Fast Cooperative Distributed Learning

Dušan Jakovetić[1,2], José M. F. Moura[1], and João Xavier[2]

[1]Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA
[2]Instituto Superior Técnico (IST), Technical University of Lisbon, Lisbon, Portugal

***Abstract*— We consider distributed optimization where N agents in a network minimize the sum $\sum_{i=1}^{N} f_i(x)$ of their individual convex costs. To solve the described problem, existing literature proposes distributed gradient-like algorithms that are attractive due to computationally simple iterations k, but have a drawback of slow convergence (in k) to a solution. We propose a distributed gradient-like algorithm, that we build from the (centralized) Nesterov gradient method. For the convex f_i's with Lipschitz continuous and bounded gradients, we show that our method converges at rate $O(\log k/k)$. The achieved rate significantly improves over the convergence rate of existing distributed gradient-like methods, while the proposed algorithm maintains the same communication cost per k and a very similar computational cost per k. We further show that the rate $O(\log k/k)$ still holds if the bounded gradients assumption is replaced by a certain linear growth assumption. We illustrate the gains obtained by our method on two simulation examples: acoustic source localization and learning a linear classifier based on l_2-regularized logistic loss.**

I. INTRODUCTION

Motivated by applications in sensor networks and distributed learning, we consider distributed optimization setup in which each agent i (out of N agents) acquires data D_i to infer a vector quantity $x^\star \in \mathbb{R}^d$. Agents are situated in a generic, connected network; agent i's own data D_i give only a partial knowledge on $x^\star$, but the quantity $x^\star$ can be successfully reconstructed based on all agents' data. More formally, each agent i has a local convex cost function $f_i(x) = f_i(x; D_i)$ of the variable x (parameterized by D_i), known only to agent i. The goal is for each agent to find $x^\star$ that solves the unconstrained problem (See Figure 1 for an illustration of the problem on a $N = 6$-agent network):

$$\text{minimize} \quad \sum_{i=1}^{N} f_i(x) =: f(x). \tag{1}$$

Application examples of (1) include distributed learning of a linear classifier, e.g., [1], or distributed acoustic source localization in sensor networks, e.g., [2]. (See also Section II for further details on the two problems.)

To solve (1) or related problems, existing literature proposes distributed iterative gradient-like algorithms, see [3], and more recent references [4], [5]. With these algorithms, agents perform local gradient and consensus-type (averaging) computations and communicate with their immediate neighbors their current solution estimates. These algorithms are attractive due to computationally inexpensive and simple iterations k, but they converge slowly (in k) to a solution.

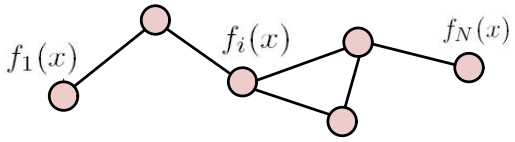

Fig. 1. An example of a connected network with $N = 6$ agents. Each agent i has a local convex cost function $f_i(x)$.

We propose a novel distributed gradient-like algorithm, based on the (centralized) Nesterov gradient method [6]. Our distributed algorithm maintains: 1) iterations' simplicity (computational cost per k); and 2) communication cost per k of existing methods [4], [5], but it significantly increases the convergence rate. Specifically, on the class of convex f_i's with Lipschitz continuous and bounded gradients, our algorithm achieves the convergence rate $O(\log k/k)$. In contrast, as shown in [1], the method in [4] cannot achieve a worst case rate better than $\Omega(1/k^{2/3})$ on the same class of cost functions. (See for details [1], equation (61).) We further show that the rate $O(\log k/k)$ is maintained if we replace the bounded gradients Assumption by a linear growth Assumption (See Assumption 3.) We corroborate numerically that our algorithm converges faster than existing methods on two examples: acoustic source localization in sensor networks and learning of a linear classifier based on l_2-regularized logistic loss.

Finally, we note that [1] proposed an algorithm with the convergence rate faster than $O(\log k/k)$; the algorithm uses the Nesterov gradient type update at the outer iteration level, and the consensus algorithm at the inner iteration level; the algorithm achieves the convergence rate $O(1/\mathcal{K}^{2-\xi})$, where $\xi > 0$ is arbitrarily small, and $\mathcal{K}$ is the number of per-agent communications. We refer to [1] for details and numerical comparisons with the algorithm studied here.

Paper organization. Section II explains the problem model and gives more details on the acoustic source localization and the classifier learning examples. Section III presents our distributed Nesterov gradient algorithm, and Section IV gives the results on its convergence rate. Section V illustrates

The work of the first and third authors is partially supported by: the Carnegie Mellon|Portugal Program under a grant from the Fundação para a Ciência e Tecnologia (FCT), Portugal; by FCT grants CMU-PT/SIA/0026/2009; and by ISR/IST plurianual funding (POSC program, FEDER). The work of the first and second authors is partially supported by NSF under grants CCF-1011903 and CCF-1018509, and by AFOSR grant FA95501010291. D. Jakovetić holds a fellowship from FCT.

the algorithm's performance on the two examples above. Finally, Section VI concludes the paper.

Notation. We denote by: $\mathbb{R}^d$ the d-dimensional real coordinate space, $d \geq 1$; A_{ij} the entry in the i-th row and j-th column of a matrix A; a_i the i-th entry of a vector a; $(\cdot)^\top$ the transpose; I the identity matrix; $\mathbf{1}$ the vector of appropriate dimension with unit entries; $J = \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ the ideal consensus matrix; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument ($\|\cdot\|$ also denotes the modulus of a scalar); $\nabla \mathcal{J}(x)$ the gradient evaluated at x of a function $\mathcal{J} : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$. Finally, for two positive sequences η_n and χ_n, $\eta_n = O(\chi_n)$ means that $\limsup_{n\to\infty} \frac{\eta_n}{\chi_n} < \infty$; $\eta_n = \Omega(\chi_n)$ means that $\liminf_{n\to\infty} \frac{\eta_n}{\chi_n} > 0$; and $\eta_n = \Theta(\chi_n)$ means that $\eta_n = O(\chi_n)$ and $\eta_n = \Omega(\chi_n)$.

II. Problem Model

Optimization model. We assume that N agents solve (1). The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only by agent i and has the following structure.

Assumption 1 (Cost functions: Basic assumption) (a) For all i, $f_i : \mathbb{R}^d \to \mathbb{R}$ is convex, and problem (1) is solvable.

(b) For all i, function f_i has Lipschitz continuous first derivative with constant $L \in [0, +\infty)$, i.e.,

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \ \forall x, y \in \mathbb{R}^d.$$

From Assumption 1 (b), e.g., [7]:

$$f_i(y) \leq f_i(x) + \nabla f_i(x)^\top (y - x) + \frac{L\|y - x\|^2}{2}, \forall x, y \in \mathbb{R}^d. \tag{2}$$

Also, under Assumption 1, the function $f(x) = \sum_{i=1}^N f_i(x)$ is also convex, and it has Lipschitz continuous gradient with constant NL. In addition to (1), we impose either one of the two following Assumptions.

Assumption 2 (Bounded gradients) There exists $G \in [0, \infty)$ such that, for all i, $\|\nabla f_i(x)\| \leq G$, for all $x \in \mathbb{R}^d$.

Assumption 3 (Linear growth) There exists a pair (b, B), $b > 0, B > 0$, such that, for all i:

$$f_i(x) \geq b\|x\| \text{ whenever } \|x\| \geq B.$$

Note that, under Assumption 3, the function f_i is coercive, i.e., $f_i(x) \to \infty$ whenever $\|x\| \to \infty$.

Denote by $x^\star$ a solution to (1), and by $f^\star = \inf_{x\in\mathbb{R}^d} f(x) = f(x^\star)$ the optimal value.

Communication model. We associate with problem (1) a network $\mathcal{V}$ of N agents, described by the graph $\mathcal{G} = (\mathcal{V}, E)$, where $E \subset \mathcal{V} \times \mathcal{V}$ is the set of links.

Assumption 4 (Network model) The graph $\mathcal{G}$ is connected, undirected, and simple (no self/multiple links.)

Denote by d_i the agent i's degree – the number of its neighbors. We associate to graph $\mathcal{G}$ a symmetric, stochastic (rows sum to one and all the entries are non-negative), $N \times N$ weight matrix W, with, for $i \neq j$, $W_{ij} > 0$ if and only if, $\{i, j\} \in E$, and $W_{ii} = 1 - \sum_{j\neq i} W_{ij}$. We require that $\mu := \|W - J\| < 1$, which is, for a connected $\mathcal{G}$, true for any W with strictly positive diagonal entries W_{ii}'s, $\forall i$. Further, we require that $\lambda_1(W) > 0$, i.e., the matrix W is positive definite. Under Assumption 4, a possible choice for W that ensures $\mu < 1, \lambda_1(W) > 0$, is: $W_{ij} = 1/(1 + 3\max\{d_i, d_j\})$, $\{i, j\} \in E$; $W_{ij} = 0$, $\{i, j\} \notin E$, $i \neq j$; and $W_{ii} = 1 - \sum_{j\neq i} W_{ij}$ (See [1] for details.) The latter weight choice requires only local knowledge at each agent, namely, the neighbors' degrees. We proceed with two application examples of problem (1) that we later study numerically in Section V.

Example 1: Acoustic source localization in sensor networks. Consider an acoustic source placed at an unknown location $\theta \in \mathbb{R}^2$. Each sensor (agents) i in a sensor network measures the received signal energy:

$$y_i = \frac{A}{\|\theta - r_i\|^\beta} + \zeta_i. \tag{3}$$

Here $r_i \in \mathbb{R}^2$ is agent i's location, known to agent i, $A > 0$ and $\beta > 0$ are constants known to all agents, and ζ_i is zero-mean additive noise. The goal is for each agent to estimate the source's position θ; see, e.g., [8]. A straightforward approach is to find the nonlinear least squares estimate $\overline{\theta} = x^\star$ by minimizing the following cost function (of the variable x):

$$\text{minimize} \quad \sum_{i=1}^N \left(y_i - \frac{A}{\|x - r_i\|^\beta}\right)^2 . \tag{4}$$

Problem (4) is nonconvex and hence difficult; still, it is possible to efficiently obtain a good estimator $\widehat{\theta}$ based on the data y_i, $i = 1, ..., N$, by solving the following *convex* problem:

$$\text{minimize} \quad \sum_{i=1}^N \operatorname{dist}^2(x, C_i) \ , \tag{5}$$

where C_i is the disk

$$C_i = \left\{ x \in \mathbb{R}^2 : \ \|x - r_i\| \leq \left(\frac{A}{y_i}\right)^{1/\beta} \right\},$$

and $\operatorname{dist}(x, C) = \inf_{y\in C} \|x - y\|$ is the distance from x to the set C. In words, (5) finds a point $\widehat{\theta}$ that has the minimal total squared distance from disks C_i, $i = 1, ..., N$. Formulation (5) is a variation on the formulation in [8]. Problem (5) fits our framework (1) with $f_i(x) = \operatorname{dist}^2(x, C_i)$.

Example 2: linear classifier based on l_2–regularized logistic loss. Training data is distributed across agents in the network; each agent has N_s data samples, $\{a_{ij}, b_{ij}\}_{j=1}^{N_s}$, where $a_{ij} \in \mathbb{R}^m$ is a feature vector and $b_{ij} \in \{-1, +1\}$ is the class label of the vector a_{ij}, e.g., [9]. For the purpose of future feature vector classifications, each agent wants to learn the linear classifier $a \mapsto \operatorname{sign}\left(a^\top x' + x''\right)$, i.e., to determine a vector $x' \in \mathbb{R}^m$ and a scalar $x'' \in \mathbb{R}$, based on all agents' data samples, that makes the best classification in a certain sense. Specifically, we seek $x' \in \mathbb{R}^m$ and $x'' \in \mathbb{R}$ that minimize a convex surrogate loss with respect

to $x = ((x')^\top, x'')^\top$:

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \lambda R(x') \ . \tag{6}$$

Here $\phi(z) = \log(1 + e^{-z})$ is the logistic loss function, $\lambda > 0$ is a parameter, and $R(x') = \|x'\|^2$ is the l_2-regularization. Problem (6) fits (1), with $f_i(x) := \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \frac{\lambda}{N} R(x')$.

III. Distributed Nesterov Gradient Method

Subsection III-A gives preliminaries on the centralized Nesterov gradient. Then, Subsection III-B presents our distributed Nesterov gradient method.

A. Fast centralized Nesterov gradient

Consider a convex differentiable function $\phi : \mathbb{R}^d \to \mathbb{R}$ that has Lipschitz continuous gradient with constant L. The goal is to find $z^\star \in \mathbb{R}^d = \arg\min_{z \in \mathbb{R}^d} \phi(z)$. (We assume that such a $z^\star$ exists.) The fast centralized gradient method [6] updates the solution estimate $z(k)$ and an auxiliary variable $w(k)$ as follows:

$$z(k) = w(k-1) - \alpha \nabla \phi(w(k-1)) \tag{7}$$

$$w(k) = z(k) + \beta_{k-1}\left(z(k) - z(k-1)\right), \tag{8}$$

for $k = 1, 2, ...$ and $z(0) = w(0) \in \mathbb{R}^d$. The constant step size $\alpha \leq 1/L$ and $\beta_k = k/(k+3)$, for $k = 0, 1, ...$ Compared with the standard gradient method $z(k) = z(k-1) - \alpha \nabla \phi(z(k-1))$, the Nesterov gradient introduces an auxiliary variable $w(k)$ and an inexpensive update (8). But, at the same time, it significantly increases the convergence rate (in the cost function optimality gap), from $O(1/k)$ to $O(1/k^2)$ [6].

B. Distributed Nesterov gradient algorithm

We now present our distributed algorithm to solve (1). The algorithm generates the sequence $(x_i(k), y_i(k))$, $k = 0, 1, 2, ...$, at each agent i, where $x_i(k)$ is agent i's solution estimate and $y_i(k)$ is an auxiliary variable (Compare with $z(k)$ and $w(k)$ with the centralized Nesterov gradient (7)–(8).) Given the initialization $x_i(0) = y_i(0)$, for all i, $k = 1, 2, ...$, our distributed Nesterov gradient algorithm at agent i is:

$$x_i(k) = \sum_{j \in O_i} W_{ij}\, y_j(k-1) - \alpha_{k-1} \nabla f_i(y_i(k-1)) \tag{9}$$

$$y_i(k) = x_i(k) + \beta_{k-1}\left(x_i(k) - x_i(k-1)\right). \tag{10}$$

Here, W_{ij} are the averaging weights (the entries of W), and O_i is the neighborhood set of agent i (including i). The step size α_k and the sequence β_k are:

$$\alpha_k = c/(k+1), \quad \beta_k = \frac{k}{k+3}, \ \ k = 0, 1, ..., \tag{11}$$

where $c > 0$ is a constant. Each agent i, at each iteration k, sends $y_i(k-1)$ to all neighbors $j \in O_i$; receives $y_j(k-1)$ from all neighbors $j \in O_i$; updates $x_i(k)$ by weight-averaging its own $y_i(k-1)$ and its neighbors variables $y_j(k-1)$, and performs a negative gradient step with respect to f_i; and updates $y_i(k)$ via (10). For notational simplicity, we assume throughout the initialization $x_i(0) = y_i(0) = x_j(0) = y_j(0) = 0$ for all i, j.

We now give an intuition behind algorithm (9)–(10) and relate it with the centralized Nesterov gradient method (7)–(8). For an exact penalty interpretation of (9)–(10), see [10]. Denote by

$$\overline{x}(k) = \frac{1}{N} \sum_{i=1}^{N} x_i(k) \tag{12}$$

$$\overline{y}(k) = \frac{1}{N} \sum_{i=1}^{N} y_i(k) \tag{13}$$

the (hypothetical) global averages of the agents' estimates. Using the fact that the matrix W is doubly stochastic, it is possible to show that $(\overline{x}(k), \overline{y}(k))$ evolve as:

$$\overline{x}(k) = \overline{y}(k-1) - \alpha_{k-1} \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(y_i(k-1)) \tag{14}$$

$$\overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right). \tag{15}$$

Thus, $(\overline{x}(k), \overline{y}(k))$ evolve "almost" according to the centralized Nesterov gradient with step-size α_{k-1}/N to minimize $f = \sum_{i=1}^{N} f_i$, except that the exact gradient $\nabla f(\overline{y}(k-1)) = \sum_{i=1}^{N} \nabla f_i(\overline{y}(k-1))$ is replaced by the inexact version $\sum_{i=1}^{N} \nabla f_i(y_i(k-1))$. The "amount" of inexactness is controlled by how close the $y_i(k-1)$'s are to the $\overline{y}(k-1)$. This insight gives us a hint on how to analyze convergence of (9)–(10): 1) establish convergence of the inexact centralized Nesterov gradient (14)–(15); and 2) establish the estimate of $\|y_i(k) - \overline{y}(k)\|$.

IV. Convergence rate results

We now present the $O(\log k/k)$ convergence rate result for algorithm (9)–(10) under: 1) Assumptions 1, 2, and 4; and 2) Assumptions 1, 3, and 4. For the proof under the first set of Assumptions, we refer to [1]; the proof under the second set of Assumptions will be pursued in a companion paper.

Theorem 1 Consider algorithm (9)–(10) with the step-size $\alpha_k = c/(k+1)$, and $c \leq 1/(2L)$. Let Assumptions 1 and 4; and either Assumption 2 or Assumption 3 hold. Then, at any agent i[1]:

$$\frac{1}{N}\left(f(x_i(k)) - f^\star\right) = O\left(\frac{\log k}{k}\right). \tag{16}$$

Under the first set of Assumptions (1, 2 and 4), an explicit expression for a constant C in the rate (16)[2] can be found in [1]. We now present how C depends on the number of agents N and the network topology. Formally, suppose that we have a sequence of $N \times N$ weight matrices $\{W^{(N)}\}_{N \geq 1}$, with $W^{(N)}$ positive definite and $\mu(N) := \|W^{(N)} - J\| < 1$, for all N, and $J = \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ with $\mathbf{1}$ the $N \times 1$ vector of unit entries.

[1] Although unnecessary, we normalize the cost optimality gap by N as is typical in the literature, e.g., [5]

[2] By a rate constant C, we refer to a quantity $C \in [0, \infty)$ independent of k that fulfills the following condition: $\frac{1}{N}\left(f(x_i(k)) - f^\star\right) \leq \frac{C \log k}{k}$, $\forall i, \forall k$.

Theorem 2 Consider algorithm (9)–(10) under Assumptions 1, 2, and 4. Further, suppose that $\alpha_k = \frac{1-\mu(N)}{k+1}$, $k = 0, 1, ...$, and that $\lambda_1(W^{(N)}) = \Omega(1)$.[3] Then, the constant C in the convergence rate (16) is $O\left(\frac{1}{(1-\mu(N))^{1+\xi}}\right)$, with $\xi > 0$ arbitrarily small.

Note that, to implement the step-size rule in Theorem 2, each agent i needs to know beforehand the quantity $\mu(N)$; see [1] for comments on how such knowledge can be obtained. From Theorem 2, we can derive how $C = C(N)$ depends on the number of agents for some commonly used network models. For example, for the weight choice below Assumption 4, we have that $C = C(N) = O(N^{2+\xi})$ (with $\xi > 0$ arbitrarily small) for a chain network, and $C = C(N) = O(1)$, for expander graphs [5].

V. Simulation examples

This Section illustrates the performance of our distributed Nesterov gradient algorithm (9)–(10) on two simulation examples: acoustic source localization and learning a linear classifier based on l_2-regularized logistic loss. Both examples demonstrate that algorithm (9)–(10) converges much faster than the standard distributed gradient method in [4].

Example 1: Acoustic source localization in sensor networks. The simulation setup is as follows. Each agent i acquires a single data sample y_i according to model (3). The coefficients $A = 1$ and $\beta = 2$; the true source's position is $(0.2, 0.2)^\top$; and the measurement noise ζ_i is zero mean, Gaussian, i.i.d. across sensors, with the standard deviation 0.5. In case that the measurement y_i is negative (due to adding a large negative noise ζ_i, we set $y_i = 0$.

The network has $N = 70$ agents (sensors) and 299 links and is modeled as a geometric graph. Sensors are deployed uniformly randomly on a unit square, and the sensor pairs whose distance is less than a radius are connected by an (undirected) edge.

Figure 2 plots the relative error averaged across agents $\left(= \frac{1}{N f^\star}\sum_{i=1}^N (f(x_i) - f^\star)\right)$, $f^\star \neq 0$, versus the iteration number k in a $\log_{10} - \log_{10}$ scale. We compare our algorithm (9)–(10) with the algorithm in [4]. With (9)–(10) , we set the step-size $\alpha_k = 1/(k+1)$; with [4], we set $\alpha_k = 1/[(k+1)^\tau]$, with $\tau \in \{1/10, 1/3, 1/2, 1\}$. We can see that our method converges much faster in k than the algorithm in [4] for any of the considered step-size choices (choices of τ). For example, for the target average relative error of 0.001, our algorithm takes about 500 iterations, while [4] requires about $14,000$ iterations. At the same time, both algorithms have the same communication cost per k and a similar computational cost per k. Also, from Figure 2, the rate of convergence (the slope of the log-log plot) is approximately $1/k^2$ with our method (9)–(10), while the best rate with [4] (among all considered choices of τ) is for $\tau = 1/2$ and is slightly worse than $1/k$.

Example 2: linear classifier based on l_2–regularized logistic loss. The simulation setup is as follows. Each agent i has one data sample $a_{ij} := a_i$. We generate a_i independently over i; each entry is drawn from the standard normal distribution. We generate the "true" vector $x^\star = ({x_1^\star}^\top, x_0^\star)^\top$ by drawing its entries independently from the standard normal distribution. The class labels are generated as

$$b_i = \text{sign}\left({x^\star}_1{}^\top a_i + {x^\star}_0 + \epsilon_i\right),$$

where the ϵ_i's are drawn independently from a normal distribution with zero mean and variance 3. The network is again a geometric network with $N = 20$ agents and 67 links.

Figure 3 plots the relative error averaged across agents $\left(= \frac{1}{N f^\star}\sum_{i=1}^N (f(x_i) - f^\star)\right)$, $f^\star \neq 0$, versus the iteration number k (in a $\log_{10} - \log_{10}$ scale) for our algorithm (9)–(10) and the algorithm in [4]. The step-sizes are chosen as in Example 1. We can see again that (9)–(10) converges faster than the method in [4]. For example, for the precision of 0.001, algorithm (9)–(10) takes about 80 iterations, while [4] requires about $1,100$ iterations.

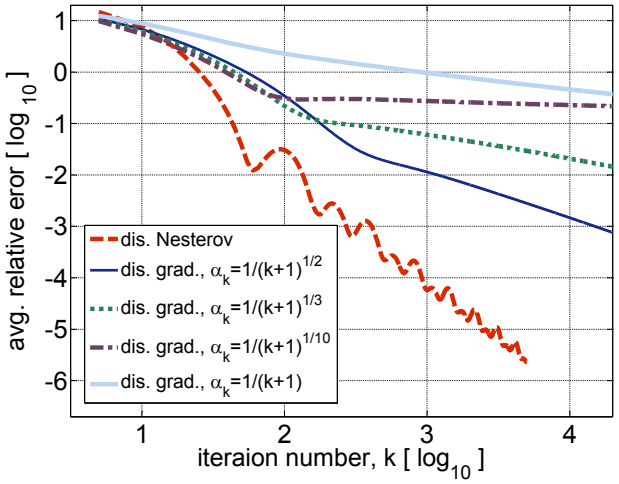

Fig. 2. **Example 1: Acoustic source localization in sensor networks**. Relative error averaged across agents: $\frac{1}{N f^\star}\sum_{i=1}^N (f(x_i) - f^\star)$, $f^\star \neq 0$, versus iteration number k in a $\log_{10} - \log_{10}$ scale for: 1) algorithm (9)–(10) with step size $\alpha_k = 1/(k+1)$; and 2) algorithm in [4] with $\alpha_k = 1/(k+1)^\tau$, $\tau \in \{1/10, 1/3, 1/2, 1\}$.

VI. Conclusion

We considered distributed optimization in networks where N agents minimize the sum of their individual convex costs. We proposed a distributed gradient-like algorithm based on the (centralized) Nesterov gradient method. We showed that the proposed algorithm converges at rate $O(\log k/k)$ when the cost functions are convex, with Lipschitz continuous gradient, and satisfy either the bounded gradients assumption or a certain linear growth assumption. We presented two simulation examples, namely acoustic source localization and learning a linear classifier based on the l_2-regularized logistic loss. Simulations corroborate the communication and computational gains of our algorithm when compared with standard distributed gradient methods.

[3] This condition is satisfied, e.g., with the weight example below Assumption 4.

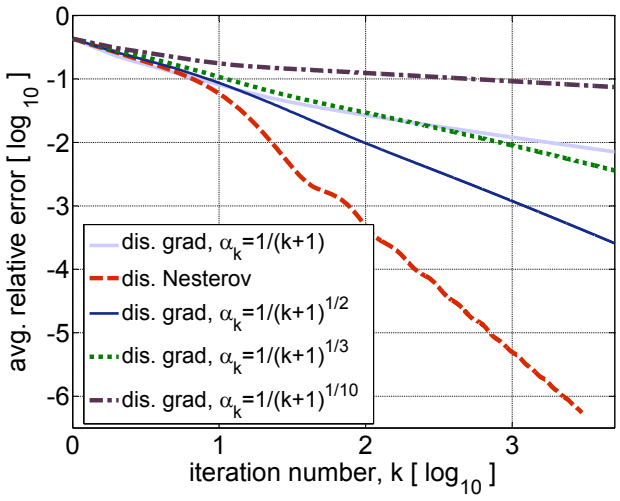

Fig. 3. **Example 2: linear classifier based on l_2–regularized logistic loss**. Relative error averaged across agents: $\frac{1}{N f^\star}\sum_{i=1}^N (f(x_i) - f^\star)$, $f^\star \neq 0$, versus iteration number k in a $\log_{10} - \log_{10}$ scale for: 1) algorithm (9)–(10) with step size $\alpha_k = 1/(k+1)$; and 2) algorithm in [4] with $\alpha_k = 1/(k+1)^\tau$, $\tau \in \{1/10, 1/3, 1/2, 1\}$.

REFERENCES

[1] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," November 2011, available at: http://arxiv.org/pdf/1112.2972.pdf.

[2] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *IPSN 2004, 3rd International Symposium on Information Processing in Sensor Networks*, Berkeley, California, USA, April 2004, pp. 20 – 27.

[3] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Contr.*, vol. 31, no. 9, pp. 803–812, Sep. 1986.

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

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

[6] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate O$(1/k^2)$," *Soviet Math. Doklady*, vol. 27, no. 2, pp. 372–376, 1983, translated from Russian by A. Rosa.

[7] L. Vandenberghe, "Optimization methods for large-scale systems," 2010, Lecture Notes, available at: http://www.ee.ucla.edu/ vandenbe/ ee236c.html.

[8] A. O. Hero and D. Blatt, "Sensor network source localization via projection onto convex sets (POCS)," in *ICASSP '05, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Philadelphia, PA, March 2005, pp. 2663–2668.

[9] 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, Michael Jordan, Editor in Chief*, vol. 3, no. 1, pp. 1–122, 2011.

[10] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Distributed Nesterov-like gradient algorithms," in *to appear in proc. CDC'12, 51st IEEE Conference on Decision and Control*, Maui, Hawaii, December 2012.