t^{-1/2} \mathbf{g}_{t+1} \right\| &\leq (1 - \epsilon) \left\| \mathbf{D}_t^{-1/2} \mathbf{g}_t \right\|\\ &\quad + \epsilon \left\| \mathbf{D}_t^{-1/2} \left[\mathbf{I} - \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \right] \mathbf{g}_t \right\|\\ &\quad + \frac{\epsilon^2 \alpha L}{2} \left\| \mathbf{D}_t^{-1/2} \right\| \left\| \hat{\mathbf{H}}_t^{-1} \mathbf{g}_t \right\|^2. \quad (58)\end{aligned}$$

Observe that the Hessian is decomposed as $\mathbf{H}_t = \mathbf{D}_t - \mathbf{B}$ and the approximate Hessian inverse is given by $\hat{\mathbf{H}}_t^{-1} := \mathbf{D}_t^{-1/2} \sum_{k=0}^{K} (\mathbf{D}_t^{-1/2} \mathbf{B} \mathbf{D}_t^{-1/2})^k \mathbf{D}_t^{-1/2}$. Considering these two relations and using the telescopic cancelation we can show that $\mathbf{I} - \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} = (\mathbf{B} \mathbf{D}_t^{-1})^{K+1}$. This result is studied with more details in Lemma 3 of [3]. Therefore, we can write

$$\left\| \mathbf{D}_t^{-1/2} \left[\mathbf{I} - \mathbf{H}_t \hat{\mathbf{H}}_t^{-1} \right] \mathbf{g}_t \right\| = \left\| \left[\mathbf{D}_t^{-1/2} \mathbf{B} \mathbf{D}_t^{-1/2} \right]^{K+1} \mathbf{D}_t^{-1/2} \mathbf{g}_t \right\| \quad (59)$$

Based on Proposition 1, the eigenvalues of the matrix $\mathbf{D}_t^{-1/2} \mathbf{B} \mathbf{D}_t^{-1/2}$ are bounded by 0 and ρ. This observation in association with the symmetry of $\mathbf{D}_t^{-1/2} \mathbf{B} \mathbf{D}_t^{-1/2}$ yields

$$\left\| \left[\mathbf{D}_t^{-1/2} \mathbf{B} \mathbf{D}_t^{-1/2} \right]^{K+1} \right\| \leq \rho^{K+1}. \quad (60)$$

The simplification in (59) and the upper bound in (60) guarantee that the norm $\|\mathbf{D}_t^{-1/2}[\mathbf{I} - \mathbf{H}_t \hat{\mathbf{H}}_t^{-1}] \mathbf{g}_t\|$ is upper bounded by

$$\left\|\mathbf{D}_t^{-1/2}\left[\mathbf{I}-\mathbf{H}_t\hat{\mathbf{H}}_t^{-1}\right]\mathbf{g}_t\right\| \leq \rho^{K+1}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\|. \tag{61}$$

Substituting the upper bound in (61) for the second summand in the right hand side of (58) implies that

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\leq (1-\epsilon)\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\| + \epsilon\rho^{K+1}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\| \\ &\quad + \frac{\alpha\epsilon^2 L}{2}\left\|\mathbf{D}_t^{-1/2}\right\|\left\|\hat{\mathbf{H}}_t^{-1}\mathbf{g}_t\right\|^2. \end{aligned} \tag{62}$$

By grouping the first two summands in (62) and using the inequality $\|\hat{\mathbf{H}}_t^{-1}\mathbf{g}_t\| \leq \|\hat{\mathbf{H}}_t^{-1}\|\|\mathbf{g}_t\|$ we can write

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\leq (1-\epsilon+\epsilon\rho^{K+1})\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\| \\ &\quad + \frac{\alpha\epsilon^2 L}{2}\left\|\mathbf{D}_t^{-1/2}\right\|\left\|\hat{\mathbf{H}}_t^{-1}\right\|^2\|\mathbf{g}_t\|^2. \end{aligned} \tag{63}$$

Now we proceed to find an upper bound for $\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\|$ in terms of $\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$ to set a recursive inequality for the sequence $\|\mathbf{D}_{i-1}^{-1/2}\mathbf{g}_i\|$. We first show that the norm of the difference $\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\|$ is bounded above as

$$\left\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\right\| \leq \left\|\mathbf{D}_t^{-1}\right\|\left\|\mathbf{D}_t-\mathbf{D}_{t-1}\right\|\left\|\mathbf{D}_{t-1}^{-1}\right\|. \tag{64}$$

Factoring $\mathbf{D}_t^{-1}$ and $\mathbf{D}_{t-1}^{-1}$ from the left and right sides of $\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}$, respectively, follows the relation in (64). Observe that the eigenvalues of the matrices $\mathbf{D}_t$ and $\mathbf{D}_{t-1}$ are bounded below by $\alpha m + 2(1-\Delta)$. Consequently, the eigenvalues of the matrices $\mathbf{D}_t^{-1}$ and $\mathbf{D}_{t-1}^{-1}$ are bounded above by $1/(\alpha m + 2(1-\Delta))$. Therefore, we can update the upper bound in (64) as

$$\left\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\right\| \leq \frac{1}{(2(1-\Delta)+\alpha m)^2}\left\|\mathbf{D}_t-\mathbf{D}_{t-1}\right\|. \tag{65}$$

The next step is to show that the block diagonal matrices $\mathbf{D}_t$ are Lipschitz continuous with parameter αL. Notice that the difference $\mathbf{D}_t-\mathbf{D}_{t-1}$ can be simplified as $\alpha(\mathbf{G}_t-\mathbf{G}_{t-1})$. Moreover, the difference of two consecutive Hessians can be simplified as $\mathbf{H}_t-\mathbf{H}_{t-1} = \alpha(\mathbf{G}_t-\mathbf{G}_{t-1})$. Therefore, we obtain that $\mathbf{D}_t-\mathbf{D}_{t-1} = \mathbf{H}_t-\mathbf{H}_{t-1}$. This observation in association with the Lipschitz continuity of the Hessians with parameter αL, i.e., $\|\mathbf{H}_t-\mathbf{H}_{t-1}\| \leq \alpha L\|\mathbf{y}_t-\mathbf{y}_{t-1}\|$, implies that

$$\|\mathbf{D}_t-\mathbf{D}_{t-1}\| \leq \alpha L\|\mathbf{y}_t-\mathbf{y}_{t-1}\|. \tag{66}$$

By substituting the upper bound in (66) for the norm $\|\mathbf{D}_t-\mathbf{D}_{t-1}\|$ in (65) we obtain that

$$\left\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\right\| \leq \frac{\alpha L}{(2(1-\Delta)+\alpha m)^2}\left\|\mathbf{y}_t-\mathbf{y}_{t-1}\right\|. \tag{67}$$

Note that the absolute value of the inner product $|\mathbf{g}_t^T(\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1})\mathbf{g}_t|$ is bounded above by the product $\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\|\|\mathbf{g}_t\|^2$. Considering the upper bound for $\|\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1}\|$ in (67), the term $|\mathbf{g}_t^T(\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1})\mathbf{g}_t|$ is bounded above by

$$\left|\mathbf{g}_t^T(\mathbf{D}_t^{-1}-\mathbf{D}_{t-1}^{-1})\mathbf{g}_t\right| \leq \frac{\alpha L\ \|\mathbf{y}_t-\mathbf{y}_{t-1}\|\|\mathbf{g}_t\|^2}{(2(1-\Delta)+\alpha m)^2}. \tag{68}$$

Considering the triangle inequality, and observing the simplifications $|\mathbf{g}_t^T\mathbf{D}_{t-1}^{-1}\mathbf{g}_t| = \|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|^2$ and $|\mathbf{g}_t^T\mathbf{D}_t^{-1}\mathbf{g}_t| = \|\mathbf{D}_t^{-1/2}\mathbf{g}_t\|^2$, we can rewrite (68) as

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\|^2 \leq \left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2 + \frac{\alpha L\ \|\mathbf{y}_t-\mathbf{y}_{t-1}\|\|\mathbf{g}_t\|^2}{(2(1-\Delta)+\alpha m)^2}. \tag{69}$$

Observe that for any three constants a, b and c, if the relation $a^2 \leq b^2 + c^2$ holds, then the inequality $|a| \leq |b| + |c|$ is valid. By setting $a := \|\mathbf{D}_t^{-1/2}\mathbf{g}_t\|$, $b := \|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$, and $c := (\alpha L\|\mathbf{y}_t-\mathbf{y}_{t-1}\|)^{1/2}\|\mathbf{g}_t\|/(2(1-\Delta)+\alpha m)$, we realize that $a^2 \leq b^2 + c^2$ holds true according to (69). Hence, we obtain that the relation $|a| \leq |b| + |c|$ holds and we obtain that

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\| \leq \left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| + \frac{(\alpha L\|\mathbf{y}_t-\mathbf{y}_{t-1}\|)^{1/2}\|\mathbf{g}_t\|}{2(1-\Delta)+\alpha m} \tag{70}$$

Considering the update of NN-K in (17) we can substitute $\mathbf{y}_t-\mathbf{y}_{t-1}$ by $-\epsilon\hat{\mathbf{H}}_{t-1}^{-1}\mathbf{g}_{t-1}$. Applying this substitution into (70) implies that

$$\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\right\| \leq \left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| + \frac{\left[\alpha\epsilon L\left\|\hat{\mathbf{H}}_{t-1}^{-1}\mathbf{g}_{t-1}\right\|\right]^{1/2}\|\mathbf{g}_t\|}{2(1-\Delta)+\alpha m}. \tag{71}$$

If we substitute $\|\mathbf{D}_t^{-1/2}\mathbf{g}_t\|$ by the upper bound in (71) and substitute $\|\hat{\mathbf{H}}_{t-1}^{-1}\mathbf{g}_{t-1}\|$ by the upper bound $\|\hat{\mathbf{H}}_{t-1}^{-1}\|\|\mathbf{g}_{t-1}\|$, the inequality in (63) can be written as

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\leq \left(1-\epsilon+\epsilon\rho^{K+1}\right)\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \\ &\quad + \frac{\left(1-\epsilon+\epsilon\rho^{K+1}\right)\left[\alpha\epsilon L\left\|\hat{\mathbf{H}}_{t-1}^{-1}\right\|\|\mathbf{g}_{t-1}\|\right]^{1/2}}{2(1-\Delta)+\alpha m}\|\mathbf{g}_t\| \\ &\quad + \frac{\alpha\epsilon^2 L}{2}\left\|\mathbf{D}_t^{-1/2}\right\|\left\|\hat{\mathbf{H}}_t^{-1}\right\|^2\|\mathbf{g}_t\|^2. \end{aligned} \tag{72}$$

Due to the fact that for a positive definite matrix the norm of its product by a vector is always larger than its minimum eigenvalue multiplied by the norm of the vector, we can write $\mu_{min}(\mathbf{D}_{t-1}^{-1/2})\|\mathbf{g}_t\| \leq \|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\|$. Rearranging the terms yields

$$\|\mathbf{g}_t\| \leq \frac{1}{\mu_{min}(\mathbf{D}_{t-1}^{-1/2})}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|. \tag{73}$$

Note that the eigenvalues of the matrix $\mathbf{D}_{t-1}$ are upper bounded by $2(1-\delta)+\alpha M$. Hence, $1/\sqrt{(2(1-\delta)+\alpha M)}$ is a lower bound for the eigenvalues of the matrix $\mathbf{D}_{t-1}^{-1/2}$. This observation implies that the upper bound in (73) can be updated as

$$\|\mathbf{g}_t\| \leq (2(1-\delta)+\alpha M)^{1/2}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|. \tag{74}$$

Substituting $\|\mathbf{g}_t\|$ by the upper bound in (74) and considering the definition $\lambda := 1/(2(1-\delta)+\alpha M)$ follows that we can update the right hand side of (72) as

$$\begin{aligned}\left\|\mathbf{D}_t^{-1/2}\mathbf{g}_{t+1}\right\| &\leq (1-\epsilon+\epsilon\rho^{K+1})\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \\ &\quad + \frac{(1-\epsilon+\epsilon\rho^{K+1})}{(2(1-\Delta)+\alpha m)}\left[\frac{\alpha\epsilon L\|\mathbf{g}_{t-1}\|\left\|\hat{\mathbf{H}}_{t-1}^{-1}\right\|}{\lambda}\right]^{1/2}\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\| \\ &\quad + \frac{\alpha\epsilon^2 L}{2\lambda}\left\|\mathbf{D}_t^{-1/2}\right\|\left\|\hat{\mathbf{H}}_t^{-1}\right\|^2\left\|\mathbf{D}_{t-1}^{-1/2}\mathbf{g}_t\right\|^2. \end{aligned} \tag{75}$$

Observe that the norms $\|\hat{\mathbf{H}}_t^{-1}\|$ and $\|\hat{\mathbf{H}}_{t-1}^{-1}\|$ are upper bounded by Λ which is defined in (27). Moreover, the norm $\|\mathbf{D}_t^{-1/2}\|$ is bound above by $1/(2(1-\Delta)+\alpha m)^{1/2}$. Substituting these bounds into (75) results in

$$\left\|\mathbf{D}_t^{-\frac{1}{2}}\mathbf{g}_{t+1}\right\| \leq \left(1-\epsilon+\epsilon\rho^{K+1}\right)\left(1+C_1\|\mathbf{g}_{t-1}\|^{\frac{1}{2}}\right)\left\|\mathbf{D}_{t-1}^{-\frac{1}{2}}\mathbf{g}_t\right\| + \frac{\alpha\epsilon^2 L\Lambda^2}{2\lambda(2(1-\Delta)+\alpha m)^{\frac{1}{2}}}\left\|\mathbf{D}_{t-1}^{-\frac{1}{2}}\mathbf{g}_t\right\|^2, \tag{76}$$

where C_1 is defined as

$$C_1 := \left[\frac{\alpha\epsilon L\Lambda}{\lambda(2(1-\Delta)+\alpha m)^2}\right]^{\frac{1}{2}}. \tag{77}$$

The next step is to establish an upper bound for $\|\mathbf{g}_{t-1}\|^{1/2}$ in terms of the objective function error $F(\mathbf{y}_t)-F(\mathbf{y}^*)$. Observe that the eigenvalues of the Hessian are bounded above by $2(1-\delta)+\alpha M$. This bound in association with the Taylor's expansion of the objective function $F(\mathbf{y})$ around $\hat{\mathbf{y}}$ leads to

$$F(\mathbf{y}) \leq\ F(\hat{\mathbf{y}})+\nabla F(\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}})+\frac{2(1-\delta)+\alpha M}{2}\|\mathbf{y}-\hat{\mathbf{y}}\|^2. \tag{78}$$

According to the definition of λ in (27) we can substitute $1/(2(1-\delta)+\alpha M)$ by λ. Applying this substitution into (78) and minimizing the both sides of (78) with respect to $\mathbf{y}$ yields

$$F(\mathbf{y}^*) \leq\ F(\hat{\mathbf{y}})-\lambda\|\nabla F(\hat{\mathbf{y}})\|^2. \tag{79}$$

Since (79) holds for any $\hat{\mathbf{y}}$, we set $\hat{\mathbf{y}} := \mathbf{y}_{t-1}$. By rearranging the terms and taking their square roots, we obtain an upper bound for the gradient norm $\|\nabla F(\mathbf{y}_{t-1})\| = \|\mathbf{g}_{t-1}\|$ as

$$\|\mathbf{g}_{t-1}\| \leq \left[\frac{1}{\lambda}[F(\mathbf{y}_{t-1})-F(\mathbf{y}^*)]\right]^{\frac{1}{2}}. \tag{80}$$

The linear convergence of the objective function error implies that $F(\mathbf{y}_{t-1})-F(\mathbf{y}^*) \leq (1-\zeta)^{t-1}(F(\mathbf{y}_0)-F(\mathbf{y}^*))$ – see Theorem 1. Considering this inequality and the relation in (80) we can write

$$\|\mathbf{g}_{t-1}\|^2 \leq \frac{(1-\zeta)^{t-1}}{\lambda}(F(\mathbf{y}_0)-F(\mathbf{y}^*)). \tag{81}$$

The upper bound for the squared norm $\|\mathbf{g}_{t-1}\|^2$ in (81) shows that $\|\mathbf{g}_{t-1}\|^{1/2}$ is upper bounded by

$$\|\mathbf{g}_{t-1}\|^{\frac{1}{2}} \leq \left[\frac{(1-\zeta)^{t-1}}{\lambda}(F(\mathbf{y}_0)-F(\mathbf{y}^*))\right]^{\frac{1}{4}}. \tag{82}$$

By considering the definition of Γ_2 in (32) and substituting the upper bound in (82) for $\|\mathbf{g}_{t-1}\|^{1/2}$, we can update the right hand of (76) as

$$\left\|\mathbf{D}_t^{-\frac{1}{2}}\mathbf{g}_{t+1}\right\| \leq \left(1-\epsilon+\epsilon\rho^{K+1}\right)\left[1+C_2(1-\zeta)^{\frac{t-1}{4}}\right]\left\|\mathbf{D}_{t-1}^{-\frac{1}{2}}\mathbf{g}_t\right\| + \epsilon^2\Gamma_2\|\mathbf{D}_{t-1}^{-\frac{1}{2}}\mathbf{g}_t\|^2, \tag{83}$$

where $C_2 := C_1[(F(\mathbf{y}_0)-F(\mathbf{y}^*))/\lambda]^{1/4}$. Considering the definition of C_1 in (77), C_2 is given by

$$C_2 := \frac{(\alpha\epsilon L\Lambda)^{\frac{1}{2}}(F(\mathbf{y}_0)-F(\mathbf{y}^*))^{\frac{1}{4}}}{\lambda^{\frac{3}{4}}(2(1-\Delta)+\alpha m)}. \tag{84}$$

The explicit expression for C_2 in (84) and the definition of Γ_1 in (32) show that $C_2 = \Gamma_1$. This observation in association with (83) leads to the claim in (31).

References

[1] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network newton," in *Proc. Asilomar Conf. on Signals Systems Computers*, vol. (to appear). Pacific Grove CA, November 2-5 2014, available at http://arxiv.org/pdf/1412.3740.pdf.

[2] ——, "An approximate newton method for distributed optimization," 2014, available at http://www.seas.upenn.edu/∼aryanm/wiki/NN-ICASSP.pdf.

[3] ——, "Network newton–part i: Algorithm and convergence," 2015.

[4] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[5] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial Informatics*, vol. 9, pp. 427–438, 2013.

[6] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *Signal Processing, IEEE Transactions on*, vol. 56, no. 7, pp. 3122–3136, 2008.

[7] A. Ribeiro, "Ergodic stochastic optimization algorithms for wireless communication and networking," *Signal Processing, IEEE Transactions on*, vol. 58, no. 12, pp. 6369–6386, 2010.

[8] ——, "Optimal resource allocation in wireless communication and networking," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, pp. 1–19, 2012.

[9] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc wsns with noisy links?part i: Distributed estimation of deterministic signals," *Signal Processing, IEEE Transactions on*, vol. 56, no. 1, pp. 350–364, 2008.

[10] U. A. Khan, S. Kar, and J. M. Moura, "Diland: An algorithm for distributed sensor localization with noisy distance measurements," *Signal Processing, IEEE Transactions on*, vol. 58, no. 3, pp. 1940–1947, 2010.

[11] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM, 2004, pp. 20–27.

[12] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[13] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning," *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pp. 1543–1550, 2012.

[14] V. Cevher, S. Becker, and M. Schmidt, "Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics," *Signal Processing Magazine, IEEE*, vol. 31, no. 5, pp. 32–43, 2014.

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

[16] D. Jakovetic, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *Automatic Control, IEEE Transactions on*, vol. 59, no. 5, pp. 1131–1146, 2014.

[17] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *arXiv preprint arXiv:1310.7063*, 2013.

[18] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *arXiv preprint arXiv:1404.6264*, 2014.

[19] Q. Ling and A. Ribeiro, "Decentralized linearized alternating direction method of multipliers," *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 5447–5451, 2014.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[21] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[22] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: convergence analysis and network scaling," *Automatic Control, IEEE Transactions on*, vol. 57, no. 3, pp. 592–606, 2012.

[23] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization." *CDC*, pp. 5453–5458, 2012.